Going deeper: the expressivity of deep neural networks

Lecture 6

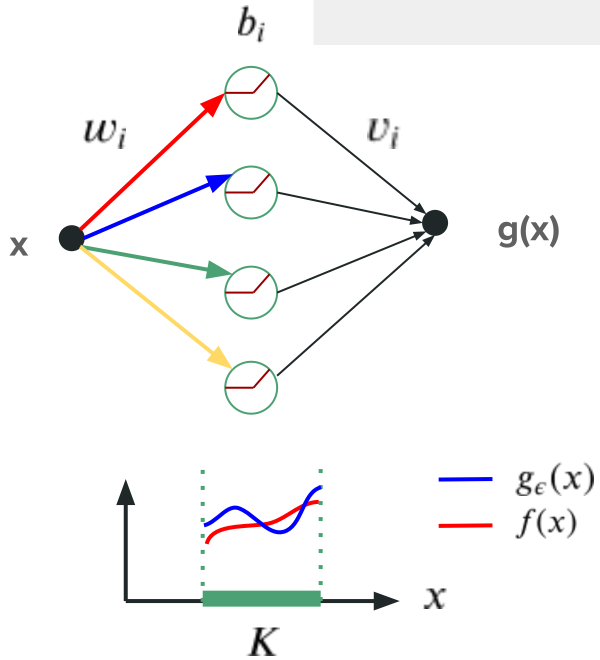
Expressivity:

What can a deep nonlinear network "say" that a shallow nonlinear network cannot?

Okay... so why do we need deep nets with more than one hidden layer?

While the universal approximation theorem says we can approximate a function to some accuracy with a one hidden layer neural network,

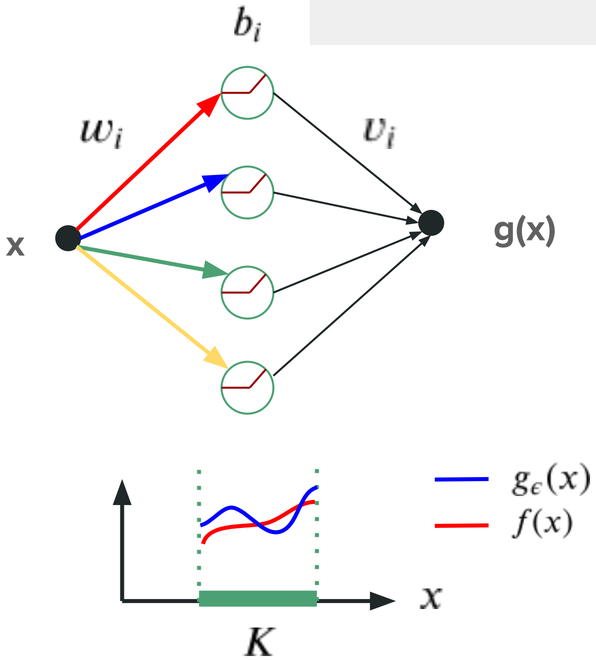
It does not tell us how many hidden neurons we will need.



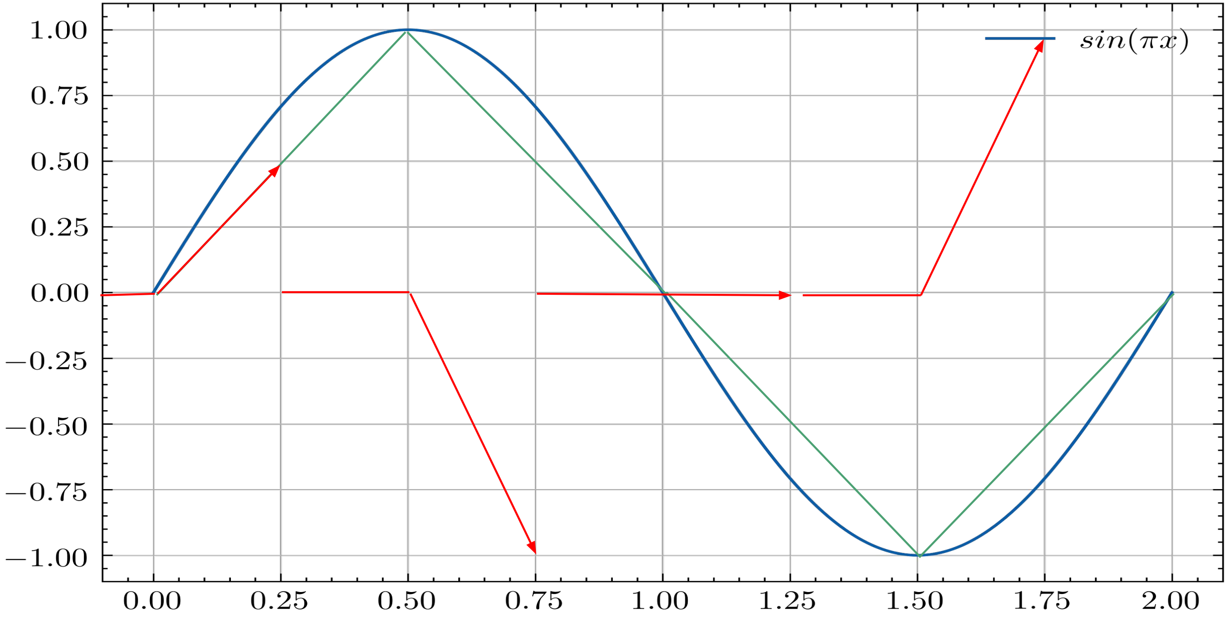
The phenomenon of deep expressivity

For example, there exist some functions which can be efficiently approximated by a deep network.

But to approximate these same functions with a 1 hidden layer network would require exponentially many more neurons.



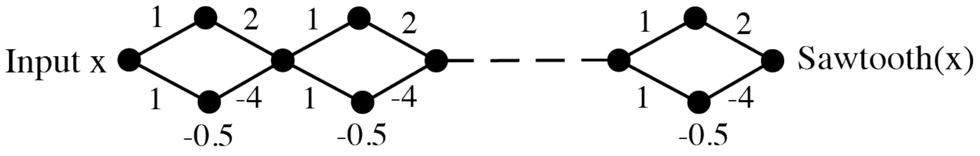
Intuition: the more wiggles in the target, the more hidden neurons needed in 1 hidden layer

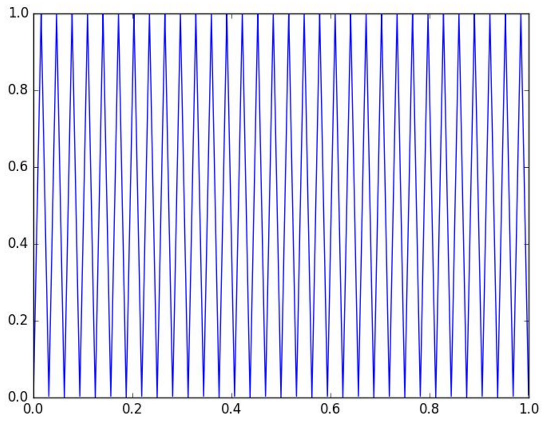


Now your turn: play around with approximating the sin function using a 1 hidden layer ReLU MLP

Sawtooth function

- 2^n linear pieces expressed with ~3n neurons (Telgarsky 2015) and depth ~2n.



