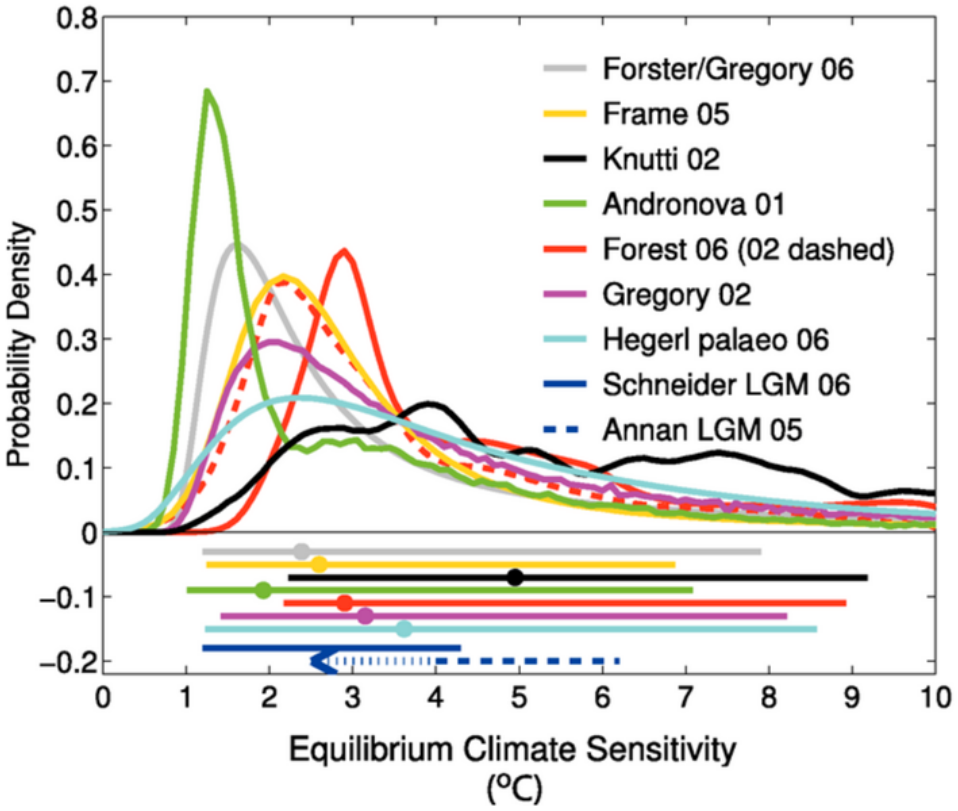
Future of Probability

Chris Piech
CS109, Stanford University



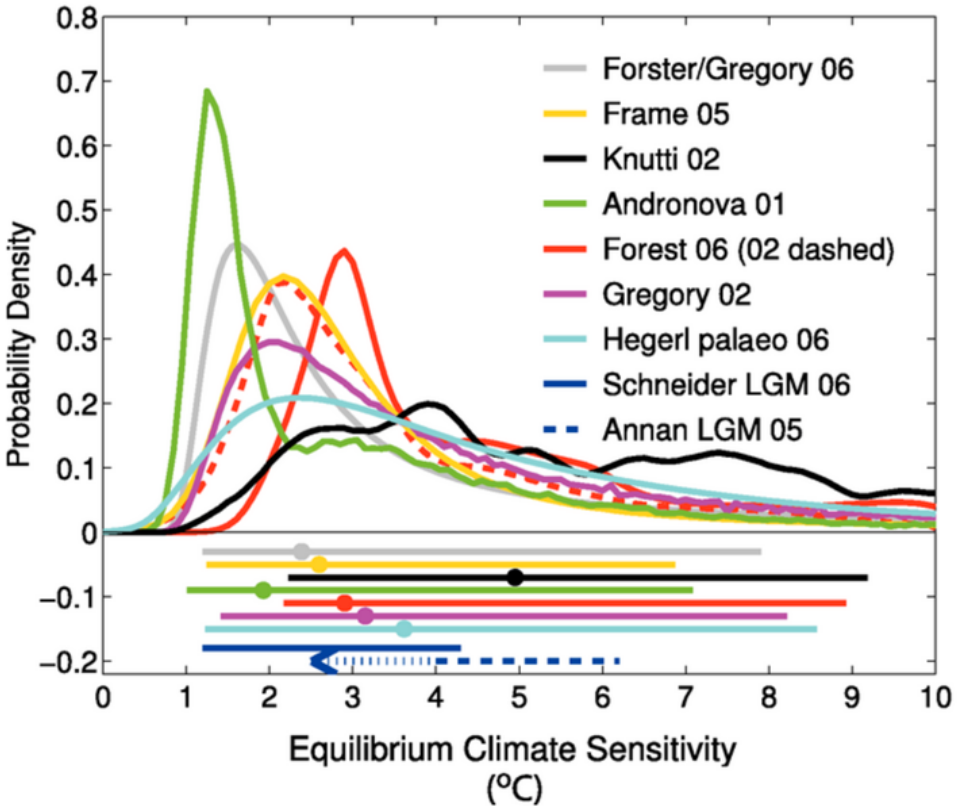
Sensitivity, S (degrees C)	0	1	2	3	4	5	6	7
IPCC Probability Mass	0.00	0.11	0.26	0.22	0.16	0.09	0.06	0.04

6% chance that sensitivity is greater than 7

Expectation is around 3-4 degrees Celsius

* This estimate was conservative (and based on experiments from ~2010). Sensitivity is probably higher

What about 2 degrees?



Sensitivity, S (degrees C)	0	1	2	3	4	5	6	7
IPCC Probability Mass	0.00	0.11	0.26	0.22	0.16	0.09	0.06	0.04

6% chance that sensitivity is greater than 7

Expectation is around 3-4 degrees Celsius

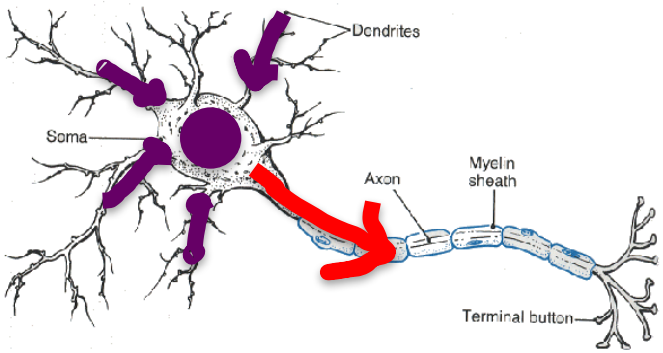
* This estimate was conservative (and based on experiments from ~2010). Sensitivity is probably higher

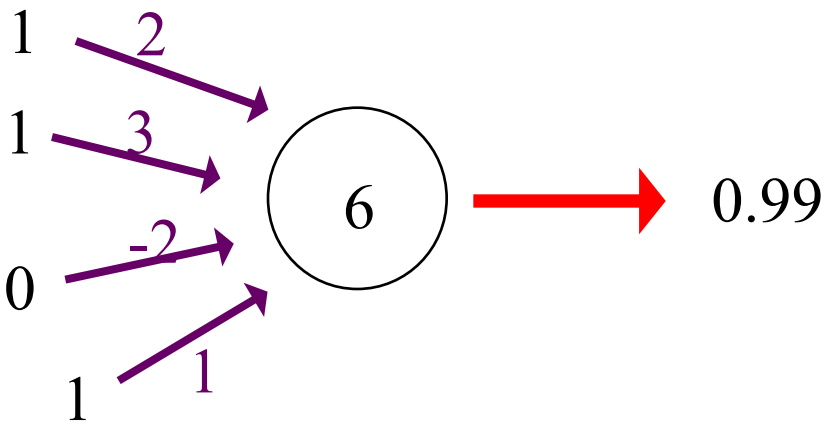
Other random variables?

We have 4 years at current CO_2 emission levels before we have a 66 percent chance of exceeding 2 degrees (C) rise in average temperature.

Missing piece: human ingenuity?

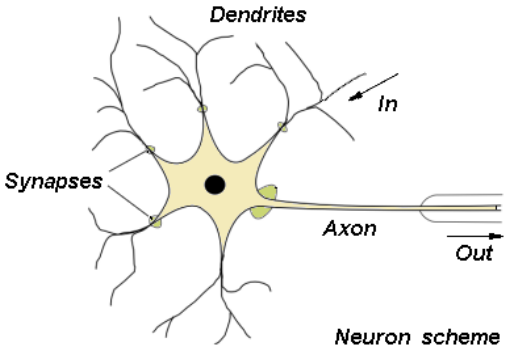
Artificial Neuron

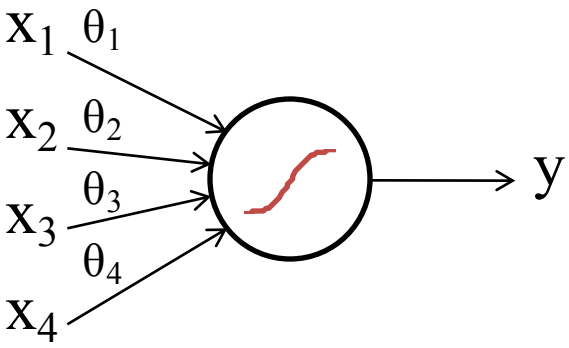




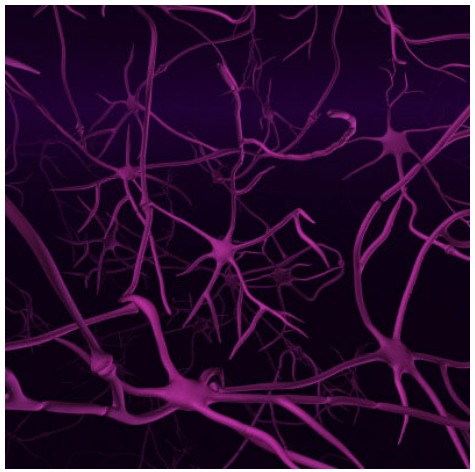
Biological Basis for Neural Networks

- A neuron

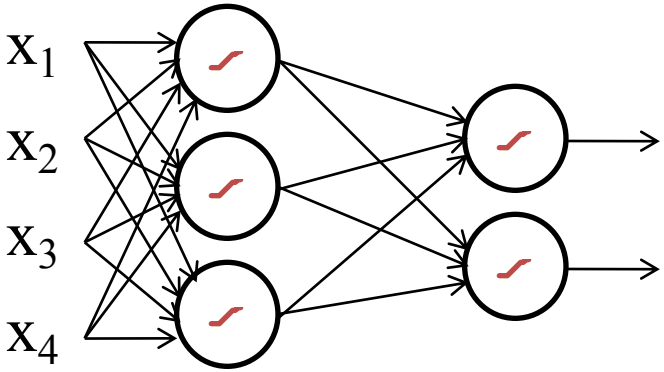




- Your brain



Actually, it's probably someone else's brain

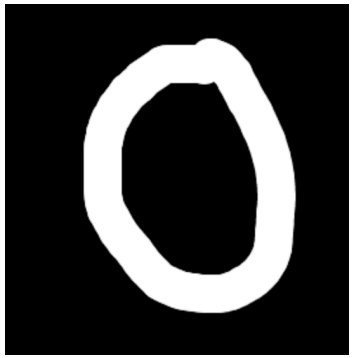


(aka Neural Networks)

Deep learning is (at its core) many logistic regression pieces stacked on top of each other.

Digit Recognition Example

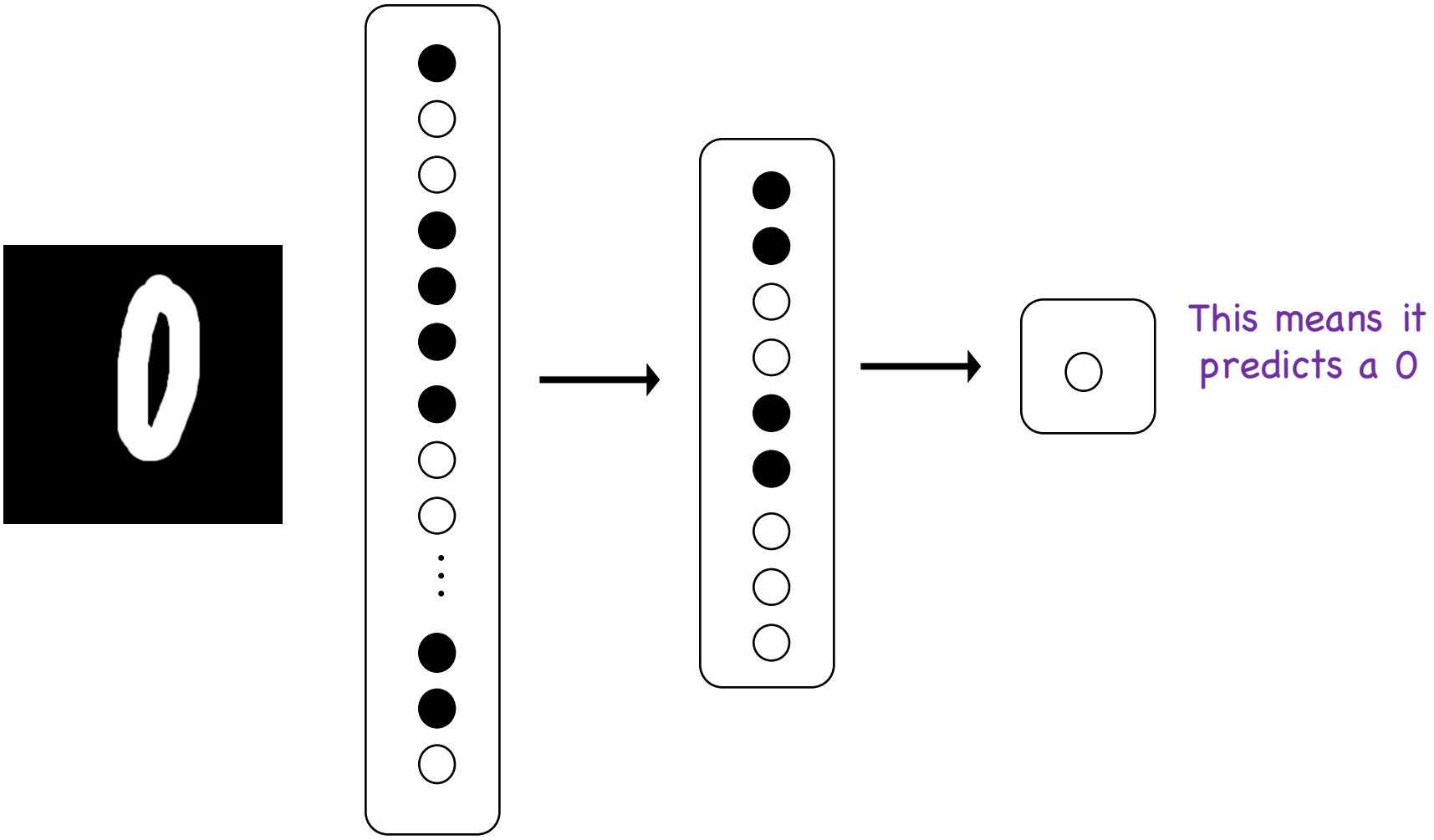
Let's make feature vectors from pictures of numbers



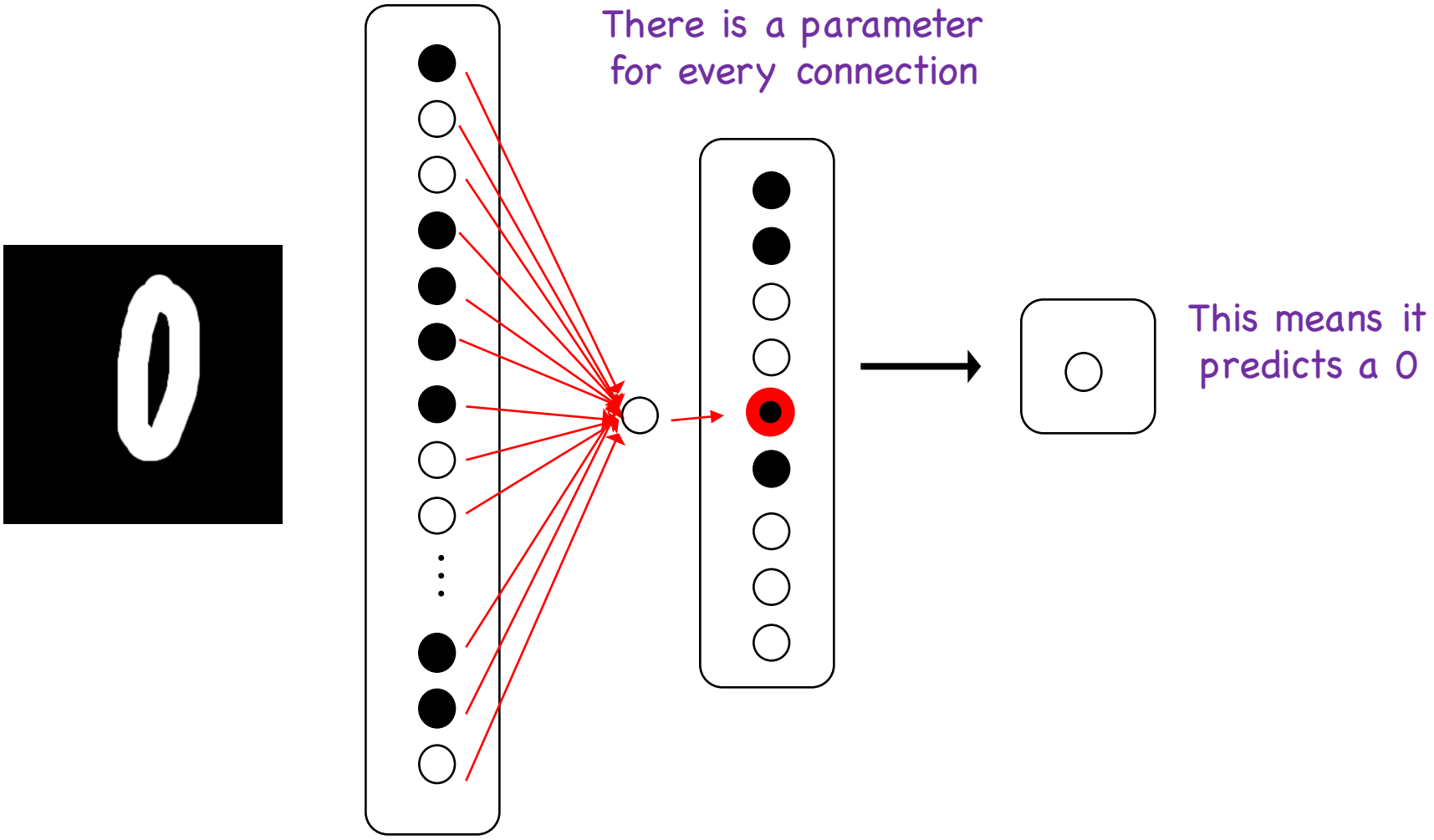
$$\mathbf{x}^{(i)} = [0, 0, 0, 0, \ldots, 1, 0, 0, 1, \ldots 0, 0, 1, 0]$$
$$y^{(i)} = 0$$

$$\mathbf{x}^{(i)} = [0, 0, 1, 1, \ldots, 0, 1, 1, 0, \ldots 0, 1, 0, 0]$$
$$y^{(i)} = 1$$

