Outline for today

- A geometric view of deep network expresivity, Raghu et al. 16' and Price et al. 19'.
- Understanding the filter activations on convolutional nets:
  The first layers are masks by which the image is compared against and responses recorded.
  The activation at a second layer is a linear combination of activations from the first layer and can be viewed as building simple components of an image.
  As depth increases the images that maximize an activation become more like specific images, sometimes referred to as a "grandmother cell".
- The first layer of CNNs are similar to wavelets, which is to be expected and is to be expected due to their near optimality and helps explain transferability

Summary expressivity of deep net: approximation theory

- Cybenko (89') showed that a single layer network with sigmoidal nonlinear activation can approximate any continuous function with arbitrary accuracy.
- Hornik (90') extended Cybenko's results to a much broader class of nonlinear activations, including ReLu.
- Telgarsky (15') used a specific deep network to construct a function and associated classification task to show that there are functions for which deep networks can exactly classify the data using polynomially many parameters (weights and bias), while exact reconstruction with a shallow network would require exponentially many parameters, otherwise has an $\mathcal{O}(1)$ fraction classification error.
- Yarotsky (17') showed ϵ approximation of smooth functions, such as x^2, on bounded domains with width being proportional to $\ln(1/\epsilon)$; extended to Sobolev spaces in $\mathbb{R}^d$.
- Bölcskei et al. (18'), Devore et al. (19') and many more...

Geometric view of data and expressivity: MNIST[2]

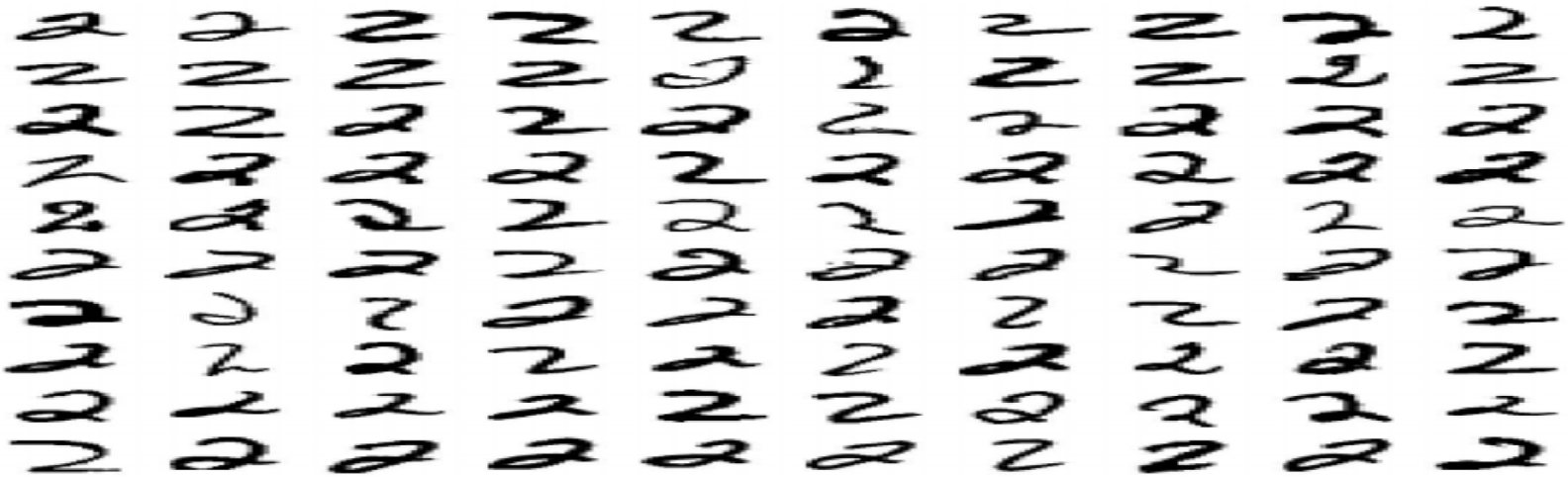


Table 7. Number of samples and estimated intrinsic dimensionality of the digits in MNIST.

1	2	3	4	5
7877	6990	7141	6824	6903
8/7/7	13/12/13	14/13/13	13/12/12	12/12/12
6	7	8	9	0
6876	7293	6825	6958	6903
11/11/11	10/10/10	14/13/13	12/11/11	12/11/11

[1]

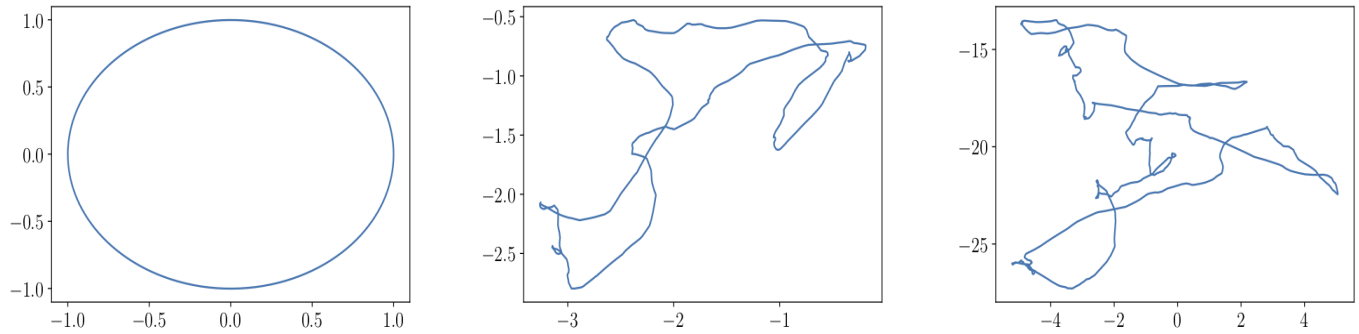
Deep nets are learning a vector valued function with prescribed values on complex, but low intrinsic dimensional, data.

