Deep Linear Neural Networks

Andrew Saxe

neuromatch academy

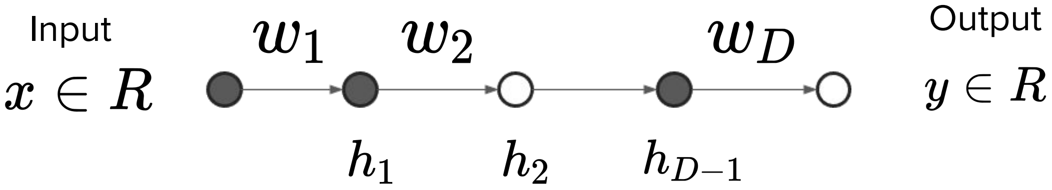
Representation learning

So far depth just seems to slow down learning. What is depth good for?

A core intuition behind deep learning is that deep nets derive their power through learning internal representations.

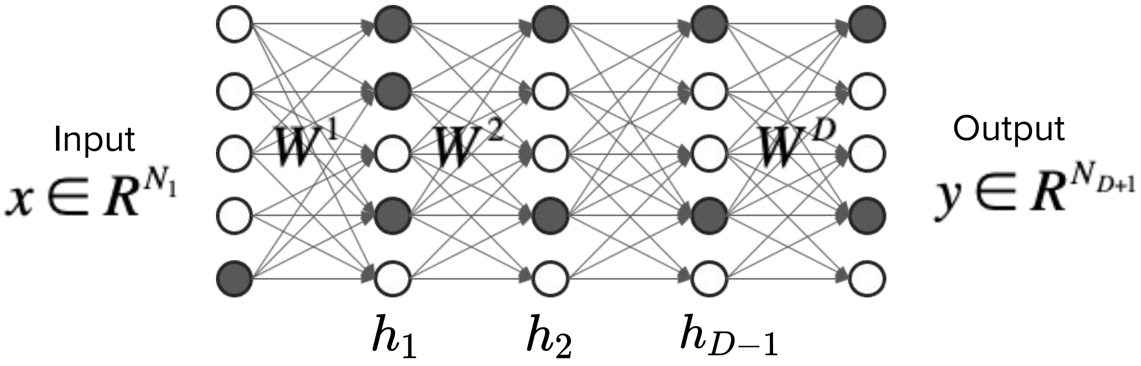
To address representation learning, we have to go beyond the 1D chain.

Deep Narrow Linear Network

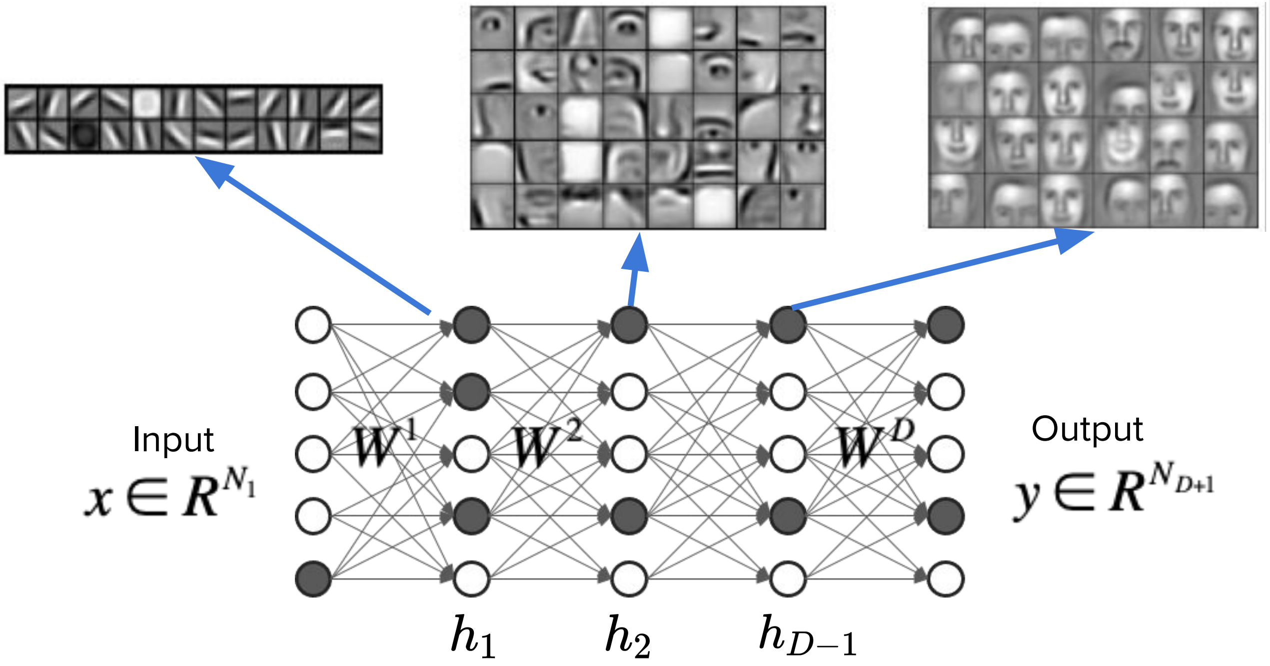


Deep Linear Network

A mixture of serial and parallel structure

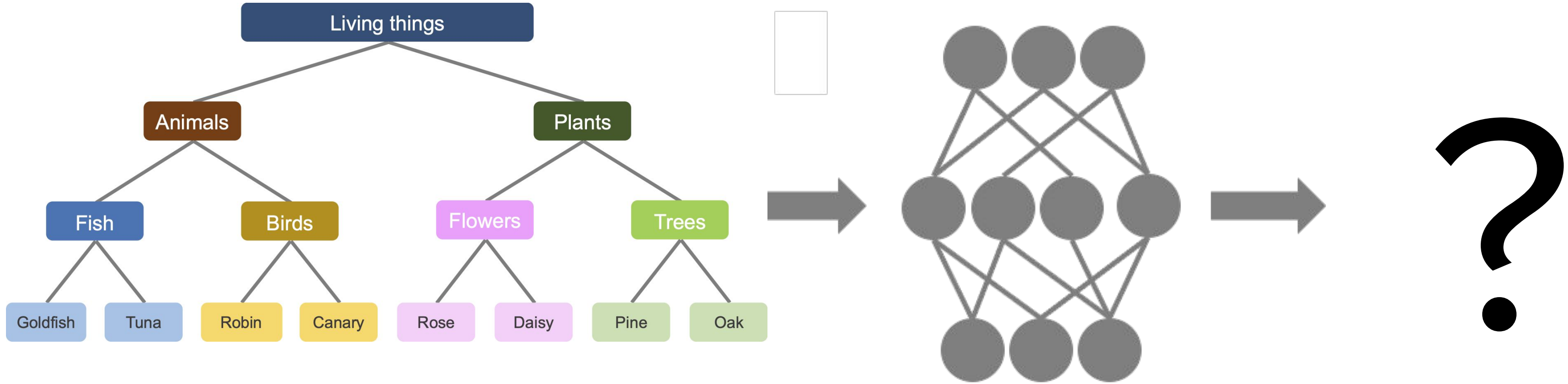


Representation learning

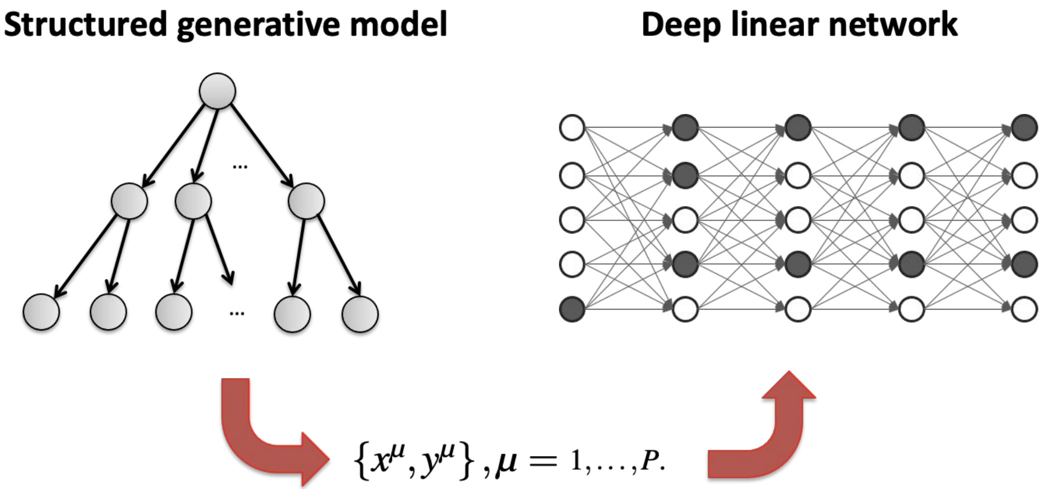


Lee et al., 2009

The role of the environment



A toy model of learning hierarchy

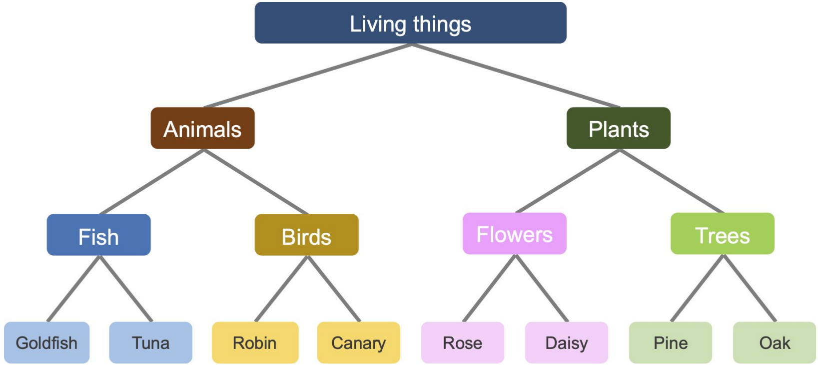


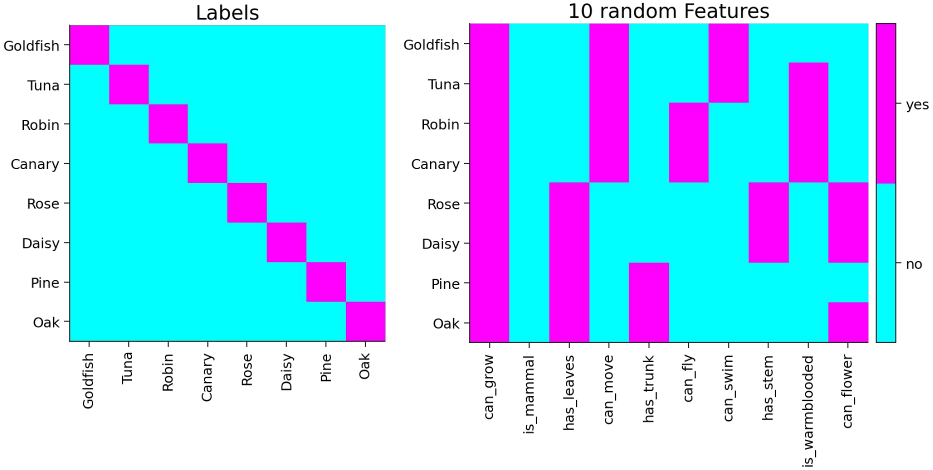
$$\{x^\mu, y^\mu\}, \mu = 1, \ldots, P.$$