We Can Put Neurons Together

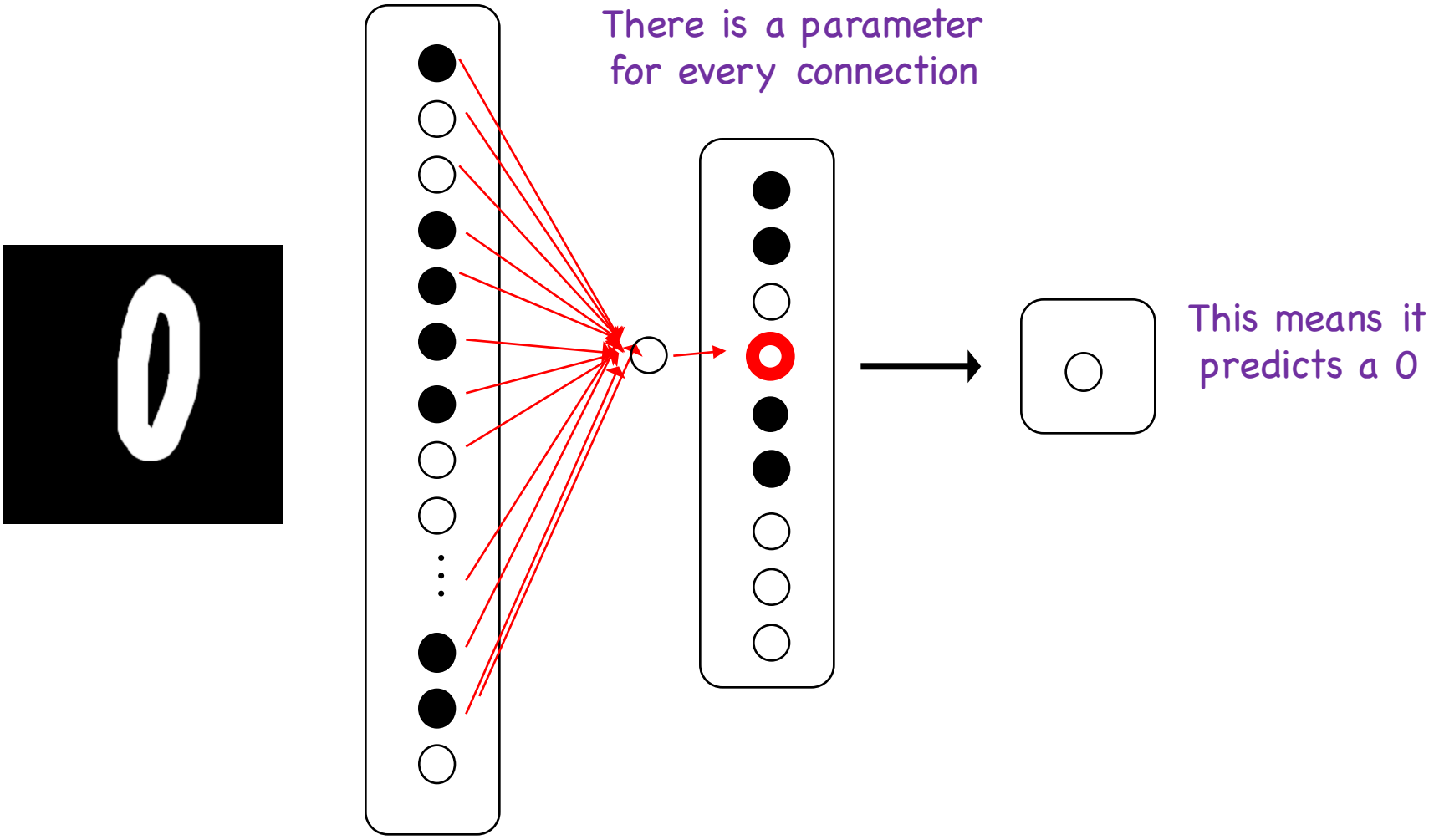


We Can Put Neurons Together



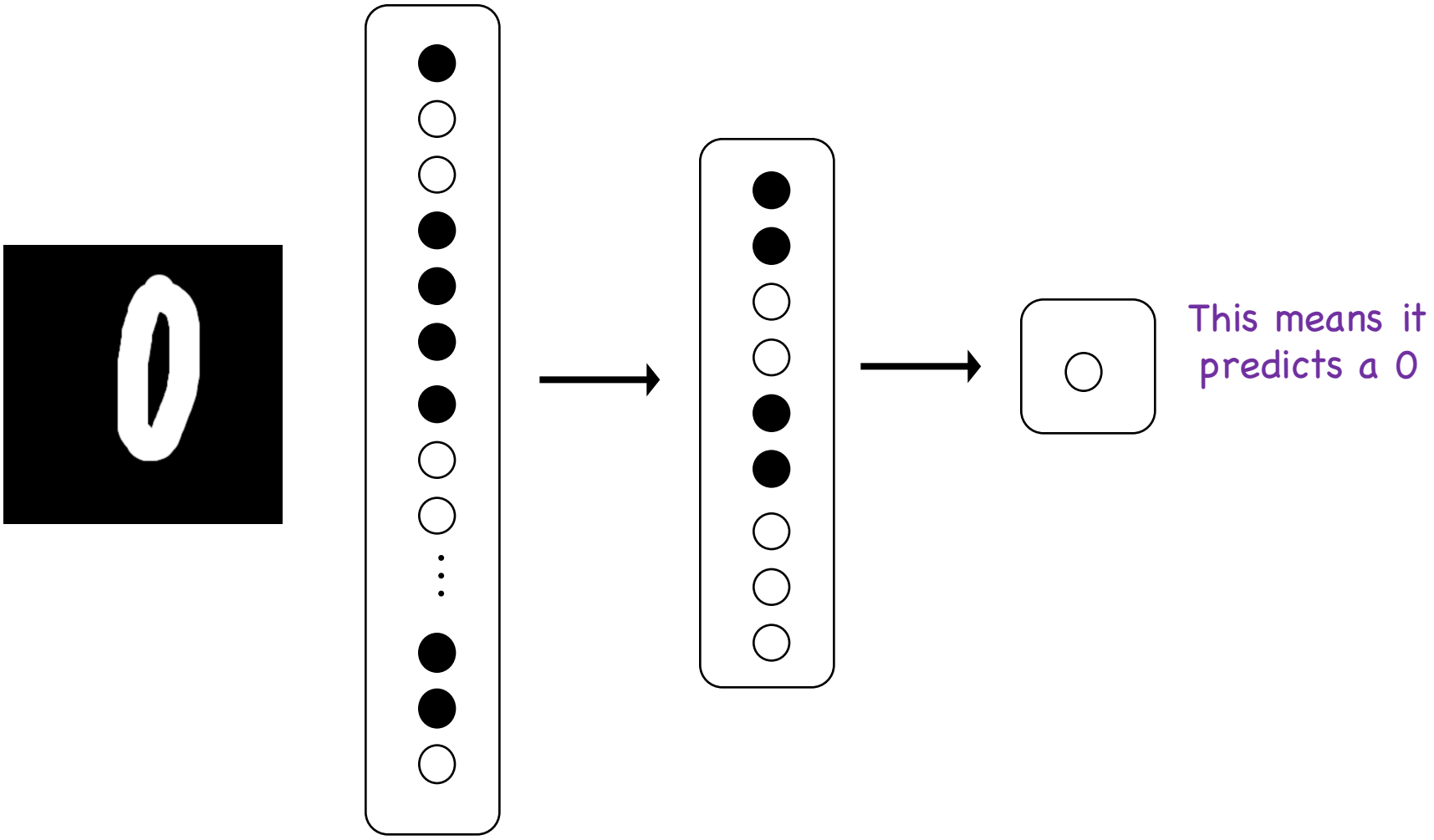
Look at a single "hidden" neuron

We Can Put Neurons Together

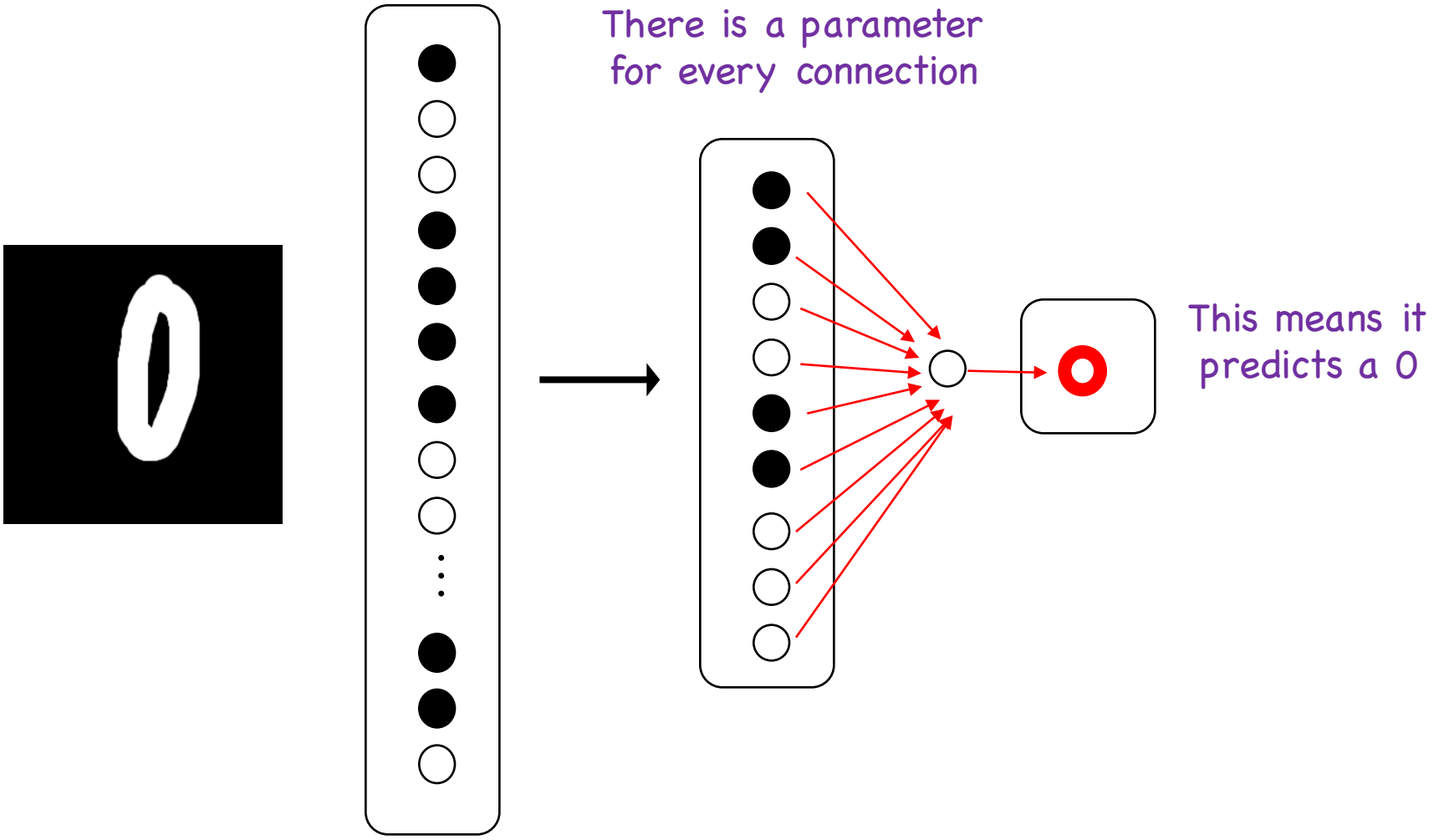


Look at another "hidden" neuron

We Can Put Neurons Together

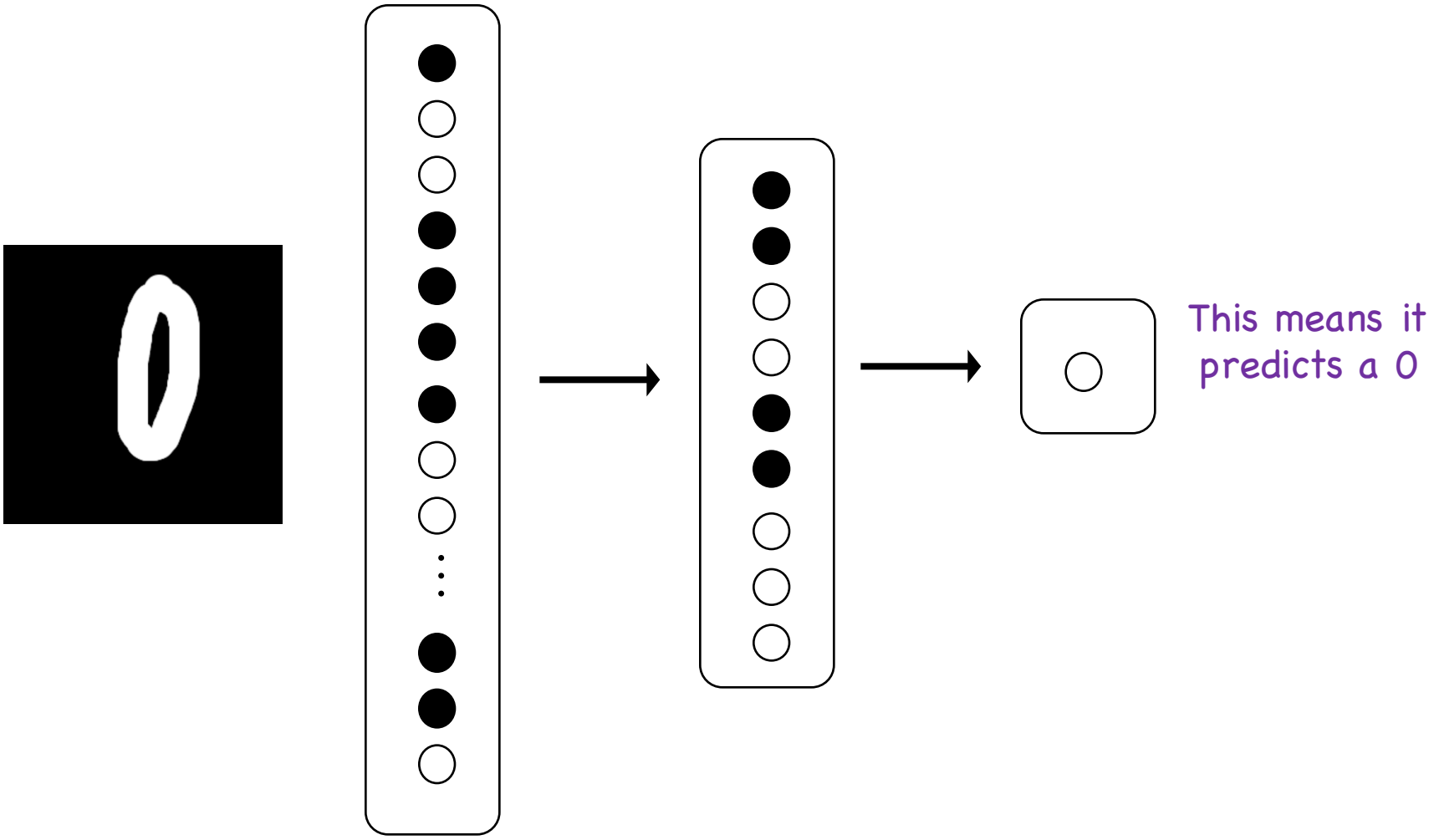


We Can Put Neurons Together



Look at another neuron

We Can Put Neurons Together



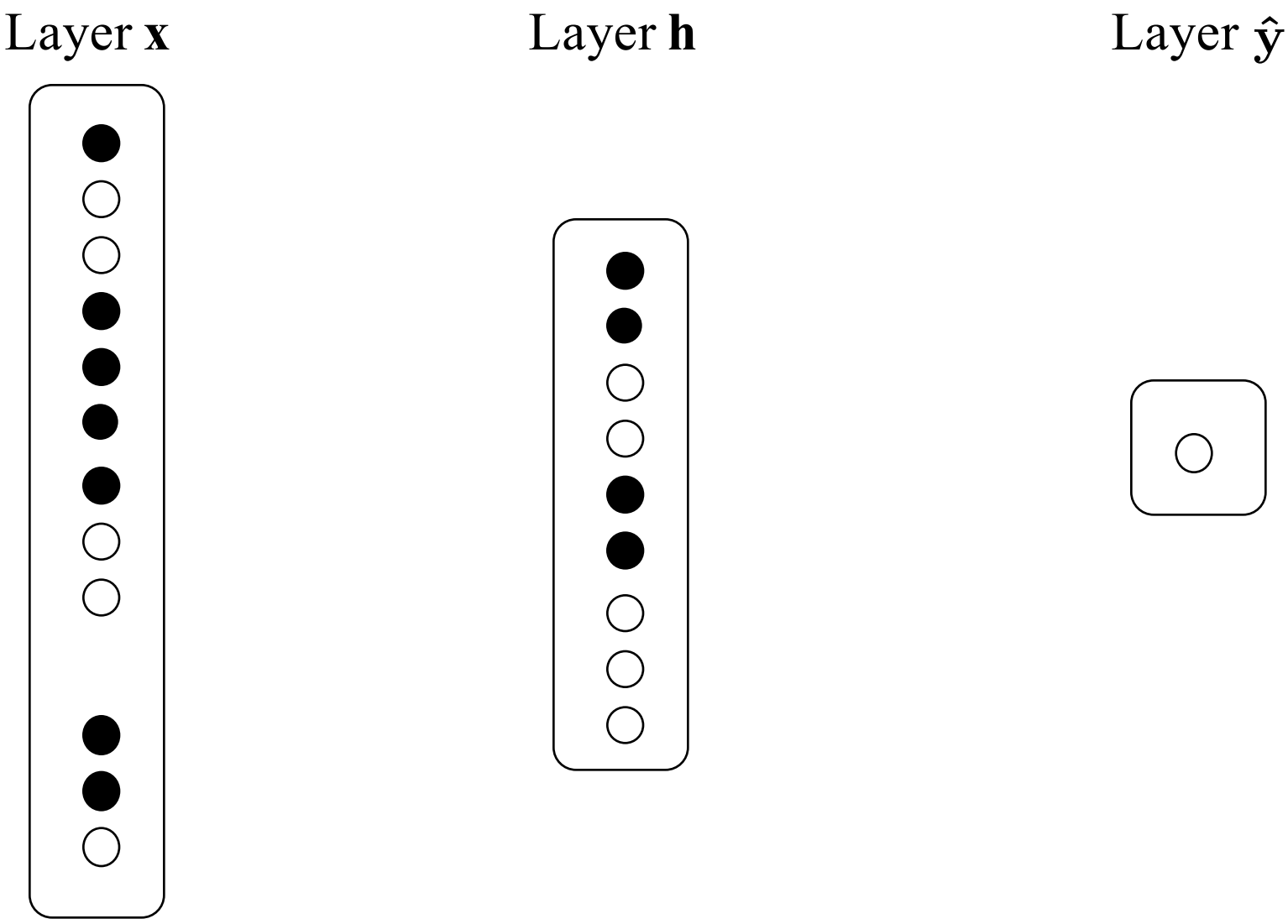
Deep Learning Assumption

- Model *conditional* likelihood P(Y | **X**) directly

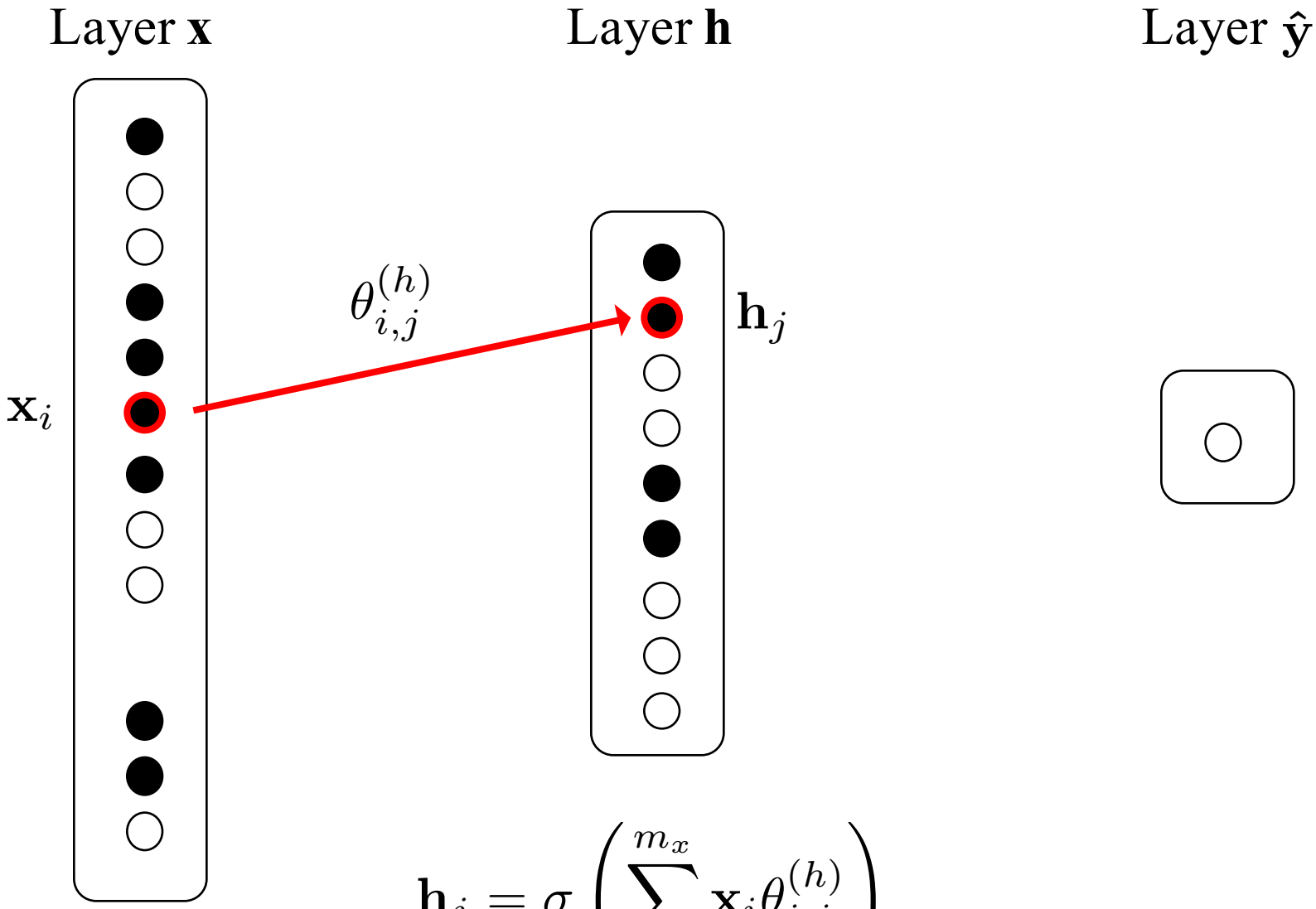
$$P(Y = 1|\mathbf{X}) =$$ The output of a chain of logistic regressions

Deep learning gets its ***intelligence*** from its thetas (aka its parameters)

New Notation

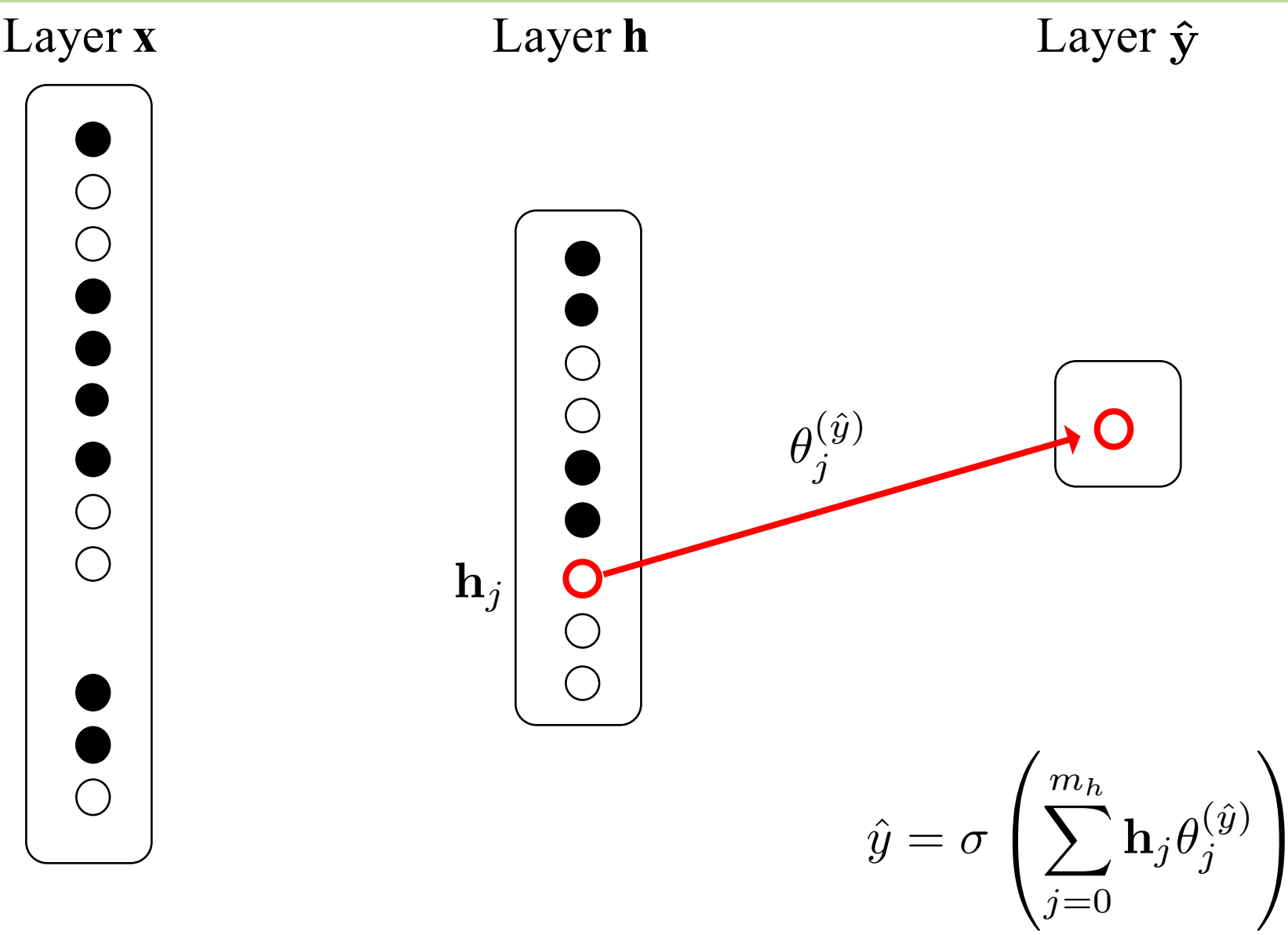


New Notation



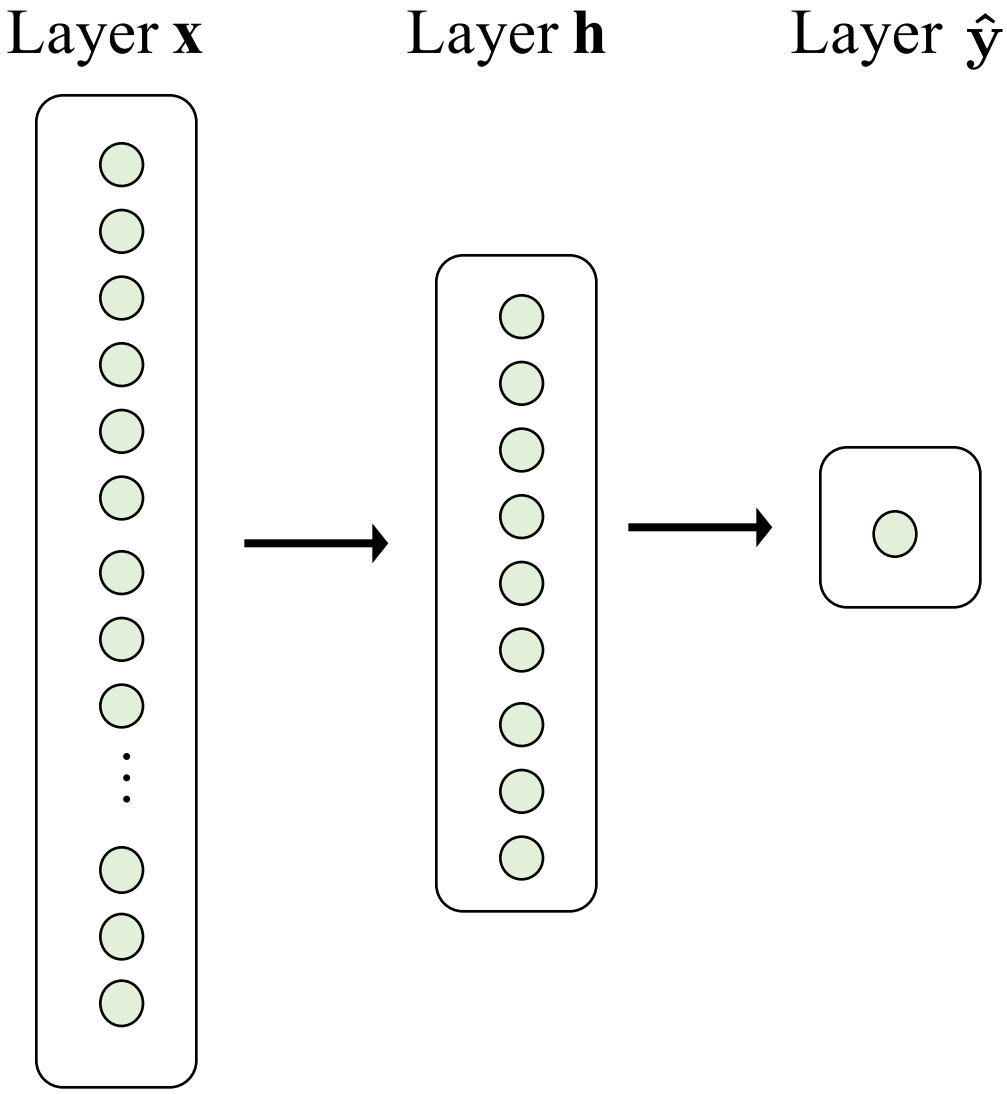
$$\mathbf{h}_j = \sigma\left(\sum_{i=0}^{m_x} \mathbf{x}_i \theta_{i,j}^{(h)}\right)$$