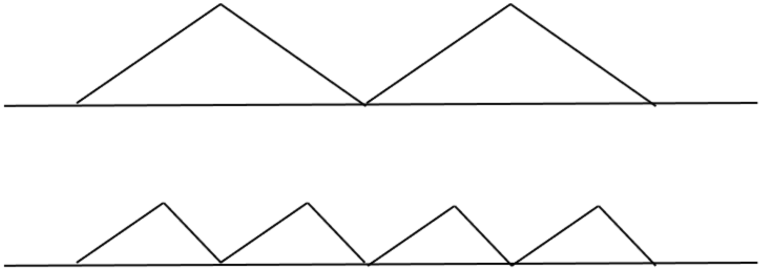


- Shallow implementation takes exponentially more neurons

Number of monomials in sum product nets

There exists a function computable by a deep ReLu network where the number of linear regions is exponential in the depth.

A shallow 1 hidden layer network would require exponentially many neurons to approximate this.

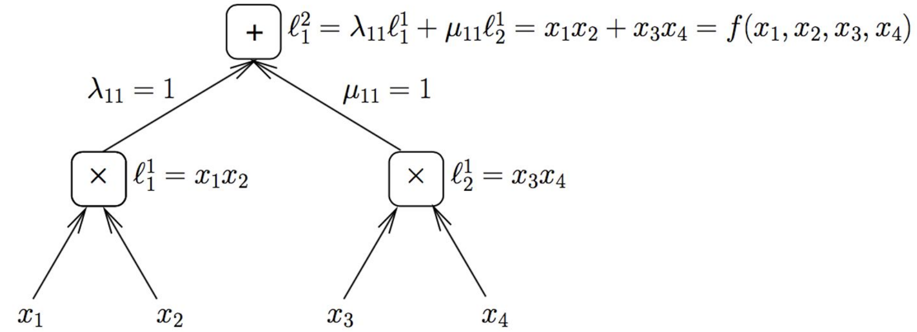


Montufar, Pascanu, Cho, Bengio, On the number of linear regions in deep neural networks NeurIPS (2014)

Number of monomials in sum-product nets

There exists a function computable by a deep sum product network where the number of monomials is exponential in the depth.

A shallow 1-hidden layer network would require exponentially many neurons to approximate this.



Delalleau and Yoshua Bengio. Shallow vs. deep sum-product networks, NeurIPS 2011.

How general is this?

These particular functions don't seem that natural?

Are they rare curiosities?

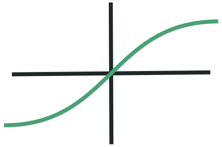
Or is some sense is **<u>any</u>** function computed by a **<u>generic</u>** deep neural network not efficiently approximated by a shallow network?

Expressivity from chaos hiding in deep nets

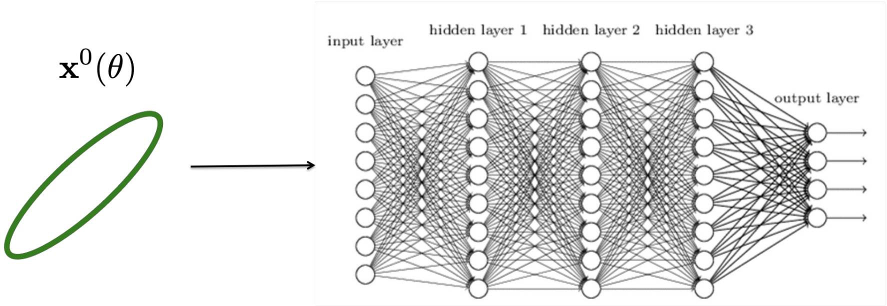
Random deep sigmoidal neural net

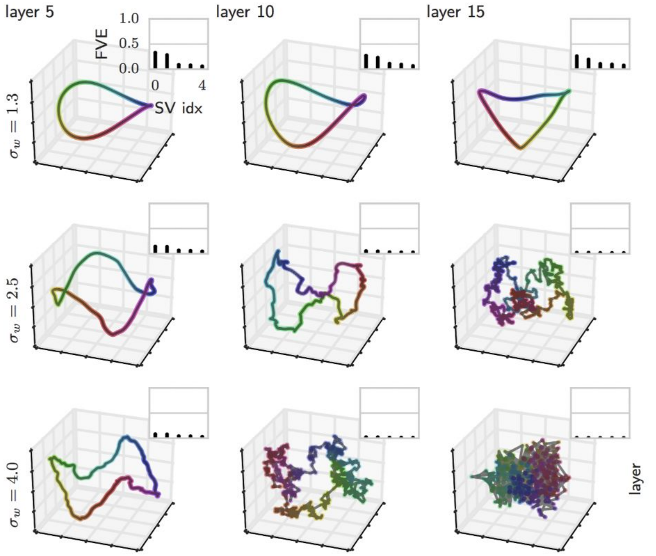
Variance of weights : $\frac{\sigma_w^2}{N}$

Variance of biases : σ_b^2



$\mathbf{x}^0(\theta)$





Small σ_w^2

Medium σ_w^2

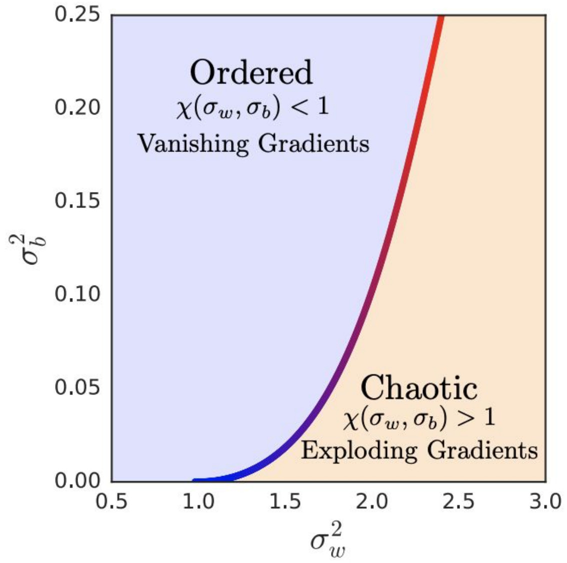
Large σ_w^2

Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli,
Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

Deep nets have an order chaos transition

A random deep network in the chaotic phase instantiates highly complex curved functions.

A shallow 1-hidden layer network would require exponentially many neurons to approximate this.

