Identity Crises: Memorisation and Generalisation under Extreme Overparametrisation

Erfan Loweimi

Centre for Speech Technology Research (CSTR),
University of Edinburgh

Listen!; 8, Sep., 2020

Identity Crises:
Memorisation and Generalisation under Extreme Overparametrisation

Erfan Loweimi

Centre for Speech Technology Research (CSTR),
University of Edinburgh

Listen!; 8, Sep., 2020

ICLR 2020

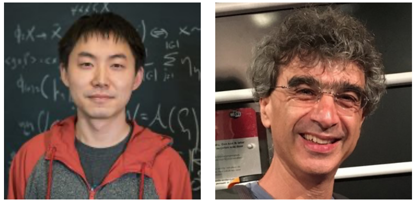
Identity Crisis: Memorization and Generalization under Extreme Overparameterization

Chiyuan Zhang & Samy Bengio
Google Research, Brain Team
Mountain View, CA 94043, USA
{chiyuan,bengio}@google.com

Moritz Hardt
University of California, Berkeley
Berkeley, CA 94720, USA
hardt@berkeley.edu

Michael C. Mozer
Google Research, Brain Team
Mountain View, CA 94043, USA
mcmozer@google.com

Yoram Singer
Princeton University
Princeton, NJ 08544, USA
y.s@princeton.edu



Google Brain

Berkeley
UNIVERSITY OF CALIFORNIA

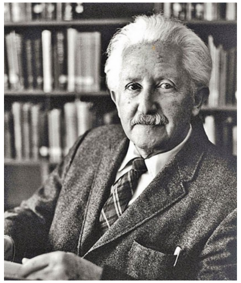
PRINCETON UNIVERSITY

Outlines

- Digression: Identity Crisis, inductive bias
- Motivation & Research Question
- Proposed Experimental Setup
- Experimental Results & Discussion
- Take-home messages

Identity Crisis

- **Term** coined by German-American **psychologi**st Erik Erikson
- **Definition**
  - *A period of uncertainty and confusion in which a person's sense of identity becomes insecure, typically due to a change in their expected aims or role in society.*



Erik Erikson
1902-1994

Inductive Bias

- **Definition**
  - A set of (implicit or explicit) assumptions made by the model to learn the target function and to generalise beyond training data
  - How a learning algorithm prioritise a solution over another, independent of data

- **Examples**
  - Linear relationship → $y = ax+b$ in the linear regression
  - Maximum Margin → SVM
  - Minimum Description Length → Simplest consistent hypothesis is the best
  - Neatest Neighbour → clustering and classification (kNN)

Occam's Razor

Motivation (1)

- [Big] Data is NOT the only reason behind success of DNNs
  - We were and still are in an overparametrised* zone!
  - Overparametrised models outperform simple models
- *“What form of inductive biases leads to better generalisation performance from highly overparametrised models?”*
- Numerous theoretical & empirical studies … BUT …
  - “… *these postmortem analyses do not identify the root source of the [inductive] bias.”*

Overparametrised: #param > #data

Why do DNNs Generalise?

- ✔ Gradient-based optimisation methods provide an implicit *bias* towards simple solutions ↔ Regularisation
  - – However, for a sufficiently large DNN Gradient methods are guaranteed to perfectly fit training set
    - Fitting could mean MEMORISATION, e.g. fitting random labels
- ✔ Generalisation guarantees for structures solved by linear or nearest neighbour classifier over original input space; Practicality?
- … and many more … BUT …
  - ➢ *“The fact that ... DNNs significantly outperform ... simpler models reveals a gap in our understanding of DNNs.”*

This paper ...

- **Goal**: Study the interplay of *memor.* and *Gener.*
- **Task**: Reconstruction of input (Regression)
  - NOT Auto-encoders, NO Bottleneck!
- **How**: Train a model using **ONLY one** training example
  - Extreme overparametrisation (#params >> #data=1)
- Question: What is the output?
  - Training example (\hat{x}), similar to input (x), sth else (???)

Output Types Analysis

- \hat{**x**} → Model learns a *constant* function
  - Mapping everything to a constant, regardless of x
  - Memorisation
- **x** → Model learns an *Identity* function
  - Identity mapping, regardless of similarity to \hat{x}
  - Generalisation
- Sth else → combination of x & \hat{x}, noise, ...

Experimental Setting

- Architectures: FCN*, CNN, ResNet (Appx. N)
- Database: digits and Fashion MNIST + CIFAR-10 (Appx. O)
- Loss function: MSE
- Optimisation:
  - Vanilla SGD (Appendix A), stepwise decay (factor: 0.2)@{30,60,80%} of training
  - Others: Adam, RMSprop, Adagrad, Adamax (Appendix I)
- Studied factors:
  - Depth, width (Appx. E), non-linearity, #channels, kernel size, Image size
  - Initialisation (Appx. I)

FCN*: Fully-Connected Net

Advantages of the Proposed Task

- Clear & unambiguous definition of memor. and gener.

- Analysis/visualisation of model behaviours & hidden layers
- Requires transmitting all input info to the output
- Investigation of architectures and hyperparameters is easy
- A simple form of conditional image generation

Trained using entire MNIST (digits)

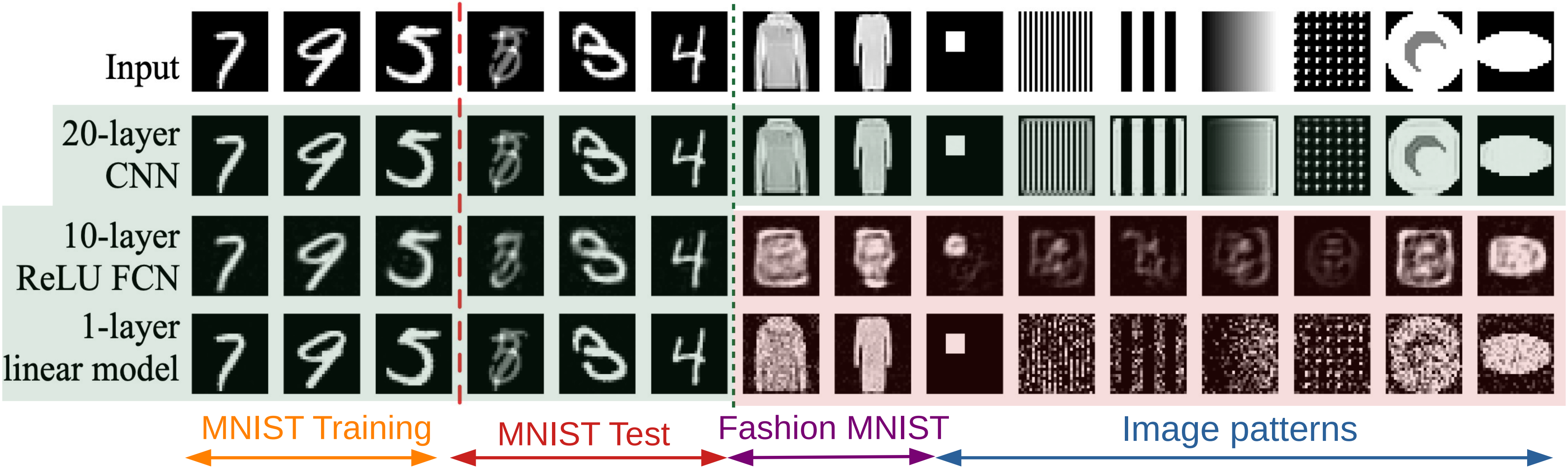


Fig. 1

- All nets work well on digits (even for blend & novel digits)
- For non-digit patterns, ONLY CNN learns identity function

Trained FCN using one digit (7)