New Notation

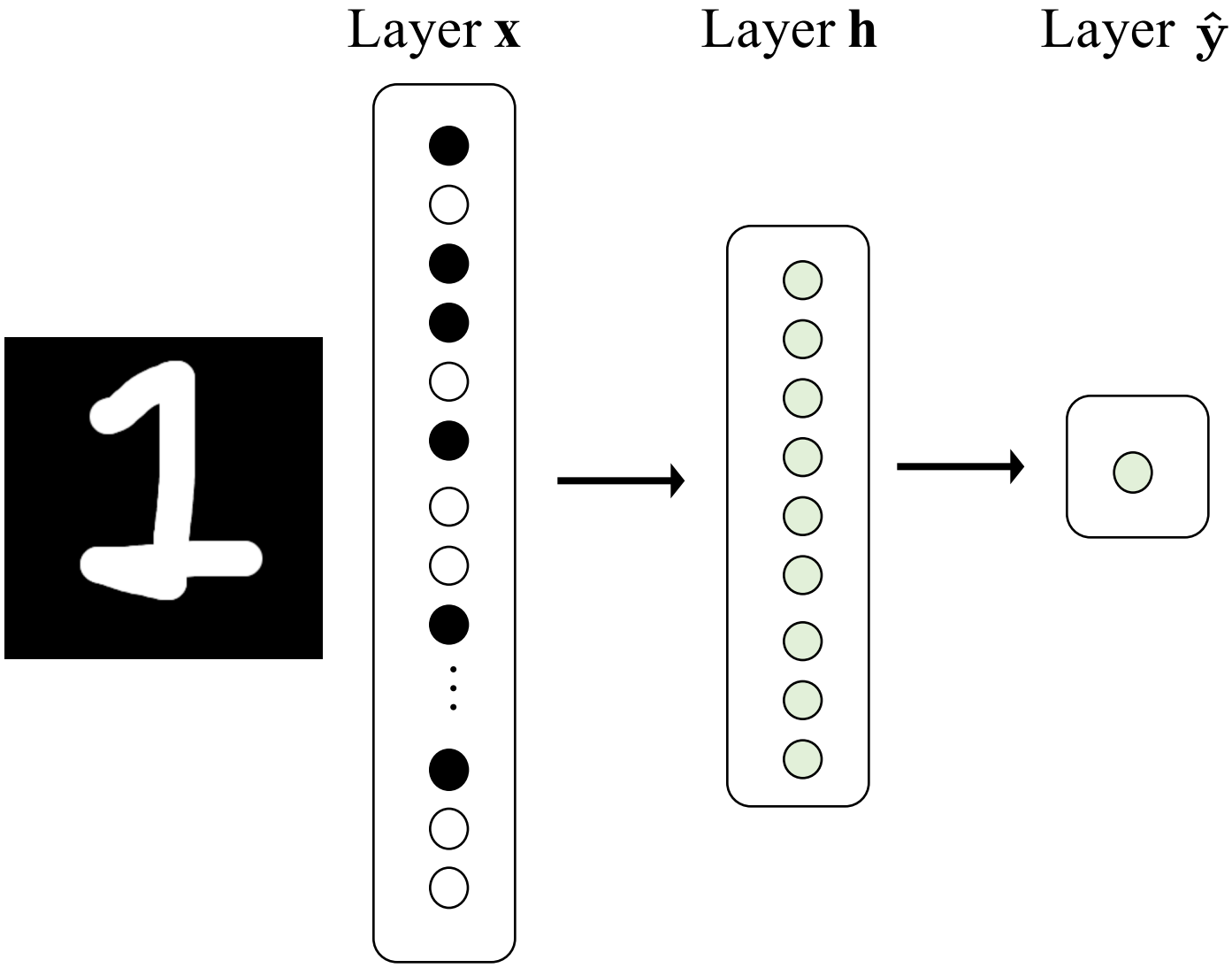


$$\hat{y} = \sigma\left(\sum_{j=0}^{m_h} \mathbf{h}_j \theta_j^{(\hat{y})}\right)$$

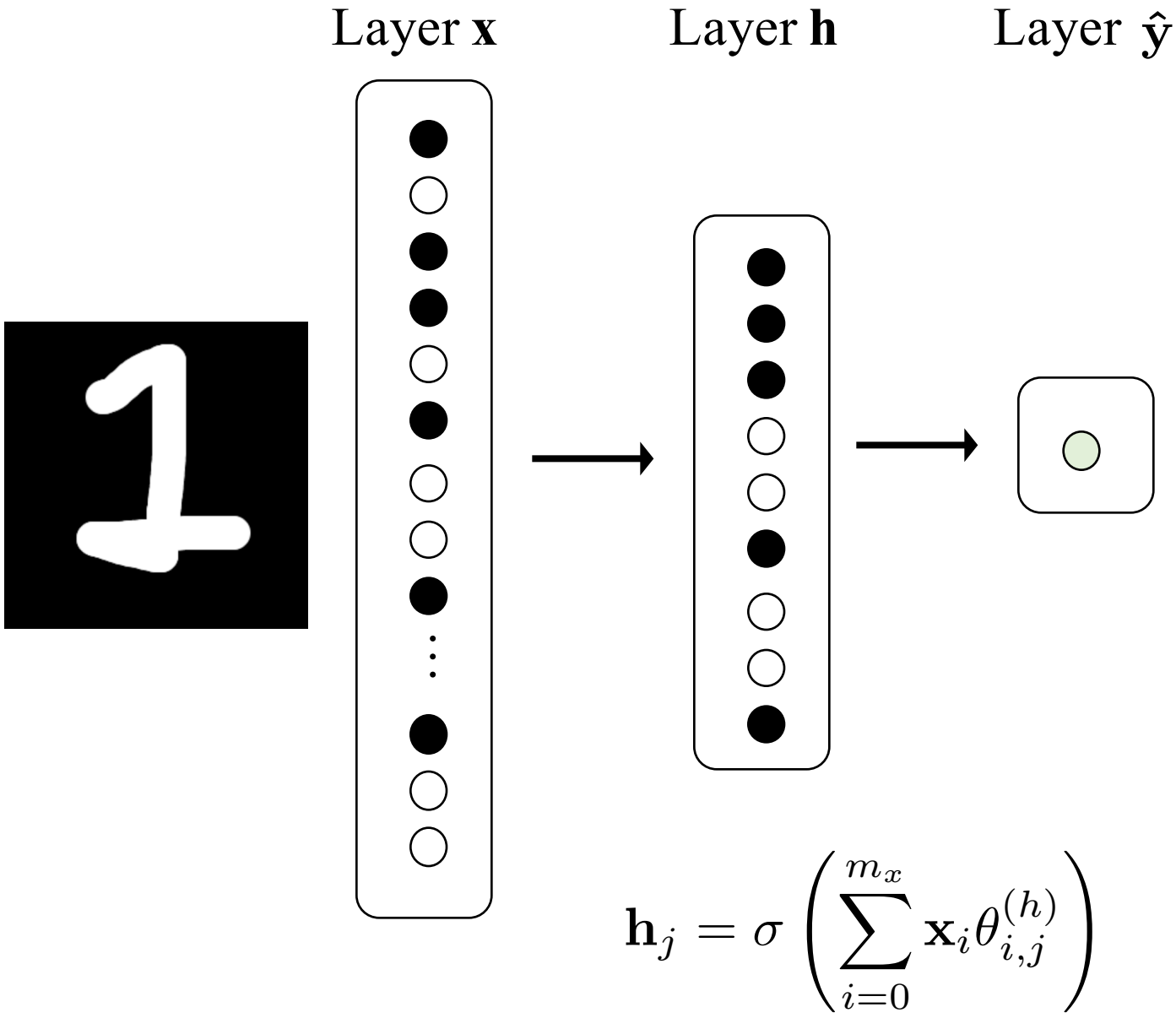
Forward Pass



Forward Pass

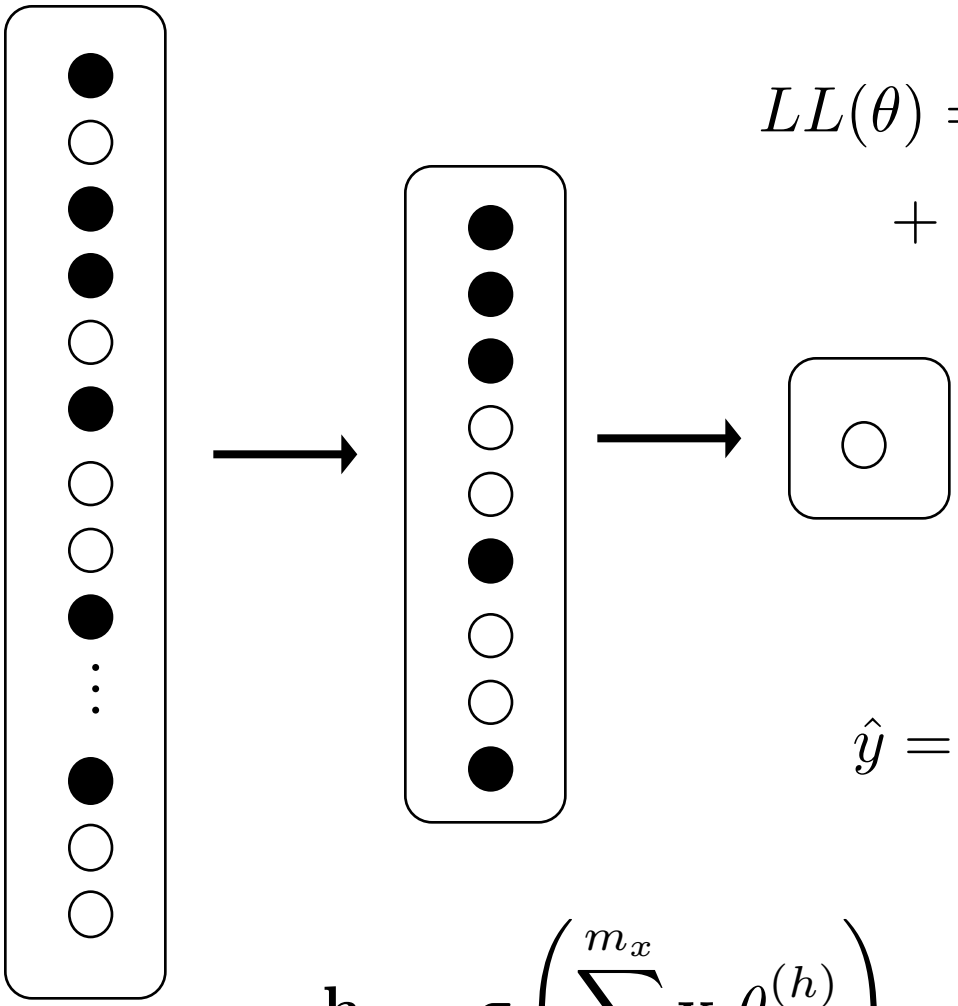


Forward Pass



$$\mathbf{h}_j = \sigma\left(\sum_{i=0}^{m_x} \mathbf{x}_i \theta_{i,j}^{(h)}\right)$$

Forward Pass



$$LL(\theta) = y \log \hat{y} + (1 - y) \log[1 - \hat{y}]$$

$$\hat{y} = \sigma\left(\sum_{j=0}^{m_h} \mathbf{h}_j \theta_j^{(\hat{y})}\right)$$

$$\mathbf{h}_j = \sigma\left(\sum_{i=0}^{m_x} \mathbf{x}_i \theta_{i,j}^{(h)}\right)$$

How do we train?

MLE of Thetas!

Why We Calculate Partial Derivatives

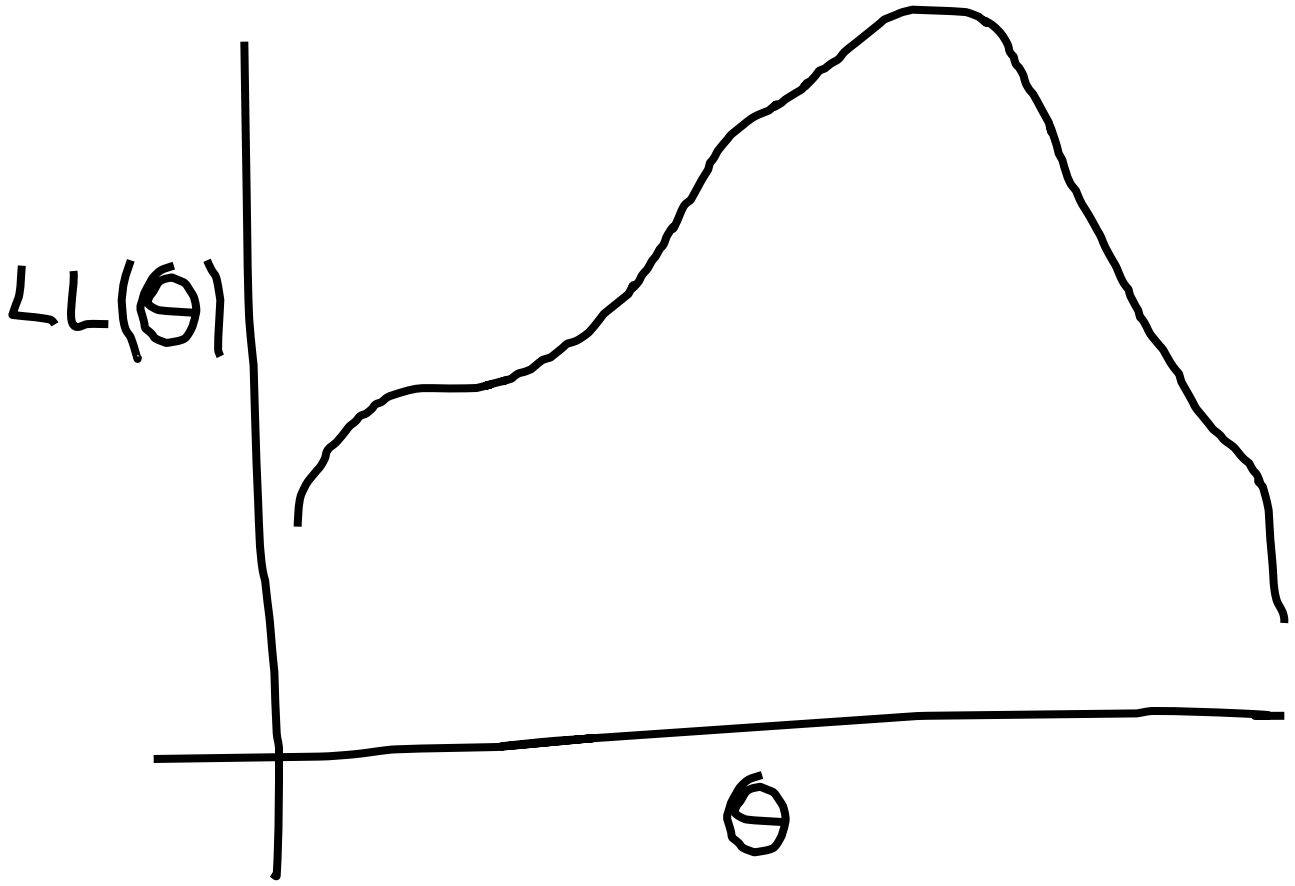
A deep learning model gets its **intelligence** by having **useful thetas**.

We can find **useful thetas**, by searching for ones that **maximize likelihood** of our training data

We can **maximize likelihood** using **optimization techniques** (such as gradient ascent).

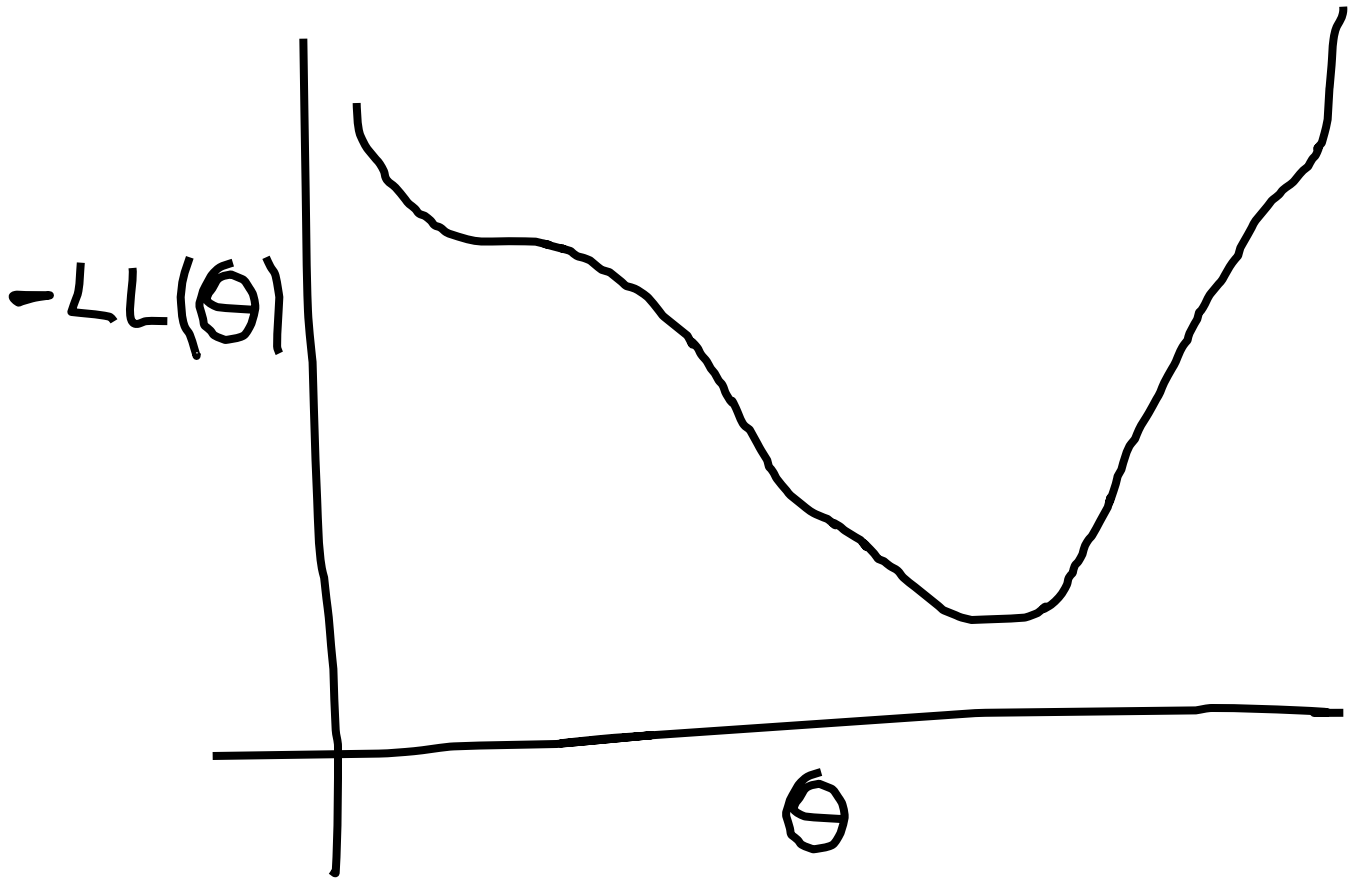
In order to use **optimization techniques**, we need to calculate the **partial derivative** of likelihood with respect to thetas.

Gradient Ascent

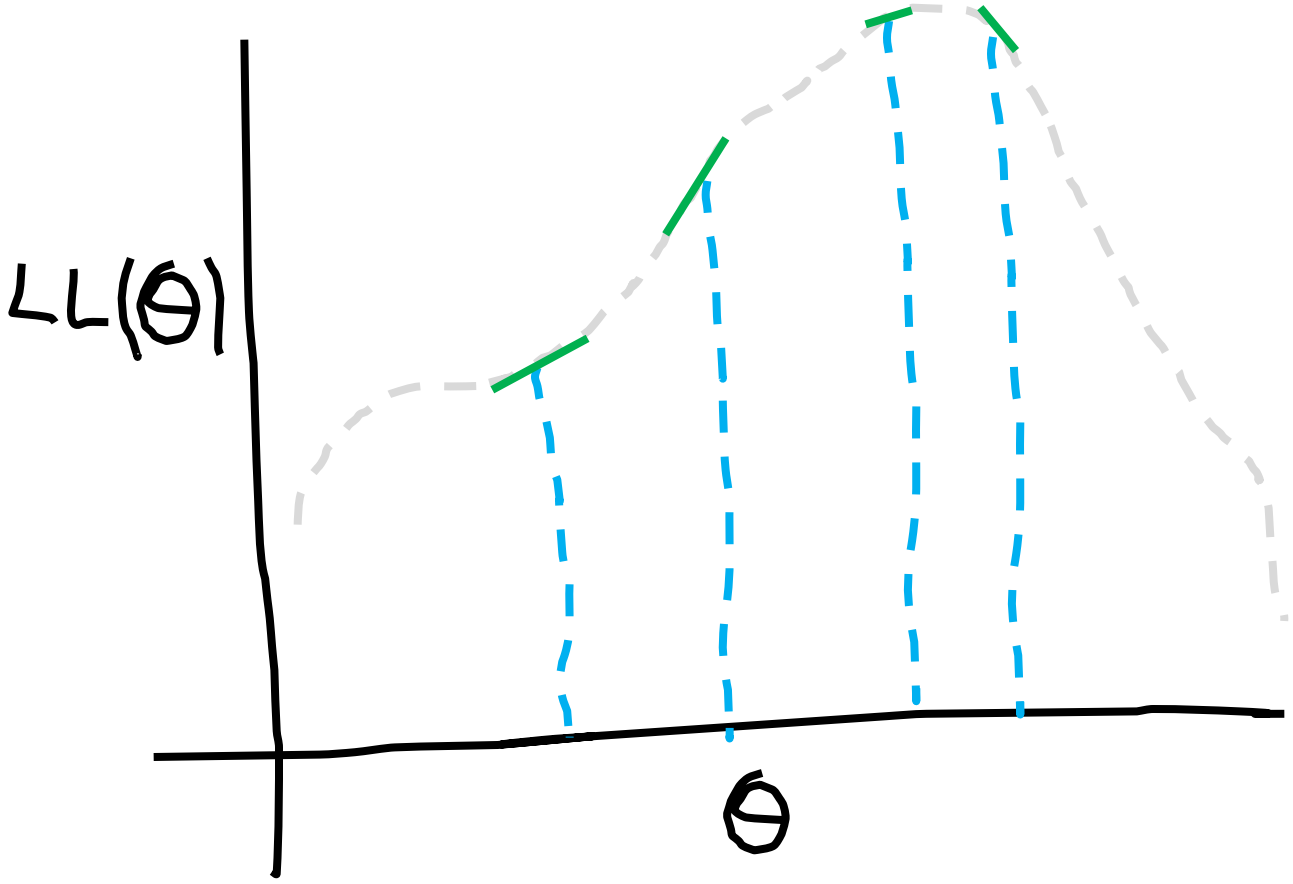


Gradient Ascent

Or is it descent?

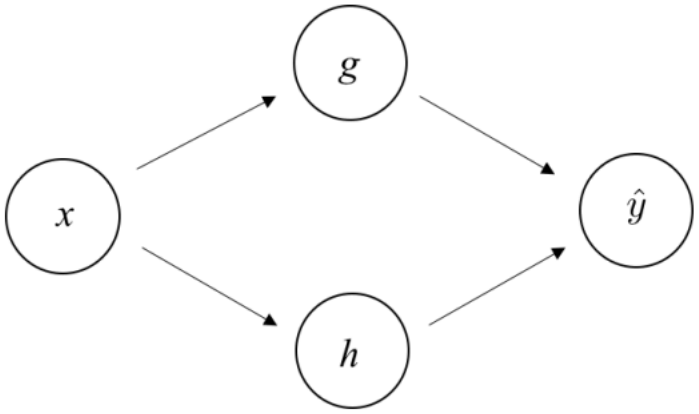


Gradient Ascent



Small step size or large step size?