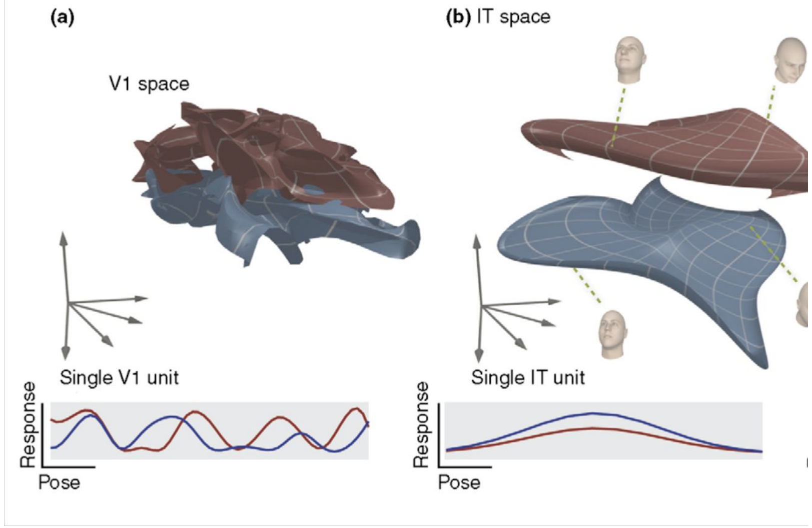


Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli,
Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

Another perspective: disentangling

Deep networks disentangle image manifolds of different classes.

They exploit a hierarchy of feature detectors (edges, sub-parts, parts, objects)



DiCarlo and Cox, Untangling invariant object recognition, Trends in Cognitive Sciences, 2007

Expressivity is not enough:
We need learnability also.

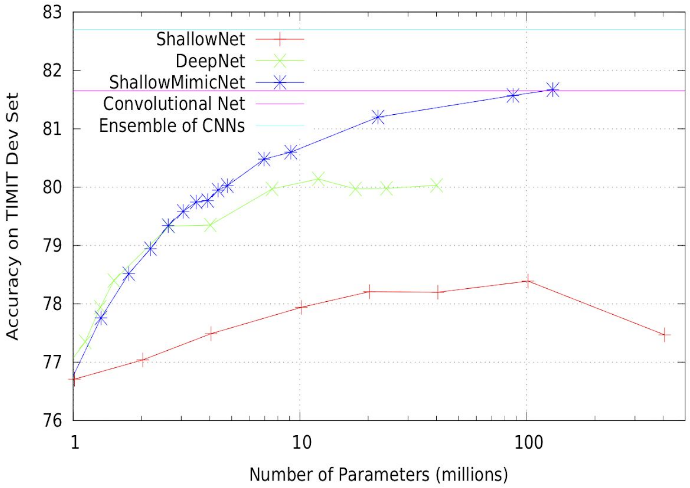
Expressivity only asks does there exist a network (of a given architecture) that approximates a desired target function well.

Existence alone does not imply we can find this function given finite data and finite computational time.

In fact, in some cases we can show a network exists, but we can’t find it in a straightforward manner.

An example where a shallow net is Expressive enough but not directly learnable.

A wide, shallow network can be trained to mimic a deep network, attaining significantly greater accuracy than training the shallow network directly on the data.



Ba & Caruana (2014)

Now your turn to experiment!

On a particular data set, lets see if:

a shallow wide network is better or

a narrow deep network is better

(while always keeping the number of parameters the same)

Might there be an ideal depth?

A case study on real data: classifying animal faces

Lecture 7

Try to classify animal faces using an MLP.

Explore data augmentation.

Explore what gets learned in the first layer.

Now, let us build a classifier for something more real

Animal faces

Images.

Image labels

Image augmentation

Load the data, take a look at it.

We generally do data augmentation when we load the data

What kind of augmentations make sense (slight rotations, horizontal flips?)

Get practice doing data augmentations with TorchVision transforms.

Train the network and inspect it

After we train a network, how does it work?

A very easy thing to do is to plot the weights in the first layer.

In an MLP, the weights into each first layer neuron can be interpreted as an image.

When viewed as images, what does the collection of first layer weights look like?

What do they reveal about how the network works?

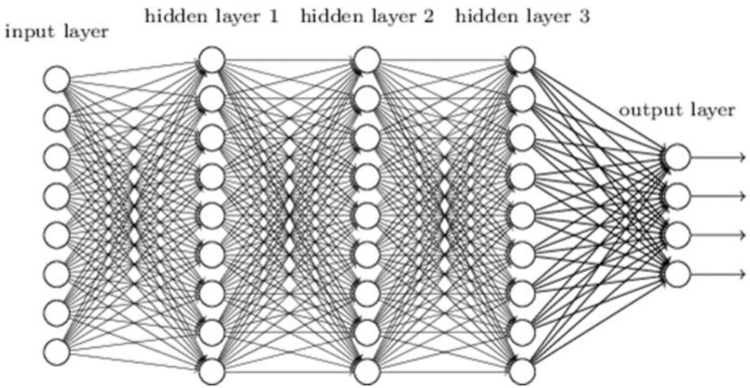
Interpretable / explainable AI: an entire field to generalize this to nonlinear settings.

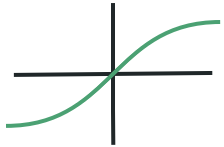
The need for good initialization

Lecture 8

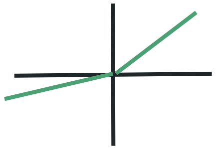
How should we choose the random weights before training?

Behavior of a randomly initialized net





Sigmoid



Leaky ReLU

Forward propagation of activity / back propagation of errors

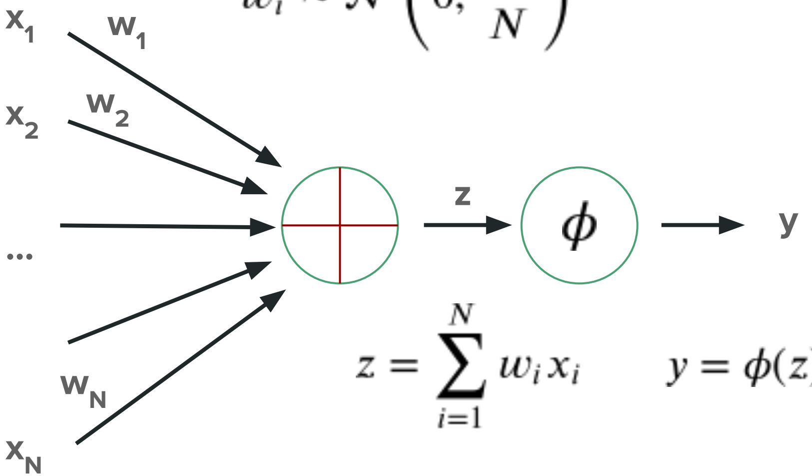
Weights too large: Could explode

Weights too small: Could vanish

Forward propagation of activity

$$x_i \sim \mathcal{N}(0, \sigma_x^2)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$



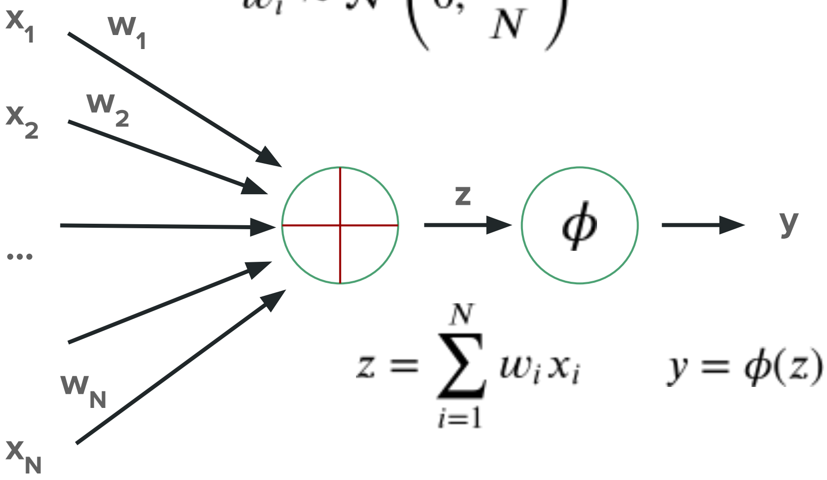
$$z = \sum_{i=1}^{N} w_i x_i \qquad y = \phi(z)$$