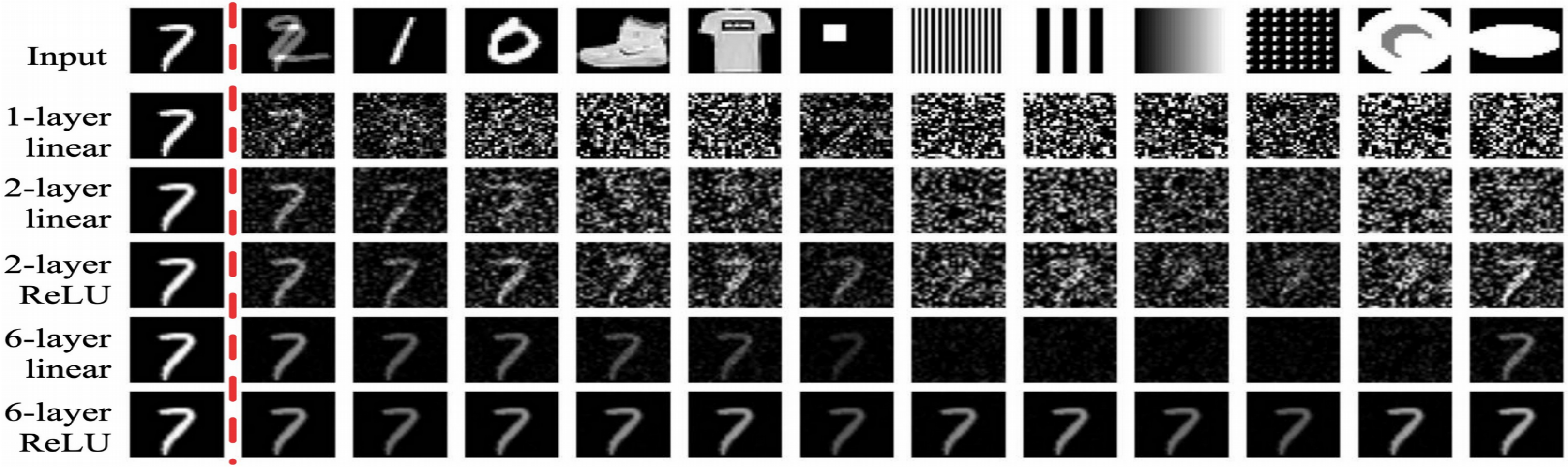


Fig. 2

- FCNs do NOT learn identity function (regardless of depth and non-lin)
- Shallower NNs biased towards outputting White noise
- Deeper NNs tends to learn a constant function (memorisation)

Theorem 1 (Proof in Appx. C)

- *A one-layer FCN, when trained with GD on a single training example \ hat{x}, converges to a solution that makes the following prediction (f(x)) on a test example x:*

R: random matrix

- *– Independent of data*
- *– Dependent on init.*

$$f(x) = \Pi_{\|}^{\hat{x}}(x) + R\Pi_{\perp}^{\hat{x}}(x)$$

$$\Pi_{\|}^{\hat{x}}(x) = x.\hat{x}\frac{\hat{x}}{\hat{x}} \quad and \quad x = \Pi_{\|}^{\hat{x}}(x) + \Pi_{\perp}^{\hat{x}}(x)$$

Parallel perpendicular decomposition

Theorem 1 for Multi-layer FCN

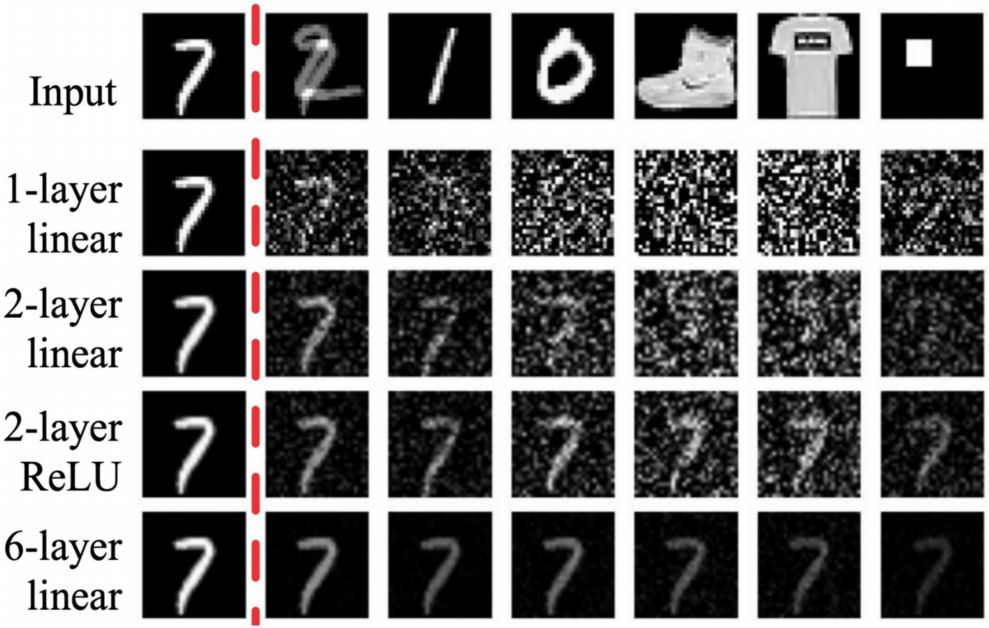
- Shallow networks tend to have similar inductive bias

$$f(x) = \Pi_{\parallel}^{\hat{x}}(x) + R\Pi_{\perp}^{\hat{x}}(x)$$

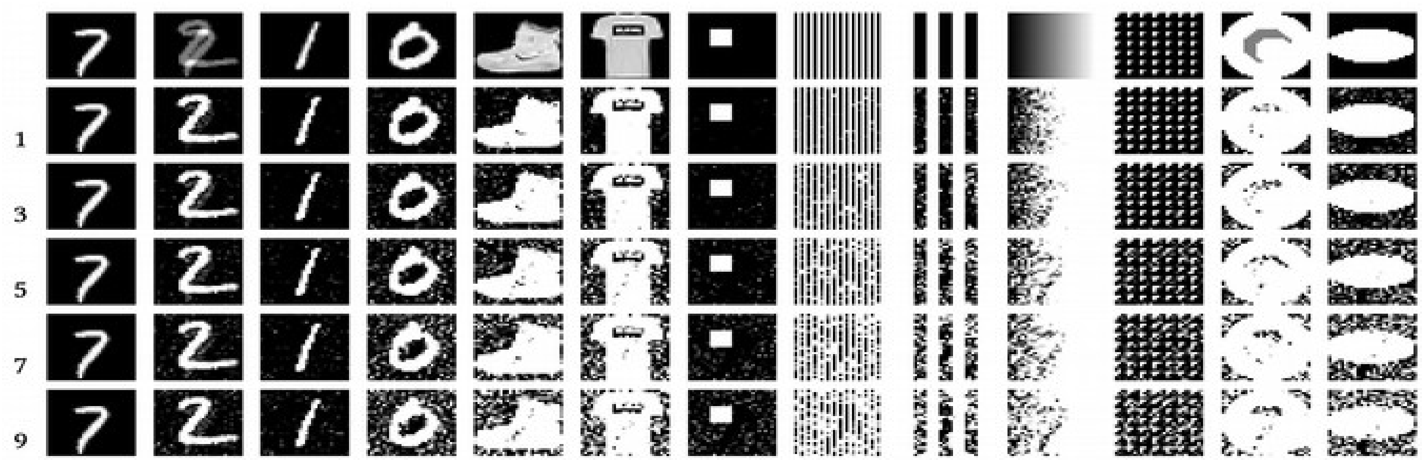
- 1L, 2L & 6L-linear FCNs have similar *representational powers* **BUT** different inductive biases!

- Shallower FCNs → noisier prediction

Fig. 2



ResNet: FCN + Skip Connection



Identity skip connection is added to every two FC layers ...