[1] https://dl.acm.org/citation.cfm?id=1102388

[2] http://yann.lecun.com/exdb/mnist/

Geometric *generative perspective of expressivity (Price 19')*

Consider passing a simple object, such as a circle, through a random deep network. Below is the pre-activation output at layers 6 and 12.



(a) Input (b) Layer 6 (c) Layer 12

Figure 1: A circular trajectory, passed through a ReLU network with $\sigma_w = 2$. The plots show the pre-activation trajectory at different layers projected down onto 2 dimensions.

One can also use networks to *generative data, GANs, and there one might consider the complexity of the manifold the GAN can generate as a measure of expressivity.*

Expressivity through path length (Raghu et al. 16[3])

Consider an (untrained) Gaussian random feedforward network, that is its weights and bias are drawn i.i.d. $W^{(\ell)}(i,j) \sim \mathcal{N}(0, \sigma_w^2/k)$ and $b^{(\ell)}(j) \sim \mathcal{N}(0, \sigma_b^2)$, with n layers, each of width k and "hard tanh" nonlinear activations $\sigma(x) = -1$ for $x < 1$, $\sigma(x) = x$ for $x \in [-1, 1]$, and $\sigma(x) = 1$ for $x > 1$.

Theorem (Raghu et al. 16')

Consider as input a one dimensional trajectory $x(t)$ with arc-length $\ell(x(t)) = \int_t \left\| \frac{dx(t)}{dt} \right\| dt$ and let $z^{(n)}(t)$ be the output of the Gaussian random feedforward network with ReLu activations, then

$$\frac{\mathcal{E}\left[\ell(z^{(n)})\right]}{\ell(x(t))} \geq \mathcal{O}\left(\left(\frac{\sigma_w}{(\sigma_w^2 + \sigma_b^2)^{1/4}} \cdot \frac{k^{1/2}}{(k + (\sigma_w^2 + \sigma_b^2)^{1/2})^{1/2}}\right)^n\right).$$

The nonlinearity introduced by the path grows exponentially in depth, n, but only polynomially in width, k.

[3] https://arxiv.org/pdf/1611.08083.pdf

Expressivity through path length (Raghu et al. 16[4])