What If You Had a Neural Network Like This?



$$g = \text{sigmoid}(\theta_1 \cdot x)$$
$$h = \text{sigmoid}(\theta_2 \cdot x)$$
$$\hat{y} = \text{sigmoid}(\theta_3 \cdot g + \theta_4 \cdot h)$$
$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

Deep Learning MLE

3 Get partial derivative of log likelihood with respect to each theta

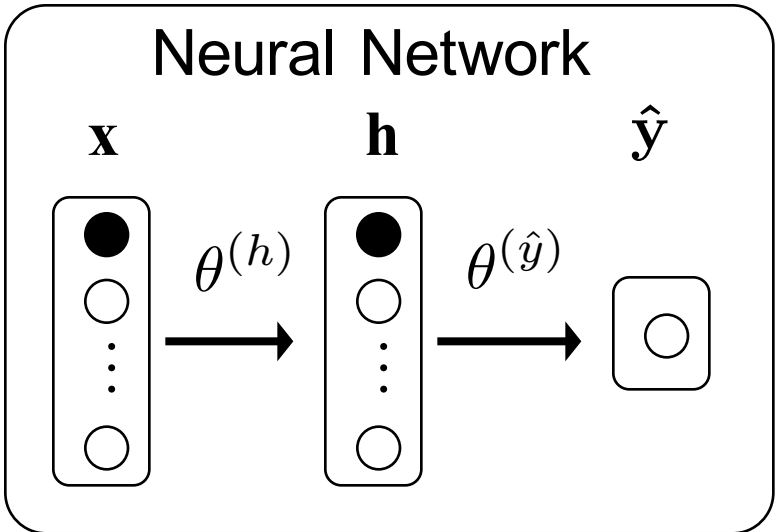
Derivative Goals

Loss with respect to output layer params

$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}}$$

Loss with respect to hidden layer params

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}}$$



Same Assumption, Same LL

$$P(Y = 1|X = \mathbf{x}) = \hat{\mathbf{y}}$$

For one datum

$$P(Y = y|X = \mathbf{X}) = (\hat{y})^y(1 - \hat{y})^{1-y}$$

For IID data

$$L(\theta) = \prod_{i=1}^{n} P(Y = y^{(i)}|X = \mathbf{x}^{(i)})$$

$$= \prod_{i=1}^{n} (\hat{y}^{(i)})^{y^{(i)}} \cdot \left[1 - (\hat{y}^{(i)})\right]^{(1-y^{(i)})}$$

Take the log

$$LL(\theta) = \sum_{i=0}^{n} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log[1 - \hat{y}^{(i)}]$$

Big Idea

Big Idea

It's called chain rule

$$\frac{\partial f(x)}{\partial x} = \frac{\partial f(z)}{\partial z} \cdot \frac{\partial z}{\partial x}$$

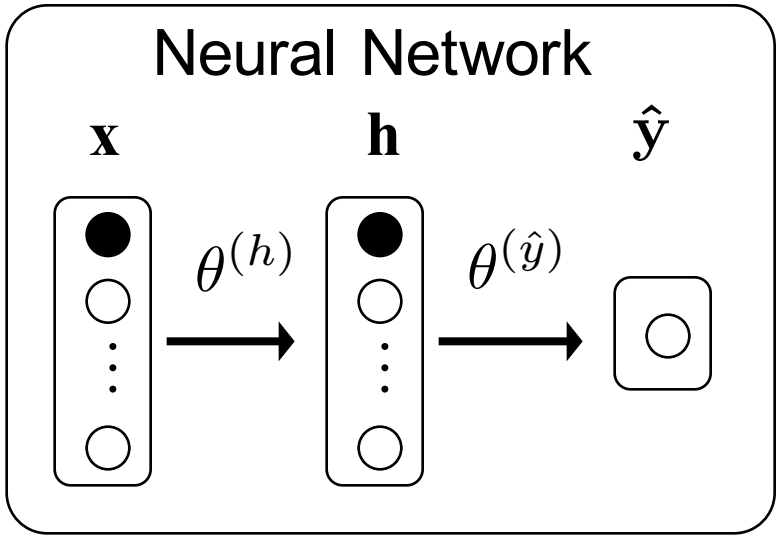
First use:

$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \theta_i^{(\hat{y})}}$$

Chain Rule Example 1

$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}}$$

Goal



Network

$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \theta_i^{(\hat{y})}}$$

Decomposition

Think About Only One Training Instance

$$LL(\theta) = \sum_{i=0}^{n} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log[1 - \hat{y}^{(i)}]$$

We only need to calculate the gradient for one training example!

$$\frac{\partial}{\partial x} \sum f(x) = \sum \frac{\partial}{\partial x} f(x)$$

We will pretend we only have one example

$$LL(\theta) = y \log \hat{y} + (1 - y) \log[1 - \hat{y}]$$

We can sum up the gradients of each example to get the correct answer

Make it Simple

$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}} = \text{  } \cdot \text{  }$$

$$\text{  } = \frac{y}{\hat{y}} - \frac{(1-y)}{(1-\hat{y})}$$

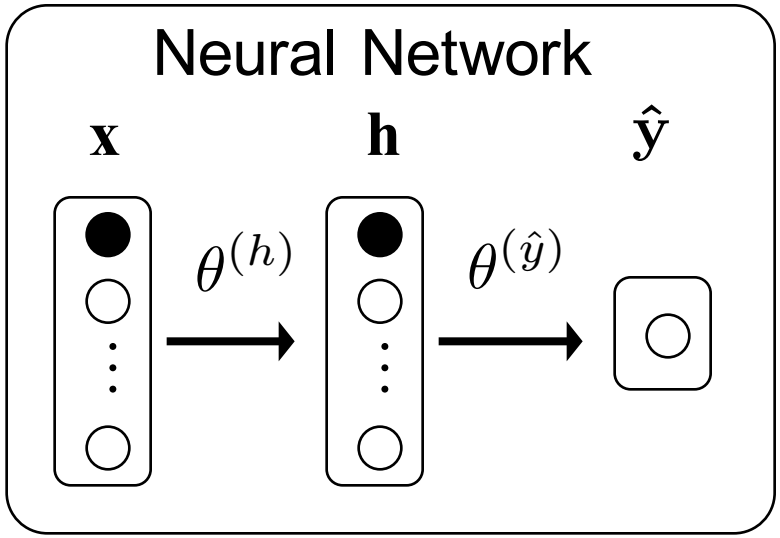
$$\text{  } = \hat{y}[1-\hat{y}] \cdot h_i$$

Where we left off

Chain Rule Example 2

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}}$$

Goal



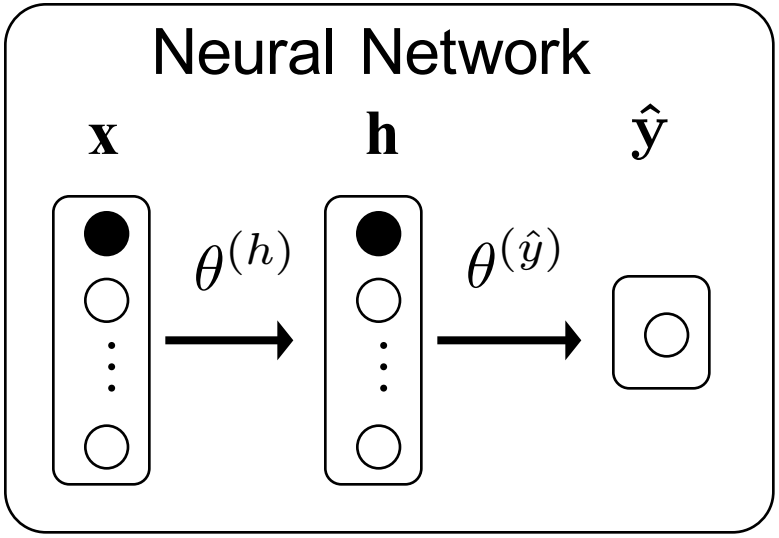
Network

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mathbf{h}_j} \cdot \frac{\partial \mathbf{h}_j}{\partial \theta_{i,j}^{(h)}}$$

Decomposition

Gradient of hidden layer params

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mathbf{h}_j} \cdot \frac{\partial \mathbf{h}_j}{\partial \theta_{i,j}^{(h)}}$$



Gradient of hidden layer params

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mathbf{h}_j} \cdot \frac{\partial \mathbf{h}_j}{\partial \theta_{i,j}^{(h)}}$$

$$\hat{y} = \sigma\left(\sum_{i=0}^{m_h} \mathbf{h}_i \theta_i^{(\hat{y})}\right)$$

Based on derivative of sigmoid

$$\frac{\partial \hat{y}}{\partial \mathbf{h}_j} = \hat{y}[1-\hat{y}]\theta_j^{(\hat{y})}$$

For all other values of i, the term in the sum is independent of h_j

Wait is it over?

Gradient of hidden layer params

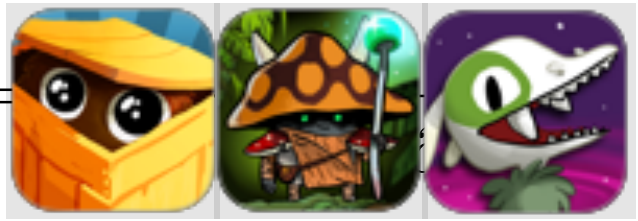
$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}} = \frac{\partial LL}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mathbf{h}_j} \cdot \frac{\partial \mathbf{h}_j}{\partial \theta_{i,j}^{(h)}}$$

$$\mathbf{h}_j = \sigma\left(\sum_{k=0}^{m_x} \mathbf{x}_k \theta_{k,j}\right)$$

$$\frac{\partial \mathbf{h}_j}{\partial \theta_{i,j}^{(h)}} = \mathbf{h}_j[1 - \mathbf{h}_j]\mathbf{x}_j$$

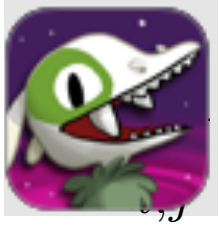
That one too?

Make it Simple

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}} =  $$

$$= \frac{y}{\hat{y}} - \frac{(1-y)}{(1-\hat{y})}$$

$$= \hat{y}[1-\hat{y}]\theta_j^{(\hat{y})}$$



$$= \mathbf{h}_j[1-\mathbf{h}_j]\mathbf{x}_j$$

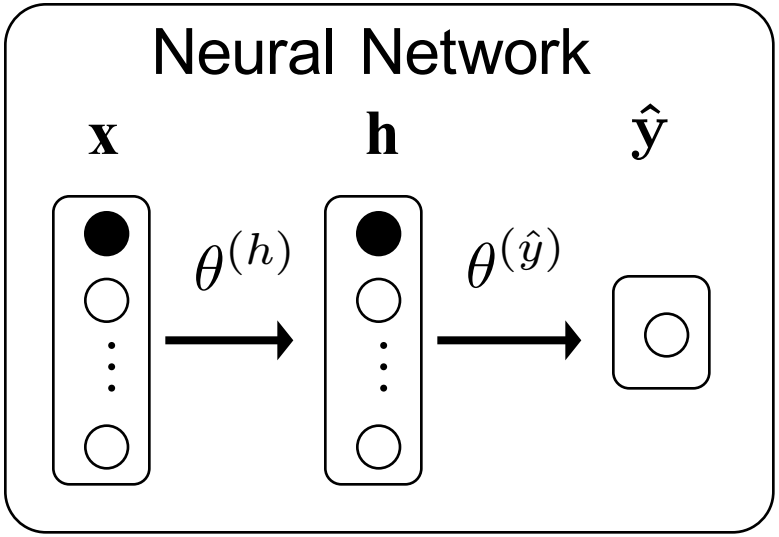
Summary: Simple Calculations For

Loss with respect to output layer params

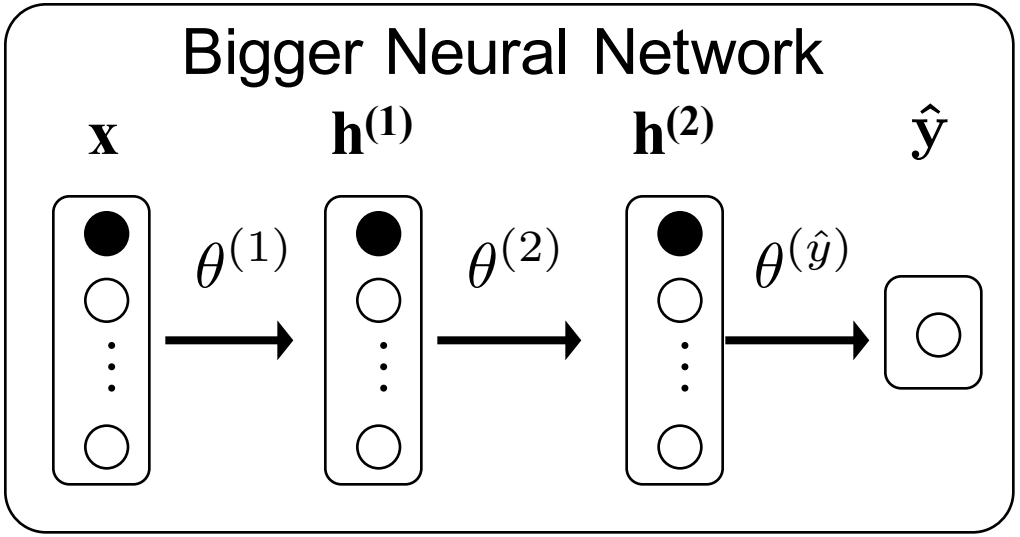
$$\frac{\partial LL(\theta)}{\partial \theta_i^{(\hat{y})}}$$

Loss with respect to hidden layer params

$$\frac{\partial LL(\theta)}{\partial \theta_{i,j}^{(h)}}$$



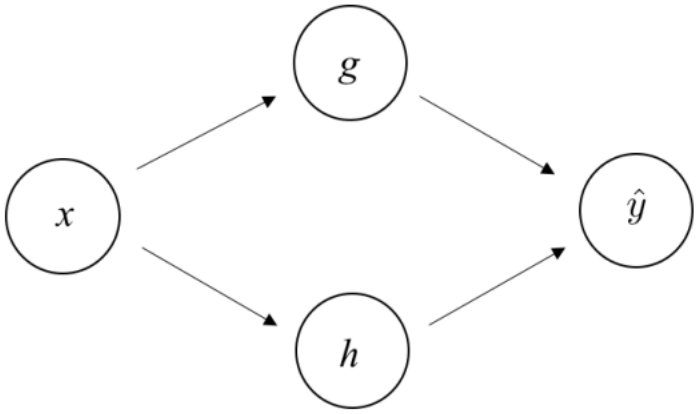
What Would You Do Here?



Chain rule:
Game changer for
artificial intelligence

Congrats. You now know backpropagation

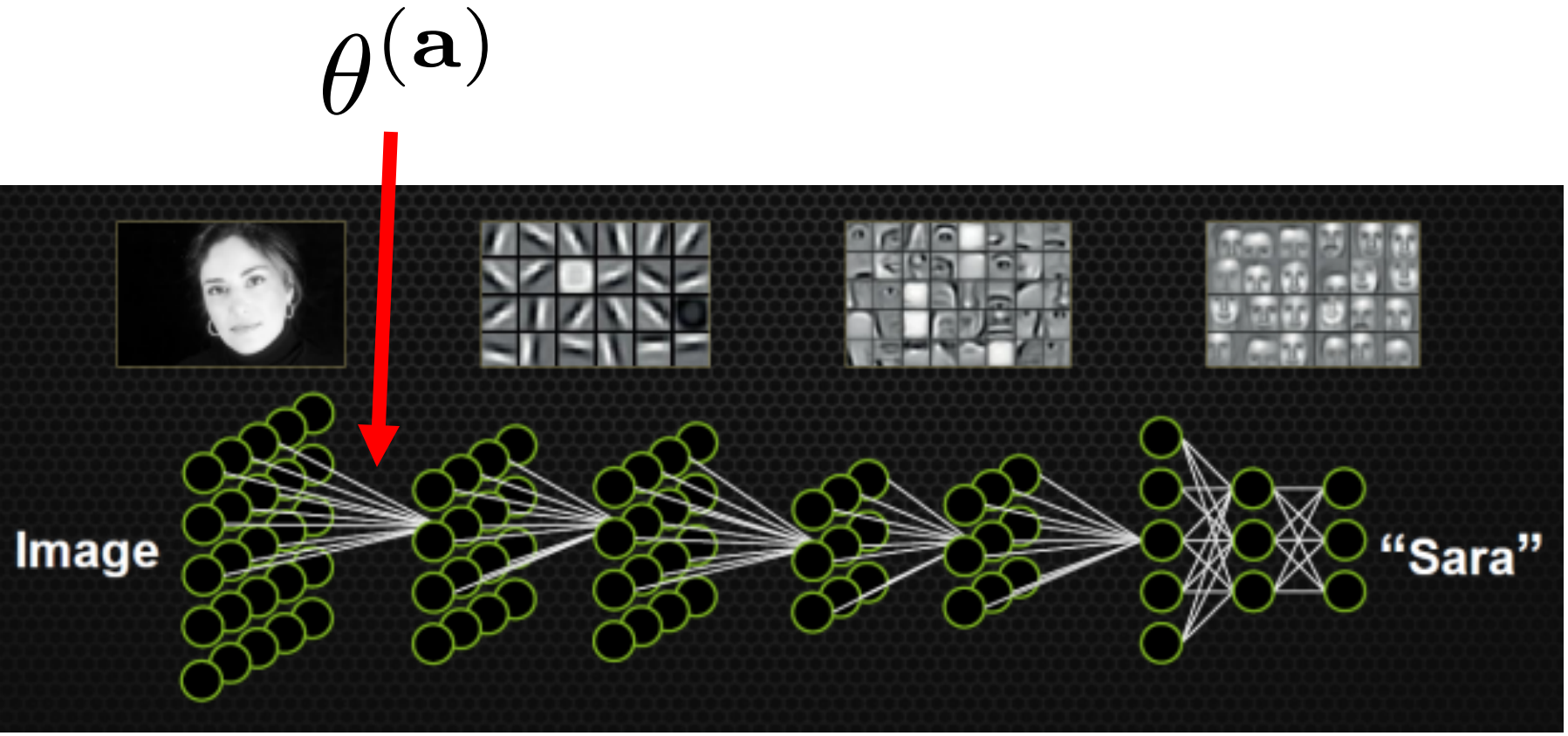
What Would You Do Here?



$$g = \text{sigmoid}(\theta_1 \cdot x)$$
$$h = \text{sigmoid}(\theta_2 \cdot x)$$
$$\hat{y} = \text{sigmoid}(\theta_3 \cdot g + \theta_4 \cdot h)$$
$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

1. Calculate partial derivative for one data instance
2. Use chain rule!
3. Sigmoid derivatives come out simple if you use the right notation
4. You don't need to give the most reduced answer

Works for any number of layers

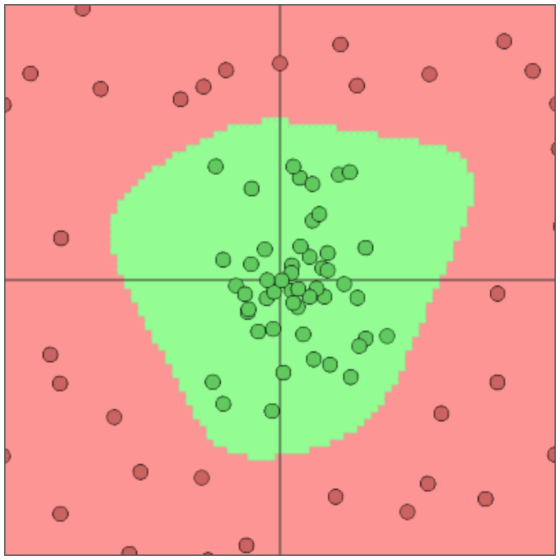


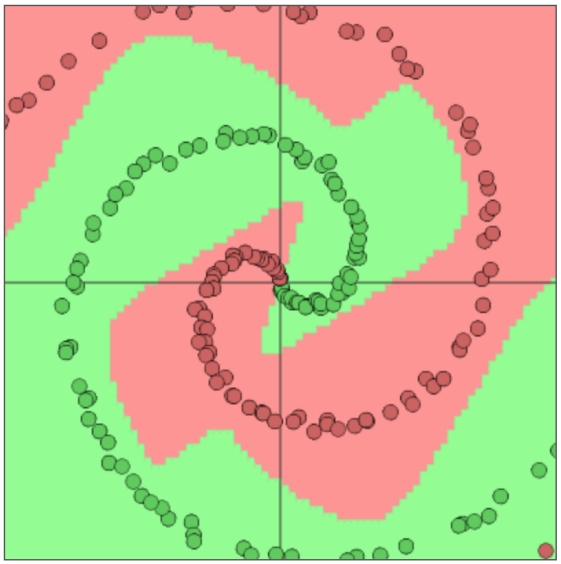
x a b c d e f g $\hat{y}$ LL

model.update(data)

Neural Networks Can Learn Complex Functions

- Some data sets/functions are not separable





  - These are classifiers learned by neural networks

http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Multiple Output Classification?

Softmax is a generalization of the sigmoid function that squashes a K -dimensional vector $\mathbf{z}$ of arbitrary real values to a K-dimensional vector $\mathrm{softmax}(\mathbf{z})$ of real values in the range [0, 1] that add up to 1.

$$P(Y = j|\mathbf{X} = \mathbf{x}) = \mathrm{softmax}(f(\mathbf{x}))_j$$

$$\mathrm{softmax}(\mathbf{z})_i = \frac{e_i^z}{\sum_j e^{z_j}}$$

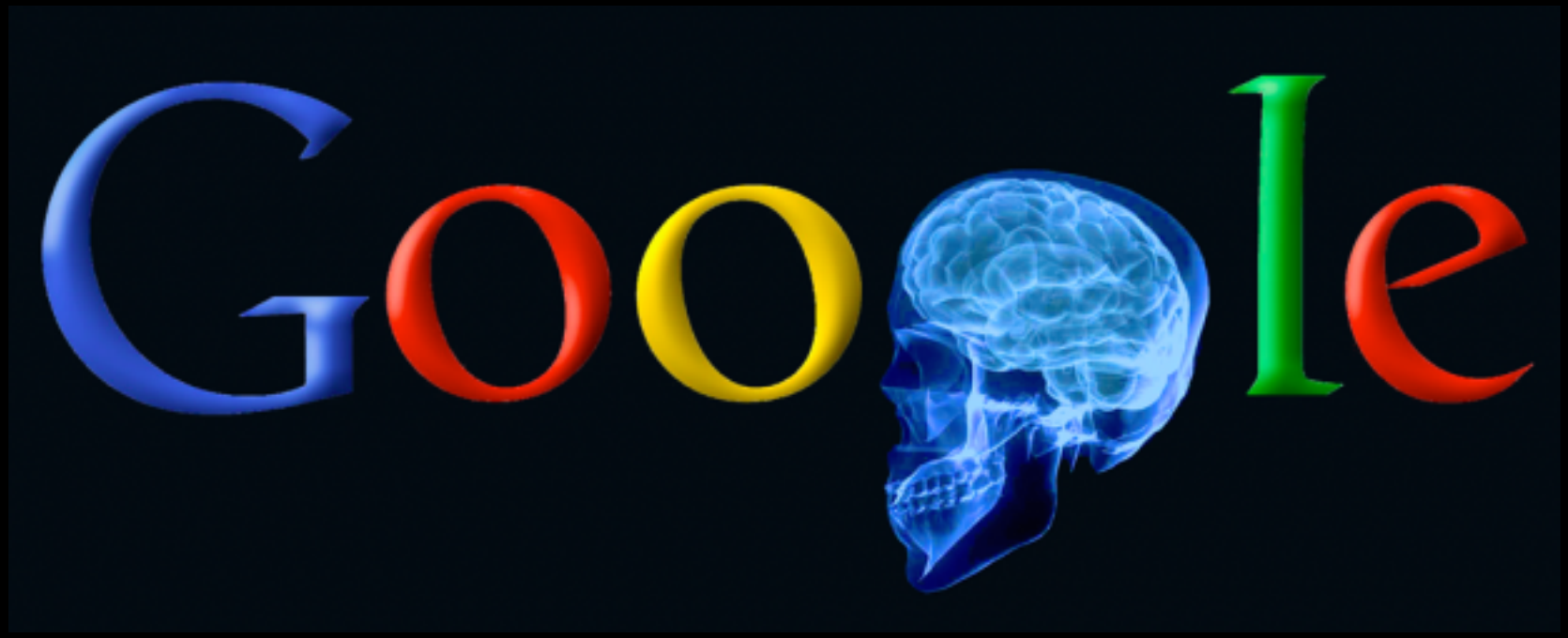
ImageNet: 22,000 Types

…
smoothhound, smoothhound shark, Mustelus mustelus
American smooth dogfish, Mustelus canis
Florida smoothhound, Mustelus norrisi
whitetip shark, reef whitetip shark, Triaenodon obseus
Atlantic spiny dogfish, Squalus acanthias
Pacific spiny dogfish, Squalus suckleyi
hammerhead, hammerhead shark
smooth hammerhead, Sphyrna zygaena
smalleye hammerhead, Sphyrna tudes
shovelhead, bonnethead, bonnet shark, Sphyrna tiburo
angel shark, angelfish, Squatina squatina, monkfish
electric ray, crampfish, numbfish, torpedo
smalltooth sawfish, Pristis pectinatus
guitarfish
roughtail stingray, Dasyatis centroura
butterfly ray
eagle ray
spotted eagle ray, spotted ray, Aetobatus narinari
cownose ray, cow-nosed ray, Rhinoptera bonasus
manta, manta ray, devilfish
Atlantic manta, Manta birostris
devil ray, Mobula hypostoma
grey skate, gray skate, Raja batis
little skate, Raja erinacea
…

Stingray

Mantaray

GoogLeNet



1 Trillion Artificial Neurons

0.005%

Random guess

1.5%

Pre Neural Networks

?