$X + \mathrm{ReLU}(W_2\,\mathrm{ReLU}(W_1 x))$

Fig. 41

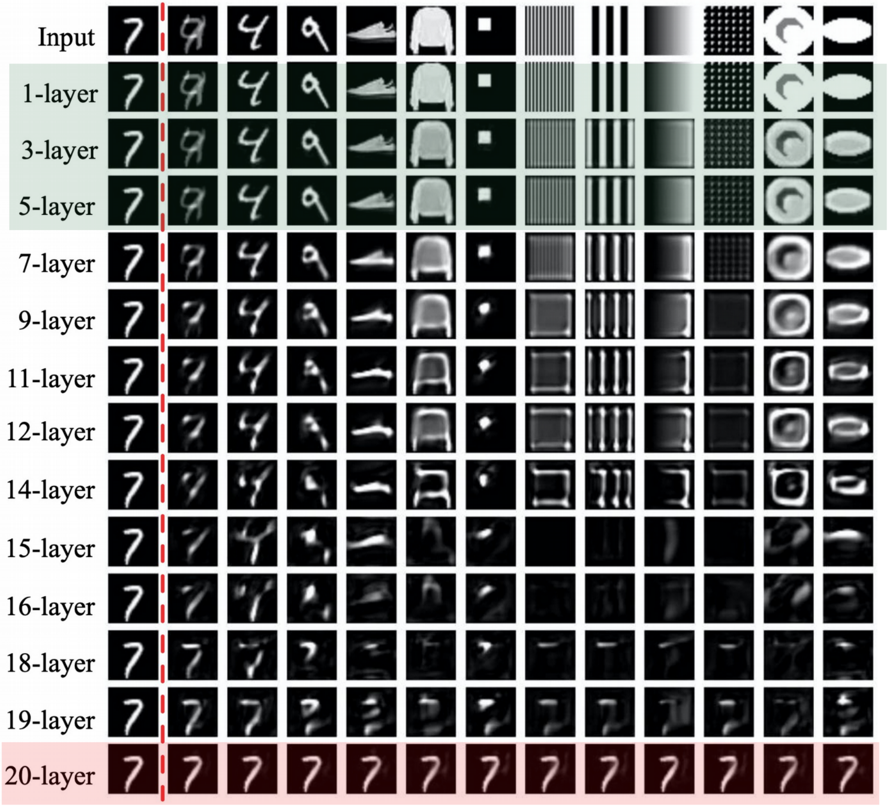
- Skip connection biases FCN towards learning identity map → better generalisation
- **Note**: Deeper structure → noisier prediction (contrary to FCN!)

Trained CNN using one digit (7)

Fig. 3

- Shallow (up to 5-layer) learns identity
- Very Deep (20-layer) learns constant
- Intermediate depth learns some edge detector (?)
  - NO White noise like FCNs!

- **Note**: output is not a continuum from identity to constant



All layers: 128 5x5 filters, stride=1, no pooling, padding=2 (with zero) [padding keeps size fixed]

Theorem 2 (Proof in Appx. D)

- *A one-layer CNN can learn the identity map from a single training example with the MSE over all output pixels bounded by*
  - m: #params ($k_w\, k_h\, C^2$), C: #channels in the image
  - r: rank of subspace formed by the span of local input patches; r ≤ m/C
    - Higher rank (richer context) → lower MSE (generalisation error (?))

$$\text{MSE} \leq \tilde{\mathcal{O}}\left(\frac{m(m/C-r)}{C}\right)$$

* *Big O tilde (Õ)* ignores log factor, e.g. for FFT → O(n log(n)) or Õ(n)

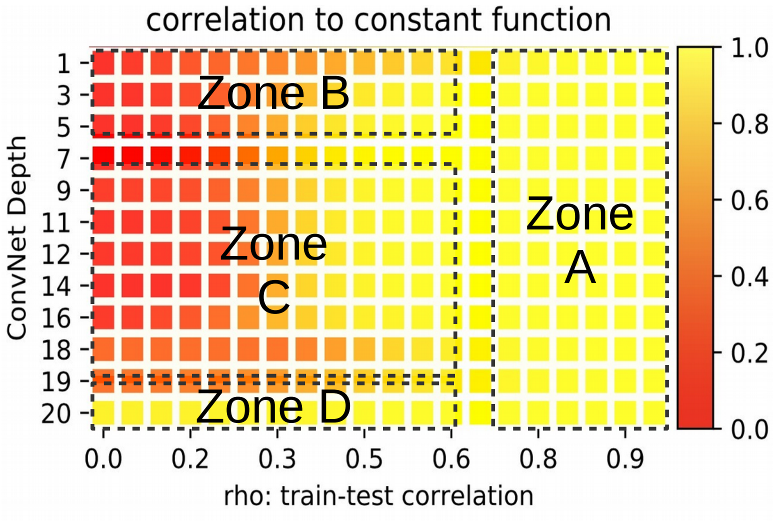
Effect of Similarity of Input & Output

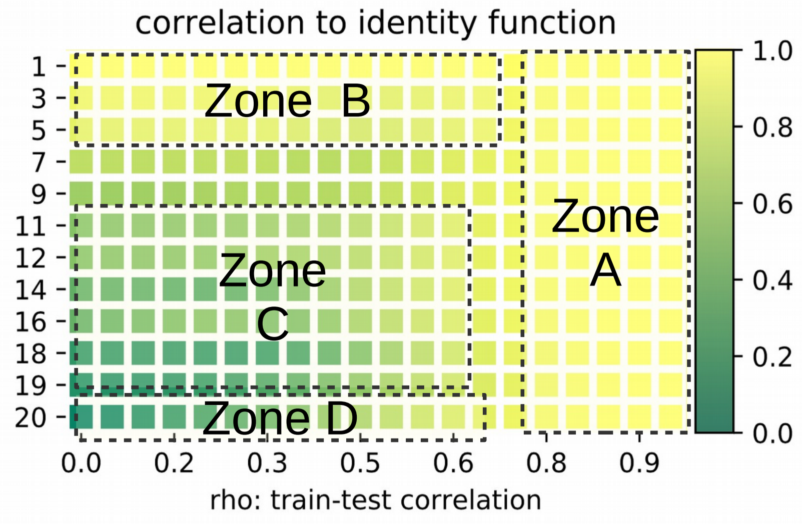
- Similarity measure: correlation
- Assume we can generate x, such that $corr(x, \hat{x}) = \rho$
  - $\rho \in [0,1]$

- Investigate
  - *Corr with identity* ↔ $corr(x, f(x))$
  - *Corr with constant* ↔ $corr(\hat{x}, f(x))$

Correlation with Constant/Identity

Fig. 4





- Zone A: depth not important, identity ≡ constant
- Zone B: Correlation w/ identity is high, w/ constant is low ↔ Generalisation
- Zone C: Correlation with constant: low; with identity: low ↔ Model hallucinates!
- Zone D: Correlation with constant: high; with identity: low ↔ Memorisation

How much info is lost across layers?

- **Goal**: Measure predictive power as a function of architecture depth and layer index
- **How** to measure this?
  - Build a similarity-weighted classifier using activations of each layer
  - Computed the classification error as a proxy for information
  - **Note**: This classifier is linear and is NOT a perfect proxy for info!
    - e.g. when data is nonlinearly-separable

Similarity-weighted Classifier

1. Feed the CNNs with (MNIST) training data: {$\boldsymbol{x}_j$, $\boldsymbol{y}_j$}
2. For each layer
    1. Dump the activations ∀ training data {$\boldsymbol{x}_j$ | $1 \leq j \leq N$}
    2. Build the *quasi-logit** ($\boldsymbol{y}_i$) for input ($\boldsymbol{x}_i$) as follows ...
    3. $c_i = \text{argmax}\ \boldsymbol{y}_i$