Forward propagation of activity

$x_i \sim \mathcal{N}(0, \sigma_x^2)$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$

$$Var[z] = N\, Var[W]\, Var[X]$$

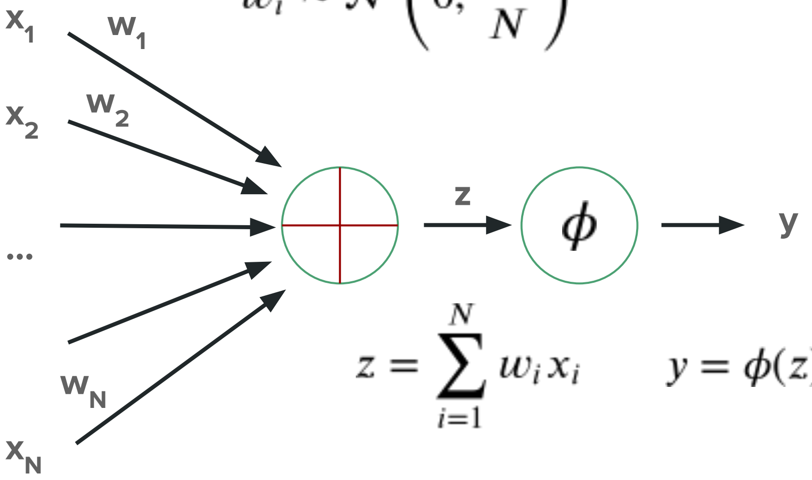


$$z = \sum_{i=1}^{N} w_i x_i \qquad y = \phi(z)$$

Forward propagation of activity

$$x_i \sim \mathcal{N}(0, \sigma_x^2)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$



$$z = \sum_{i=1}^{N} w_i x_i \qquad y = \phi(z)$$

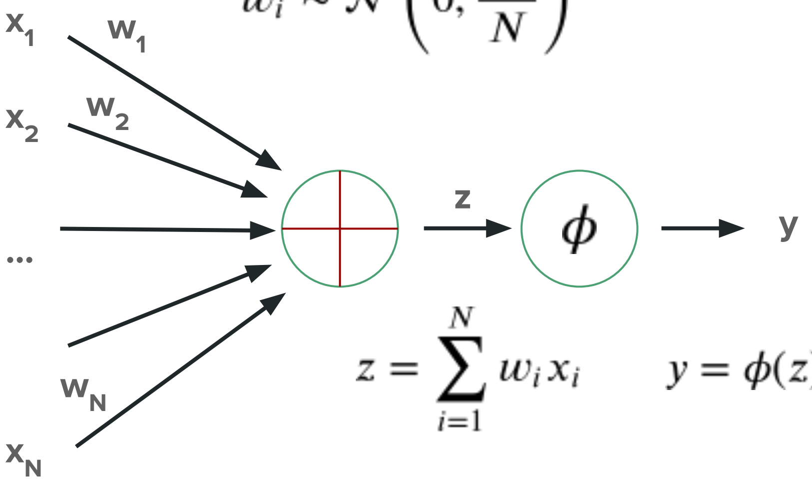
$$Var[z] = N\, Var[W]\, Var[X]$$

$$Var[z] = N\, \frac{\sigma_w^2}{N}\, \sigma_x^2 = \sigma_w^2 \sigma_x^2$$

Forward propagation of activity

$$x_i \sim \mathcal{N}(0, \sigma_x^2)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$



$$z = \sum_{i=1}^{N} w_i x_i \qquad y = \phi(z)$$

$$Var[z] = N\, Var[W]\, Var[X]$$

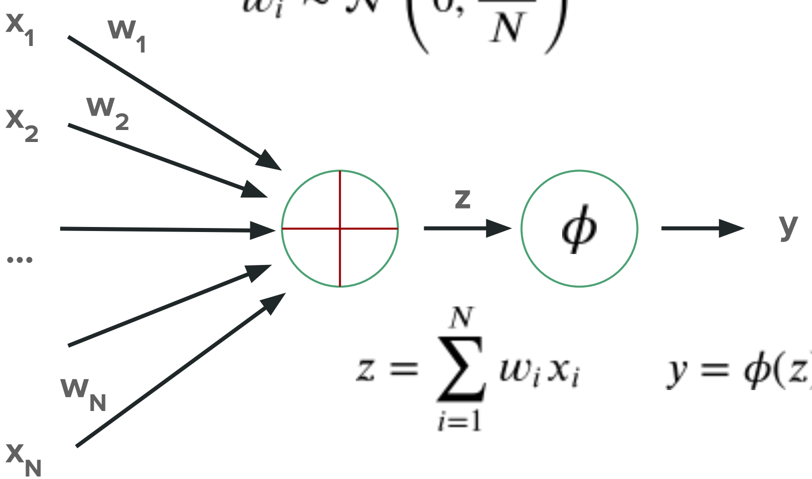
$$Var[z] = N \frac{\sigma_w^2}{N} \sigma_x^2 = \sigma_w^2 \sigma_x^2$$

$$Var[y] = \int dz P(z)\, \phi(z)^2 \quad \text{where} \quad P(z) = \mathcal{N}(0, \sigma_w^2 \sigma_x^2)$$

Forward propagation of activity

$$x_i \sim \mathcal{N}(0, \sigma_x^2)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$

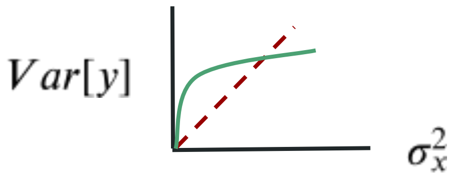


$$z = \sum_{i=1}^{N} w_i x_i \qquad y = \phi(z)$$

$$Var[z] = N\, Var[W]\, Var[X]$$

$$Var[z] = N \frac{\sigma_w^2}{N} \sigma_x^2 = \sigma_w^2 \sigma_x^2$$

$$Var[y] = \int dz P(z)\, \phi(z)^2 \quad \text{where} \quad P(z) = \mathcal{N}(0, \sigma_w^2 \sigma_x^2)$$



Stable forward propagation in ReLU nets

ReLU

$$Var[y] = \int_0^\infty dz P(z) z^2 = \frac{1}{2} Var[z]$$

$$Var[z] = N \, Var[W] \, Var[X]$$

$$Var[z] = N \, \frac{\sigma_w^2}{N} \, \sigma_x^2 = \sigma_w^2 \sigma_x^2$$

$$Var[y] = \int dz P(z) \, \phi(z)^2 \quad \text{where} \quad P(z) = \mathcal{N}(0, \sigma_w^2 \sigma_x^2)$$