Saxe et al., 2019

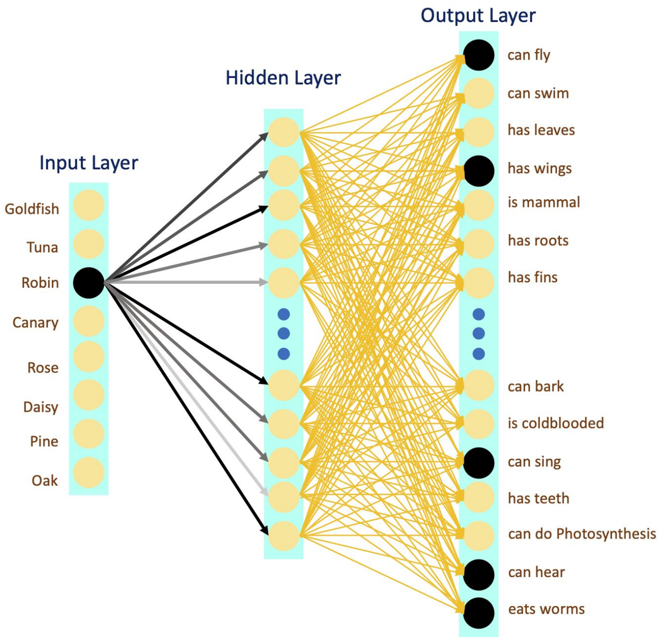
Hierarchical generative model creates data for deep linear network.

A hierarchical generative model



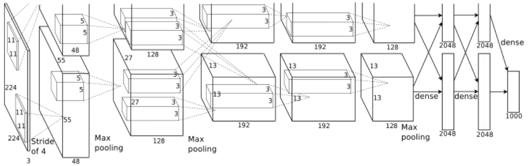


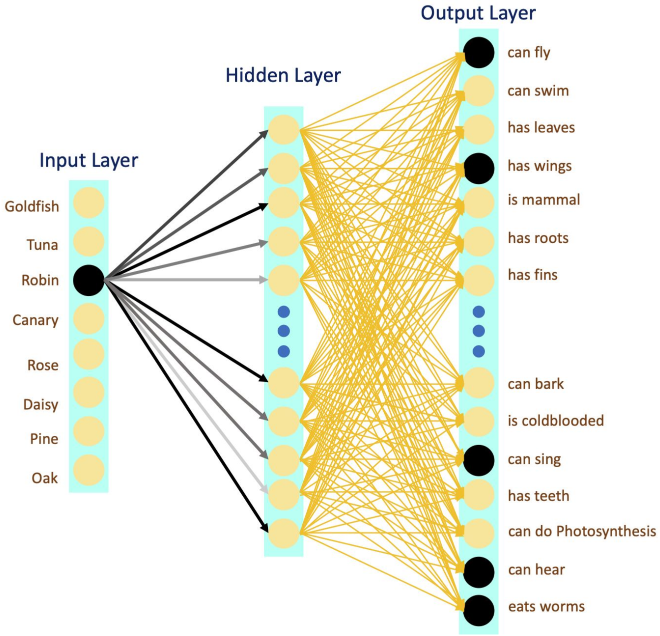
Learning semantic properties



Saxe et al., 2019
Rogers & McClelland, 2004
Rumelhart & Todd, 1993

Beyond class labels



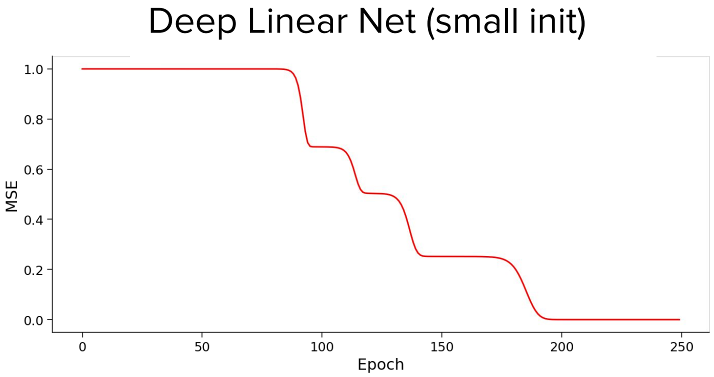


Deep Linear Network

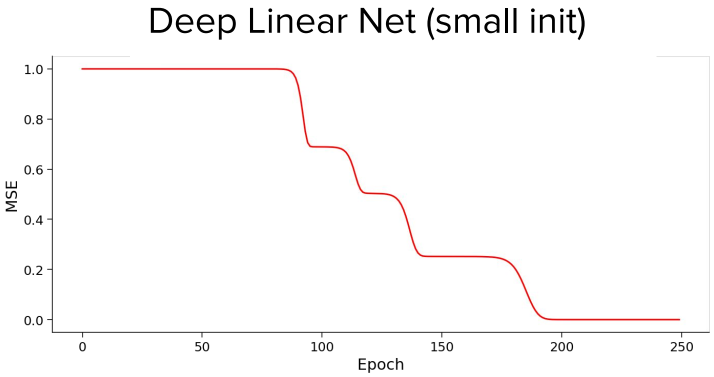
To investigate how representations change to help perform a task, we'll need a network with many hidden neurons.

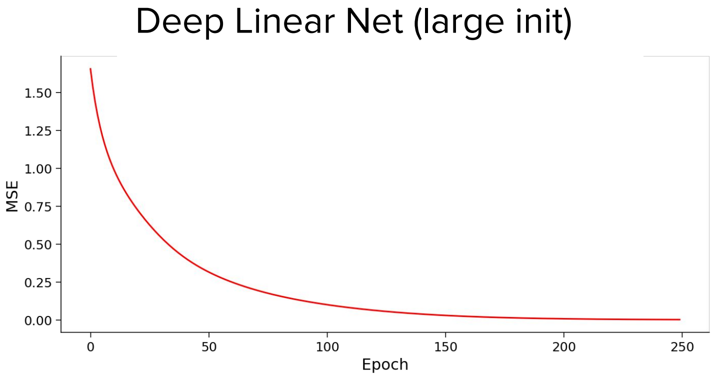
Implement a deep linear network and train it in a hierarchical world.

Training error dynamics



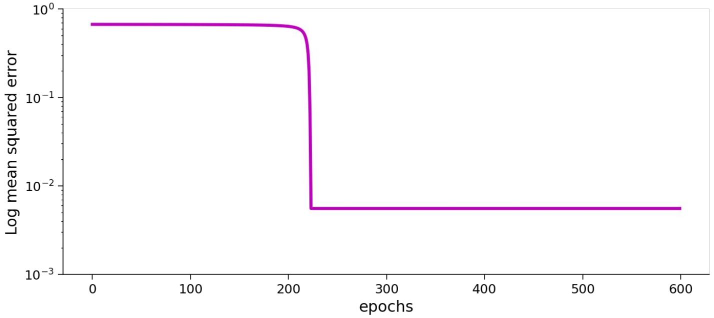
Training error dynamics



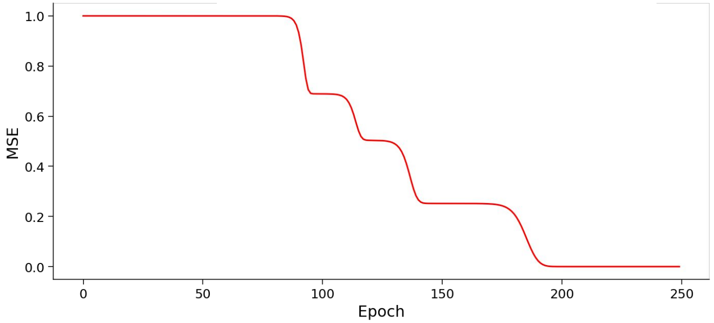


Training error dynamics

Deep Narrow Linear Net (Tutorial 2)



Deep Linear Net



Decomposing the trajectory with Singular Value Decomposition

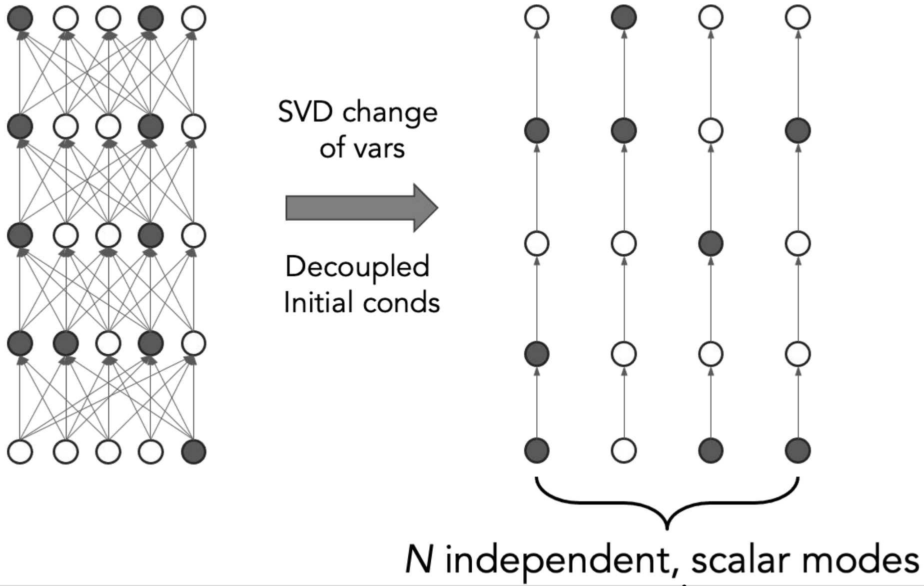
It turns out that the dynamics really are the sum of several Deep Narrow Linear Networks, if we know how to look.