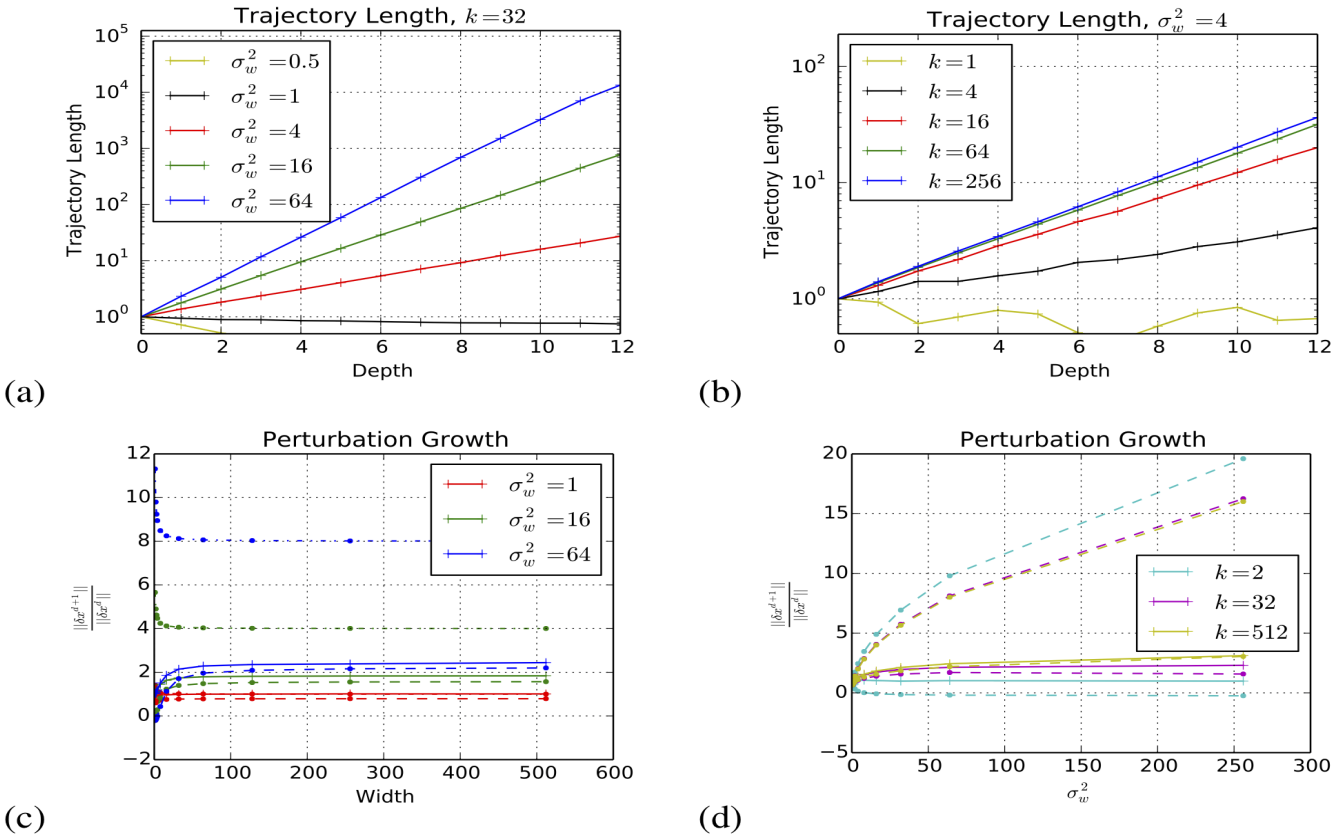


Figure 2: The exponential growth of trajectory length with depth, in a random deep network with hard-tanh nonlinearities. A circular trajectory is chosen between two random vectors. The image of that trajectory is taken at each layer of the network, and its length measured. *(a,b)* The trajectory length vs. layer, in terms of the network width k and weight variance σ_w^2, both of which determine its growth rate. *(c,d)* The average ratio of a trajectory's length in layer $d+1$ relative to its length in layer d. The solid line shows simulated data, while the dashed lines show upper and lower bounds (Theorem 1). Growth rate is a function of layer width k, and weight variance σ_w^2.

[4] https://arxiv.org/pdf/1611.08083.pdf

Expressivity through path length (Price et al. 19) pt. 1

Let the d-th post-activation layer of a ReLU deep net be $z^{(d)}$, and the subsequent pre-activation layer as $h^{(d)}$, such that

$$h^{(d)} = W^{(d)} z^{(d)} + b^{(d)}, \qquad z^{(d+1)} = \phi(h^{(d)}),$$

where $\phi(x) := \max(x, 0)$ is applied elementwise. Let $f_{NN}(x; \mathcal{P}, \mathcal{Q})$ denote a random feedforward deep net with weight matrices $W^{(d)}$ entries sampled iid from $\mathcal{P}$, and bias $b^{(d)}$ entries iid from $\mathcal{Q}$.

Theorem (Price et al. 19')

Let $f_{NN}(x; \alpha, \mathcal{P}, \mathcal{Q})$ be a random sparse net with layers of width k. Then, if $\mathbb{E}[|u^T \hat{w}_i|] \geq M\|u\|$, where $\hat{w}_i$ is restriction of any row of a weight matrix to its $\mathcal{P}$ distributed entries, and u and M are constants, then $\mathbb{E}[l(z^{(d)}(t))] \geq \left(\frac{\alpha M \sqrt{k}}{2}\right)^d \cdot l(x(t))$

for $x(t)$ a 1-dimensional trajectory in input space.

Expressivity through path length (Price et al. 19) pt. 2

Theorem (Price et al. 19')

Let $f_{NN}(x; \alpha, \mathcal{P}, \mathcal{Q})$ be a random sparse net with layers of width k. Then, if $\mathbb{E}[|u^T \hat{w}_i|] \geq M\|u\|$, where $\hat{w}_i$ is restriction of any row of a weight matrix to its $\mathcal{P}$ distributed entries, and u and M are constants, then

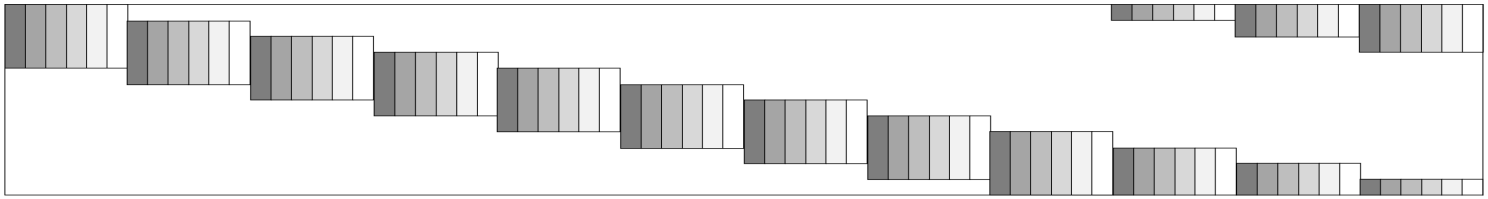
$$\mathbb{E}[l(z^{(d)}(t))] \geq \left(\frac{\alpha M \sqrt{k}}{2}\right)^d \cdot l(x(t))$$

for $x(t)$ a 1-dimensional trajectory in input space.