Stable forward propagation in ReLU nets

ReLU

$$Var[y] = \int_0^\infty dz P(z) z^2 = \frac{1}{2} Var[z]$$

$$Var[y] = \frac{1}{2}\sigma_w^2 \sigma_x^2$$

$$Var[z] = N\, Var[W]\, Var[X]$$

$$Var[z] = N \frac{\sigma_w^2}{N} \sigma_x^2 = \sigma_w^2 \sigma_x^2$$

$$Var[y] = \int dz P(z)\, \phi(z)^2 \quad \text{where} \quad P(z) = \mathcal{N}(0, \sigma_w^2 \sigma_x^2)$$

Stable forward propagation in ReLU nets

ReLU

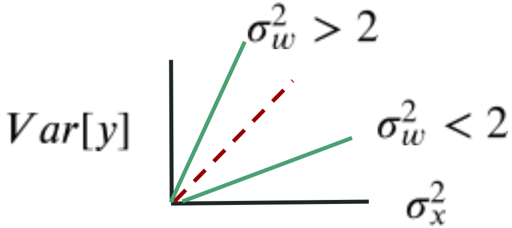
$$Var[z] = N\, Var[W]\, Var[X]$$

$$Var[z] = N\,\frac{\sigma_w^2}{N}\,\sigma_x^2 = \sigma_w^2\sigma_x^2$$

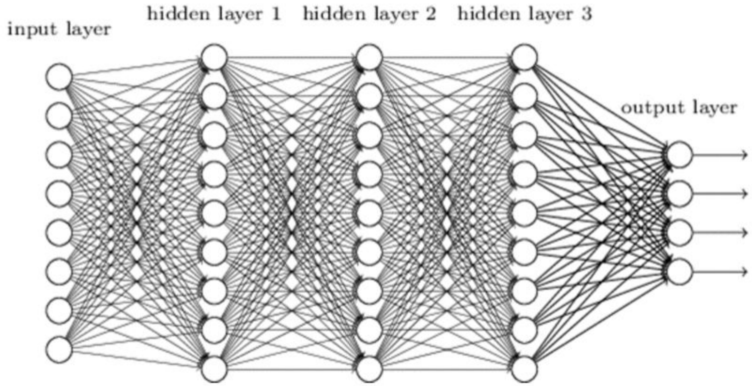
$$Var[y] = \int_0^\infty dz P(z) z^2 = \frac{1}{2} Var[z]$$

$$Var[y] = \frac{1}{2}\sigma_w^2\sigma_x^2$$

$$Var[y] = \int dz P(z)\,\phi(z)^2 \quad \text{where} \quad P(z) = \mathcal{N}(0, \sigma_w^2\sigma_x^2)$$



Simple intuition for ReLU forward prop:

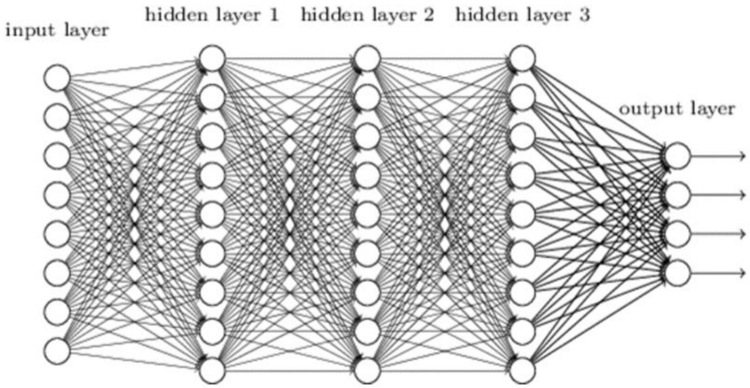


The ReLU reduces variance by killing half the units randomly.

So restore lost variance by cranking up the variance of the weights by a factor of 2/N

You work out the case for leaky ReLU

Now consider backpropagation



$$J_{ij} = W_{ij}\phi'(x_j)$$

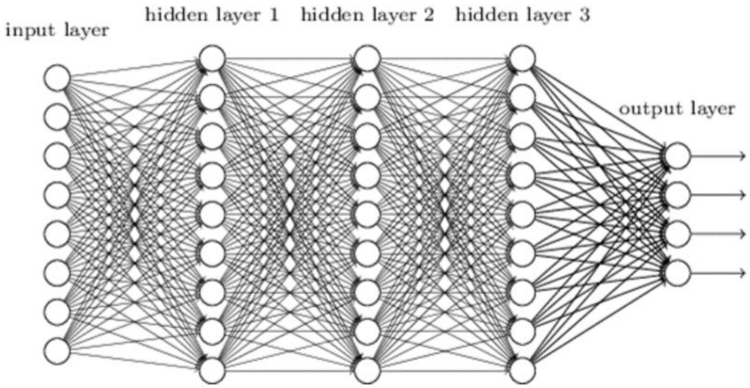
$$1 = \sigma_w^2 \int dz P(z) \phi'(z)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$

Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli, Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

Schoenholz, Gilmer, Ganguli, Sohl-Dickstein, Deep information propagation, ICLR 2017.

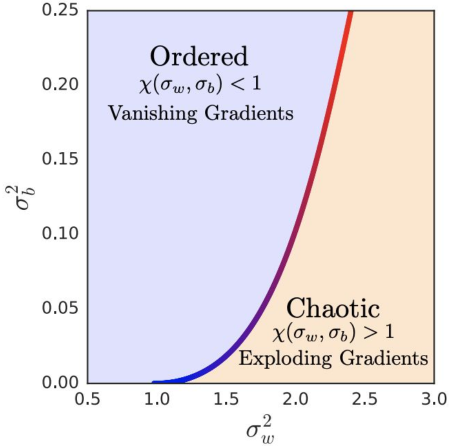
Now consider backpropagation



$$J_{ij} = W_{ij}\phi'(x_j)$$

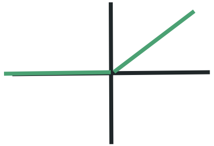
$$1 = \sigma_w^2 \int dz P(z) \phi'(z)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$



Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli, Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.
Schoenholz, Gilmer, Ganguli, Sohl-Dickstein, Deep information propagation, ICLR 2017.

Now consider backpropagation for ReLU



ReLU

$$J_{ij} = W_{ij}\phi'(x_j)$$

$$1 = \frac{\sigma_w^2}{2}$$

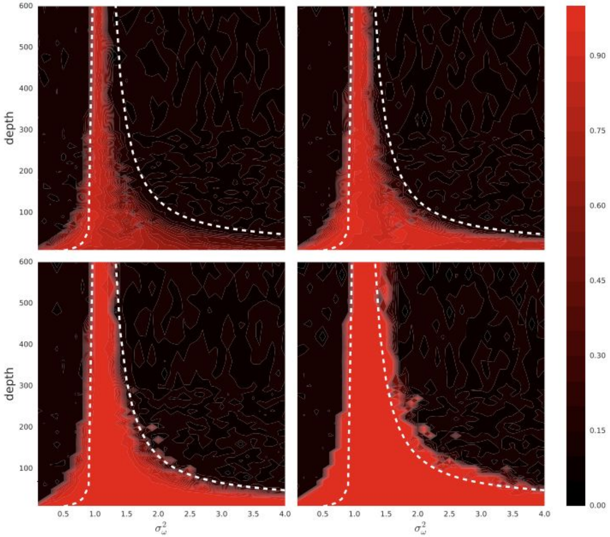
$$1 = \sigma_w^2 \int dz P(z)\phi'(z)$$

$$w_i \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{N}\right)$$

Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli, Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

Schoenholz, Gilmer, Ganguli, Sohl-Dickstein, Deep information propagation, ICLR 2017.