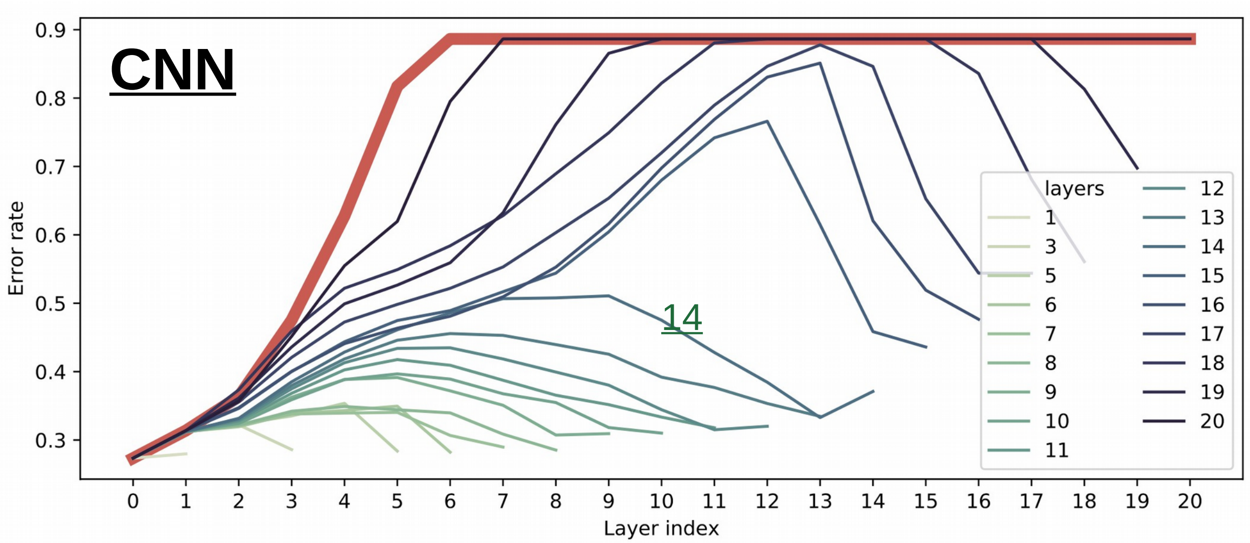
$$\mathbf{y}_i = \sum_{j=1}^{N} w_j \mathbf{y}_j, \quad \text{where} \quad w_j = \frac{\mathbf{x}_j^T \mathbf{x}_i}{|\mathbf{x}_j|\ |\mathbf{x}_i|}$$

one-hot

N: #training_data

* My term ;-)

Error vs Depth & Layer Index



90% Error [10 classes]

Red curve: untrained 20-layer CNN

Fig. 5

- Error vs L-index: first up (info lost), then down (info recovered)
- Deeper structure → further info loss at intermediate layers → less recovery chance
- Info loss across layers does NOT necessarily hinder reconstruction (redundancy)

Visualisation of intermediate Layers

7-layer trained

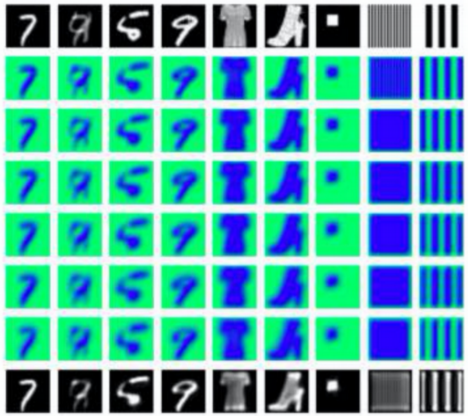
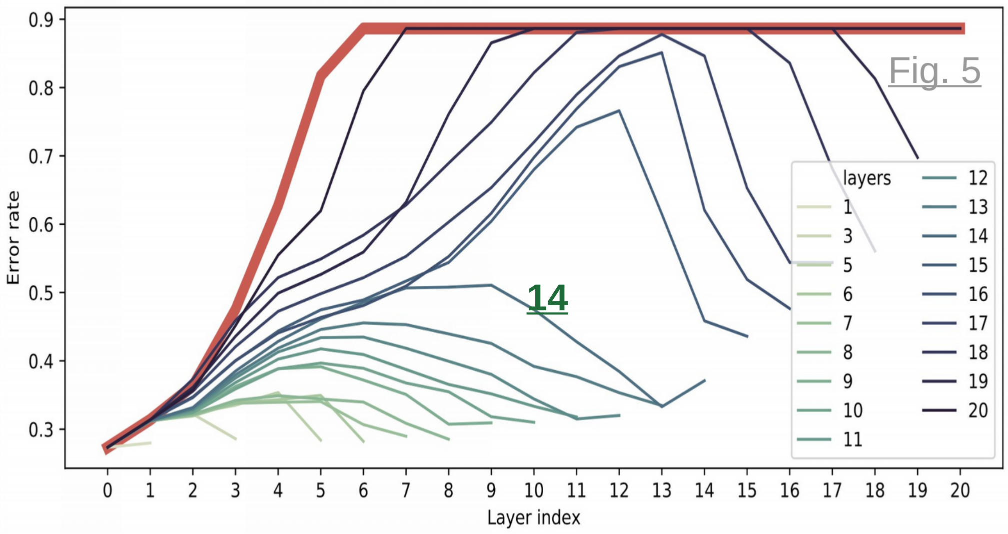


Fig. 15



14-layer trained

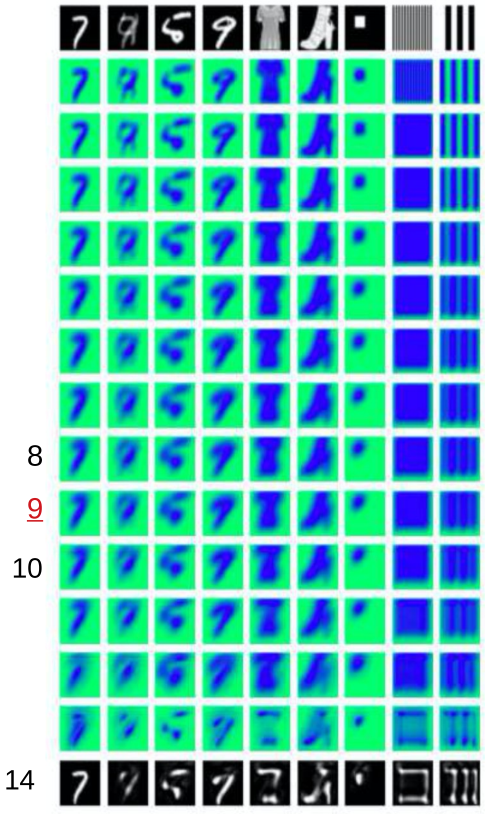


Fig. 15

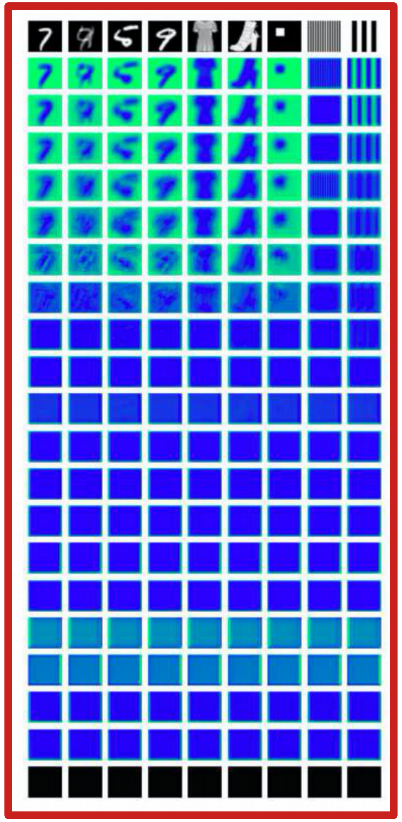
- Shallower CNNs → Intermediate layers are more active
- Reliability of error rate as an info proxy? Error-L9 is max, but ...