We can reveal these using the SVD: $W^{tot} = U \Sigma V^T$

$$U^T U = V^T V = I$$

Σ is diagonal

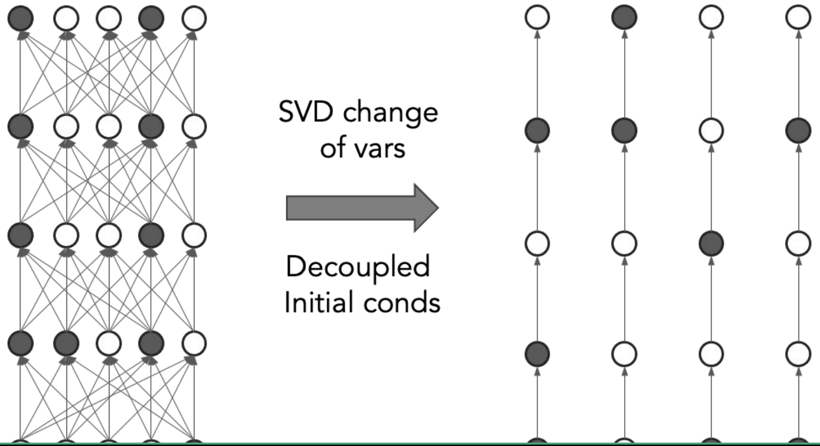
Decomposing the trajectory with Singular Value Decomposition



$$W^{tot} = U\Sigma(t)V^T$$

Diagonal!

Decomposing the trajectory with Singular Value Decomposition

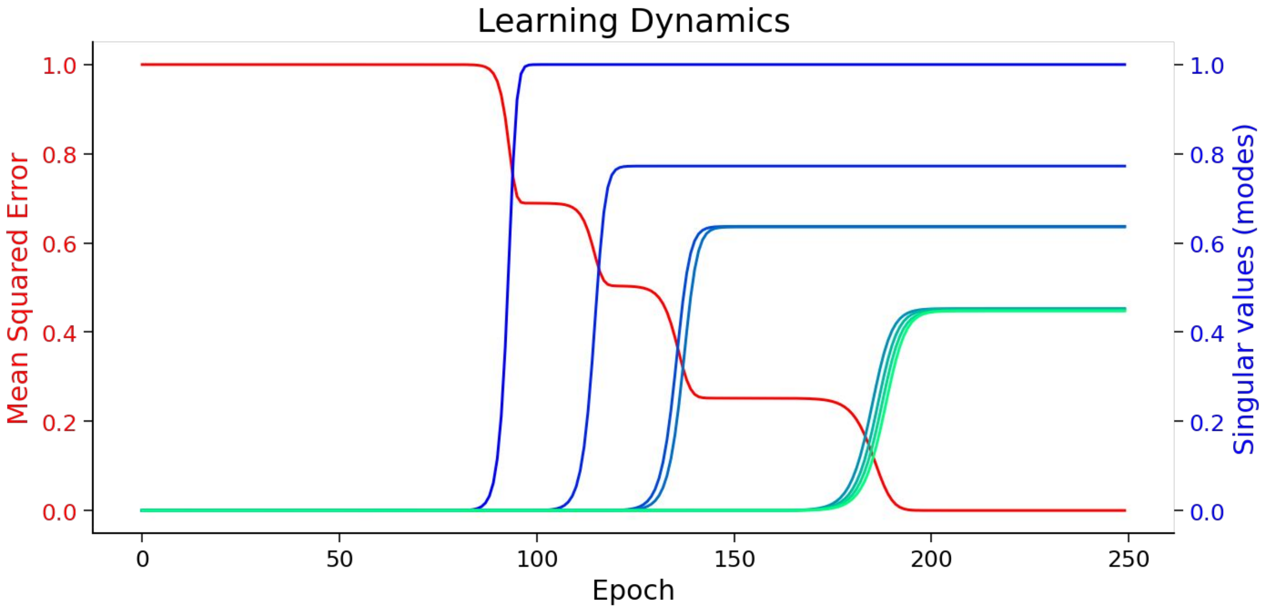


$$W^{tot} = U\Sigma(t)V^T$$

Reveal hidden Deep Narrow Linear Network dynamics with the SVD.

N independent, scalar modes

Deep Linear Network Dynamics

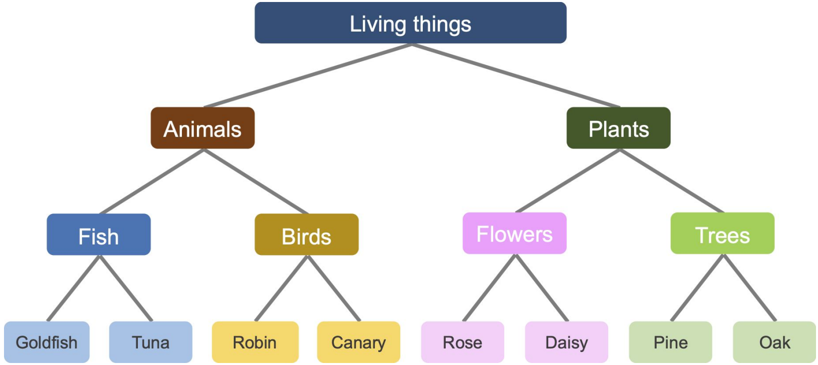


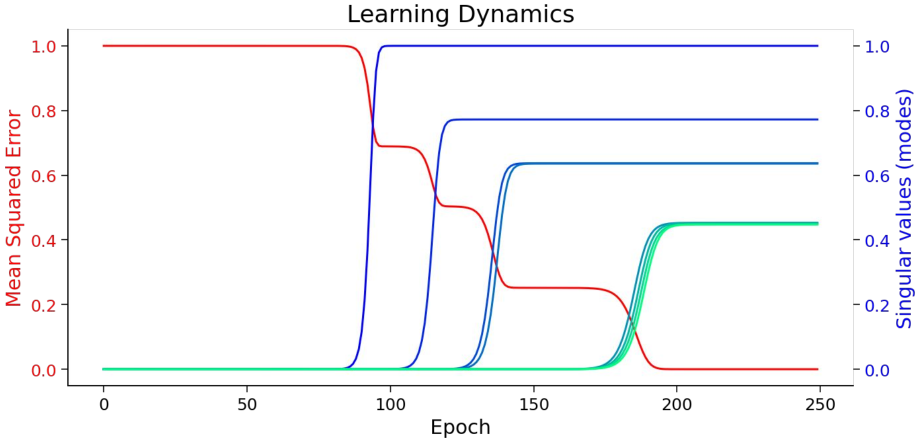
Deep Linear Network Dynamics

Therefore, everything you've learned about depth, learning rates, initializations, and interactions carries over