GoogLeNet

Le, et al., *Building high-level features using large-scale unsupervised*

0.005%

Random guess

1.5%

Pre Neural Networks

43.9%

GoogLeNet

Szegedy et al, Going Deeper With Convolutions, CVPR 2015

0.005%

Random guess

1.5%

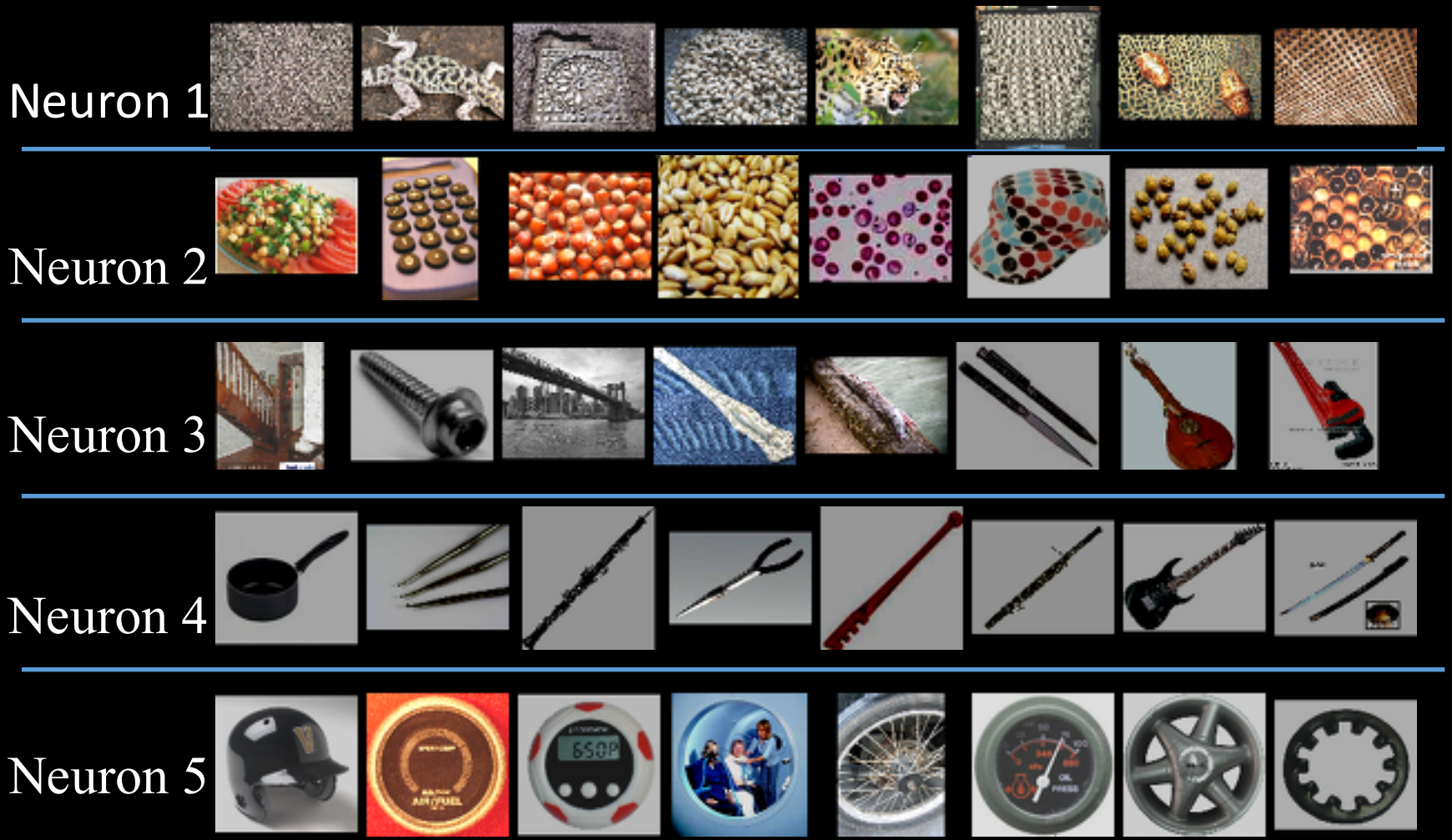
Pre Neural Networks

66.3%

2016

http://image-net.org/challenges/LSVRC/2016/results

Best Neuron Stimuli



Le, et al., *Building high-level features using large-scale unsupervised*

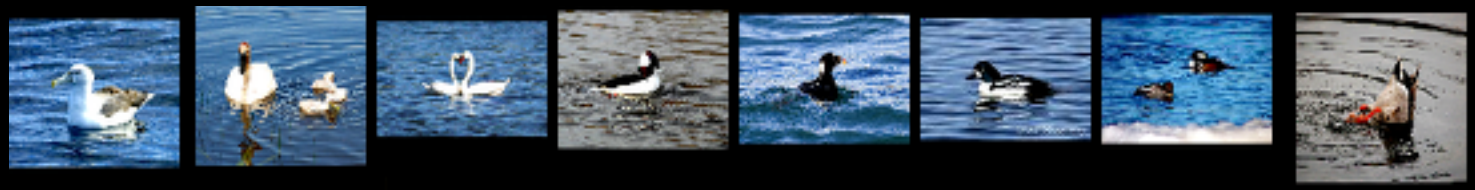
Best Neuron Stimuli

Neuron 6

Neuron 7

Neuron 8

Neuron 9



Le, et al., *Building high-level features using large-scale unsupervised*

Best Neuron Stimuli

Neuron 10

Neuron 11

Neuron 12

Neuron 13

Le, et al., *Building high-level features using large-scale unsupervised*

The Cat Neuron

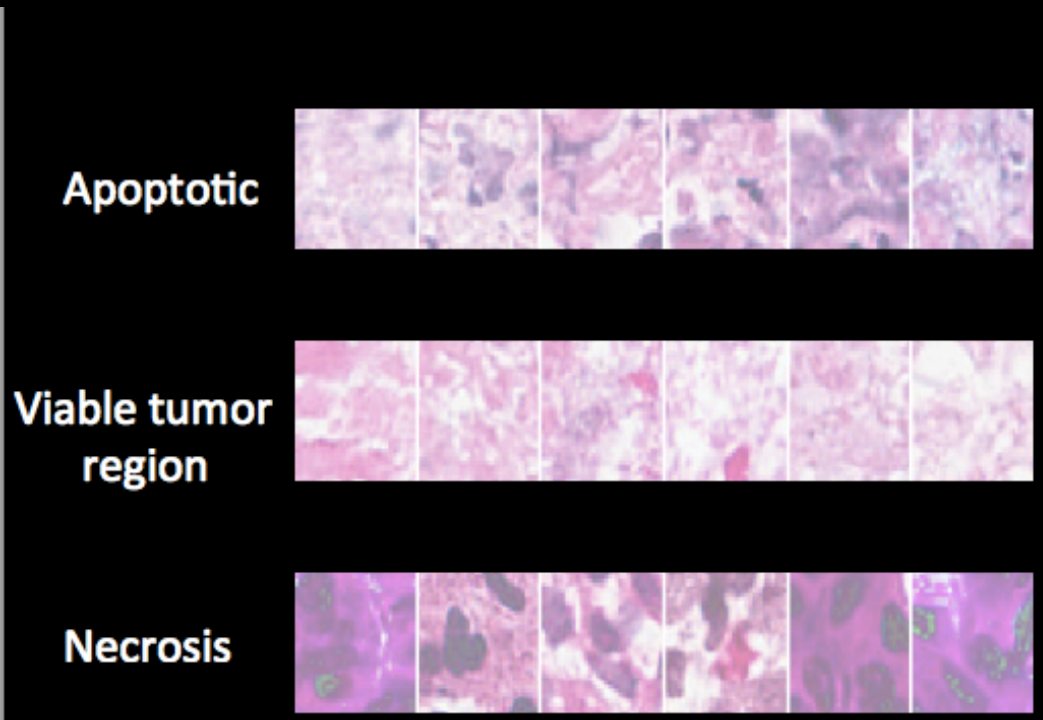
Top stimuli from the test set

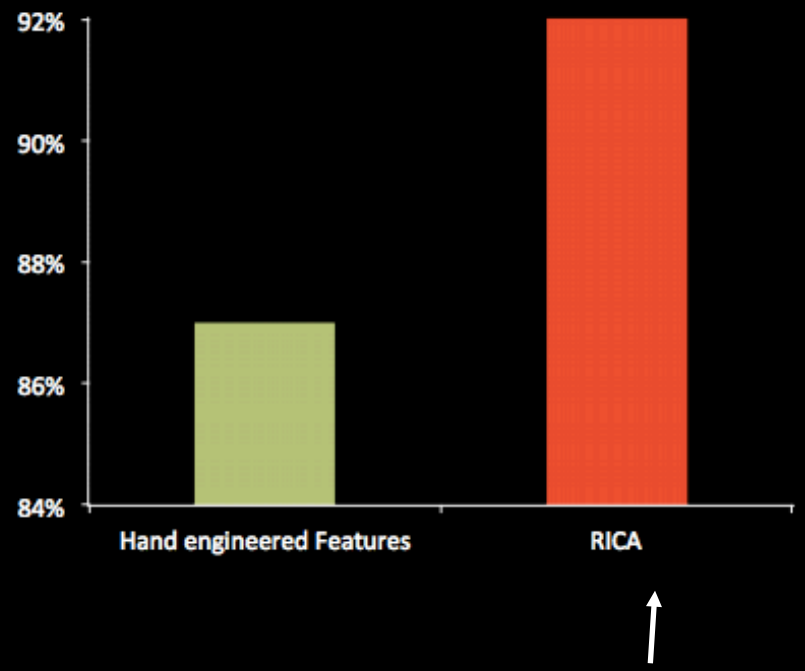
Optimal stimulus
by numerical optimization

Le, et al., *Building high-level features using large-scale unsupervised*


Hire the smartest people in the world
Invent cat detector

Vision Has Social Implications

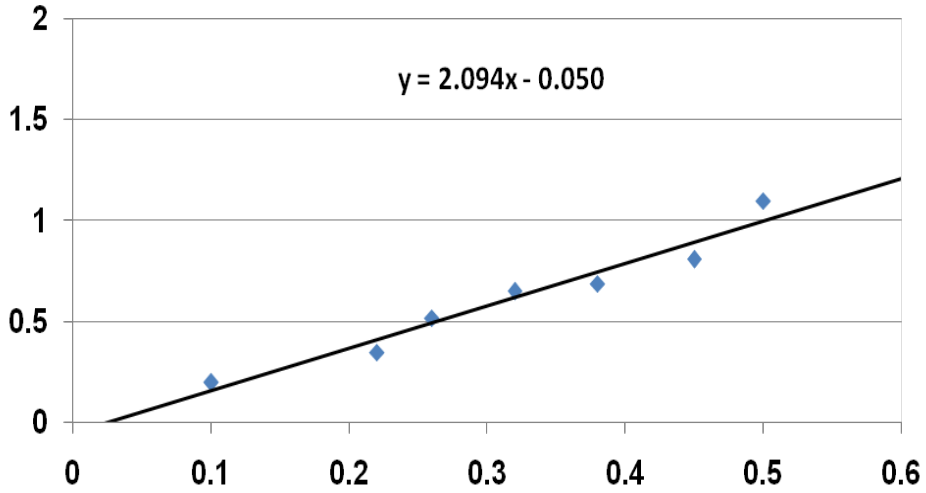


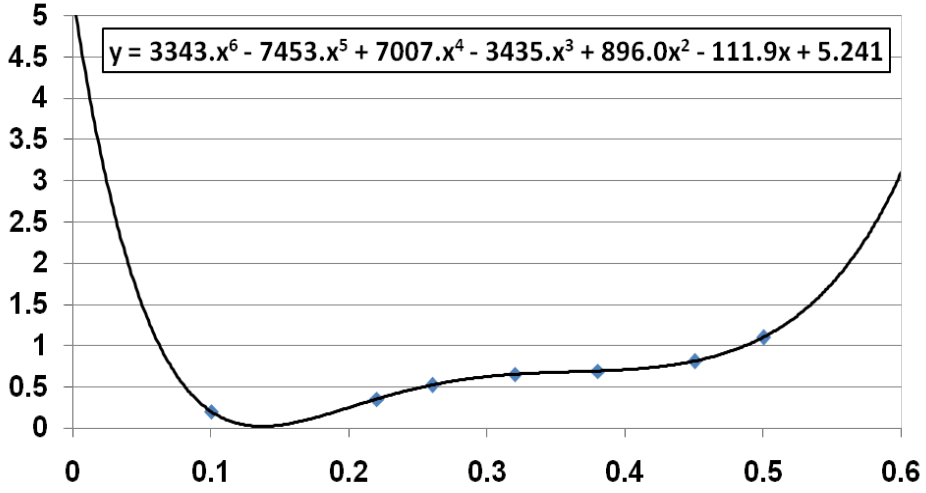


Neural network

Good ML = Generalization

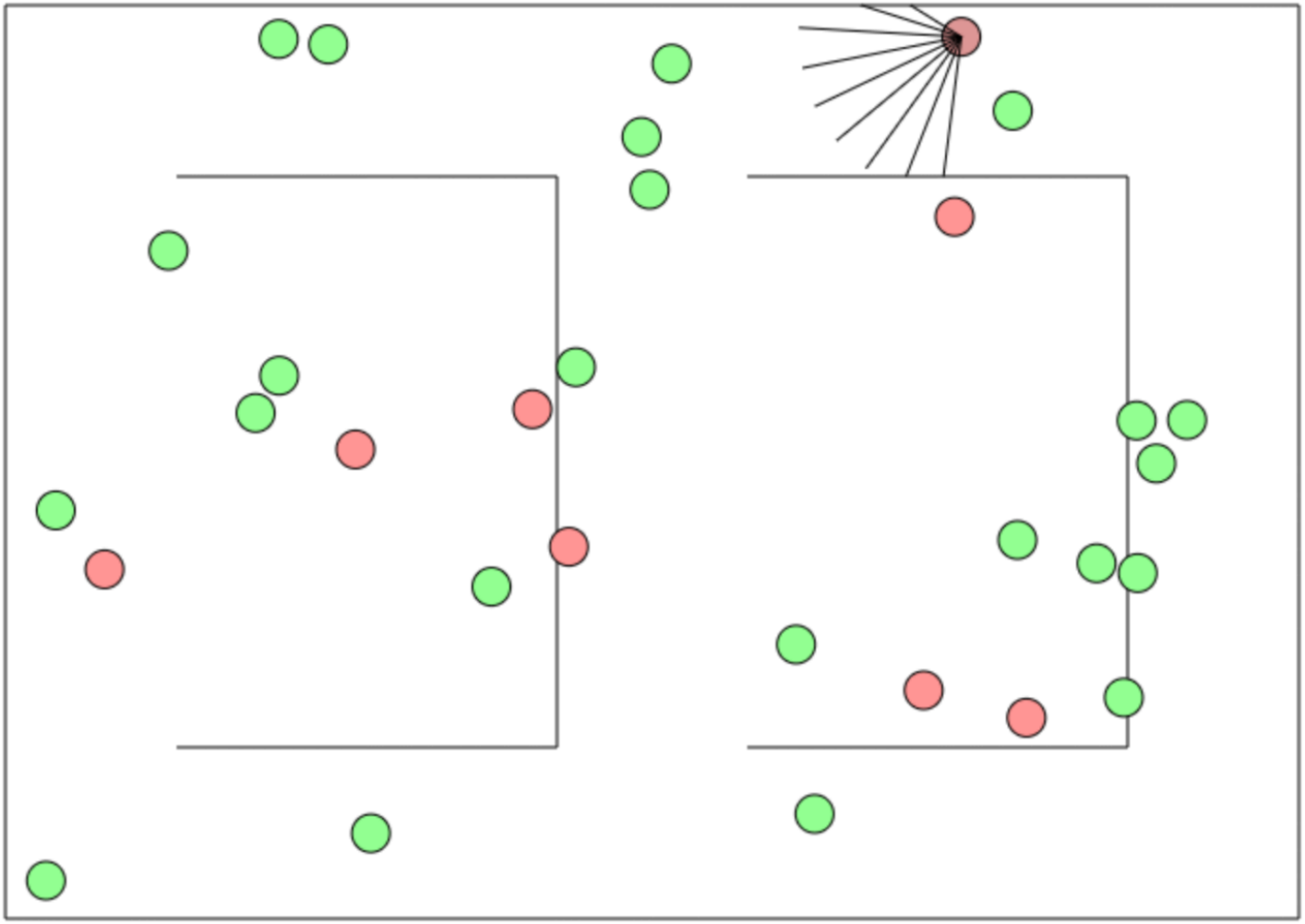
- Goal of machine learning: build models that ***generalize*** well to predicting new data
  - "Overfitting": fitting the training data too well, so we lose generality of model





  - Polynomial on the right fits training data perfectly!
  - Which would you rather use to predict a new data point?

Deep Reinforcement Learning

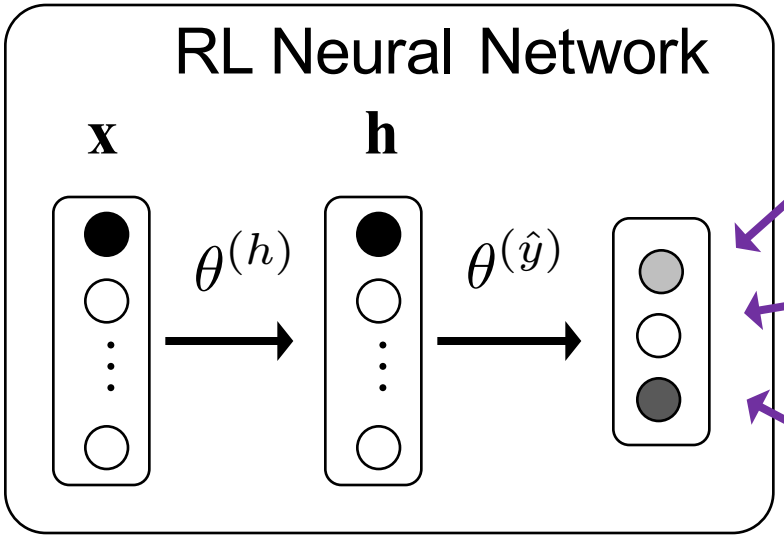


http://cs.stanford.edu/people/karpathy/convnetjs/demo/rldemo.html

Deep Reinforcement Learning

R is a reward and A_i is a legal action

Input is a representation of current state (S)



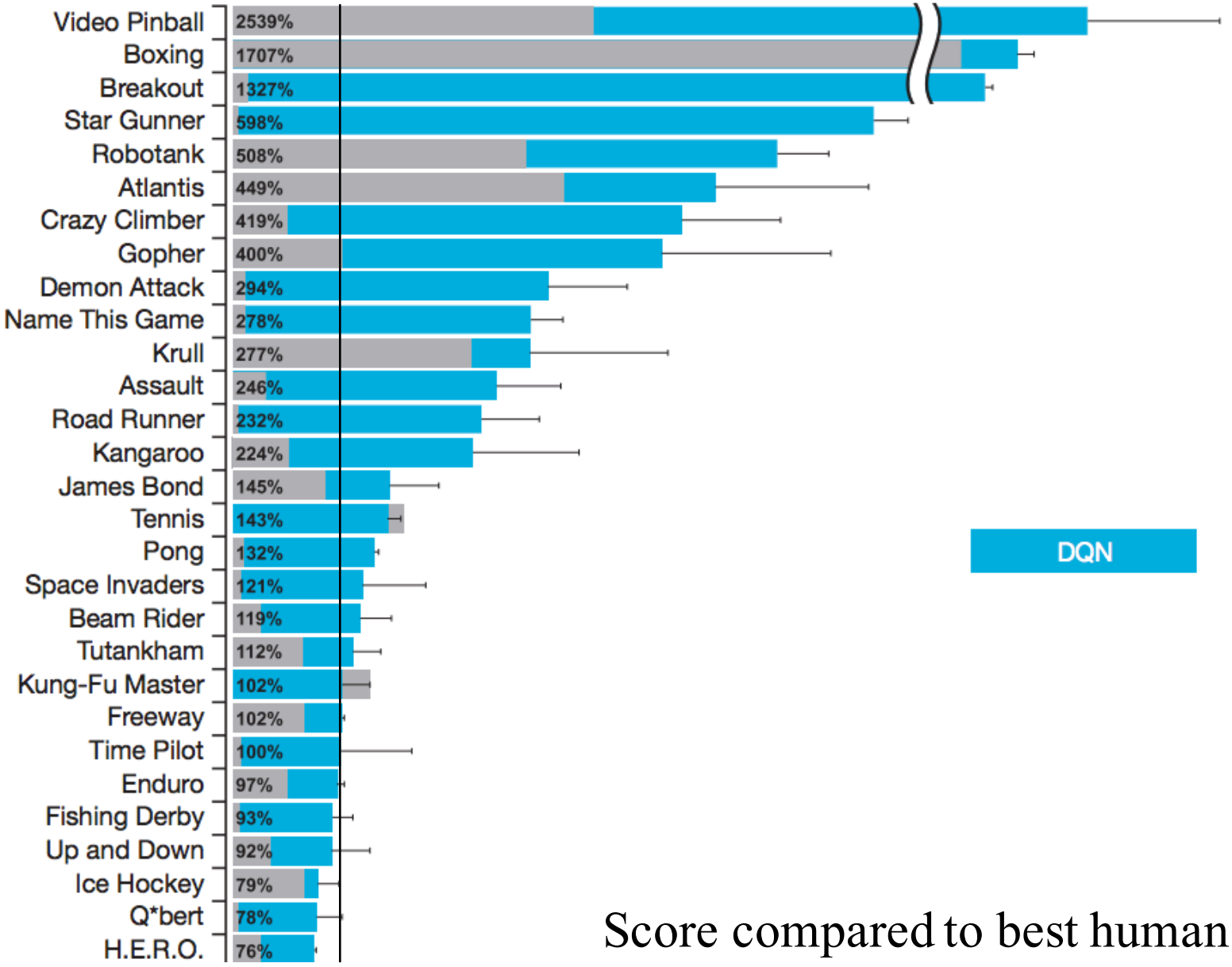
E[R | A_1, S]

E[R | A_2, S]

E[R | A_3, S]