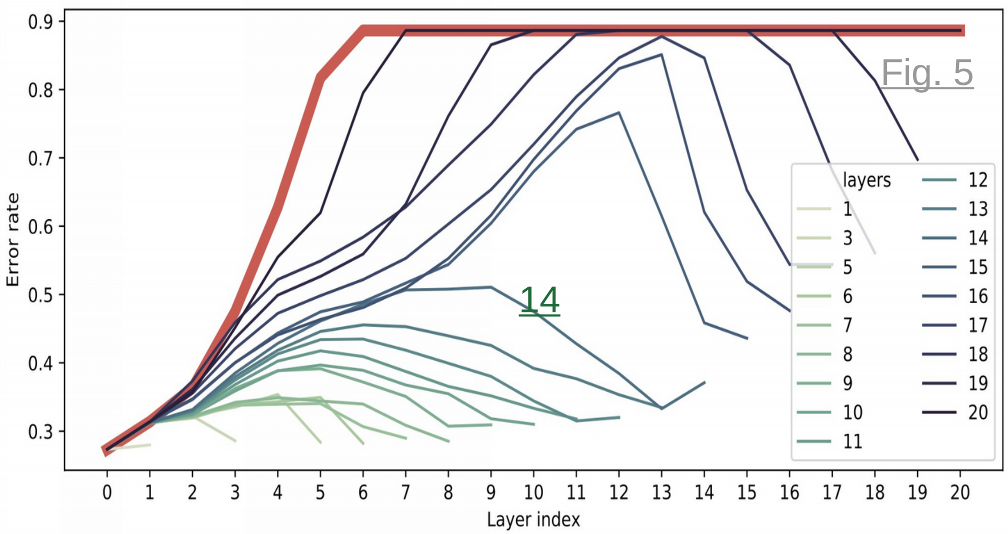
E. Loweimi

Visualisation of intermediate Layers

20-layer untrained

Fig. 15





- Intermediate layers are off (memorisation?)
- Only last layers are involved in generating constant output

20-layer trained

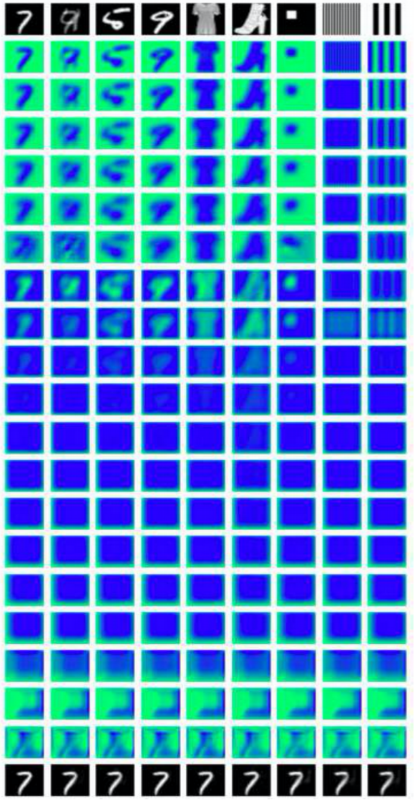
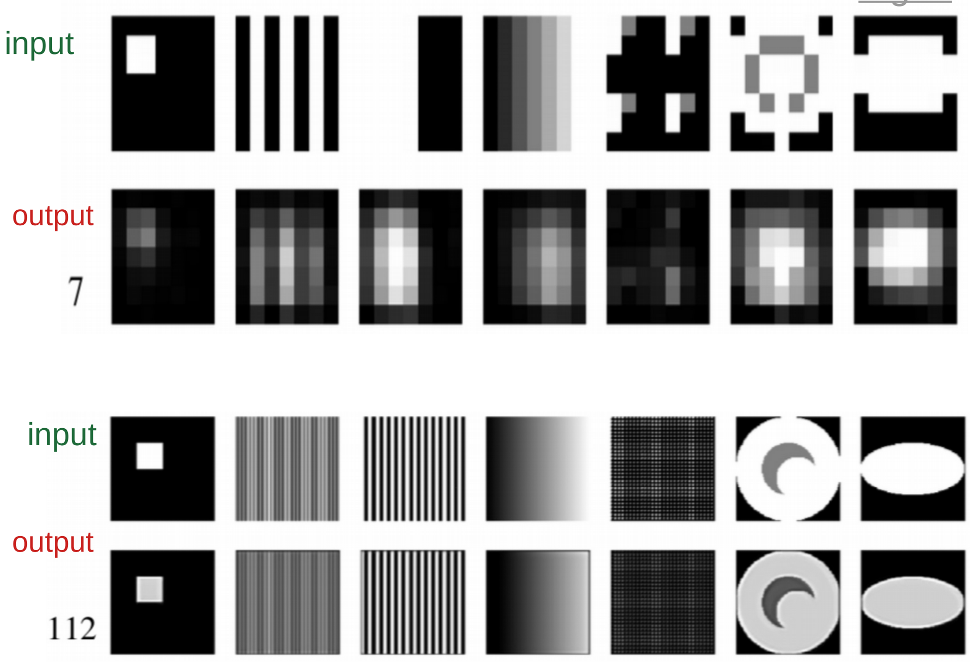


Fig. 15

Robustness to Image Size Change (1)

- 5-layer CNN trained with 28x28 images (learned identity mapping)
- Test with 7x7 and 112x112 images
- The learned identity mapping ...
  - **Disturbed** for smaller-than-trained input
  - **Held** for *larger-than-trained* input

Fig. 6



Robustness to Image Size Change (2)

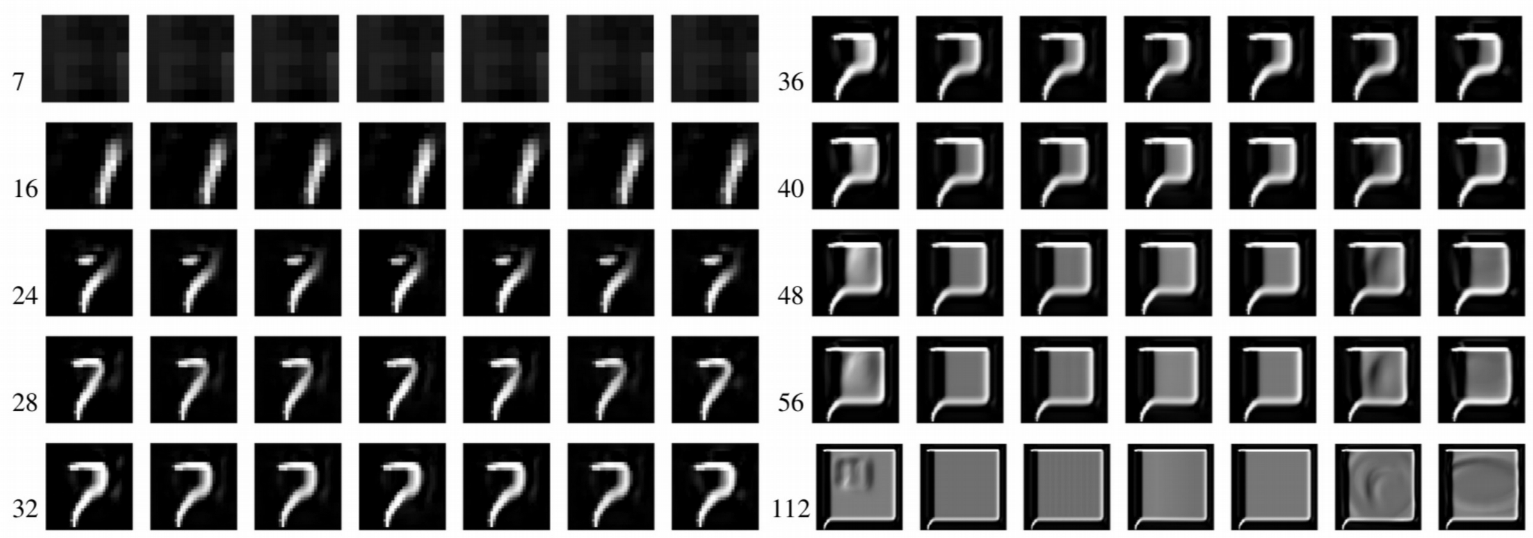


Fig. 7

- 20-layer CNN, trained on 28x28, learned constant function
- Smaller images → constant, but not exactly 7
- Larger images → constant, but distorted 7 (especially@corners, 0-padding?)