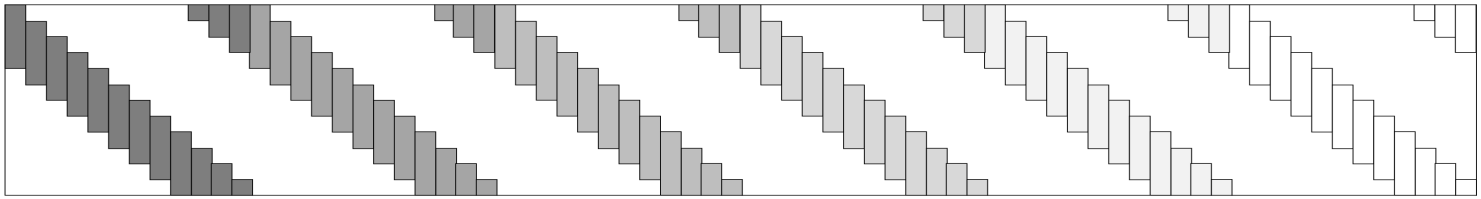
This theorem bounds exponential growth with depth for random initializations such as Gaussian, uniform, and discrete; e.g. for Gaussian with variance σ_w we have $M = \sigma_w \sqrt{2/\pi}$.

Convolutional Neural Networks:

Convolutional neural network layers impose a structure on $W^{(i)}$:



(a) A convolutional matrix.



(b) A concatenation of banded and Circulant matrices.

5

$W^{(i)}$ is composed of a "mask" (usually of compact support, say just living on 9 pixels) translated by some amount which is referred to as "stride." These "masks" are sometimes referred to as "features." Additional nonlinearities include "max pooling."

[5] https://arxiv.org/pdf/1607.08194.pdf

Exemplar CNN structure: LeNet5 LeCun et. al. 1998

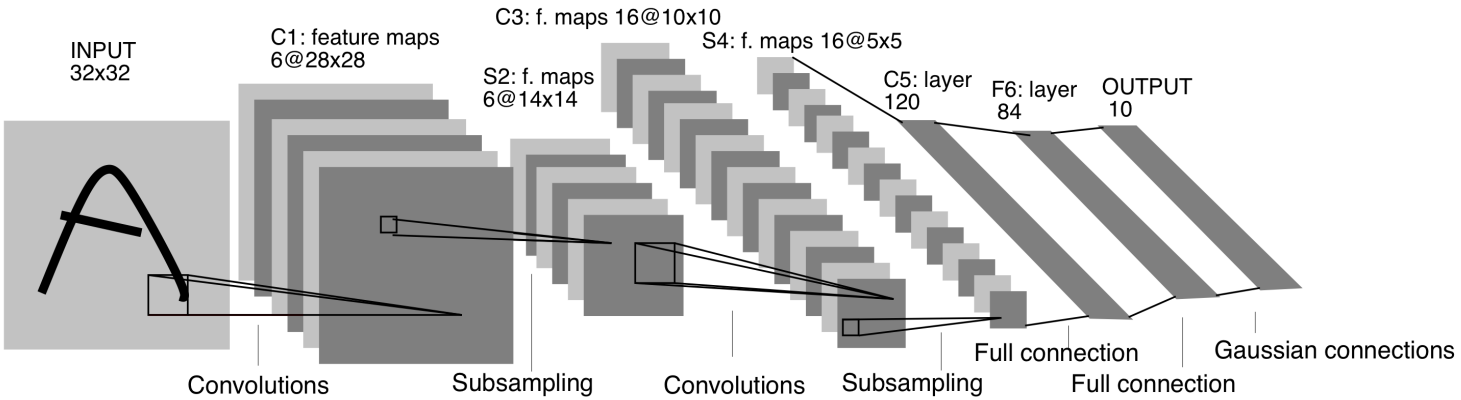


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

[6]

C1: conv. layer with 6 feature maps, 5 by 5 support, stride 1.

S2 (and S4): non-overlapping 2 by 2 blocks which equally sum values, mult by weight and add bias.

C3: conv. layer with 16 features, 5 by 5 support, partial connected.

C5: 120 features, 5 by 5 support, no stride; i.e. fully connected.

F6: fully connected, $W \in \mathbb{R}^{84\times 120}$.

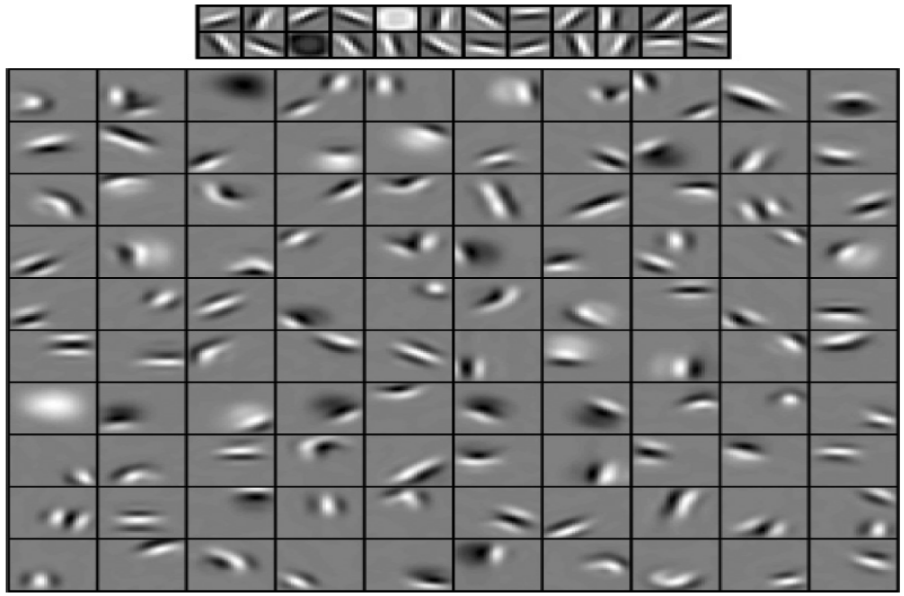
CNNs trained on a data set typically work well on other, related, data sets with different labels by retraining just the last layer. Why might this be? What do the conv. filters look like?