But now we have several 1D chains going in parallel and summing together

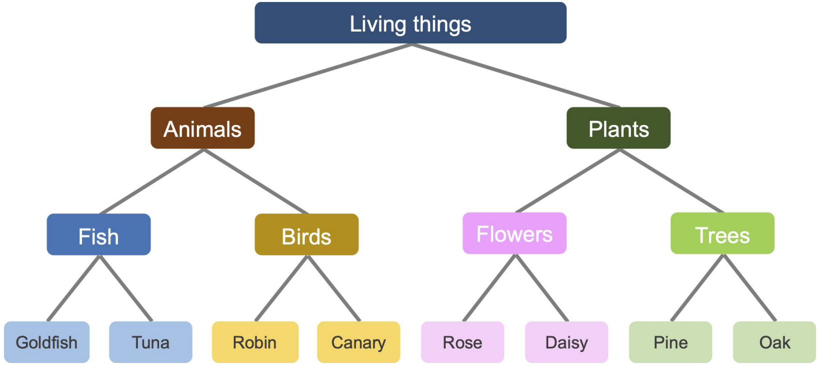
Training error dynamics

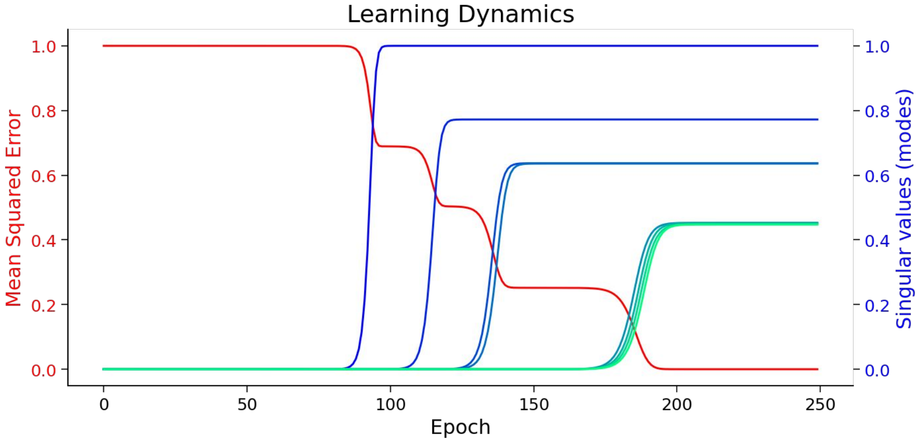




Training error dynamics

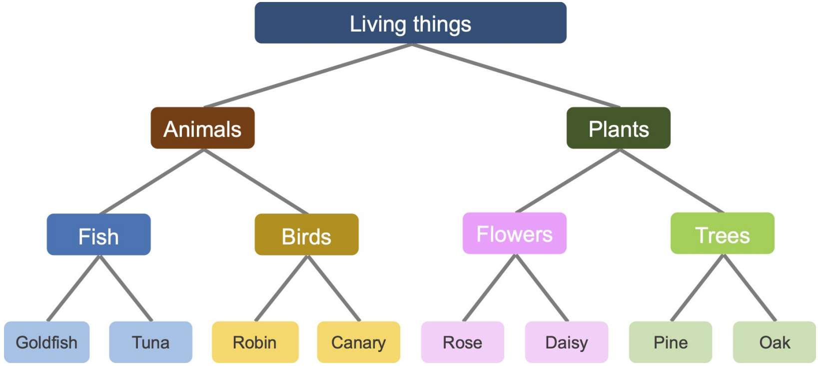
4 Levels



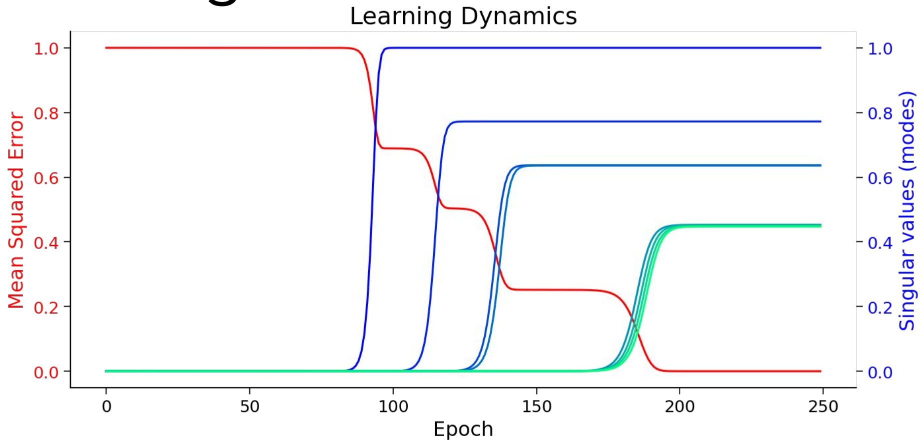


Training error dynamics

4 Levels



4 Stages!

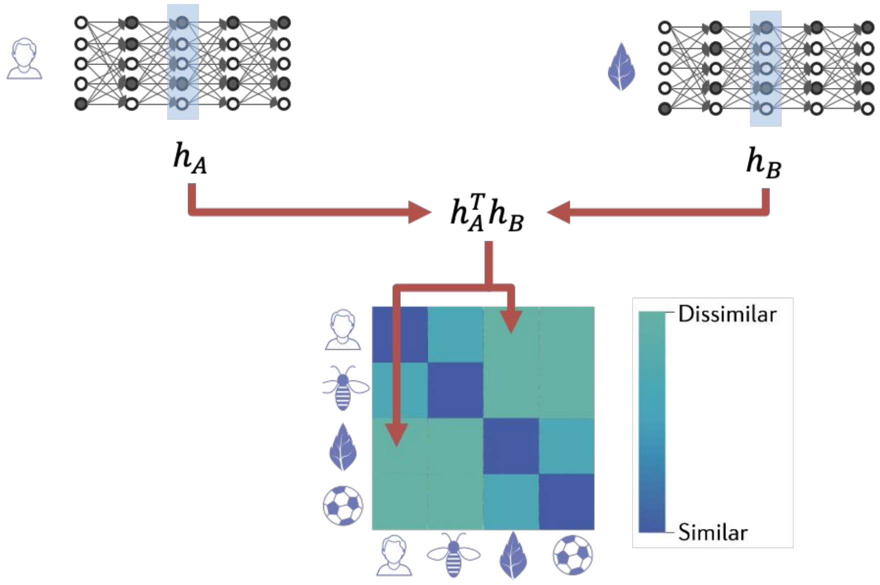


Representational Similarity Analysis (RSA)

How can we peek inside to see how the network has structured its internal representations?

Apply analysis methods familiar in neuroscience!

RSA



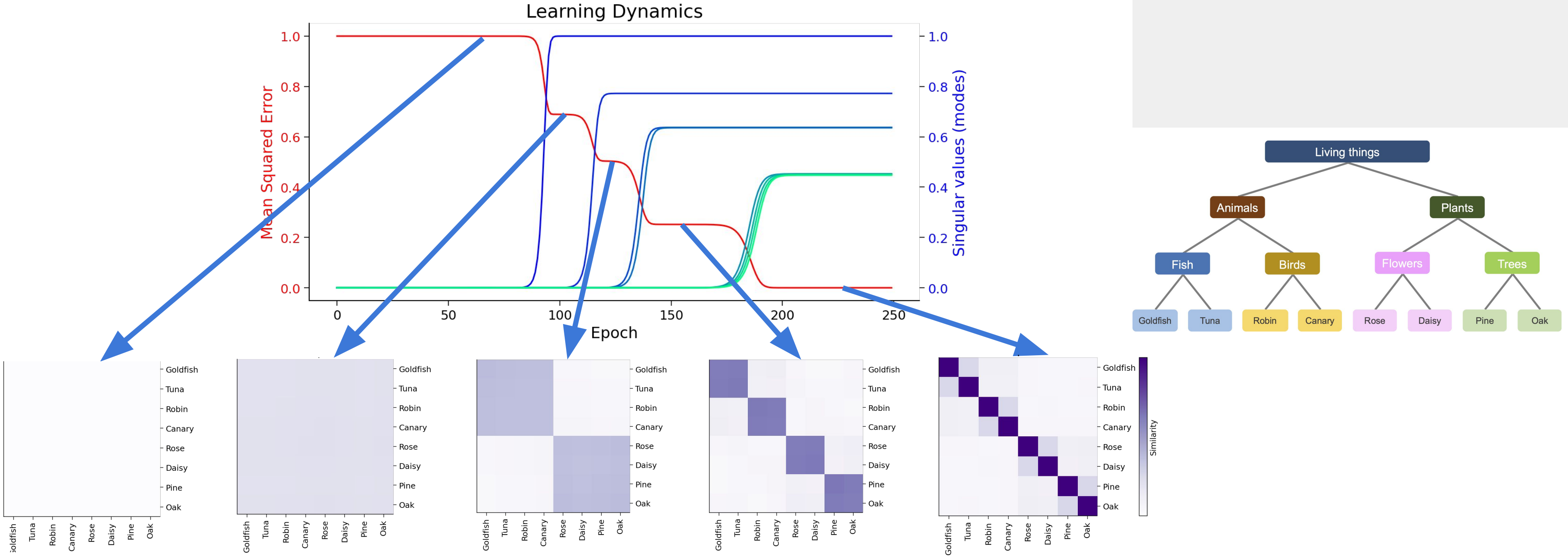
Kriegeskorte et al., 2008

Representational similarity over learning

Let's use RSA to better understand internal representations in our hierarchical dataset.

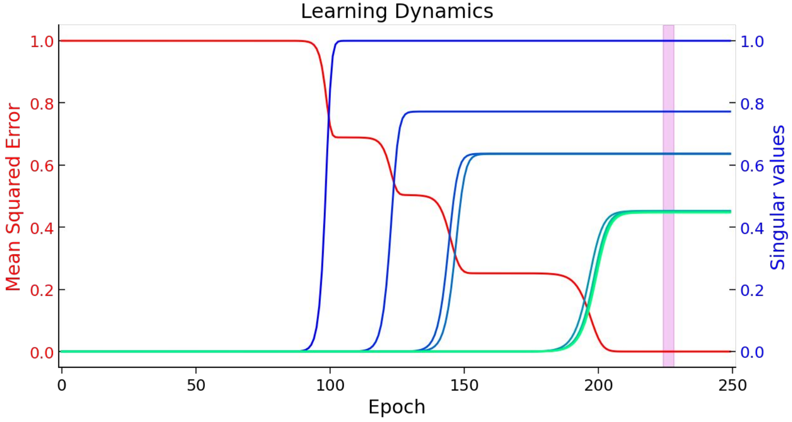
Use RSA to visualize how representations emerge through learning.