What you can do with a great init



Xiao, Bahri, Sohl-Dickstein, Schoenholz, Pennington, Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks, ICML 2018

Now your turn!

Work through the analysis of the leaky ReLU network.

Go beyond theory to practice: scan the variance of the initial weights as a hyperparameter.

Does the best performance actually occur at the scale of init predicted by theory for avoiding vanishing / exploding propagation?

Ethics: Hype in AI

Lecture 9

“All of this has happened before and all of this will happen again.” - Battlestar Galactica

A history of hype in AI

A steely eyed view of AI accomplishments.

A history of hype in AI

Melanie Mitchell. Why AI is harder than we think.

1958 Perceptron. NY Times: “..Navy revealed the embryo of an electronic computer that it expects will be able to walk, talk, see, write, reproduce itself, and be conscious.”

1960: Herbert Simon: “Machines will be capable, within twenty years, of doing any work that a [hu]man can do.”

1961: Claude Shannon: “I confidently expect that within a matter of 10 or 15 year something will emerge … which is not too far from the robot of science fiction fame.”

1960’s Marvin Minsky: “Within a generation… the problems of creating AI will be substantially solved.

OK... so what happened next?

OK… so what happened next?

An AI winter (the first one)

Melanie Mitchell. Why AI is harder than we think.

1969 Minsky and Papert book: Perceptrons

1973: Government reports in the US and the UK: extremely negative reports about the prospects for AI.

Lead to sharp decreases in both funding and enthusiasm for AI in the 70's.

An AI spring and winter (the 2nd one)

Melanie Mitchell. Why AI is harder than we think.

1980's Rise of expert systems and parallel distributed computing

Early 80's: large increases in funding and promises

Promises not delivered; expert systems brittle; neural nets can't scale

Lead to sharp decreases in both funding and enthusiasm for AI in the 90's.

People told not to put "artificial intelligence" on their CV's in the 90's

That’s okay - we’ve learned our lesson, right?

Melanie Mitchell. Why AI is harder than we think.

2015: The Guardian: “In 2020 you will be a permanent back seat driver.”

2016: Business Insider: “10 million self-driving cars will be on the road by 2020”

2019: Elon Musk: “A year from now, we’ll have over a million cars with full self-driving, software…everything.”

Several automobile companies announced 2020 as a target for self-driving cars.

Some reading on AI history and hype

Melanie Mitchell:Why AI is harder than we think

Michael Jordan: Artificial Intelligence— The Revolution Hasn't Happened Yet

Amanda Gefter: On Walter Pitts, The Man Who Tried to Redeem the World with Logic

Jeremy Bernstein: Marvin Minsky's vision of the future, New Yorker

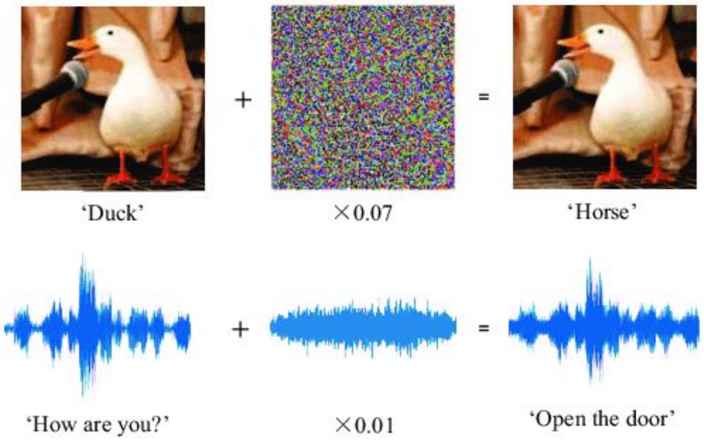
To combat hype: recall major performance gaps between humans and machines

THE INTERTWINED QUEST FOR UNDERSTANDING BIOLOGICAL INTELLIGENCE AND CREATING ARTIFICIAL INTELLIGENCE,

Surya Ganguli
HAI.STANFORD.EDU/BLOG

Robustness: humans versus machines

Adversarial examples: deep nets in vision, speech, language and RL are fooled by illusions that never fool humans.



Why do adversarial examples exist?

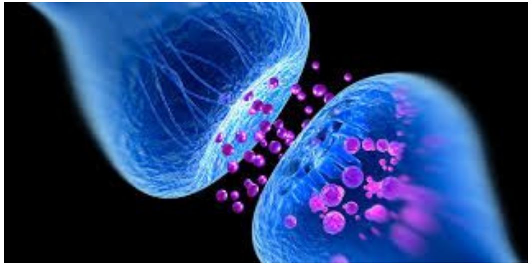
How does the brain avoid them?

Energy cost: humans versus machines

The human brain spends 20 watts of energy.

Our supercomputers consume 10 million times more power.

Machines: fast and precise control of digital bits.



Biology: slow and noisy but just good enough.

Credit assignment: humans versus machines

You saw you hit the tennis ball wrong.

Which one of your hundred trillion synapses screwed up?

How does the brain figure out how to fix the wrong ones,

using a local learning rule?

Neither neuroscience nor AI knows how to solve this problem.

Data hunger: humans versus machines

Early speech recognition: 16 years of someone reading you text 2 hours every day.

Alpha Go: practice 450 games a day every day for 30 years.

Visual question answering: receive answers to 100 questions about images every day for 274 years.

We need better algorithms for unsupervised learning, transfer learning, curriculum design.

Active learning: humans versus machines

Even babies:

Build complex internal models of world dynamics

Pay attention to events that violate the world model

Perform active experiments to test the model and gather their own data.

Use the model to plan and imagine alternate futures.

Takes actions to make alternate futures a reality.

Some reading on AI history and hype

Melanie Mitchell:Why AI is harder than we think

Michael Jordan: Artificial Intelligence— The Revolution Hasn’t Happened Yet

Amanda Gefter: On Walter Pitts, The Man Who Tried to Redeem the World with Logic

Jeremy Bernstein: Marvin Minsky’s vision of the future, New Yorker

Surya Ganguli: Intertwined quest for understanding biological intelligence and creating artificial intelligence

Multilayer perceptrons: outro

Surya Ganguli and Arash Ash

neuromatch academy

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

3 major puzzles of deep learning

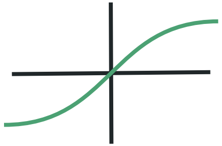
- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
    - Highly complex / chaotic functions
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