[6] http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

Convolutional Deep Belief Networks(H. Lee et al. 11'[8])

We omit the details of this somewhat different architecture, which is stylistically similar to a deep CNN.

Figure 3. The first layer bases (top) and the second layer bases (bottom) learned from natural images. Each second layer basis (filter) was visualized as a weighted linear combination of the first layer bases.



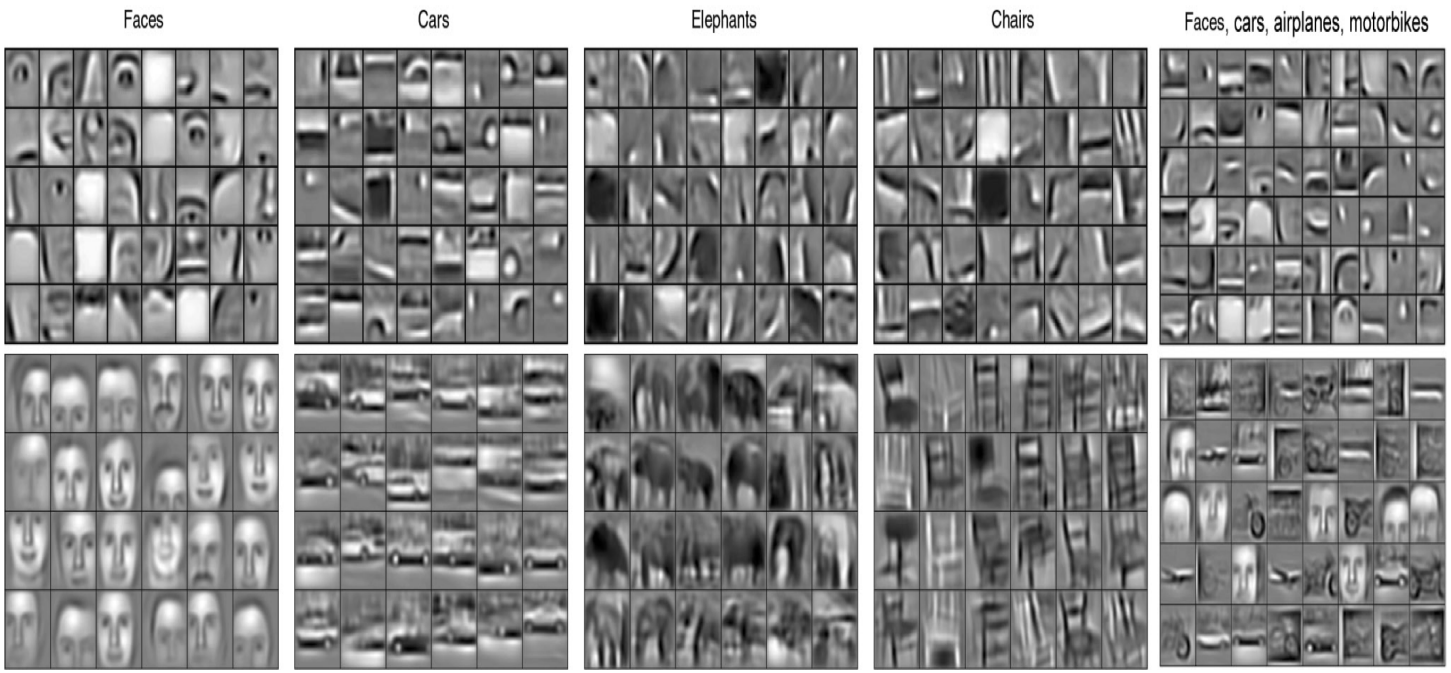
Display of the convolutional masks in layers 1 and 2, trained from Kyoto natural image database.[7] Note wavelet structures.

[7] http://eizaburo-doi.github.io/kyoto_natim/

[8] http://www.cs.utoronto.ca/~rgrosse/cacm2011-cdbn.pdf

Convolutional Deep Belief Networks(H. Lee et al. 11'[10])

Figure 4. Columns 1–4: the second layer bases (top) and the third layer bases (bottom) learned from specific object categories. Column 5: the second layer bases (top) and the third layer bases (bottom) learned from a mixture of four object categories (faces, cars, airplanes, motorbikes).