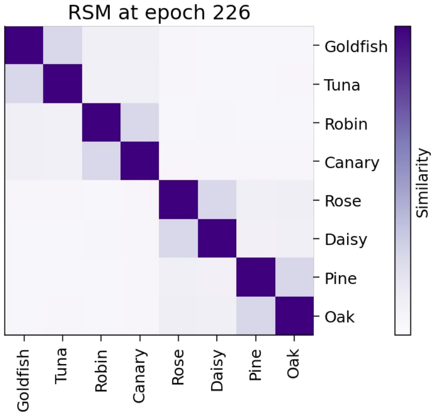
Progressive differentiation



Representing structure: initialization matters

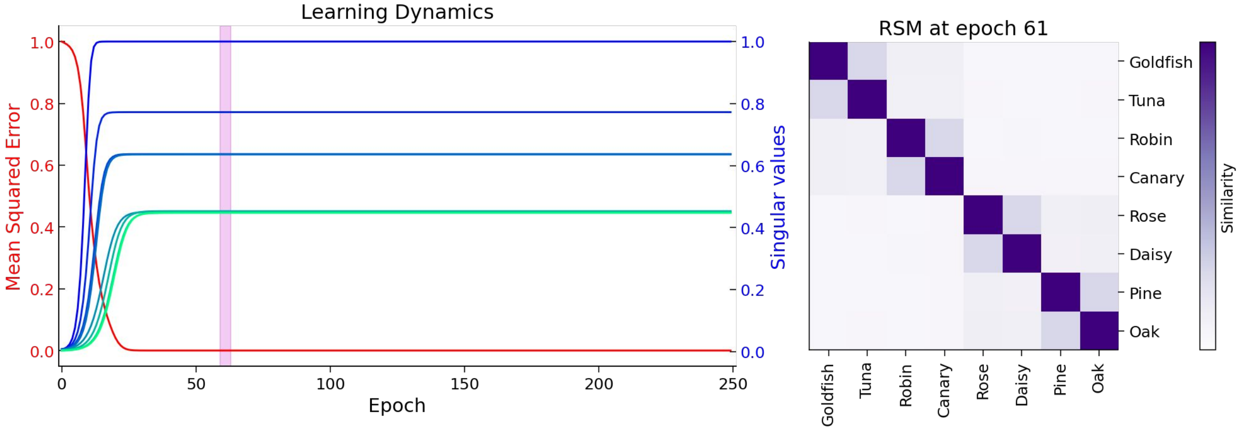
Small init $\gamma = 10^{-12}$





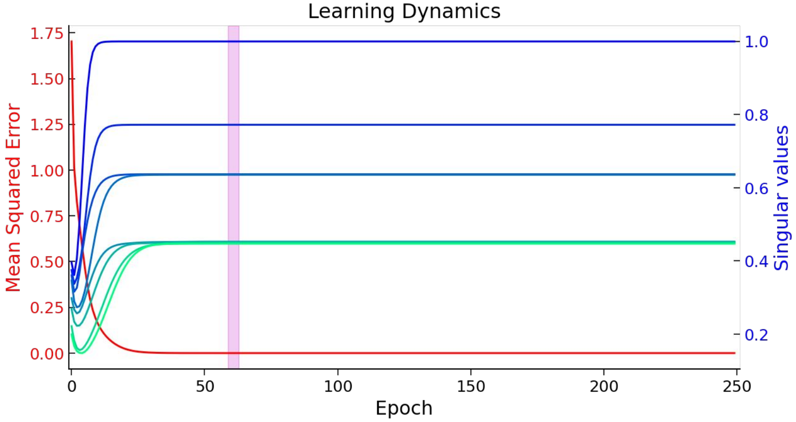
Representing structure: initialization matters

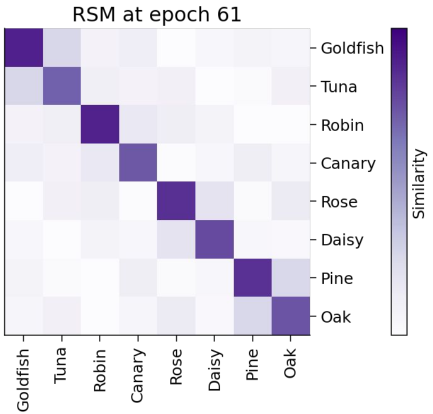
Dynamic Isometry init $\gamma = 1$



Representing structure: initialization matters

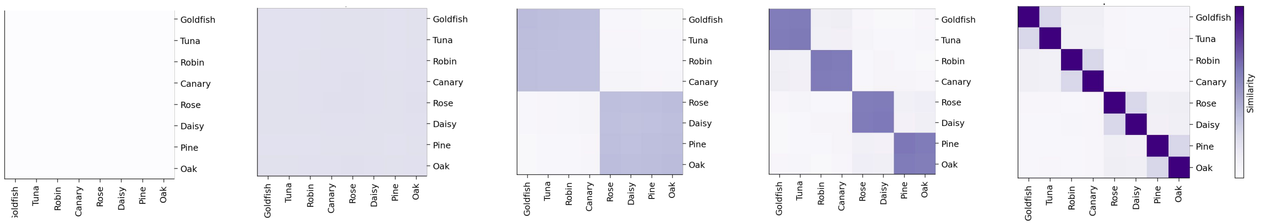
Very large init $\gamma = 10$

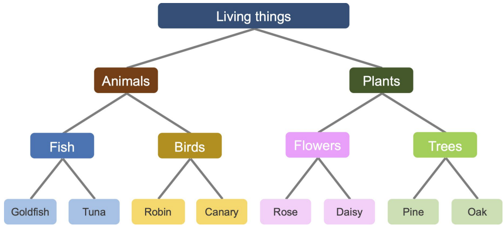




Similarity-based reasoning

Deep networks generalise based on learned similarity between inputs.





Illusory correlations

Similarity-based generalization can work, but it’s a big risk.

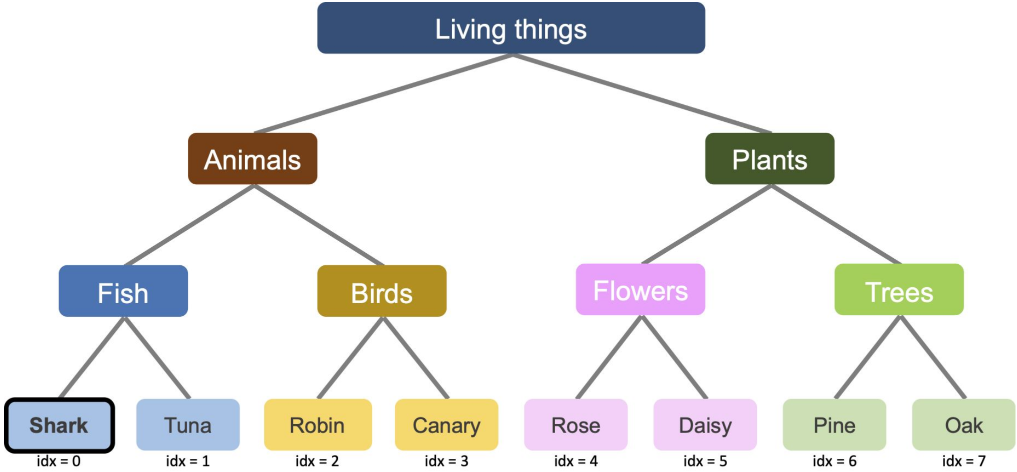
Similarity is not causality.

Beyond the evidence

"Does a shark have bones?"