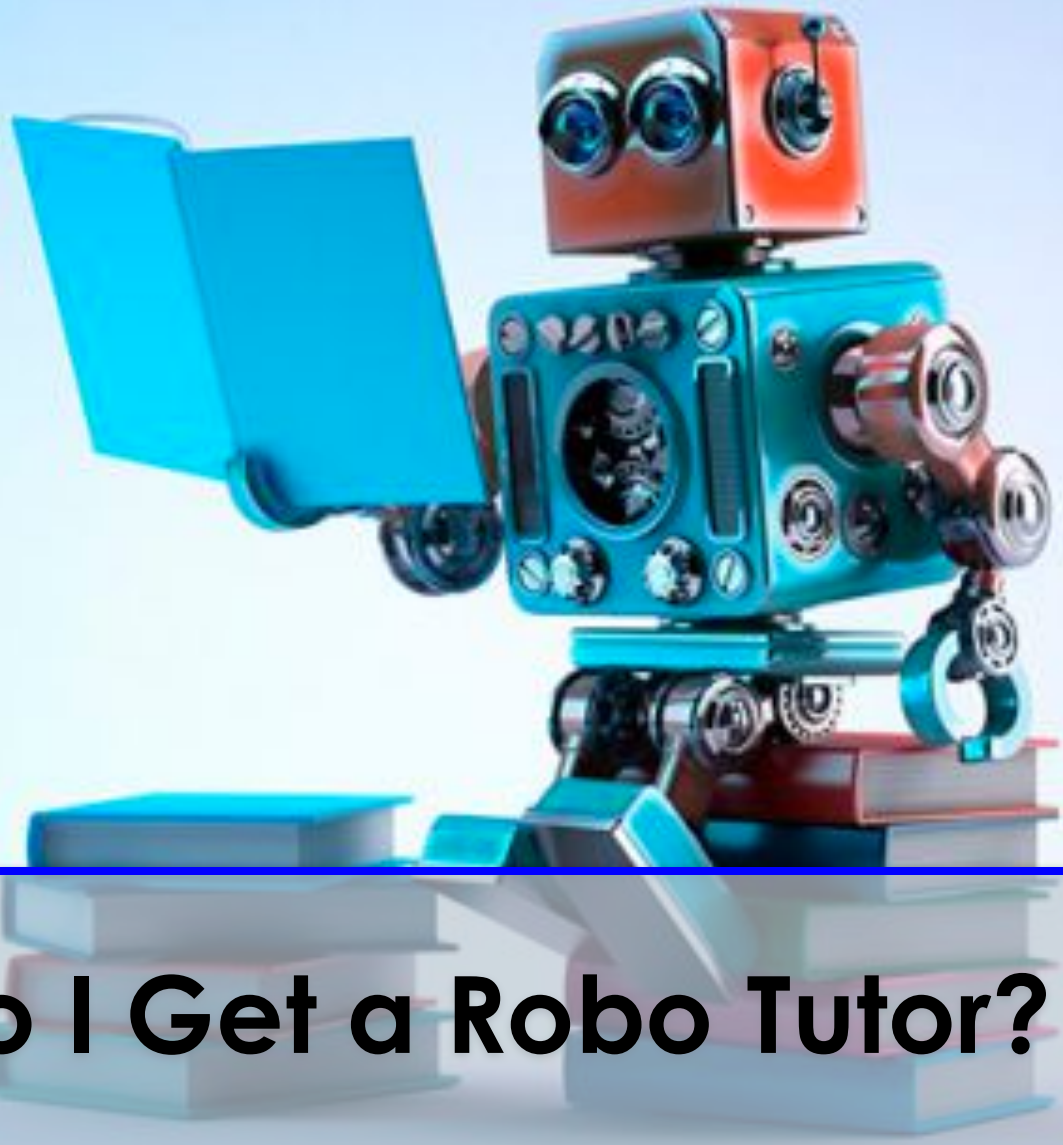
Interpret outputs as expected reward for a given action

Deep Mind Atari Games



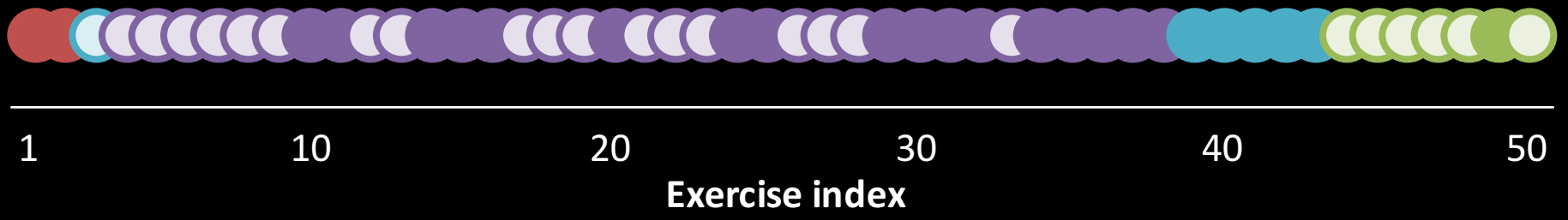
Score compared to best human

Alpha GO



When Do I Get a Robo Tutor?

Story of Riley



Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley

Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley

Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley

Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley

Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley

Exercise Type:

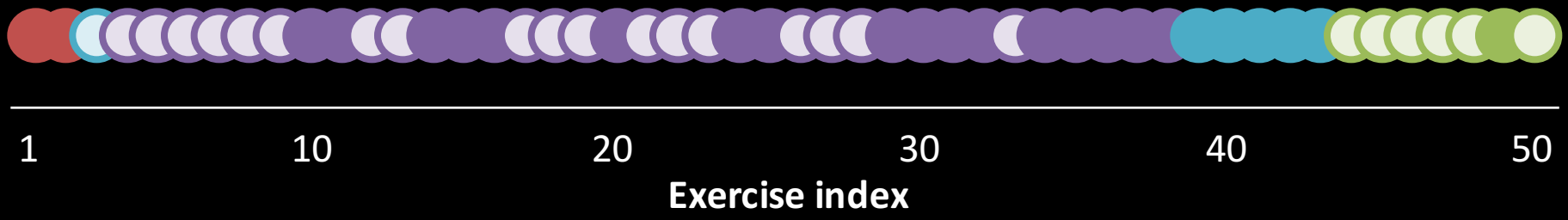
- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley



What does Riley know?

Exercise Type:

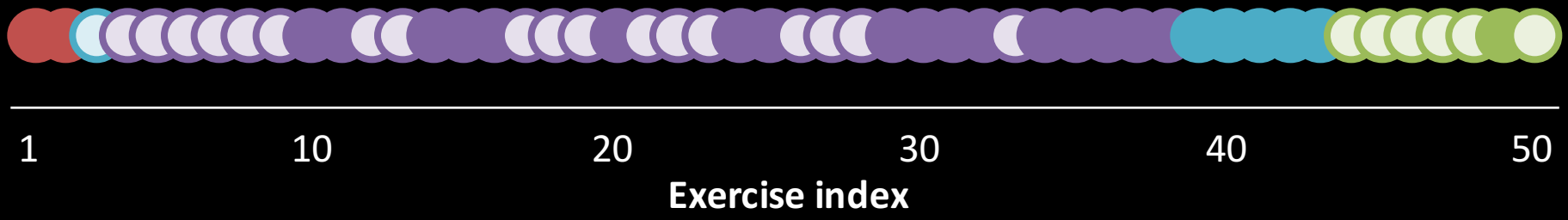
- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley



What should Riley do next?

Exercise Type:

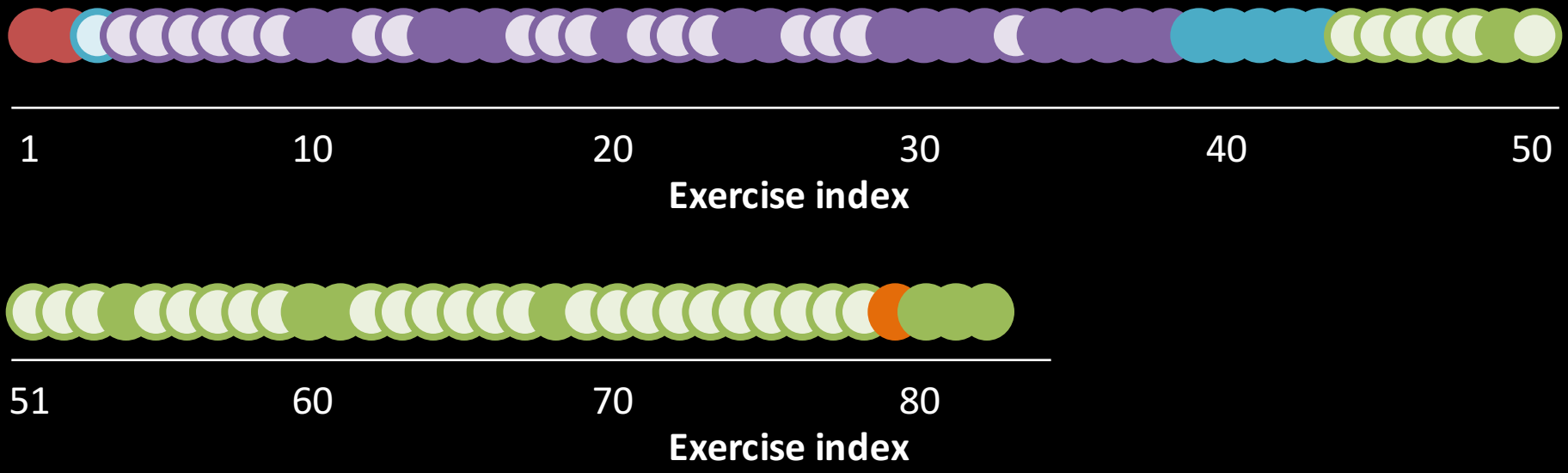
- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

KHAN ACADEMY

Story of Riley



Exercise Type:

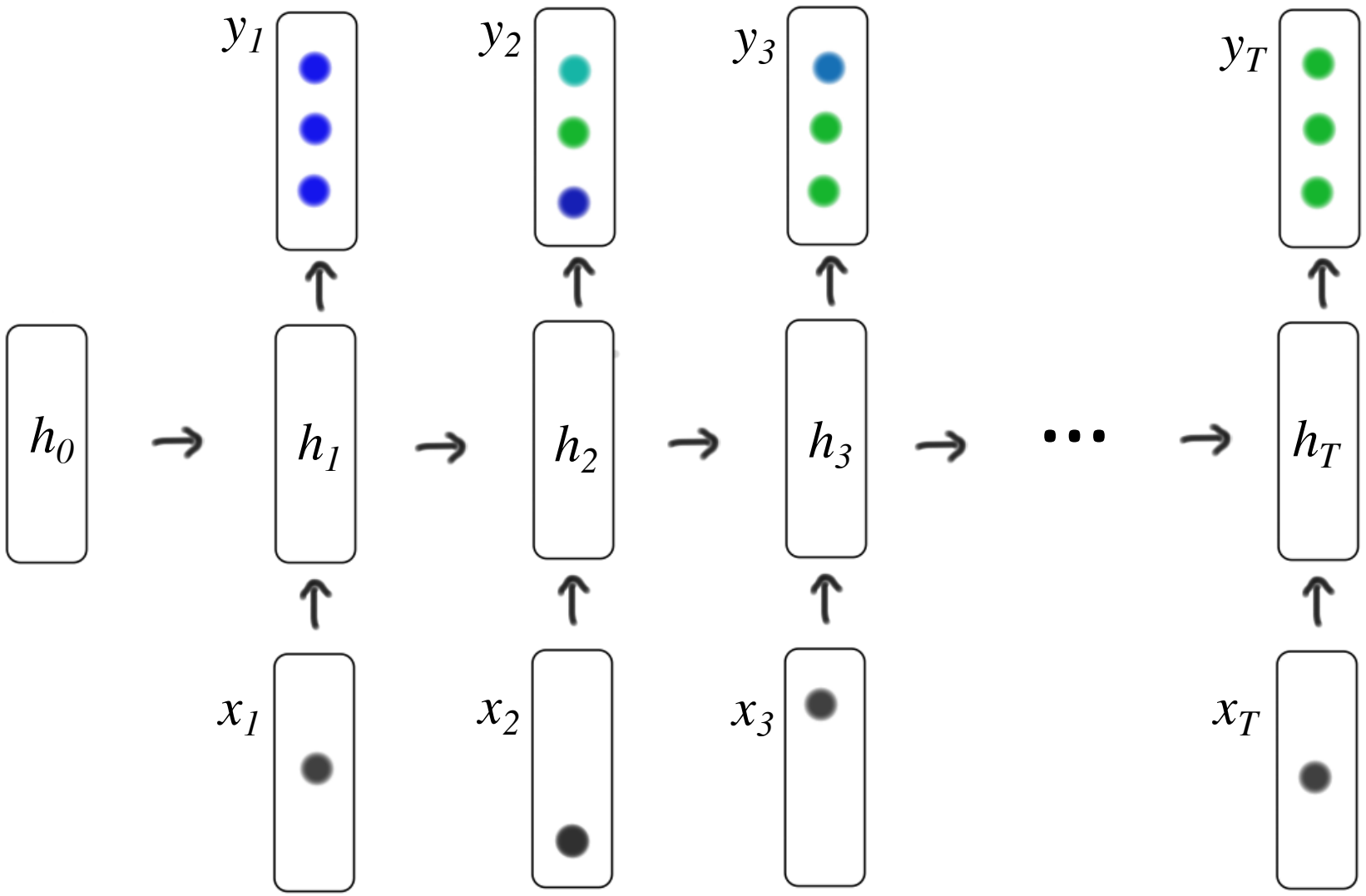
- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

Answer:

- Correct
- Incorrect

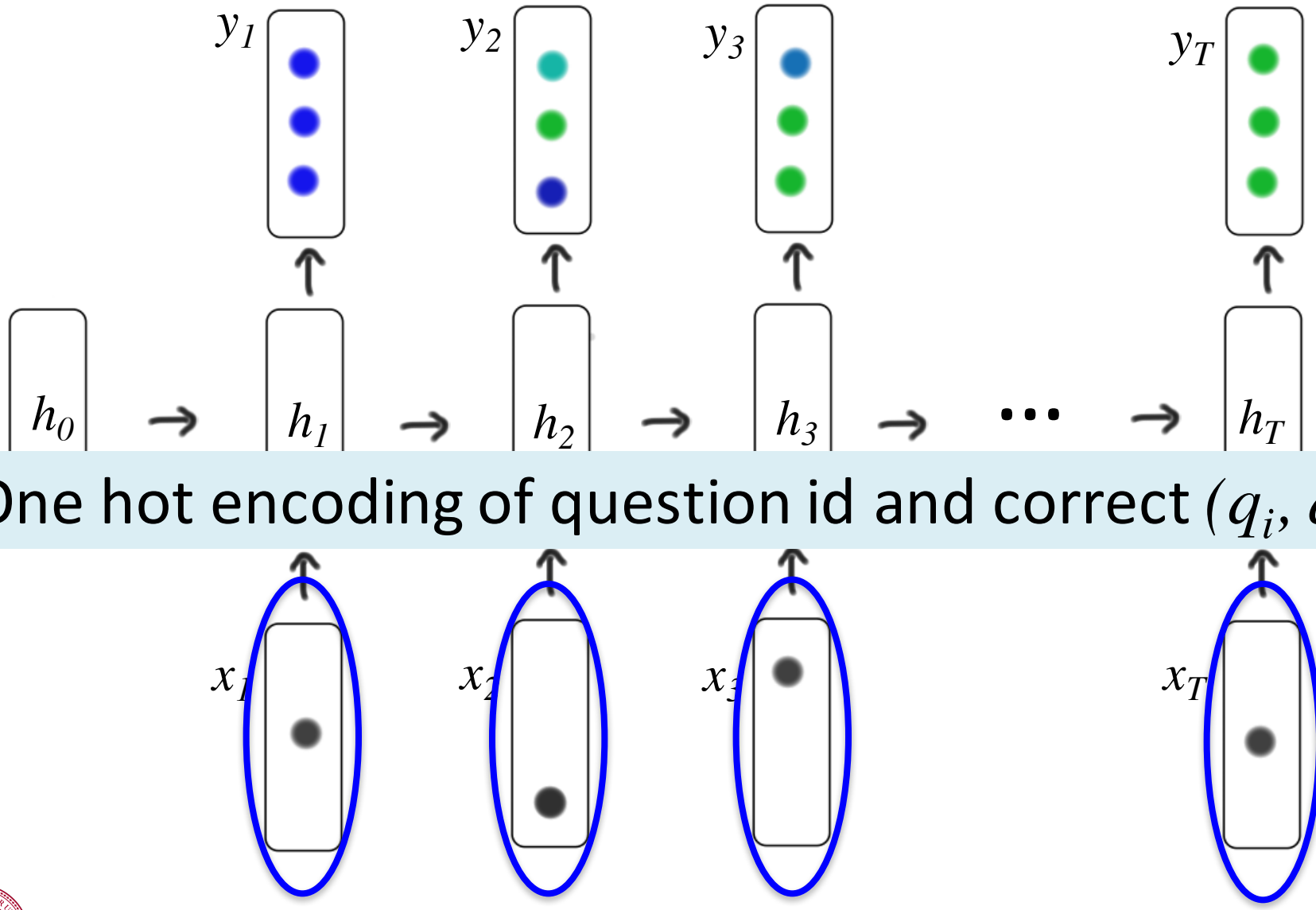
KHAN ACADEMY

Recurrent Neural Network



Piech

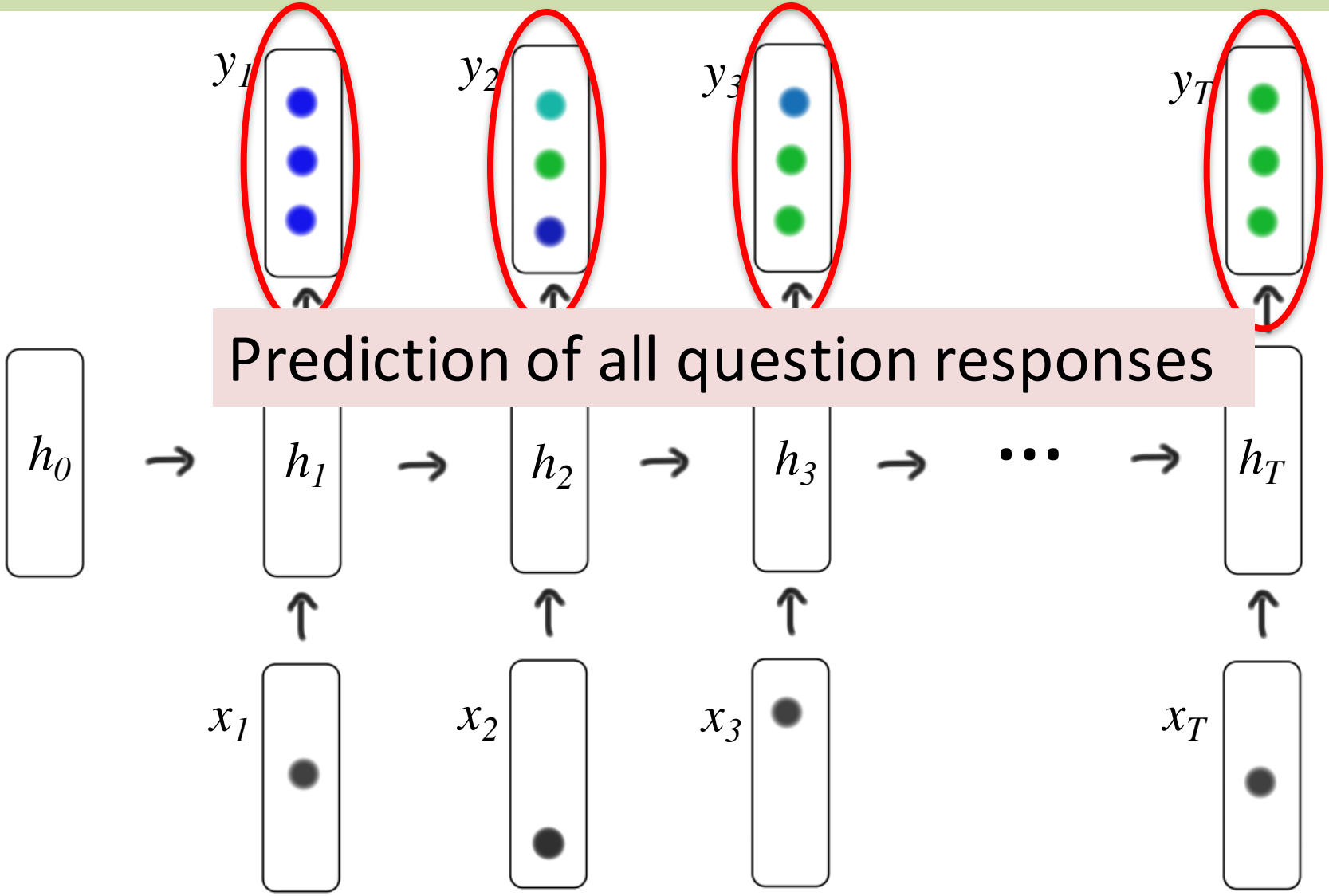
Deep Knowledge Tracing



One hot encoding of question id and correct (q_i, a_i)