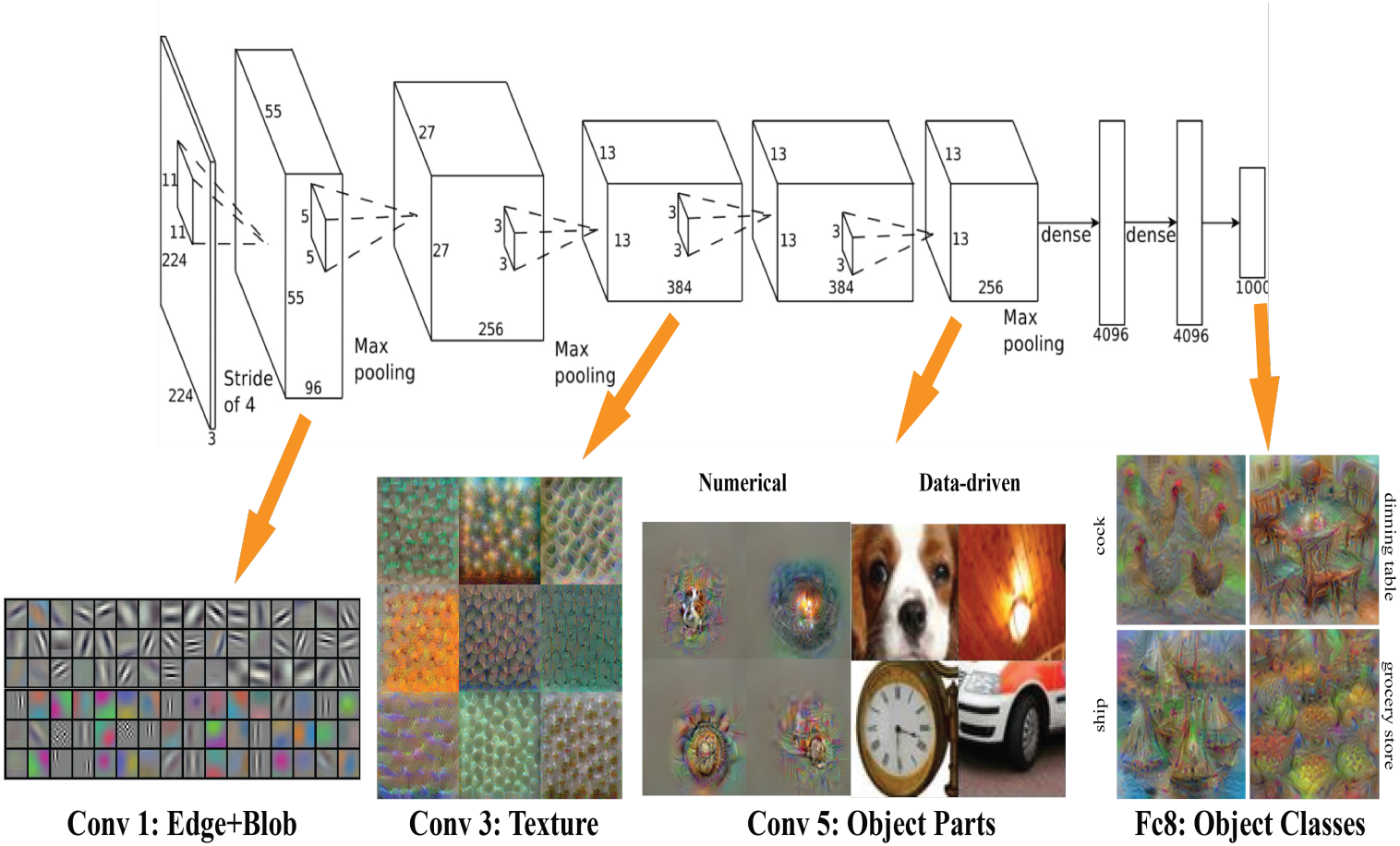


The third and fourth layers develop bases which represent features or objects, trained on CalTech 101 dataset.[9].

[9] http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[10] http://www.cs.utoronto.ca/~rgrosse/cacm2011-cdbn.pdf

Deep CNN, AlexNet(Krizhevsky et al. 12'[11])



Images are those that maximize specific activation responses.

Layer 1 are masks, subsequent layers are their linear combinations.

[11] http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Deep CNN, VGG(Mahendran et al. 16'[12])