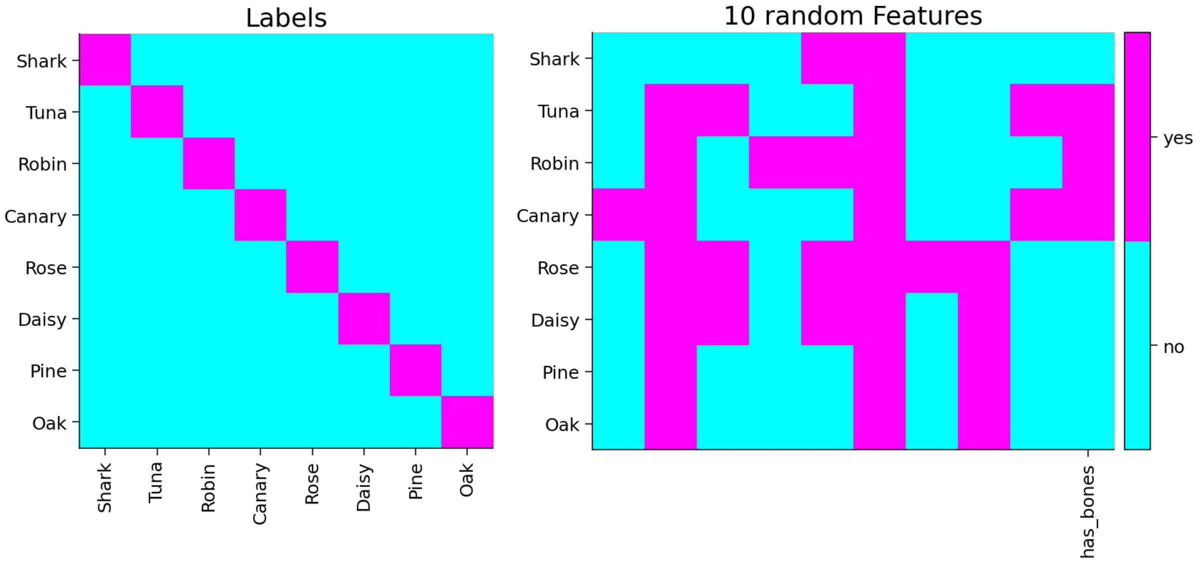
Carey, 1985

Let’s test this in our toy hierarchy



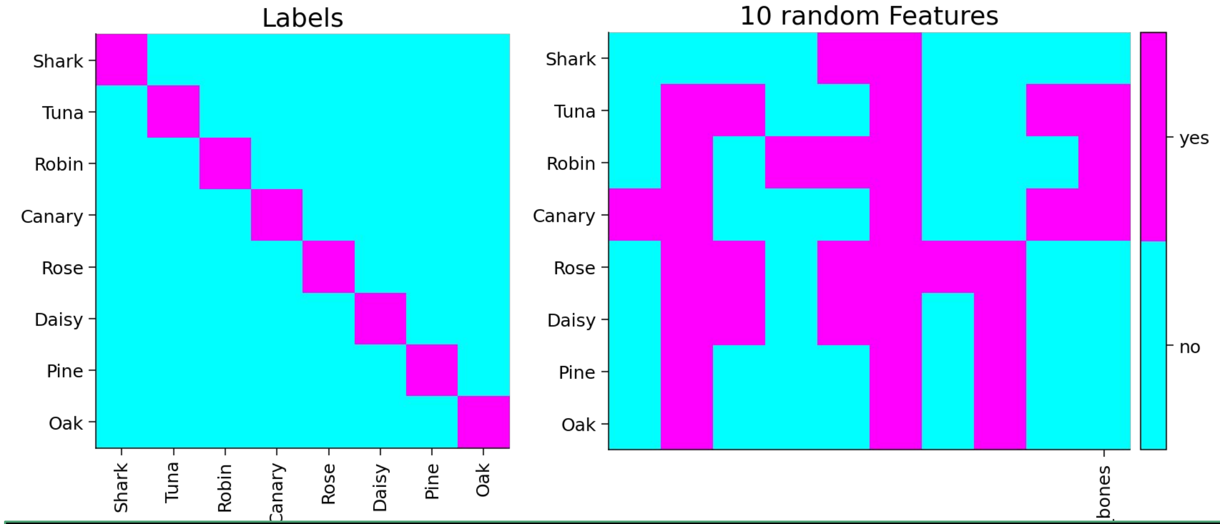
Has bones: -1 1 1 1 -1 -1 -1 -1

Let’s test this in our toy hierarchy



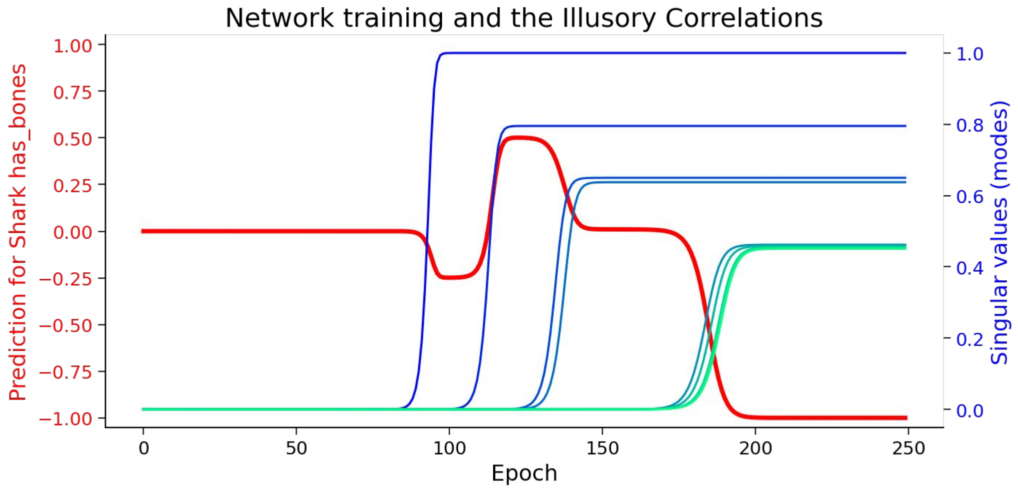
Every time the network sees the *shark* input, it is told that the shark *does not have bones*

Let’s test this in our toy hierarchy

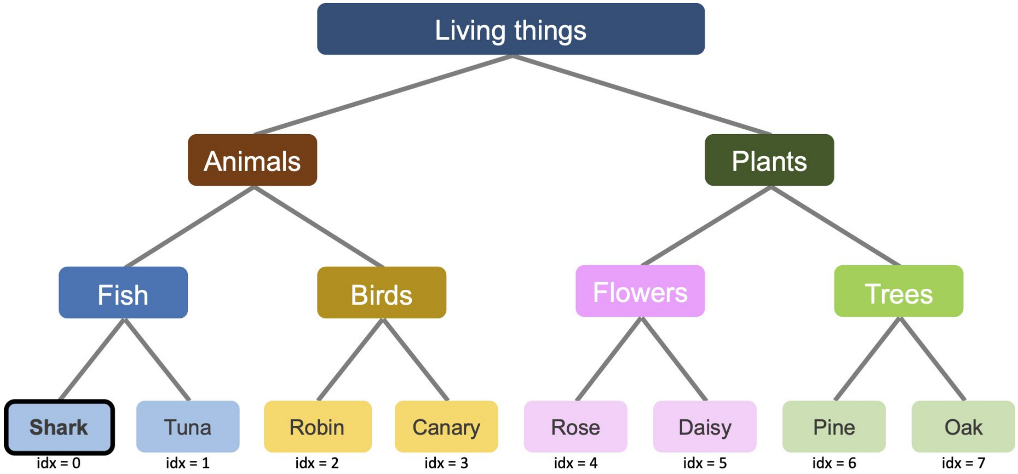


Test deep and shallow network’s predictions for ‘shark has bones’

Illusory Correlations

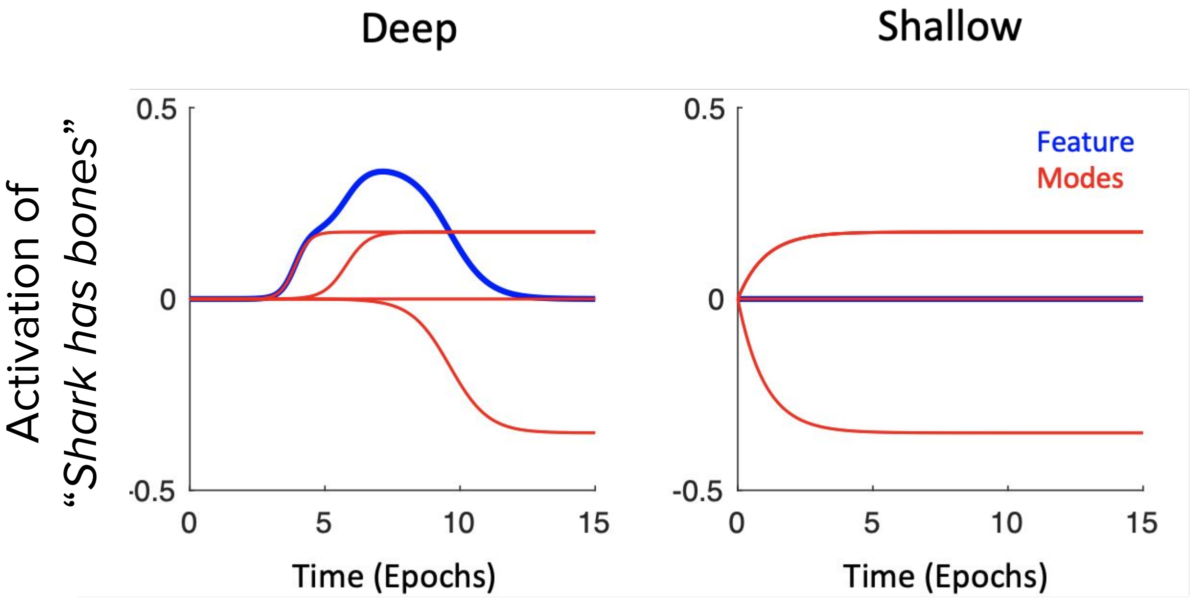


Transient overgeneralizations

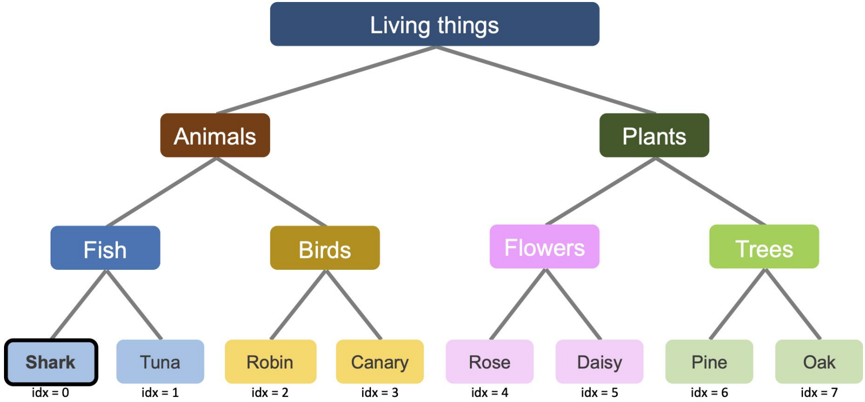


Has bones: -1 1 1 1 -1 -1 -1 -1

Illusory Correlations



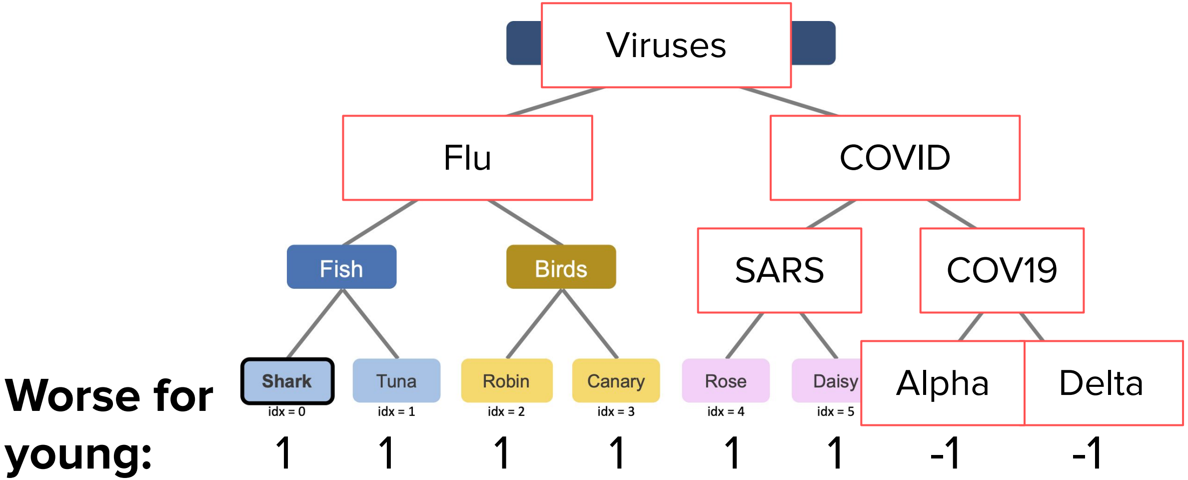
Beyond the evidence



Has bones: -1 1 1 1 -1 -1 -1 -1

Rename nodes to come up with your own example. What are the risks?

Beyond the evidence



Rename nodes to come up with your own example. What are the risks?