Training CNN with Different Image Size

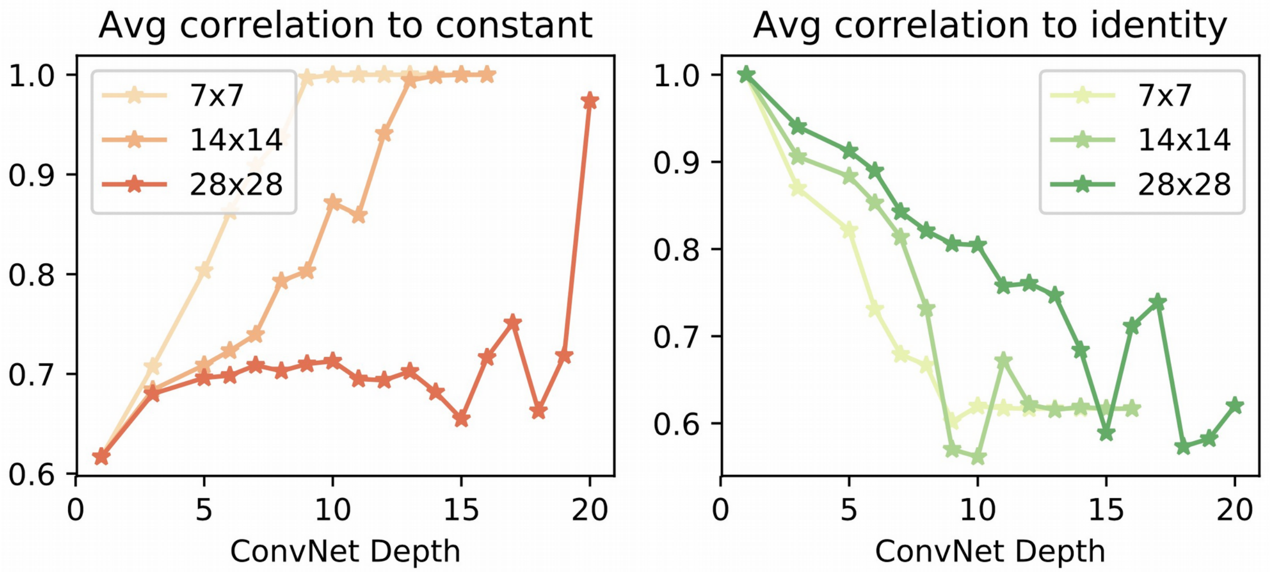
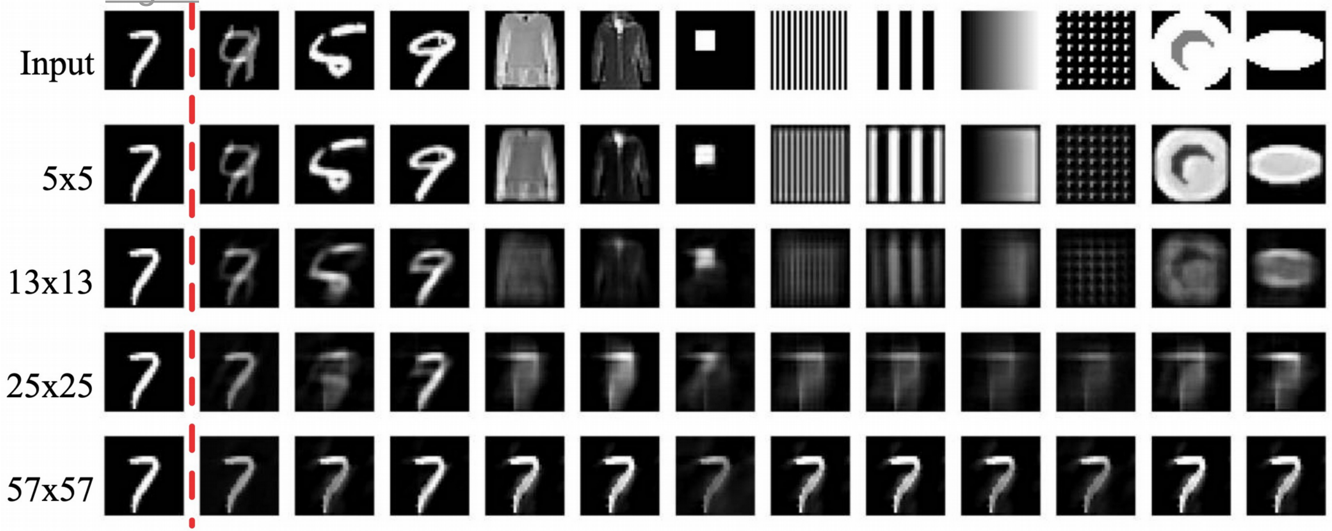


Fig. 8

- Training with smaller images → less spatial regularity/constraint
- Bias towards … const function increases ... identity decreases

Effect of Filter Size (5-layer CNN)

Fig. 9

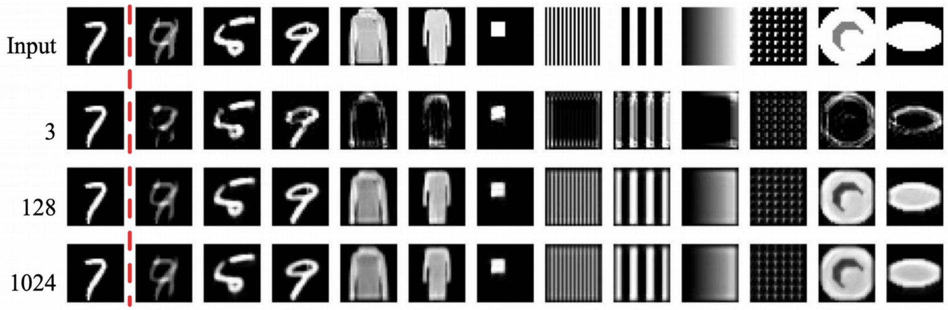


#params
1M
8M
30M
160M

- Larger filter size …
  - Blurrier prediction + Getting closer to a constant function

Effect of Number of Filters

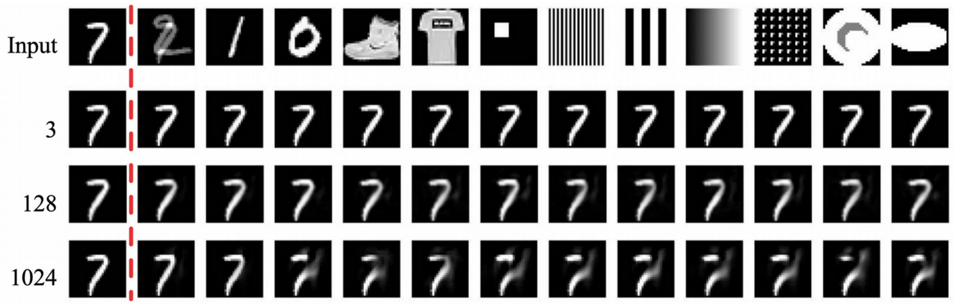
5-layer CNN



#params: 3→825, 128→1M, 1024→79M

20-layer CNN

Fig. 10



#params: 3→4.2k, 128→7M, 1024→471M

- Too deep net biased towards const function, regardless of #filters
- With proper depth, #filters does not affects bias towards identity
- **Note:** Model with 79M params generalises BUT one with 7M memorises

Takeaway Messages

- Why overparameterised DNNs magically avoid overfitting and generalise well?
- Task: input reconstruction (regression) using ONLY one training example
  - Learning … const map ↔ MEMORISATION; Identity ↔ GENERALISATION
- Shallow CNNs learn identity mapping; deep CNN learn const function
- FCNs, cannot learn identity function → more biased towards memorisation
- Skip connections help FCNs to learn identity mapping → improve gener.
- Increasing width/#channels cannot lead to overfit, contrary to increasing depth
- #params does NOT strongly correlates with generalisation performance

That's It!