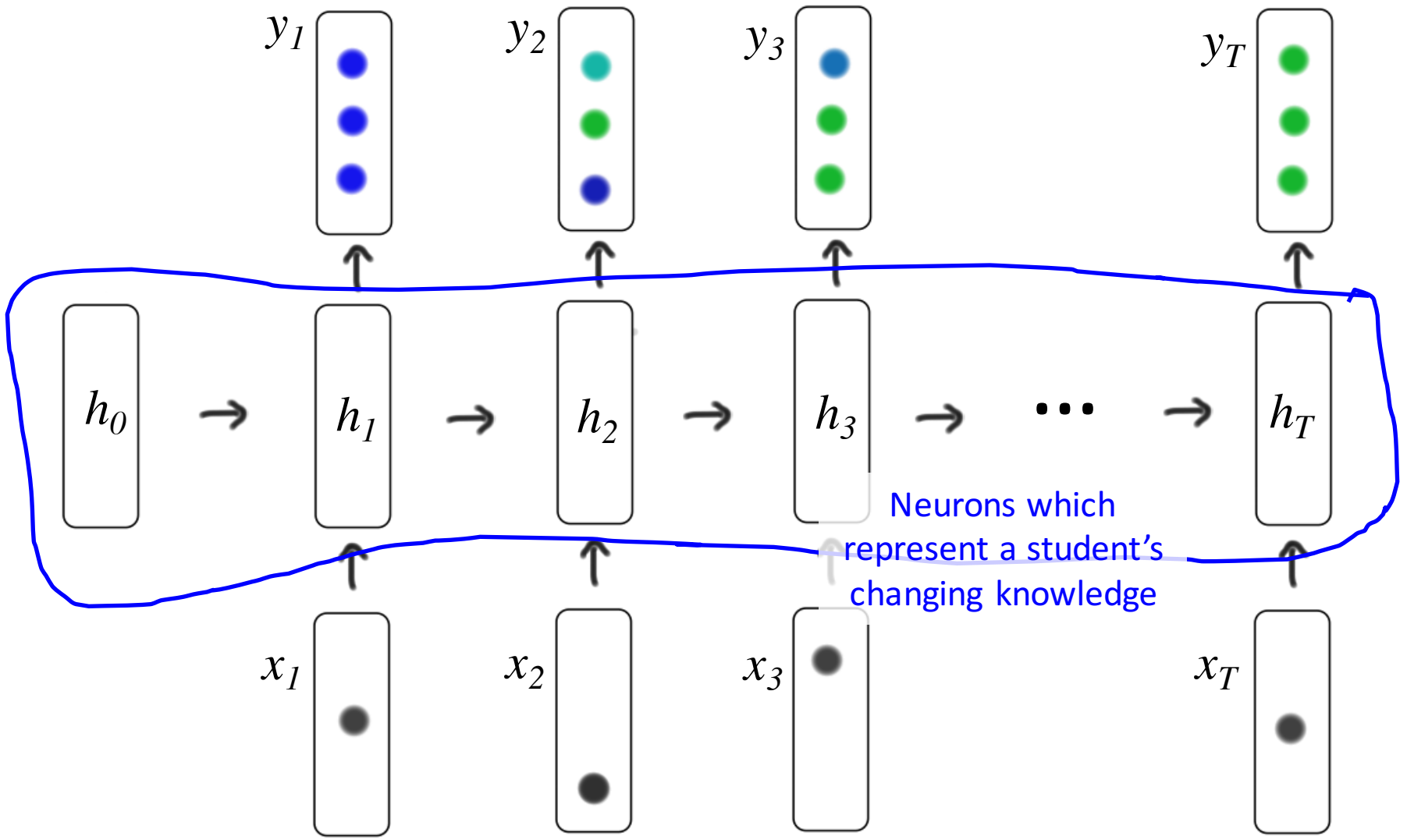
Piech

Deep Knowledge Tracing



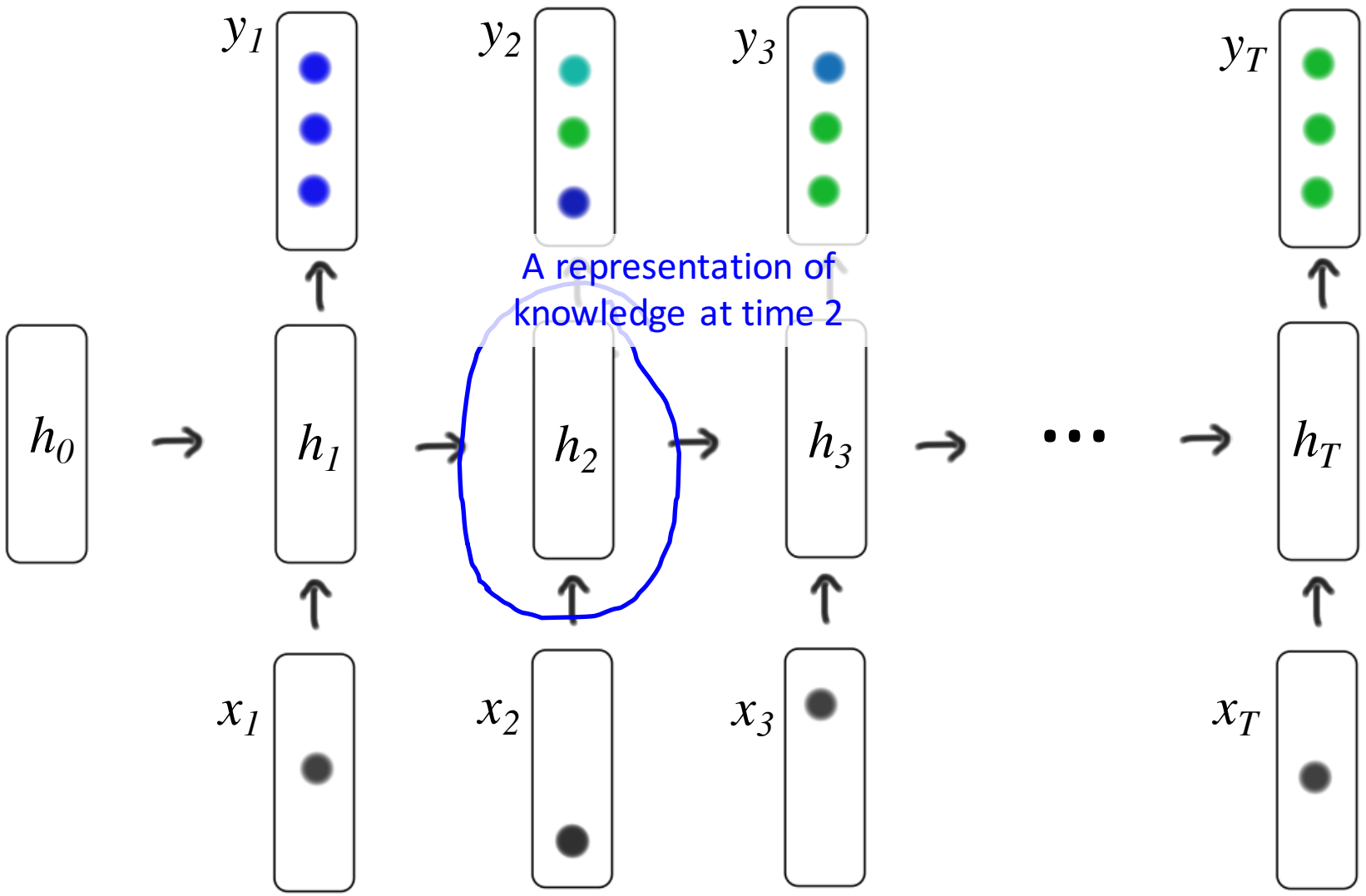
Piech

Recurrent Neural Network



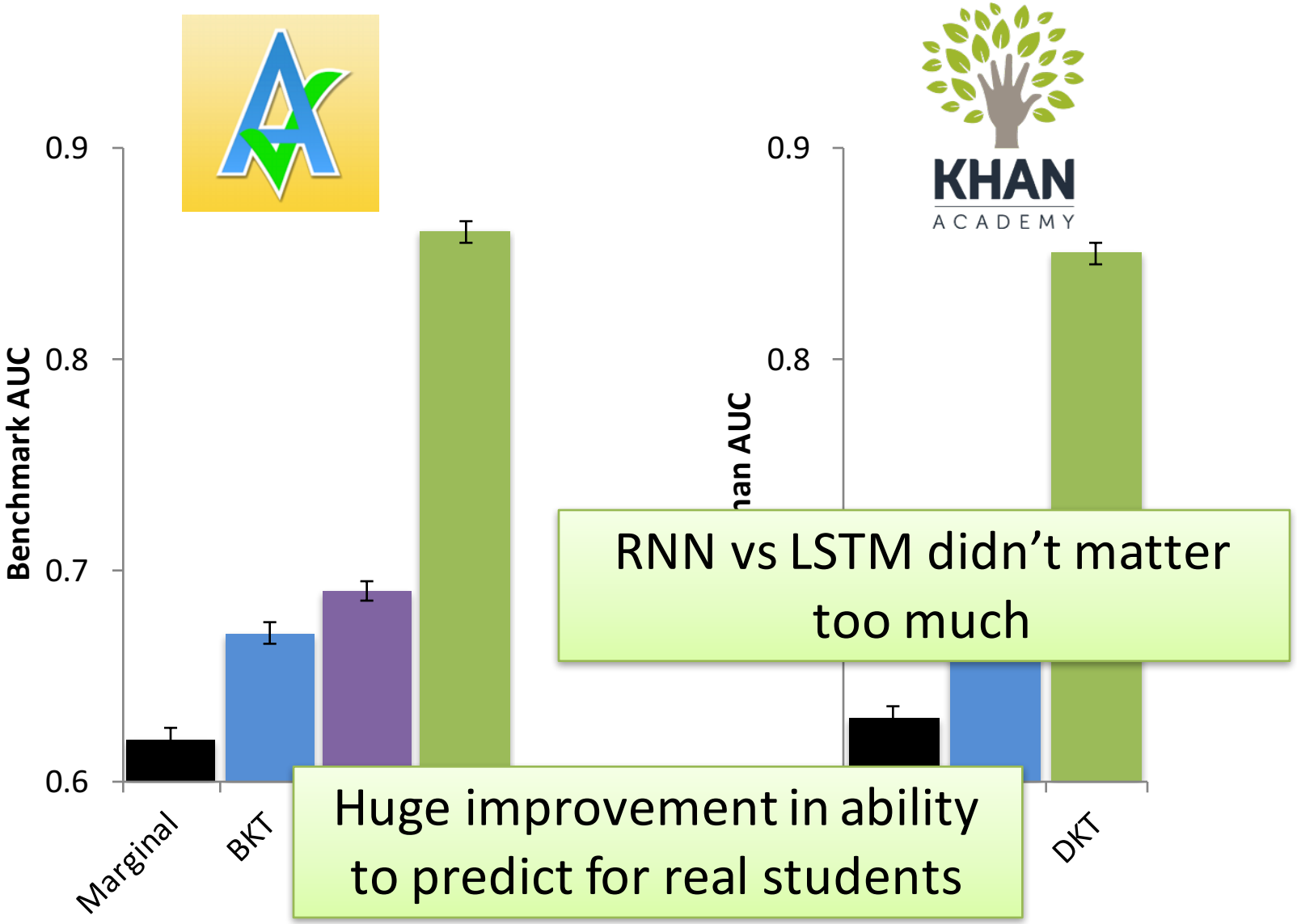
Piech

Recurrent Neural Network



Piech

Prediction Results



RNN vs LSTM didn't matter too much

Huge improvement in ability to predict for real students

Piech

Interpretation

If you can't do problem *X*

Then you can't do problem *Y* either.

But, if you can do problem *X*

Then you can do problem *Y* too.

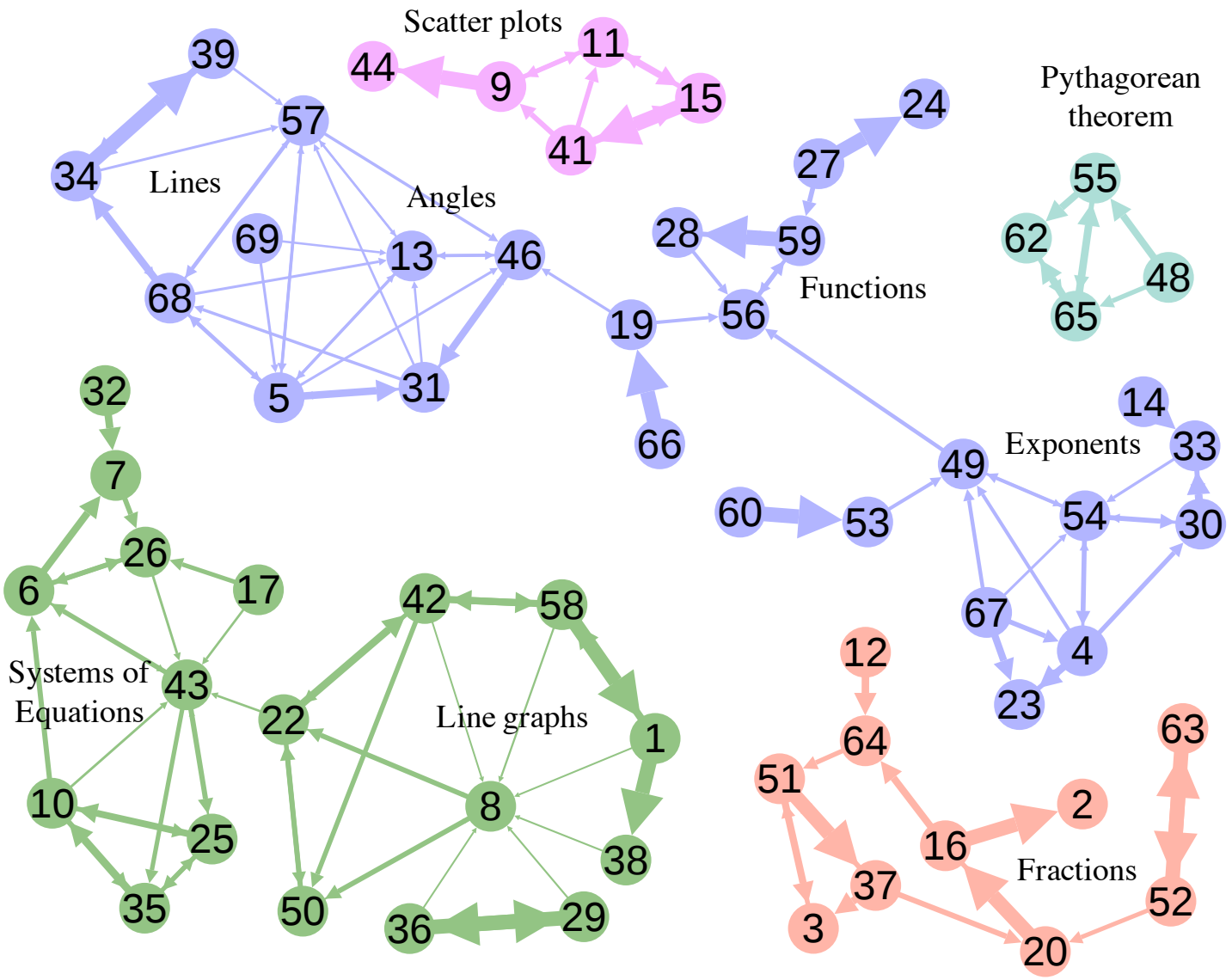
$$\rho(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var(X)Var(Y)}}}$$