Wrap up to Deep Linear Networks Day

We’ve used the simplest possible networks to understand:

- The basics of gradient descent
- The effect of depth on training dynamics
- The internal representations that deep networks learn

The deep learning framework

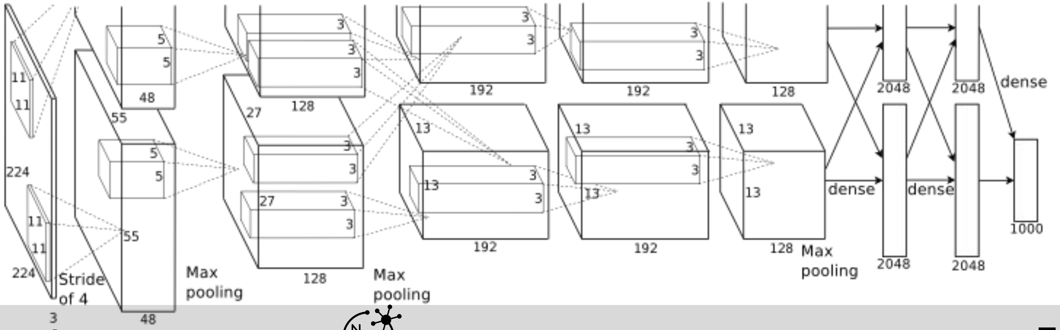
Objective function: Cross entropy loss

Learning rule: Gradient descent with momentum

Architecture: Deep convolutional ReLU network

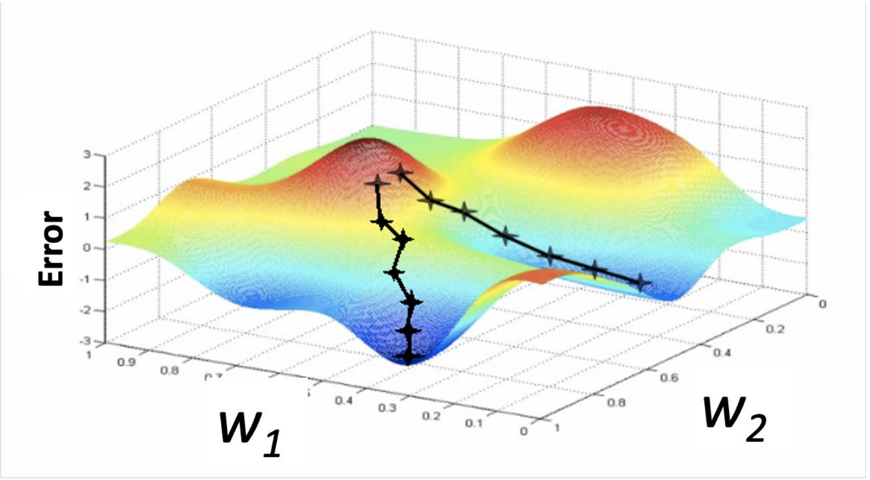
Initialisation: He et al. (Scaled Gaussian)

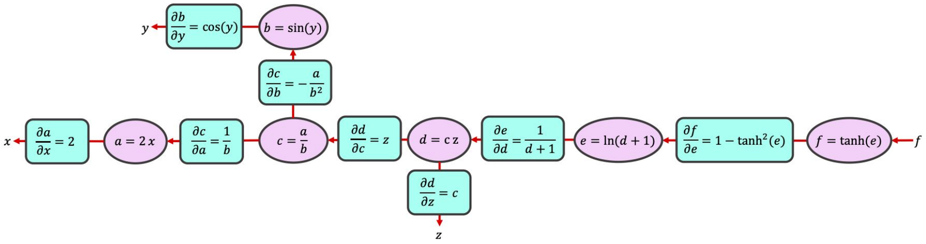
Environment: ImageNet dataset



Output:
Cat
Target:
Dog

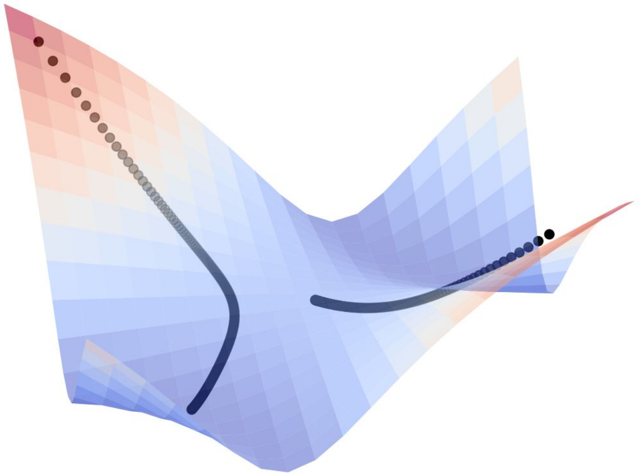
Gradient descent & autograd

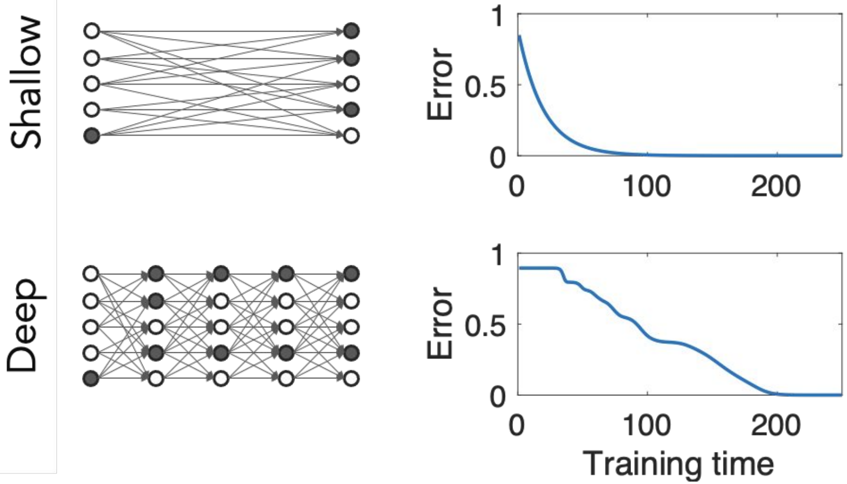




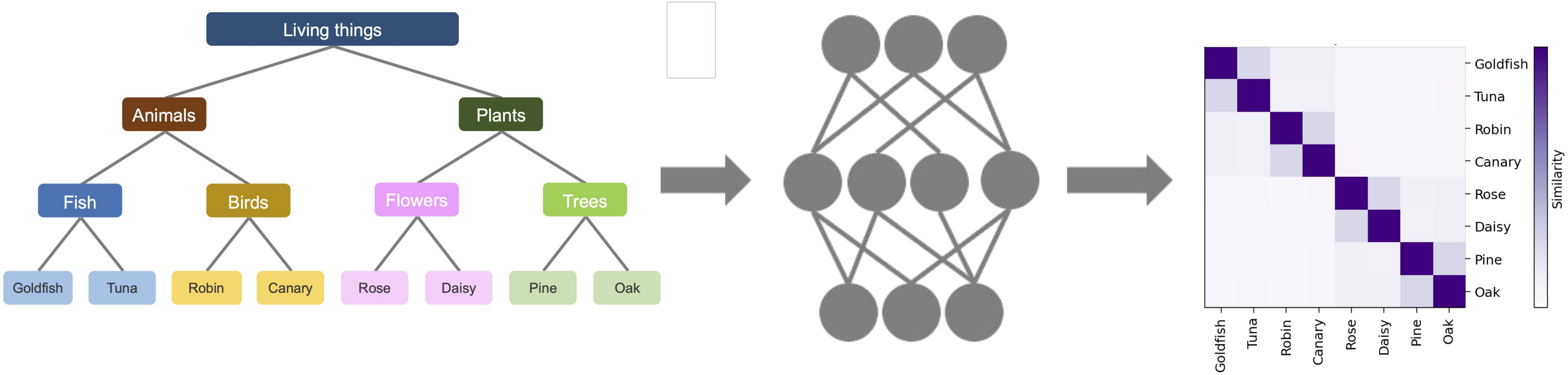
http://blog.datumbox.com/wp-content/uploads/2013/10/gradient-descent.png

The effect of depth on training





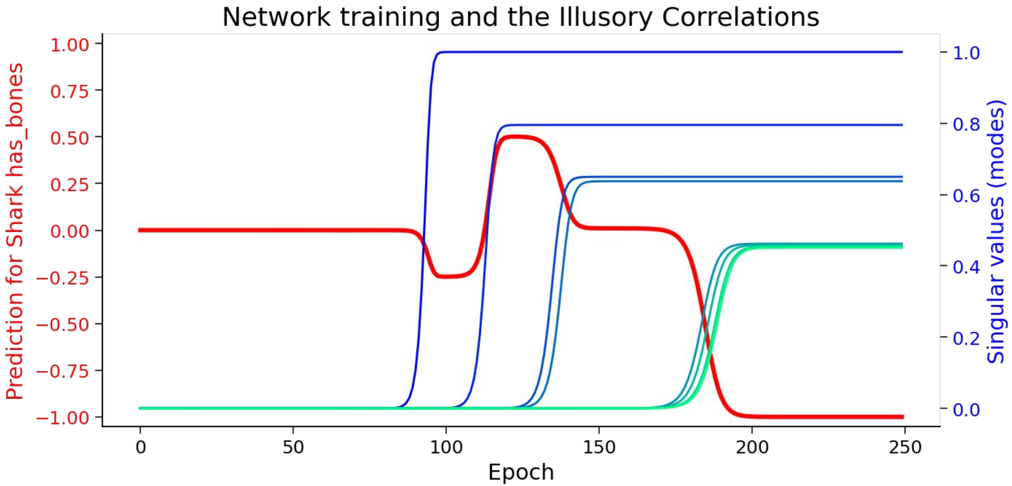
Emergence of internal representations



Illusory Correlations

"Does a shark have bones?"

Carey, 1985



Tuning up training

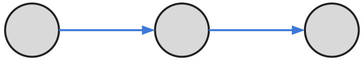
Generally you want to be in the **deep, wide, smallish-initialisation variance, maximum stable learning rate regime**, but your mileage may vary

You'll know you're there when

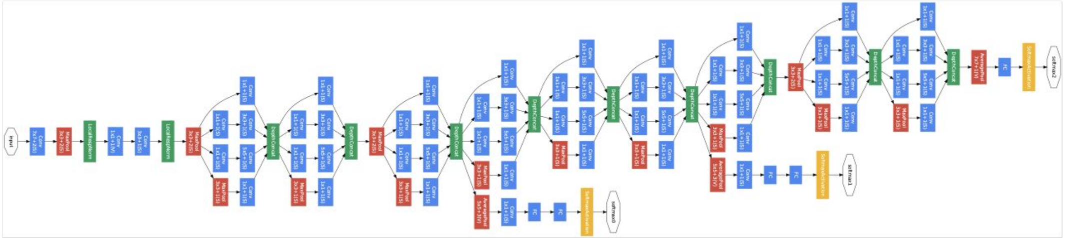
- you see a hint of a sigmoidal learning trajectory
- you see internal reps change substantially through learning
- multiple retrainings yield nearly identical trajectories and internal reps

Simple models

Today

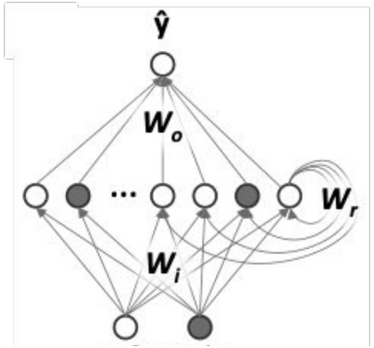


The rest of your career



Szegedy et al., CVPR 2015

What carries over?