- Thanks for your attention!
- Q/A?
- Appendices
  - A1. Initialisation Effect
  - A2. Optimisation Effect
  - A3. Training with two examples
  - A4. Training with three examples
  - A5. CIFAR-10

Initialisation Methods

$$\textbf{Yann}_n : \mathcal{N}(0, \frac{1}{f_i})$$

$$\textbf{Yann}_u : \mathcal{U}(-l, l) \leftarrow l = \sqrt{\frac{3}{f_i}}$$

$$\textbf{Orthogonal} : \mathcal{N}(0,1) \rightarrow \text{SVD} \rightarrow U * \text{scale}$$

$$\textbf{Default} : \mathcal{N}(0, \frac{1}{f_i f_o})$$

$$\textbf{Xavier}_n : \mathcal{N}(0, \frac{2}{f_i + f_o})$$

$$\textbf{Xavier}_u : \mathcal{U}(-l, l) \leftarrow l = \sqrt{\frac{6}{f_i + f_o}}$$

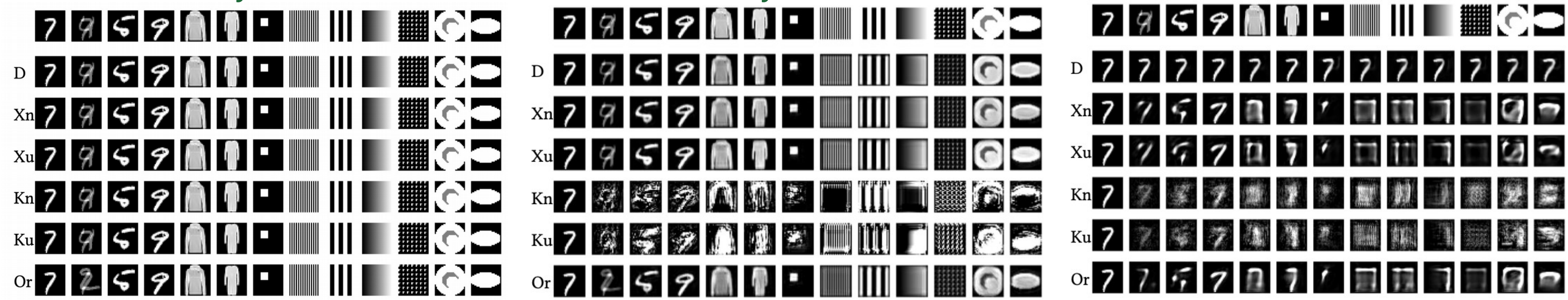
$$\textbf{Kaiming}_n : \mathcal{N}(0, \frac{2}{f_i})$$

$$\textbf{Kaiming}_u : \mathcal{U}(-l, l) \leftarrow l = \sqrt{\frac{6}{f_i}}$$

- *Yann Lecun et al., 1998*
- *Xavier Glorot et al., 2010*
- Orthogonal [*Andrew Saxe et al., 2014*]
- *Kaiming He et al., 2010*

Initialisation Effect – CNN

1-layer 5-layer 20-layer Fig. 31



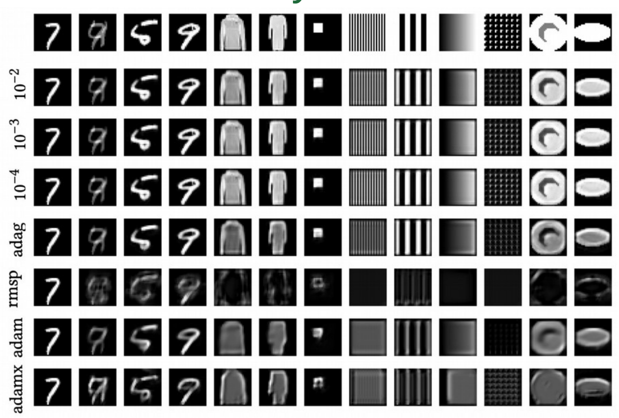
20-layer CNNs

- Initialisation matters … especially for deeper networks (?)
  - Xn, Xu and Orthogonal init. are equally good
  - Kaiming init. (Kn and Ku) creates some artifacts

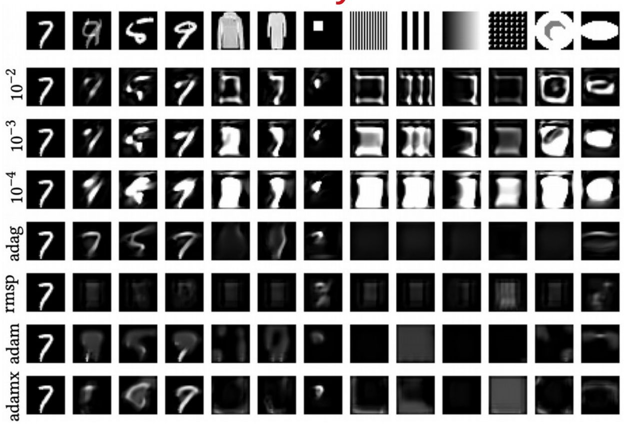
Optimisation Effect – CNN

Fig. 32

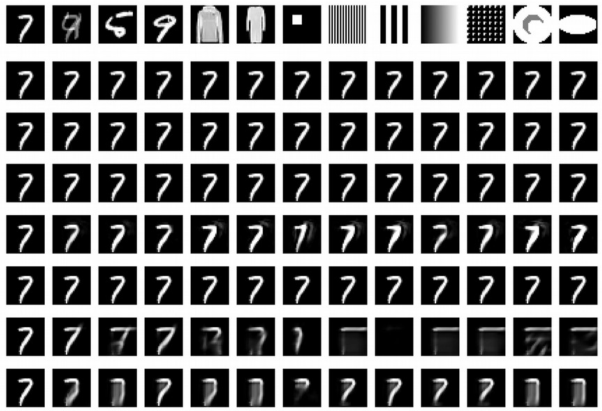
5-layer



14-layer



24-layer

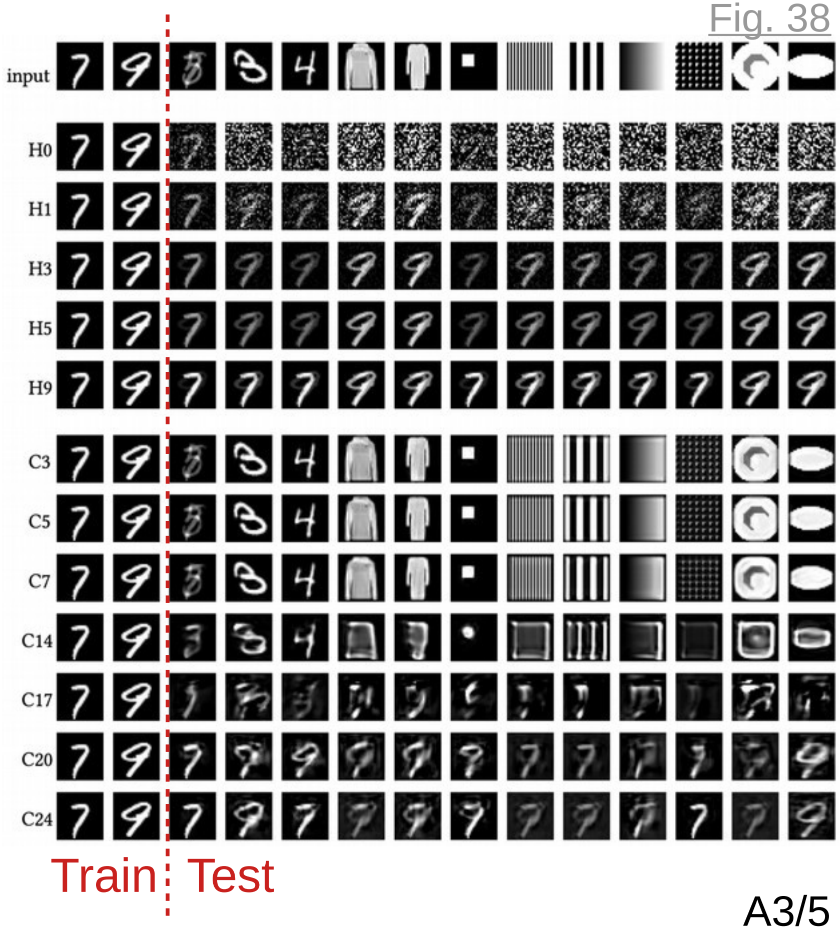


- SGD is better than fancier methods in terms of Generalisation
- … **BUT** … they have a better dynamics (converge faster)

Training with **Two** Examples; Similar ...

- FCNs learn **const** + noise
- CNNs learn ...
  - Shallow ↔ identity
  - Deep ↔ const
  - Int. ↔ edge detector (?)
- What is the const, here?
  - Interpolation, simpler pattern or ...