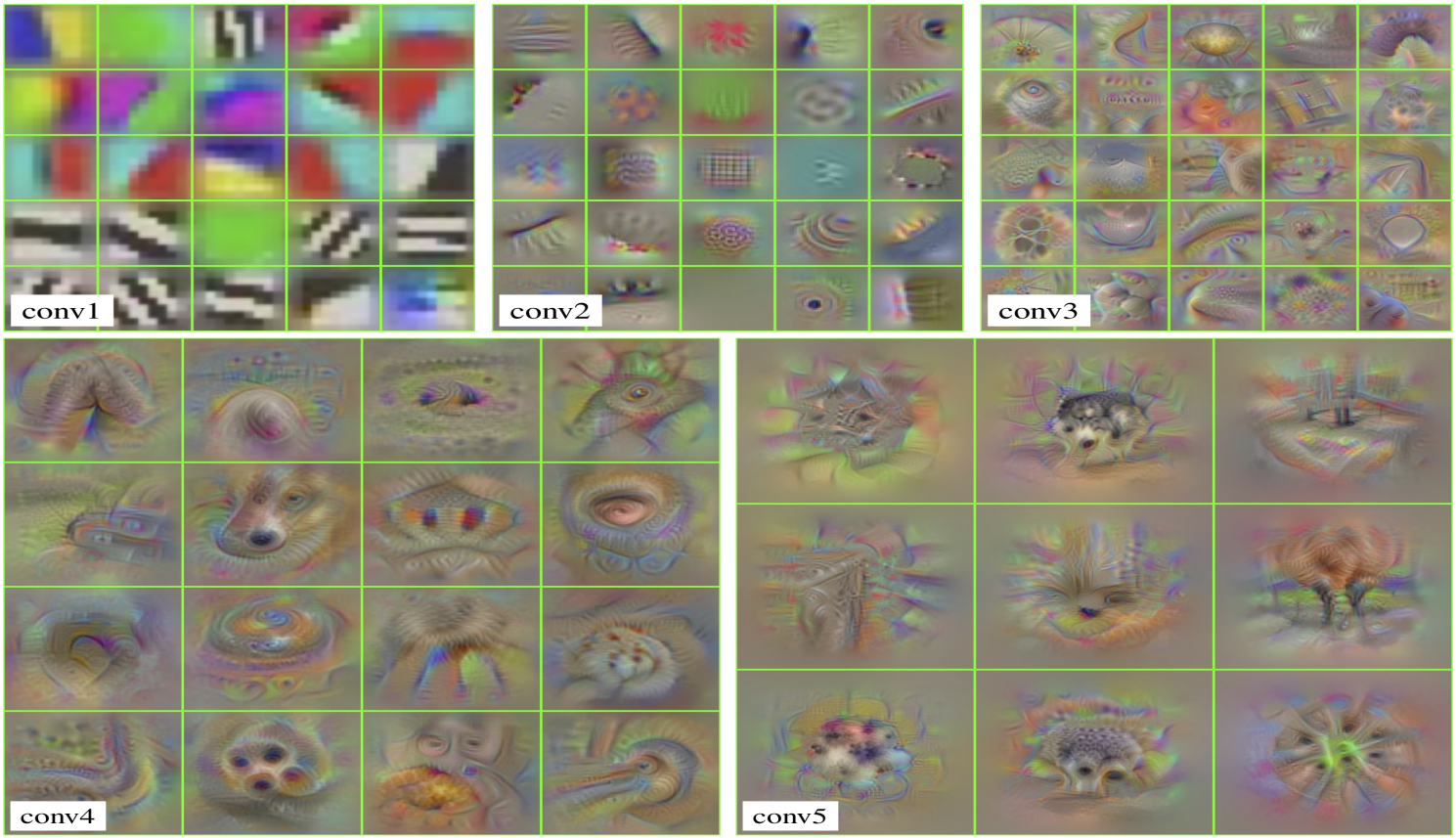


Figure 16: Activation maximization of the first filters of each convolutional layer in VGG-M.

Note, again we observe the same pattern, the initial filters are similar to Gabor/Wavelet filters and later layers are image components.

[12] https://arxiv.org/abs/1512.02017

Summary: similarity and importance of initial layers

We observe the initial layer of CNNs to be similar to one another, and to exhibit wavelet like representations. This is to be expected.

