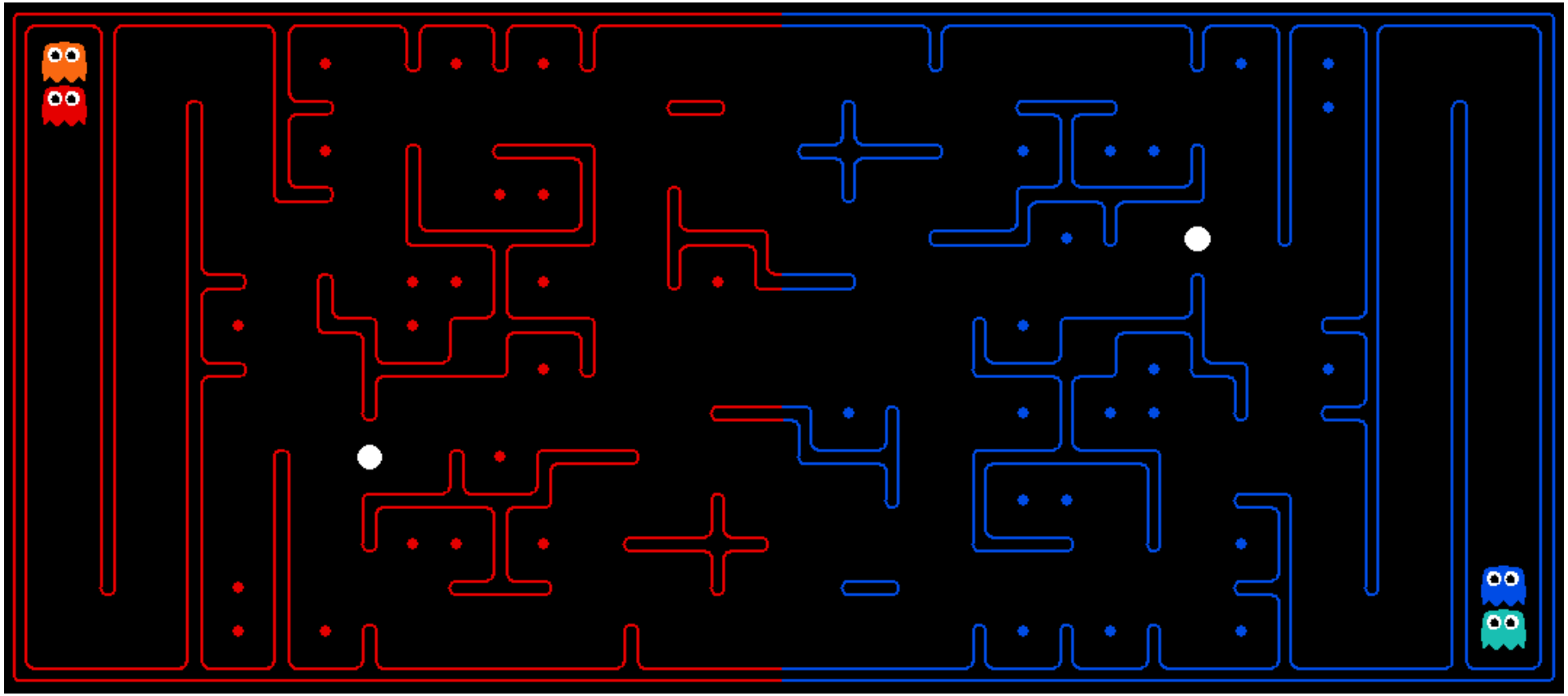
Announcements

- Project 5 Ghostbusters
  - Due Friday 4/15 at 5pm
- Homework 9
  - Released soon, due Monday 4/18 at 11:59pm
- Cal Day – Robot Learning Lab Open House
  - Saturday 10am-1pm
  - 3rd floor Sutardja Dai Hall
  - Robot demos of knot tying, high-fives, fist-pumps, hugs

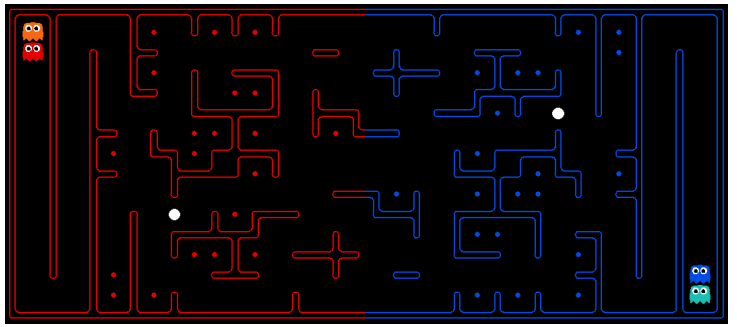
Final Contest!