Expressivity from chaos hiding in deep nets

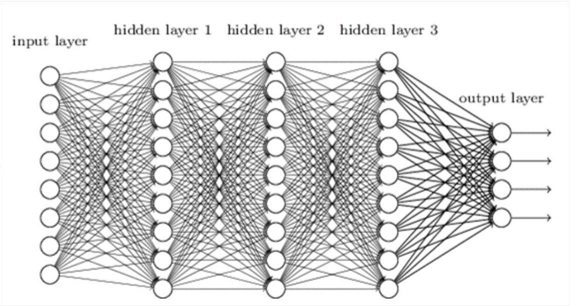
Random deep sigmoidal neural net

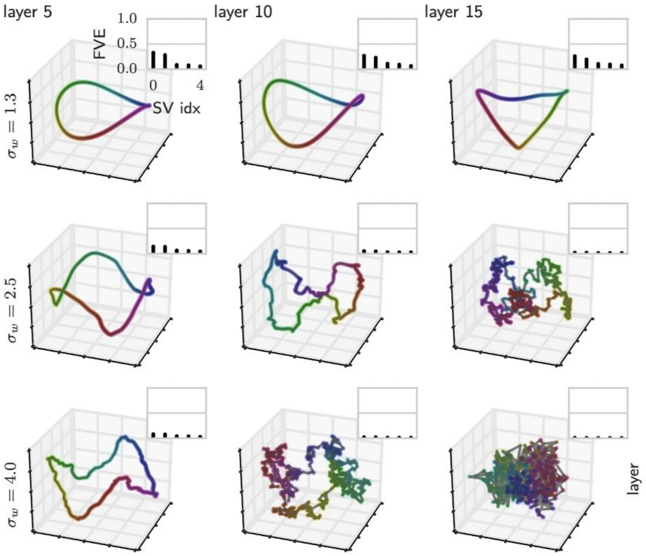
Variance of weights : $\frac{\sigma_w^2}{N}$

Variance of biases : σ_b^2



$\mathbf{x}^0(\theta)$





Small σ_w^2

Medium σ_w^2

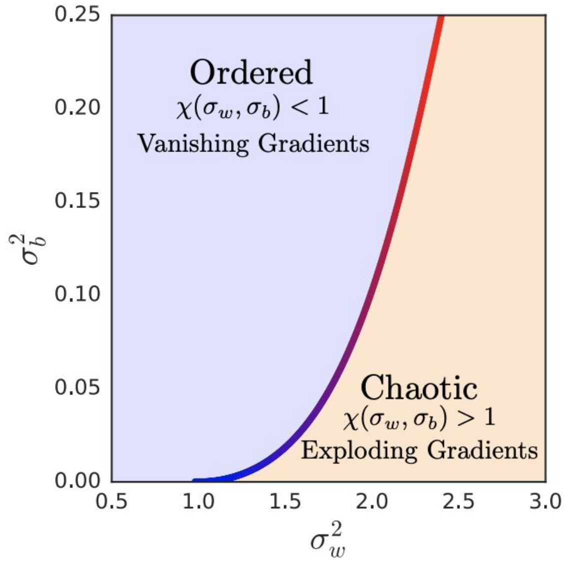
Large σ_w^2

Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli,
Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

Deep nets have an order chaos transition

A random deep network in the chaotic phase instantiates highly complex curved functions.

A shallow 1-hidden layer network would require exponentially many neurons to approximate this.



Poole, Lahiri, Raghu, Sohl-Dickstein, Ganguli,
Exponential expressivity in deep neural networks through transient chaos, NeurIPS 2016.

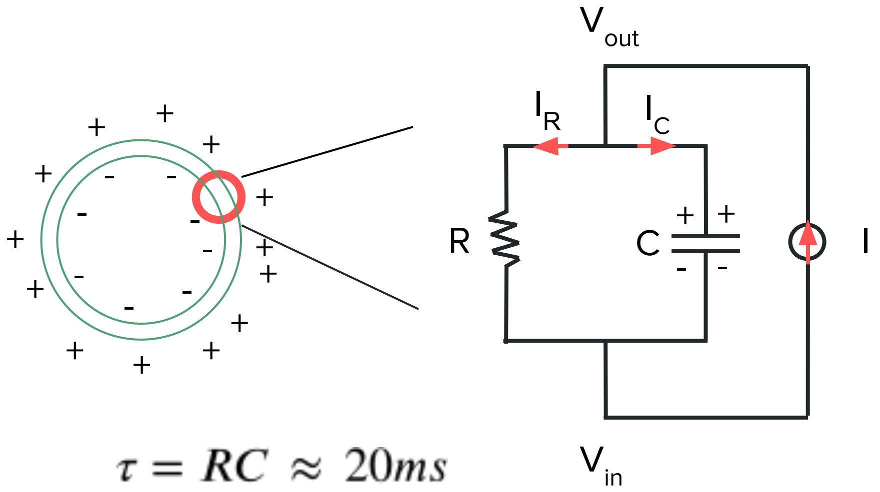
3 major puzzles of deep learning

- **Expressivity**:
    - What kinds of functions can deep nets efficiently compute or express?
    - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
    - How should we initialize them before we optimize them?
    - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
    - How do we help deep nets generalize to new held out data?

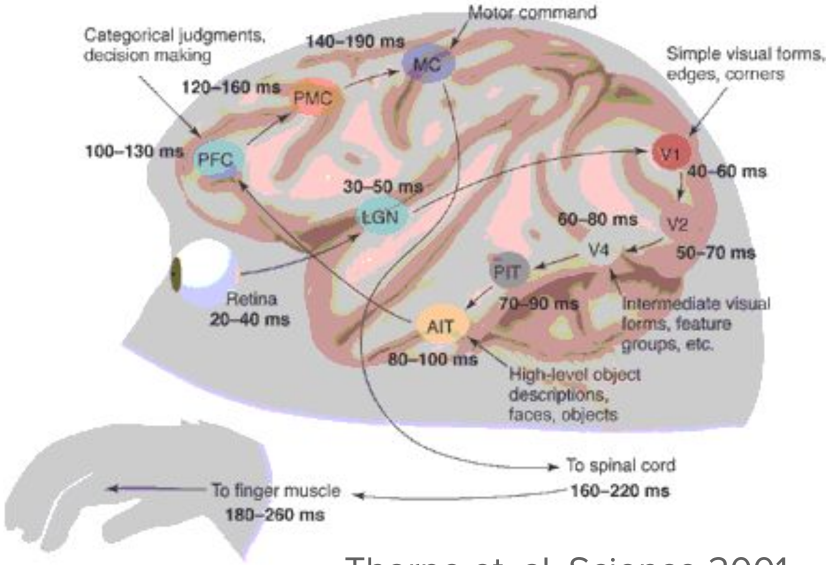
3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
    - Temporal signal propagation through the brain may not be too deep
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

From biophysics to psychology: the “clock” speed of the brain and mind



$$\tau = RC \approx 20ms$$



Thorpe et. al. Science 2001

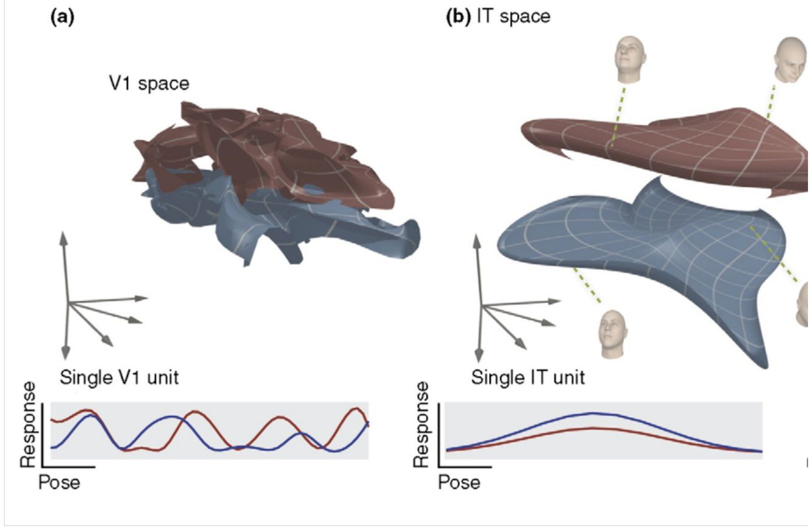
3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
    - Deep nets can disentangle a hierarchy of features
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

Another perspective: disentangling

Deep networks disentangle image manifolds of different classes.