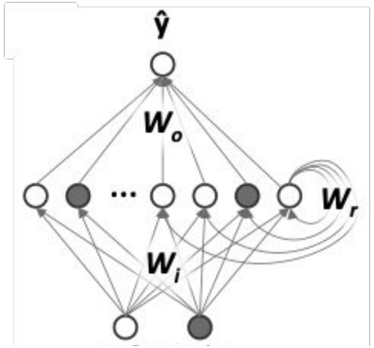


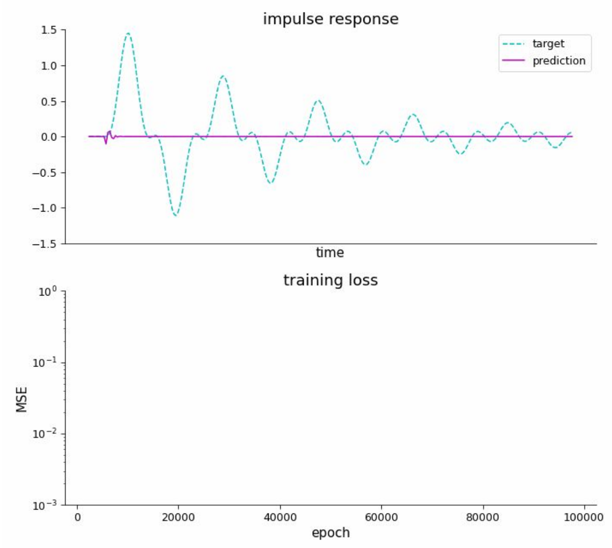
One more complicated example:

- **Recurrent** network
- **Nonlinear** network

Trained to produce complex temporal response

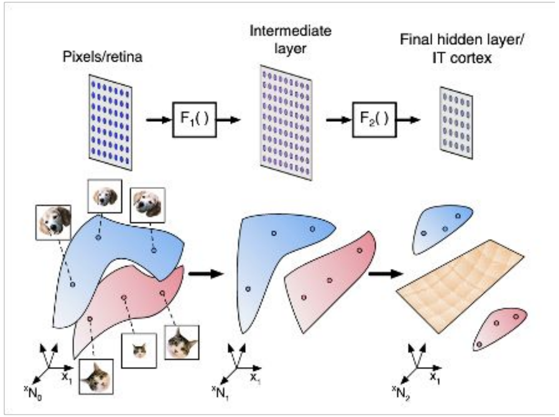
What carries over?



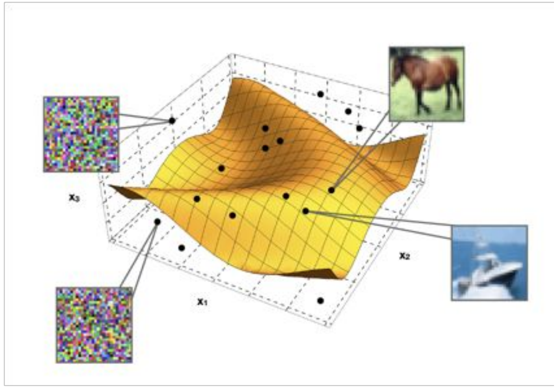


Theory

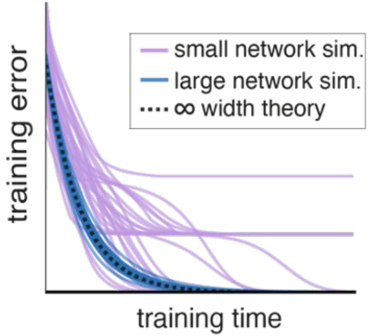
What will it take to understand deep learning?



Chung et al., 2018;
Cohen, 2020

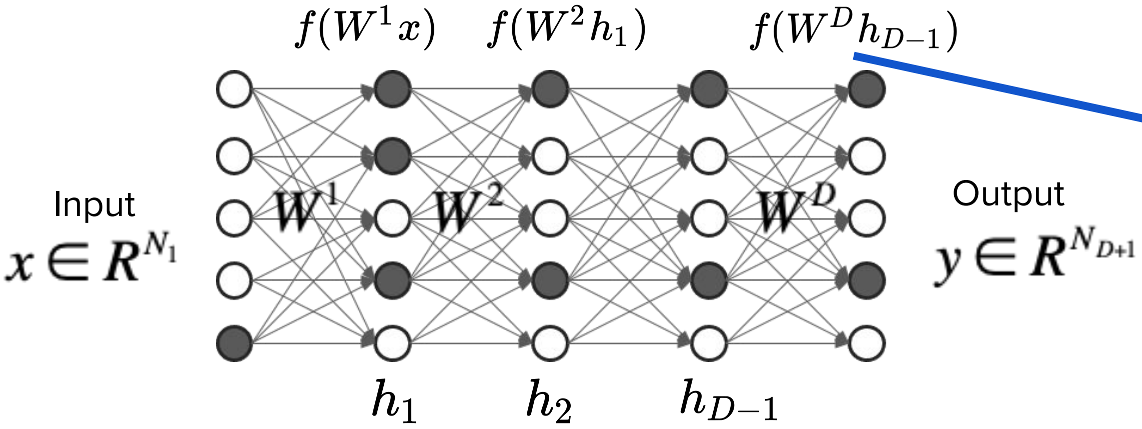


Goldt et al., 2019

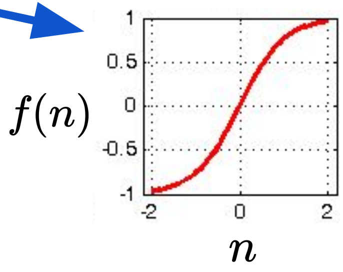


Jacot et al., 2018;
Lee et al., 2019;
Arora et al., 2019

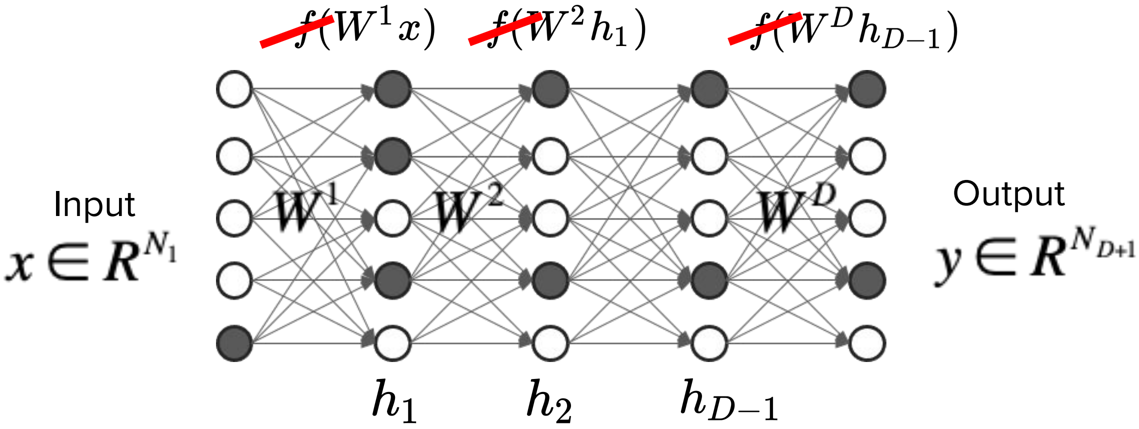
Main assumption: linearity



Neural nonlinearity



Deep *Linear* Network

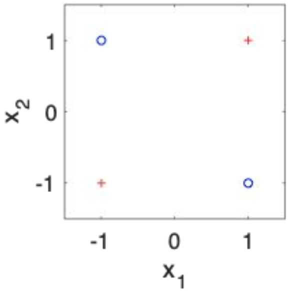


Main assumption: linearity

Deep linear networks can only implement linear functions

Famously, they cannot solve even simple nonlinear problems like XoR

Real world problems are nonlinear



Onward to nonlinear networks!

Today you've heard about how depth affects **dynamics**, even in linear nets.

Depth also impacts the **set of functions a network can implement**, when the network is nonlinear.

Much more on that, beginning tomorrow.

Bonus

neuromatch academy

Isn’t this just linear regression?

Input-output map is always linear:

$$\hat{y} = \left(\prod_{i=1}^{D} W_i\right)x = W^{tot}x$$

Linear regression; analytical solution

For linear regression, the optimal weights are:

$$W^{tot} = YX^T(XX^T)^{-1}$$

At convergence, do total weights of the DLN match linear regression?

Dynamics vs asymptotic performance

Deep linear networks typically end up at the linear regression solution

$$W^{tot} = YX^T(XX^T)^{-1}$$

But they take very different trajectories to get there.

Gradient descent is *not invariant* to parametrization

Linear map, nonlinear learning

Input-output map: Linear

$$\hat{y} = \left(\prod_{i=1}^{D} W^i\right) x \equiv W^{tot} x$$

Error function: Nonlinear

$$\sum_{\mu} \left\| y^\mu - \left(\prod_{i=1}^{D} W^i\right) x^\mu \right\|^2$$

Learning problem: Nonconvex (for D>1)

$$\min_{W_1,\cdots,W_D} \sum_{\mu} \left\| y^\mu - \left(\prod_{i=1}^{D} W^i\right) x^\mu \right\|^2$$

Shallow vs Deep Networks