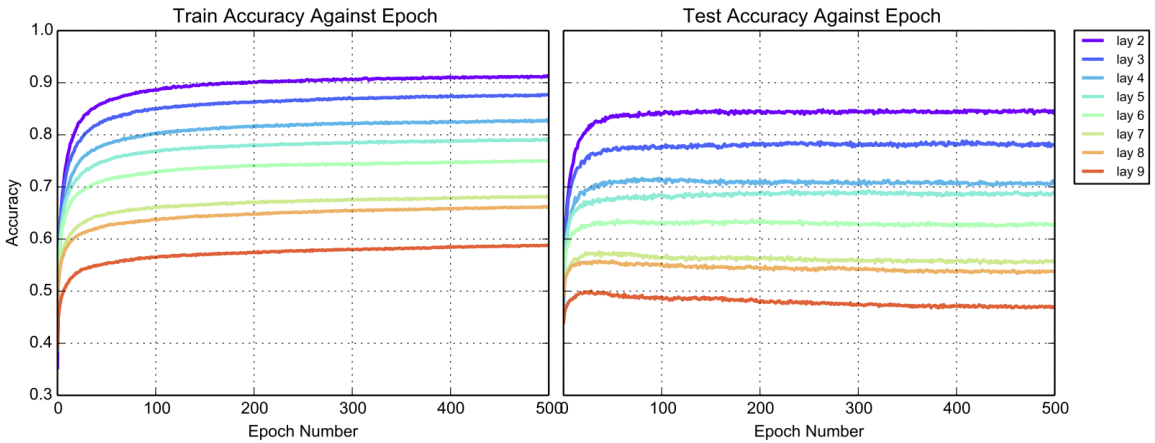


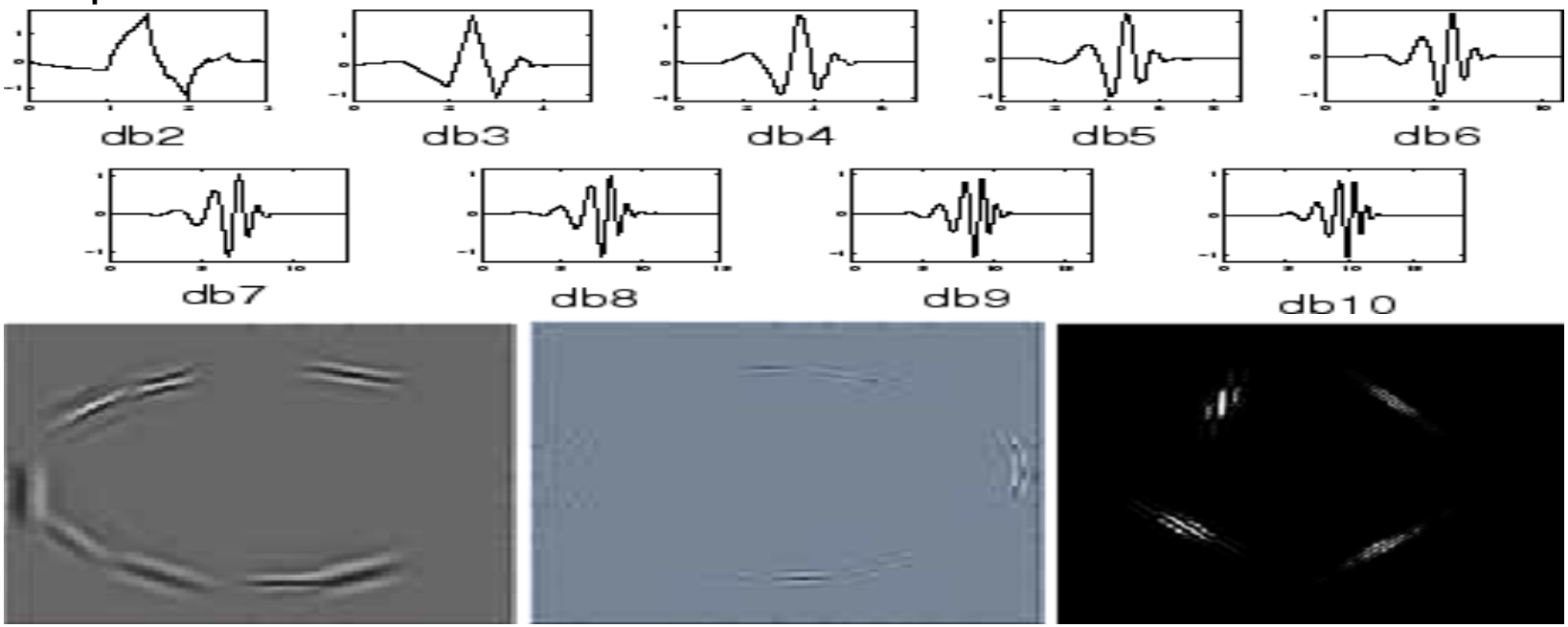
Figure 6: Demonstration of expressive power of remaining depth on MNIST. Here we plot train and test accuracy achieved by training exactly one layer of a fully connected neural net on MNIST. The different lines are generated by varying the hidden layer chosen to train. All other layers are kept frozen after random initialization.

Accuracy of a random network is improved most by training earlier layers (Raghu 16'[13]).

[13] https://arxiv.org/pdf/1611.08083.pdf

Wavelet, curvelet, and contourlet: fixed representations

Applied and computational harmonic analysis community developed representations with optimal approximation properties for piecewise smooth functions.



Most notable are the Daubechies wavelets and Curvelets/Contourlets pioneered by Candes and Donoho.
While optimal, in a certain sense, for a specific class of functions, they can typically be improved upon for any particular data set.

Optimality of curvelets in 2D

Theorem (Candes and Donoho 02'[a])

[a] http://www.curvelet.org/papers/CurveEdges.pdf

Let f be a two dimensional function that is piecewise C^2 with a boundary that is also C^2. Let f_n^F, f_n^W, and f_n^C be the best approximation of f using n terms of the Fourier, Wavelet and Curvelet representation respectively. Then their approximation error satisfy $\|f - f_n^F\|_{L^2}^2 = \mathcal{O}(n^{-1/2})$, $\|f - f_n^W\|_{L^2}^2 = \mathcal{O}(n^{-1})$, and $\|f - f_n^C\|_{L^2}^2 = \mathcal{O}(n^{-2}\log^3(n))$; moreover, no fixed representation can have a rate exceeding $\mathcal{O}(n^{-2})$.

Near optimality of such representation suggest a good first layer.

Should we be deterministic or use learning?

The first layer of a net is seemingly the most important, and if we have good prior knowledge of the data we can probably guess near optimal candidate weights.

One can perform classification based on two layer net with:

layer 1: $h_2(x) = \sigma(W^{(1)}x + b^{(1)})$ where $W^{(1)}$ is a fixed transform of x to, say, the wavelet domain and $\sigma(\cdot)$ project to keep just the largest entries with hard or soft thresholding;

$$\sigma_{hard}(x;\tau) = \begin{cases} x & x > \tau \\ 0 & |x| \leq \tau \\ -x & x < -\tau \end{cases}, \quad \sigma_{soft}(x;\tau) = \begin{cases} x-\tau & x > \tau \\ 0 & |x| \leq \tau \\ -x+\tau & x < -\tau \end{cases}$$

layer2: $h_3 = \sigma(W^{(2)}h_2 + b^{(2)})$ with $W^{(2)}$ learned as the classifier based on the sparse codes h_2.

However, h_2 does not build in invariance we would desire in classification, such as dilation, rotation, translation, etc... Depth remains important to generate these.