They exploit a hierarchy of feature detectors (edges, sub-parts, parts, objects)



DiCarlo and Cox, Untangling invariant object recognition, Trends in Cognitive Sciences, 2007

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
    - Deep generative models of data succeed? (Week 2 Day 5)
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

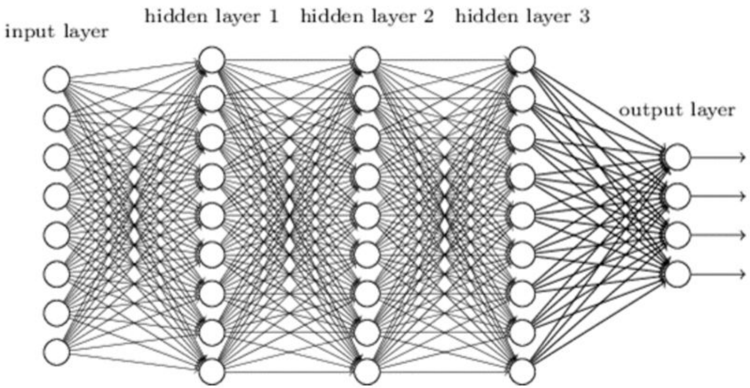
3 major puzzles of deep learning

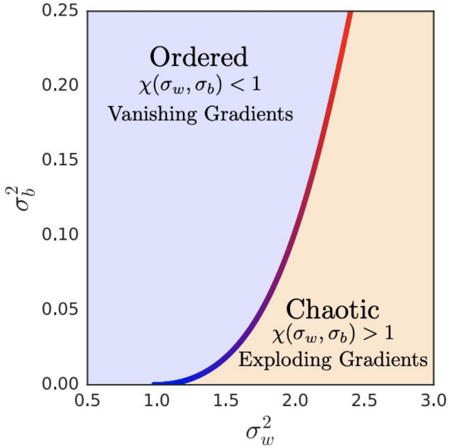
- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
    - Initialize at the edge of chaos!
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

Tutorial 8: The need for a good initialization: stick to the edge of chaos!





Forward propagation of activity / back propagation of errors

Weights too large:	Could explode
Weights too small:	Could vanish

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
    - Optimization (Week 1 Day 4)
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?

3 major puzzles of deep learning

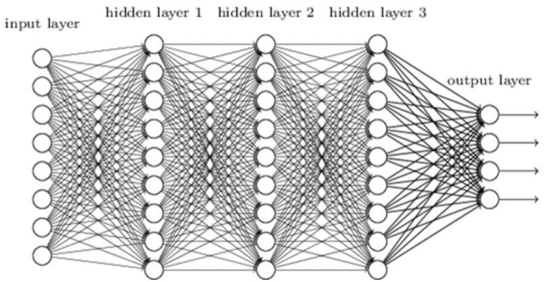
- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?
    - Regularization (Week 1 Day 5)

3 major puzzles of deep learning

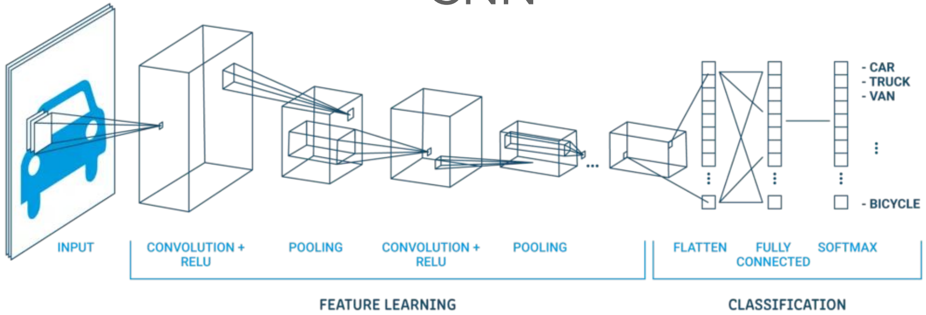
- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?
    - Build in inductive biases; CNNs, RNNs (Week 2 Days 1-3)

MLPs are a basis for CNNs and RNNs

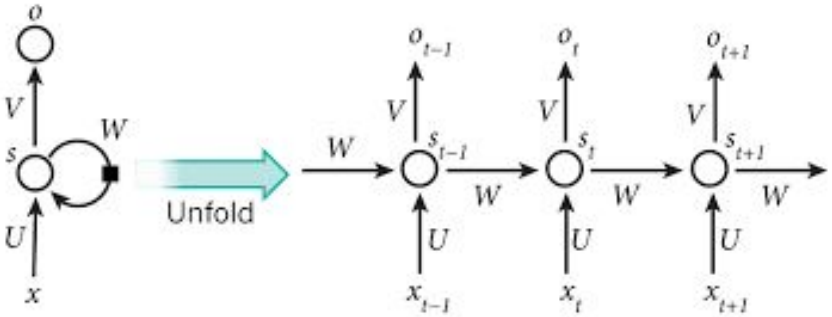
MLP



CNN



RNN



3 major puzzles of deep learning

- **Expressivity**:
  - What kinds of functions can deep nets efficiently compute or express?
  - Why are the natural functions that we care about computing in AI easily computable by deep nets (or are they)?
- **Trainability**:
  - How should we initialize them before we optimize them?
  - How can we optimize deep nets to achieve zero training error?
- **Generalization:**
  - How do we help deep nets generalize to new held out data?
    - Or scale up training data sets w/ Transformers (Week 2 Day 4)

Training and evaluation

Best practices for training and evaluating, as well as pitfalls:

What is the training set?

Does the split between train and test match your use case?

Is your metric for evaluation reasonable for real world deployment?

Will future validation data drift from train and test settings?

Are there biases due to problem selection, training data, algorithm design, evaluation metrics, or anywhere in ML pipeline?

Ethics: beware of hype in AI

Melanie Mitchell:Why AI is harder than we think

Michael Jordan: Artificial Intelligence— The Revolution Hasn’t Happened Yet

Amanda Gefter: On Walter Pitts, The Man Who Tried to Redeem the World with Logic

Jeremy Bernstein: Marvin Minsky’s vision of the future, New Yorker

Surya Ganguli: Intertwined quest for understanding biological intelligence and creating artificial intelligence