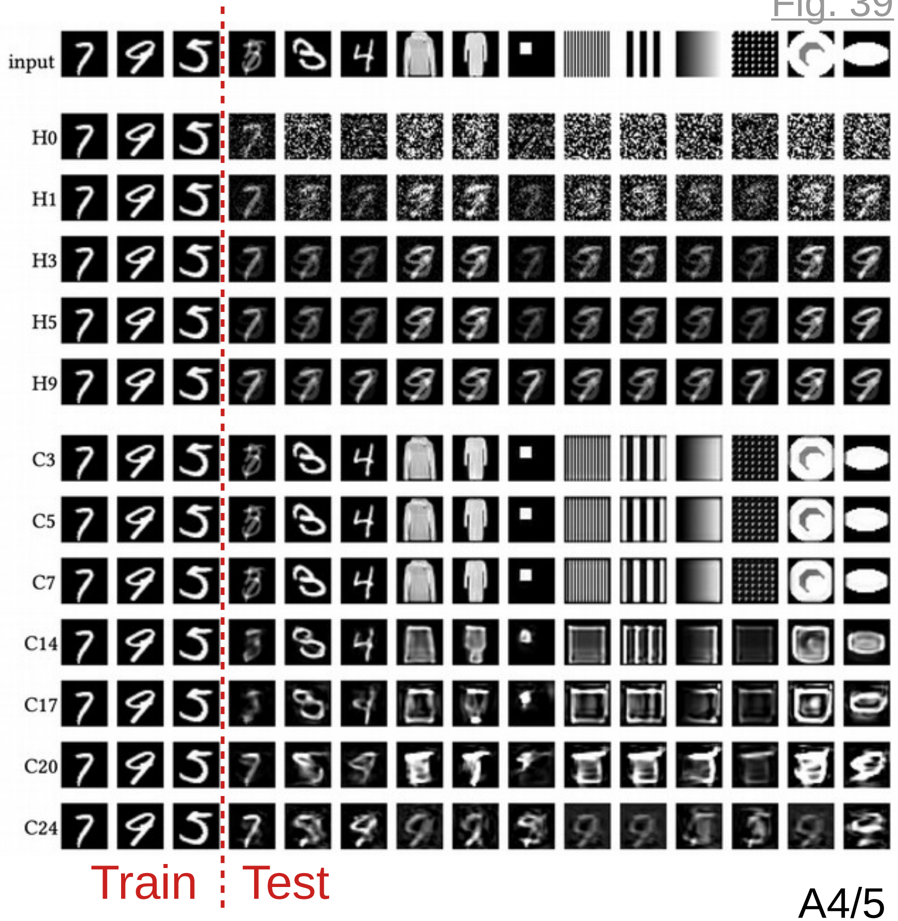
Fig. 38



Training with **Three** Examples; Similar ...

- FCNs learn **const** + noise
- CNNs learn ...
  - Shallow ↔ identity
  - Deep ↔ const
  - Int. ↔ edge detector (?)
- What is the const, here?
  - Interpolation, simpler pattern or ...

Fig. 39



CIFAR-10 – FCNs

i in Hi:
#hidden_Lay

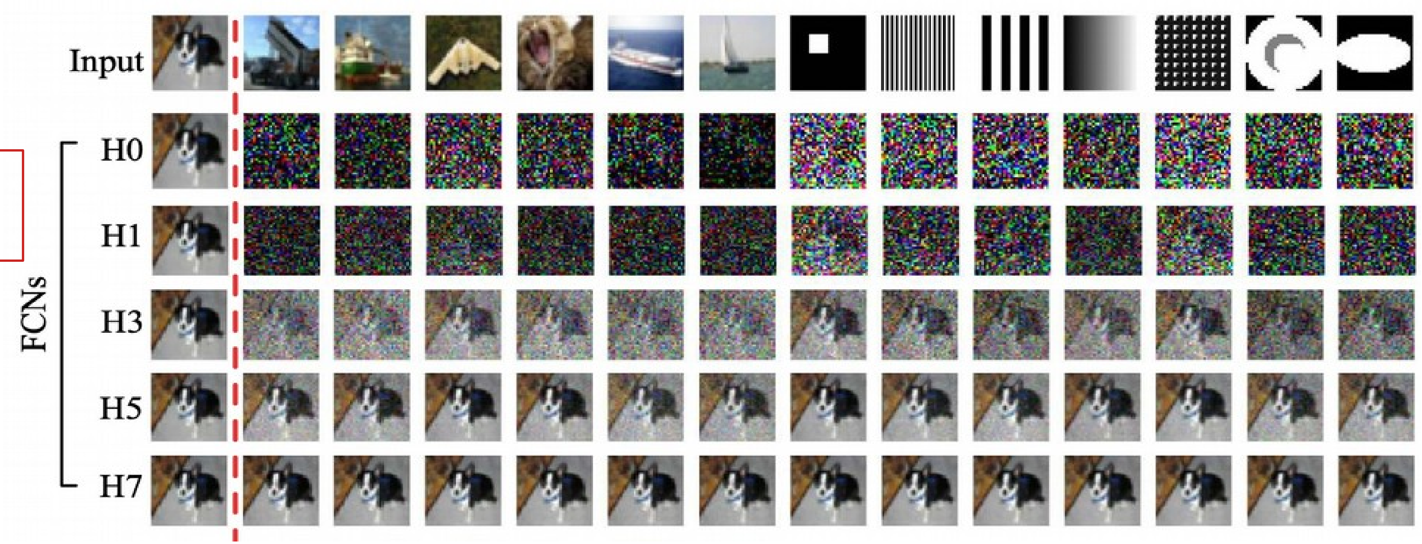


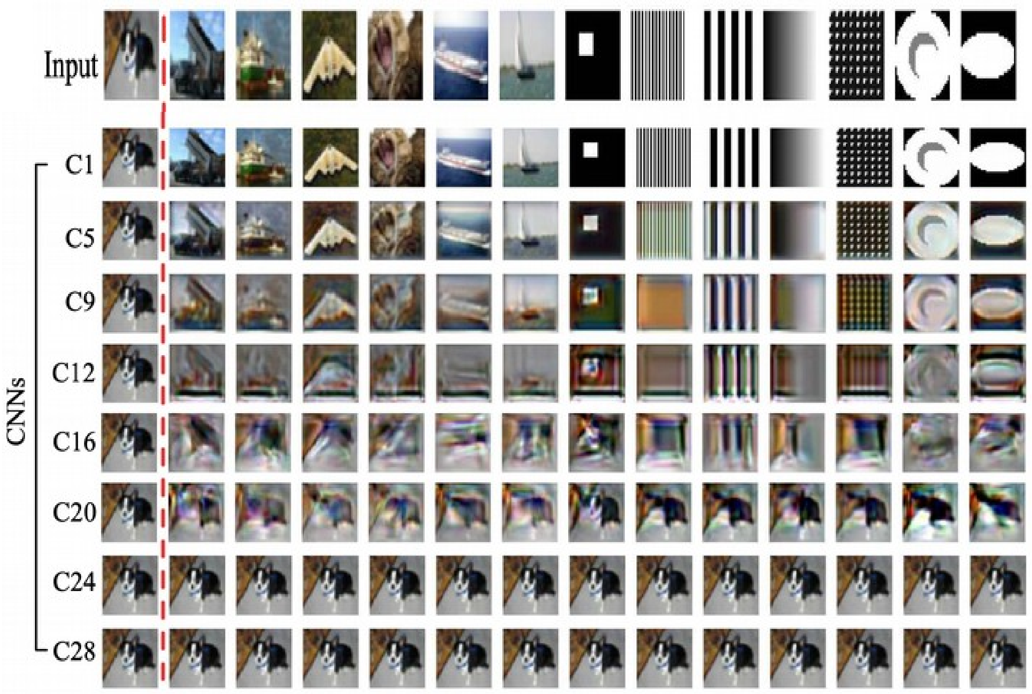
Fig. 42

- Similar to MNIST → output = training example + White noise
  - No Chance for learning identity mapping (generalisation)
  - Shallow network: White noise is dominant (hallucination)
  - Deeper network: training example is dominant (memorisation)

CIFAR-10 – CNNs

- Similar to MNIST ...
  - Shallow → learns identity
    - Generalisation
  - Deep → learns constant
    - Memorisation
  - Intermediate → edge detector
    - Hallucination

Fig. 42



i in Ci: #hidden_Layers 128 5x5 channels