$$\mathrm{Cov}(X,Y) = E[XY] - E[X]E[Y]$$

$$= P(XY) - P(X)P(Y)$$

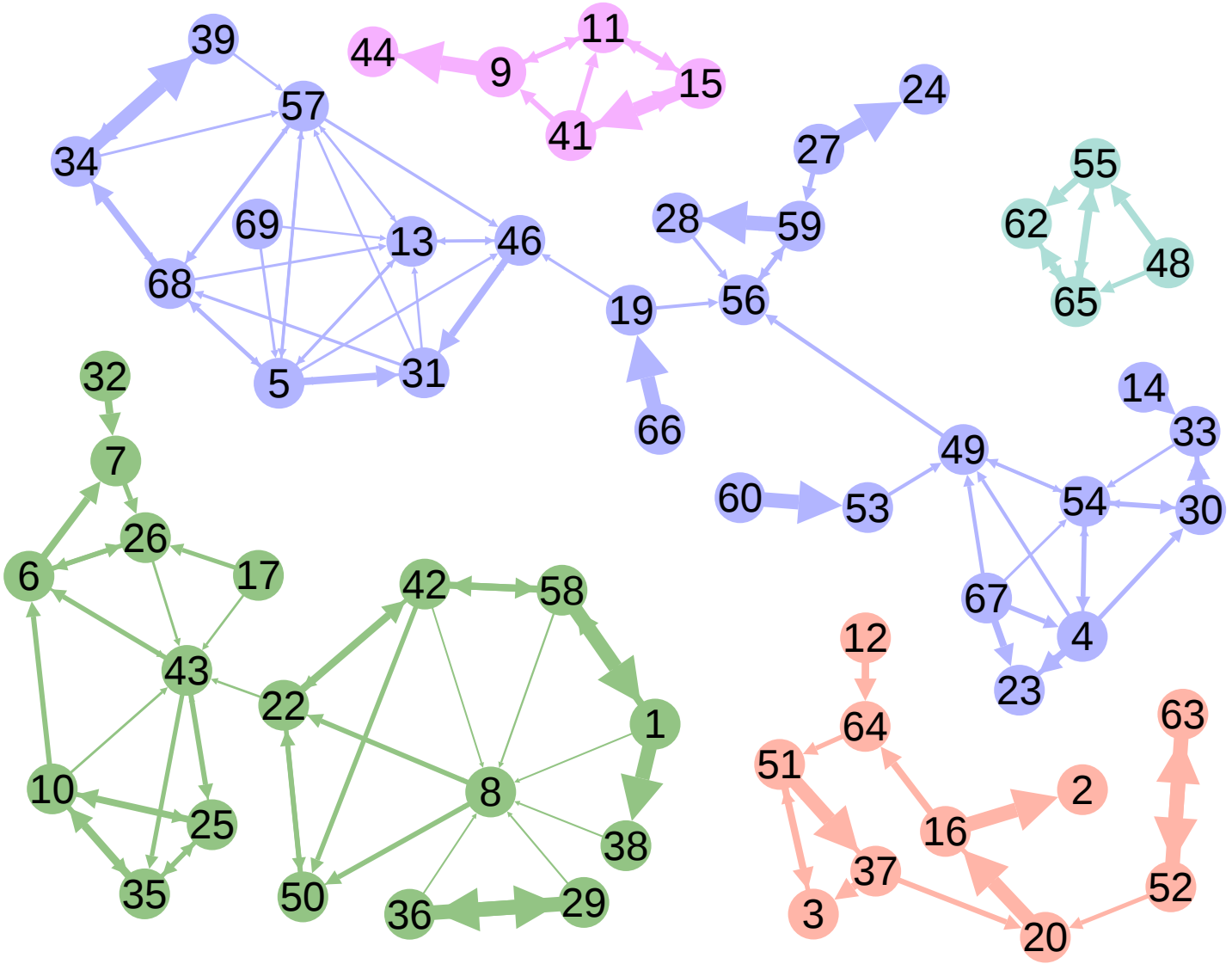
Piech

Learns Concept Relationships



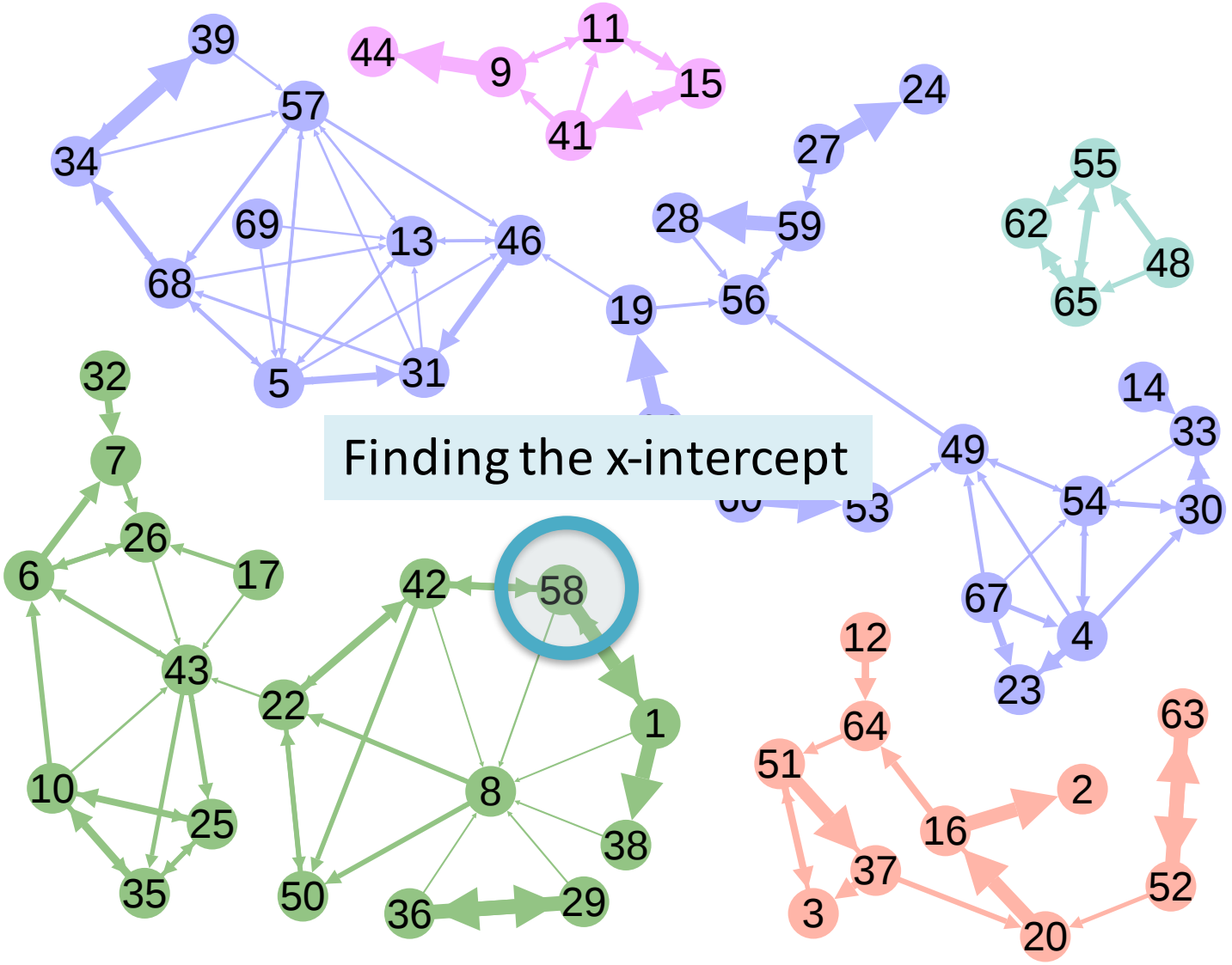
Piech

Learns Concept Relationships



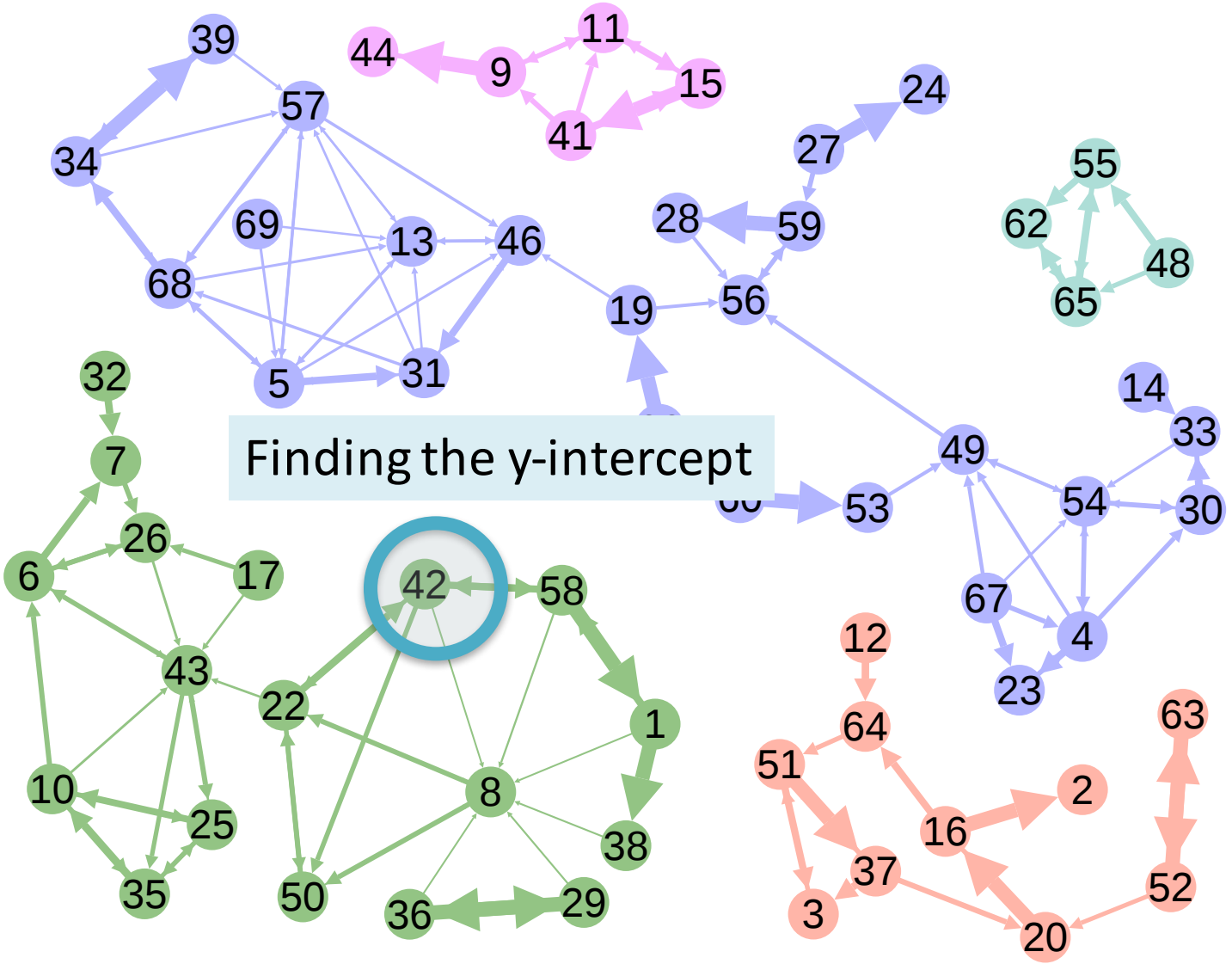
Piech

Learns Concept Relationships



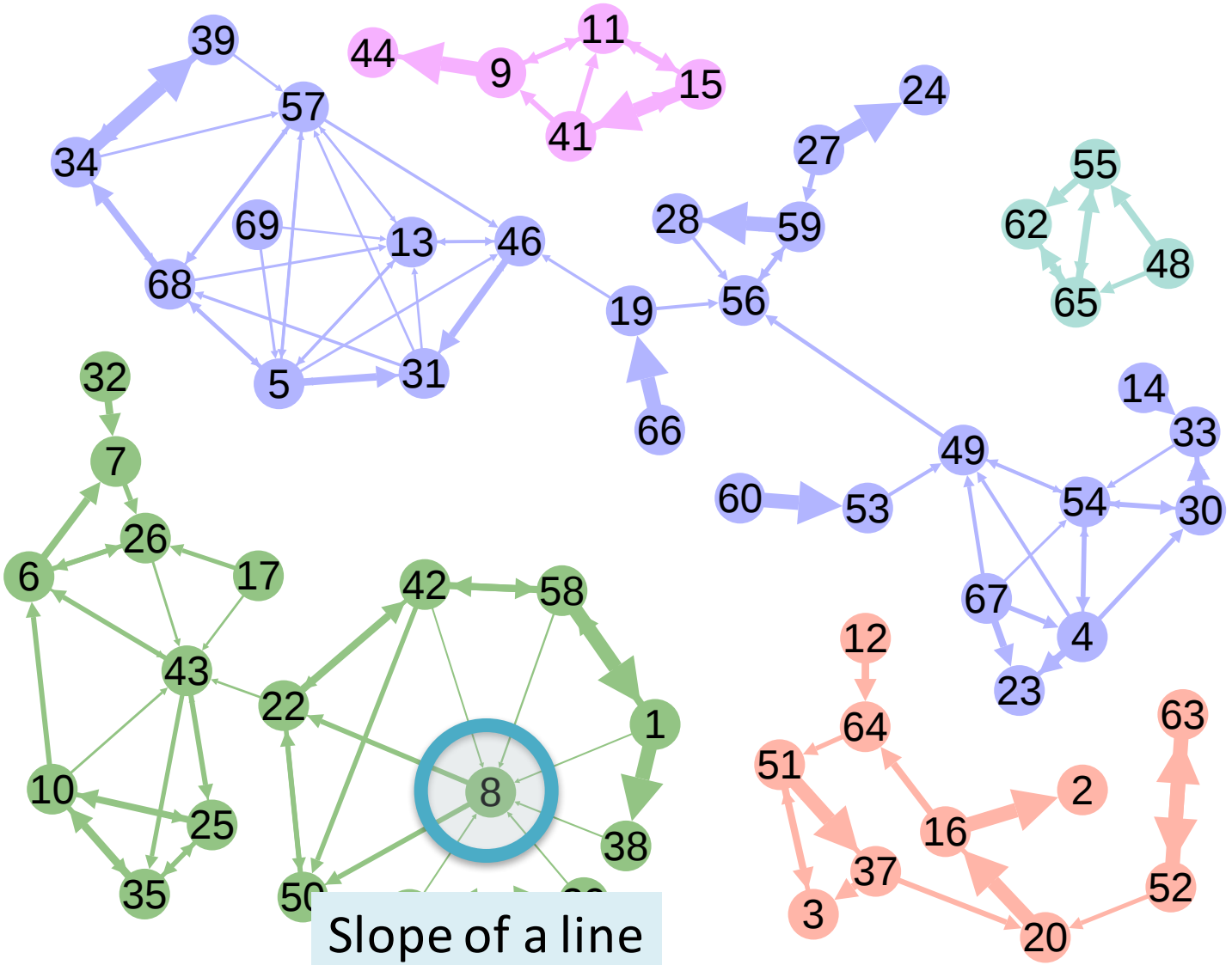
Piech

Learns Concept Relationships



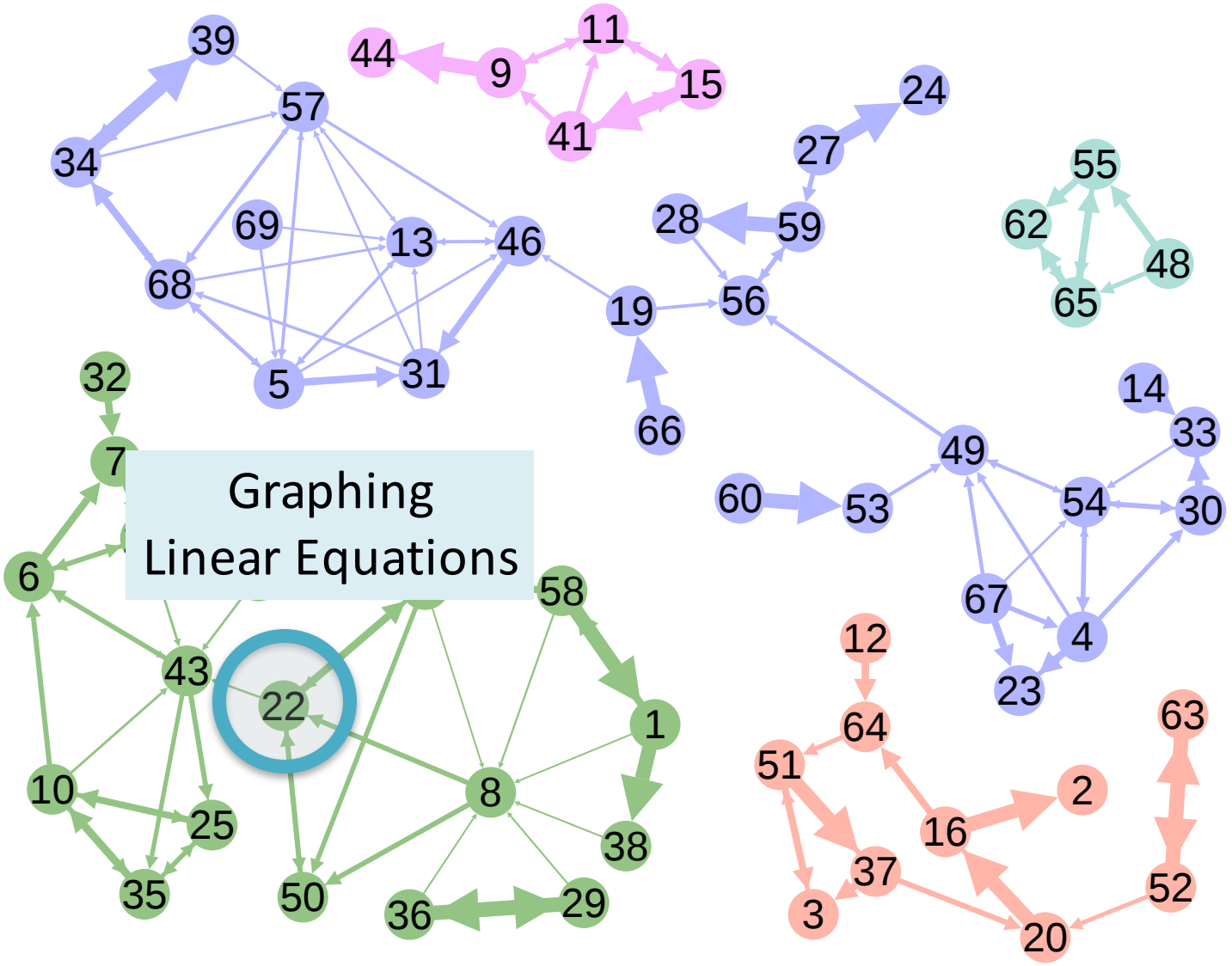
Piech

Learns Concept Relationships



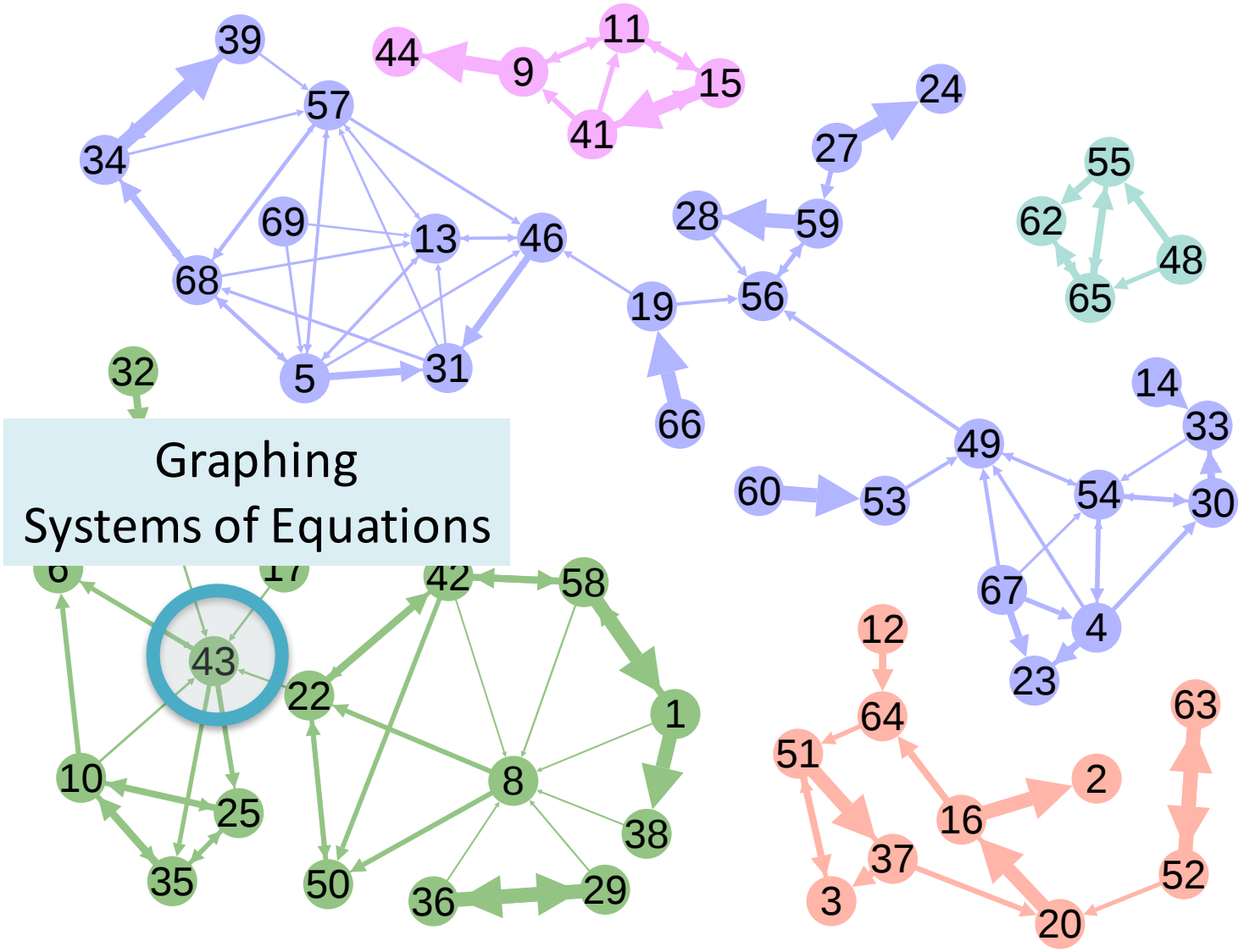
Piech

Learns Concept Relationships



Piech

Learns Concept Relationships



Graphing
Systems of Equations

Piech

Let’s try it!

Piech

Only type of AI worth knowing?

No!

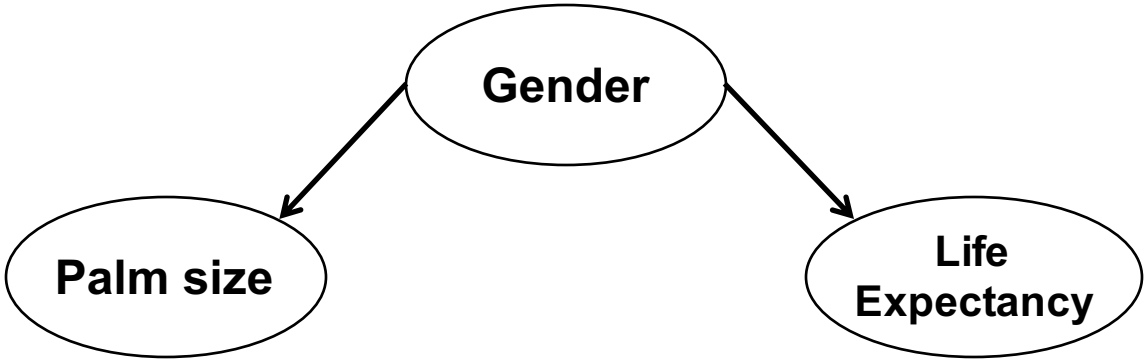
Predictions vs Understanding

- In machine learning, maintain critical perspective
  - Making predictions is only part of the story
  - Humans solve many different tasks. Don't just predict.
- Example
  - True statement: palm size negatively correlates with life expectancy
    - The larger your palm size, the shorter your life (on average)
  - Why?
    - Women have smaller palms than men on average
    - Women live 5 years longer than men on average
  - Sometimes you need better model of your domain!

Piech

Bayesian Networks

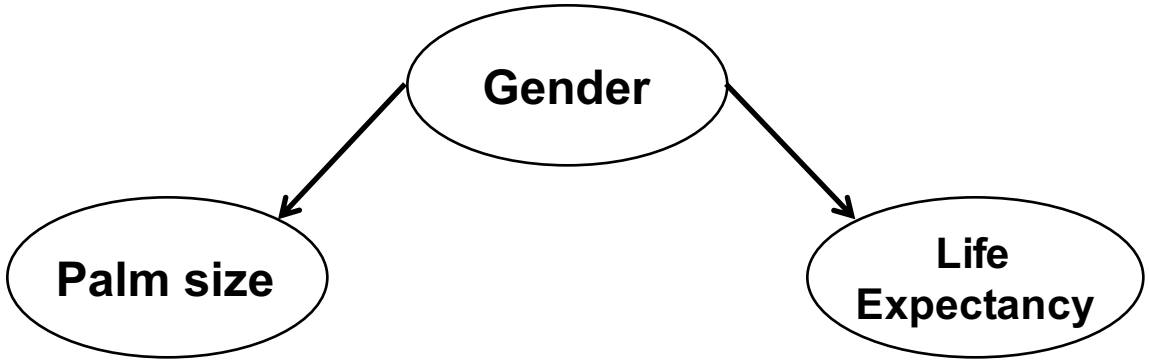
- Graphical representation of joint probability distribution



  - Node: random variable
  - Arc (X, Y): variable X has direct influence on variable Y
    - Call X a "parent" of Y
  - Each node X has conditional probability: P(X | parents(X))
  - Graph has no cycles (loops by following arcs)
    - Called "Directed Acyclic Graph" (DAG)

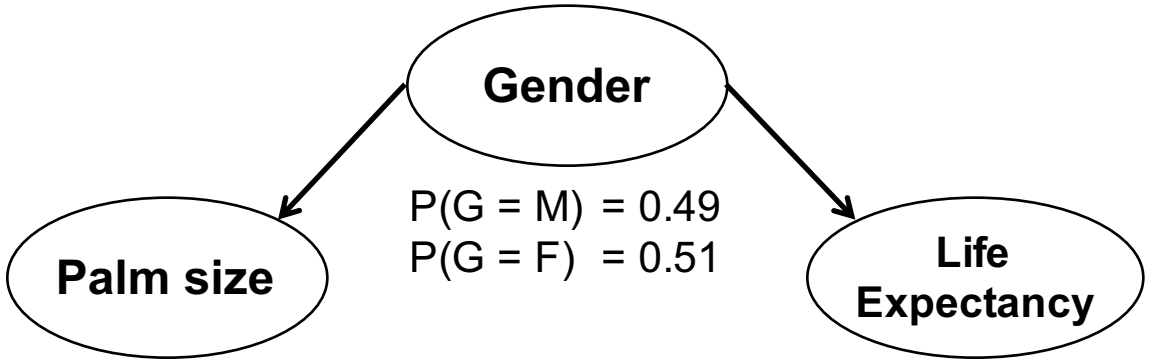
Piech

Shows Conditional Independence



- Conditional independence encoded in network
  - Each node (variable) is conditionally independent of its non-descendants, given its parents
  - In network above, Palm Size (PS) and Life Expectancy (LE) are conditionally independent, given Gender (G)
    - Formally: P(PS, LE | G) = P(PS | G) P(LE | G)
- Network structure provides insight about domain

Conditional Probability Tables



P(PS = L | G = M) = 0.45
P(PS = M | G = M) = 0.35
P(PS = S | G = M) = 0.20
P(PS = L | G = F) = 0.10
P(PS = M | G = F) = 0.30
P(PS = S | G = F) = 0.60

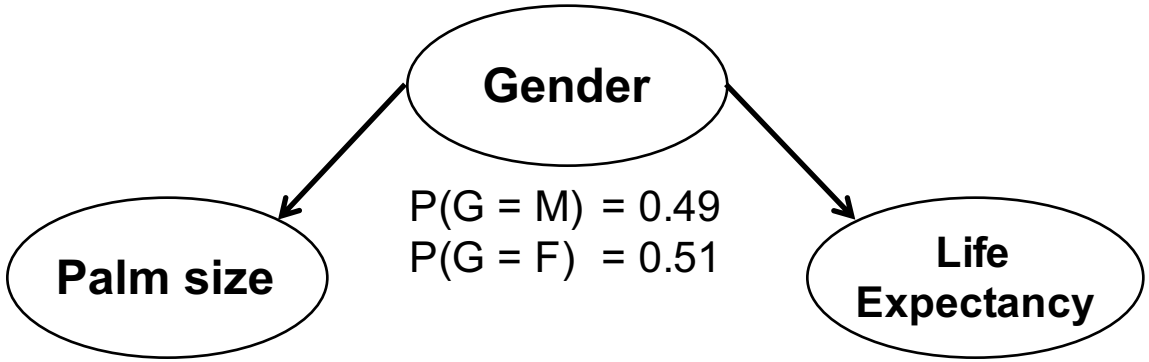
P(LE = < 70 | G = M) = 0.20
P(LE = 70-80 | G = M) = 0.50
P(LE = > 80 | G = M) = 0.30
P(LE = < 70 | G = F) = 0.10
P(LE = 70-80 | G = F) = 0.35
P(LE = > 80 | G = F) = 0.55

- Each node has conditional probability table (CPT)
  - For node X: P(X | Parents(X))
  - Conditional independence modularizes joint probability:

$$P(X_1, X_2, \ldots, X_m) = \prod_{i=1}^{m} P(X_i \mid \text{Parents}(X_i))$$

Piech

Full Joint, Fewer Parameters



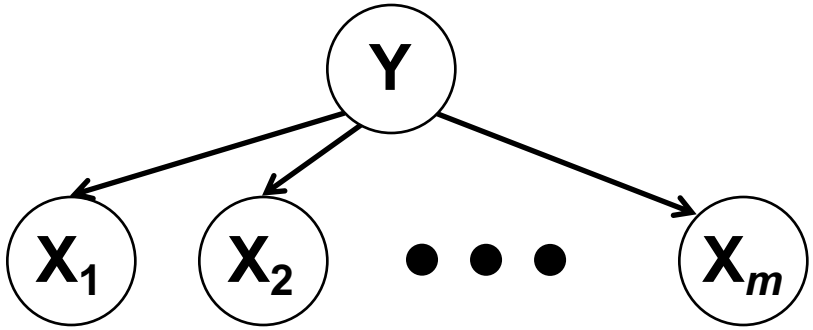
P(PS = L | G = M) = 0.45
P(PS = M | G = M) = 0.35
P(PS = S | G = M) = 0.20
P(PS = L | G = F) = 0.10
P(PS = M | G = F) = 0.30
P(PS = S | G = F) = 0.60

P(LE = < 70 | G = M) = 0.20
P(LE = 70-80 | G = M) = 0.50
P(LE = > 80 | G = M) = 0.30
P(LE = < 70 | G = F) = 0.10
P(LE = 70-80 | G = F) = 0.35
P(LE = > 80 | G = F) = 0.55

- Each node has conditional probability table (CPT)
  - Reduces number of parameters needed in model
  - Normally, need 2 x 3 x 3 – 1 = 18 – 1 = 17 parameters
  - Here, need (2 – 1) + (6 – 2) + (6 – 2) = 9 parameters

Naïve Bayes is a Bayesian Network

- Welcome back, Naïve Bayes…
  - Now with new and improved "Bayesian Network" flavor!



  - Network structure encodes assumption:

$$P(\boldsymbol{X} \mid Y) = P(X_1, X_2, \ldots, X_m \mid Y) = \prod_{i=1}^{m} P(X_i \mid Y)$$

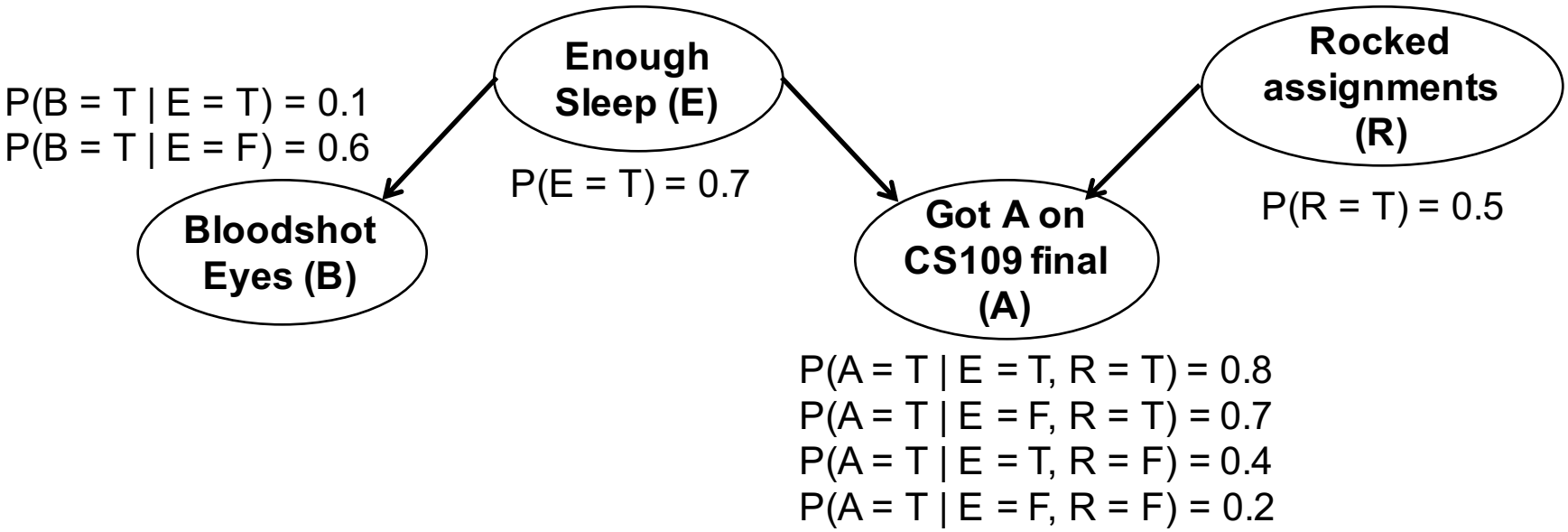
  - Full joint distribution can be computed as:

$$P(\boldsymbol{X}, Y) = P(Y)P(\boldsymbol{X} \mid Y) = P(Y)\prod_{i=1}^{m} P(X_i \mid Y)$$

Piech

Answer Many Questions

- Consider the following Bayes Net:



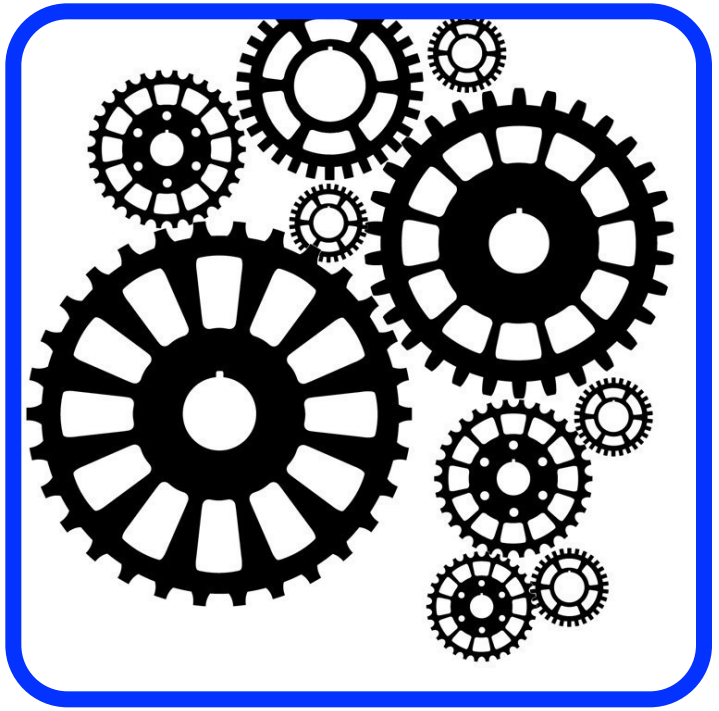
- Determine $P(A = T \mid B = T, R = T)$

When Do I Get a Robo Tutor Take 2?

Encoding a Simulator is Reasonable

Student Knowledge, experience

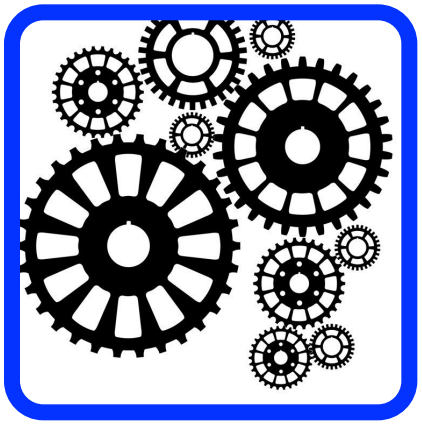
Problem



Student solution

Piech

Samples from Joint -> The data we need



Infinite samples from Joint Distribution of Knowledge and observable

Learn to understand

Piech

All the Breakouts

We have 10,000,000 labelled breakouts

Piech

The future of artificial intelligence is uncertain

Open questions:

Natural Language Processing?

One Shot Learning?

General AI?

How will these ideas help improve the quality of human life?

You tell me.

Probability is a phenomenal toolset