- Challenges: Long term strategy, multiple agents, adversarial utilities, uncertainty about other agents' positions, plans, etc.

Final Contest Extra Credit

- Extra Credit Based on Final Ranking (deadline 4/24 11:59pm)
  - 1st place: 2 points on the final
  - 2nd and 3rd place: 1.5 points on the final
  - 4th to 10th place: 1 point on the final
  - 0.25 points on the final per staff bot beaten in the final ranking.

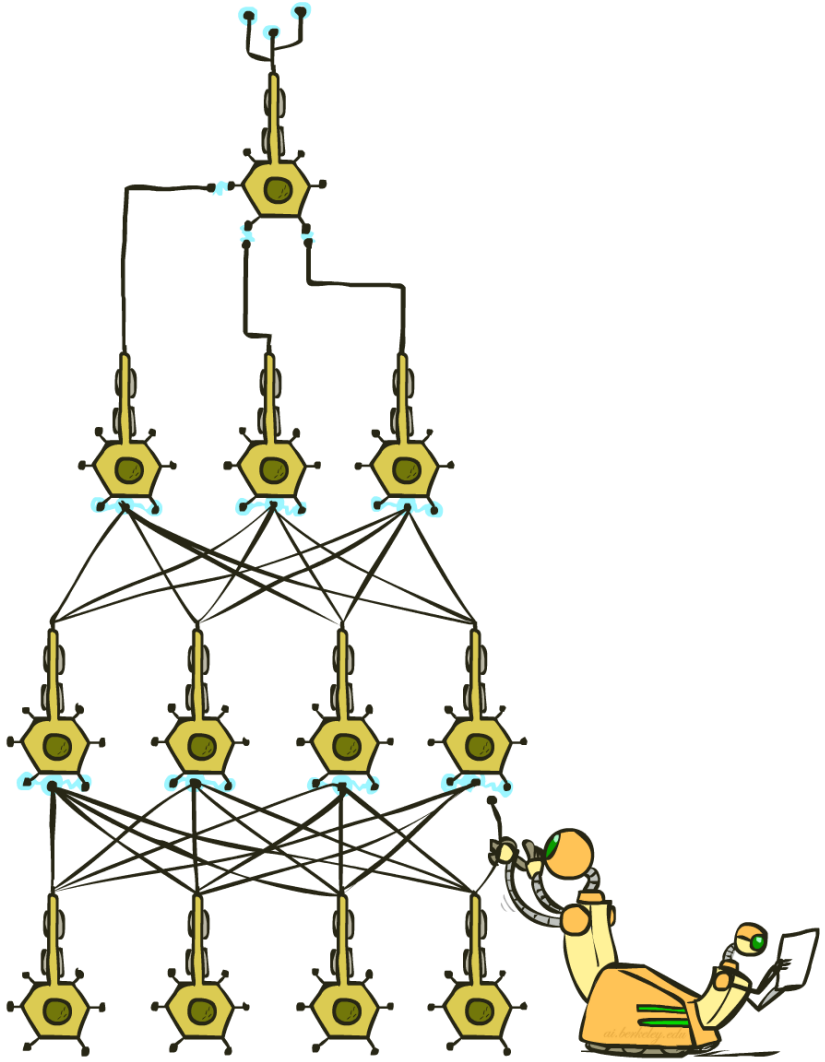


- In addition, every night from 4/17 through 4/24 at 11:59pm we will temporarily close submissions to compute an intermediate ranking, and there is also extra credit:
  - Top 20: 0.5 points on the final
  - Top 10: 1 point on the final

- Note that the ranking rewards are not cumulative, you will only get points for the highest ranking category you achieve during the contest.

Announcements

- Piazza: Don’t discuss exam!
  - People still taking exam Friday morning.
- Project 6: Out next Thursday.



CS 188: Artificial Intelligence

Deep Learning

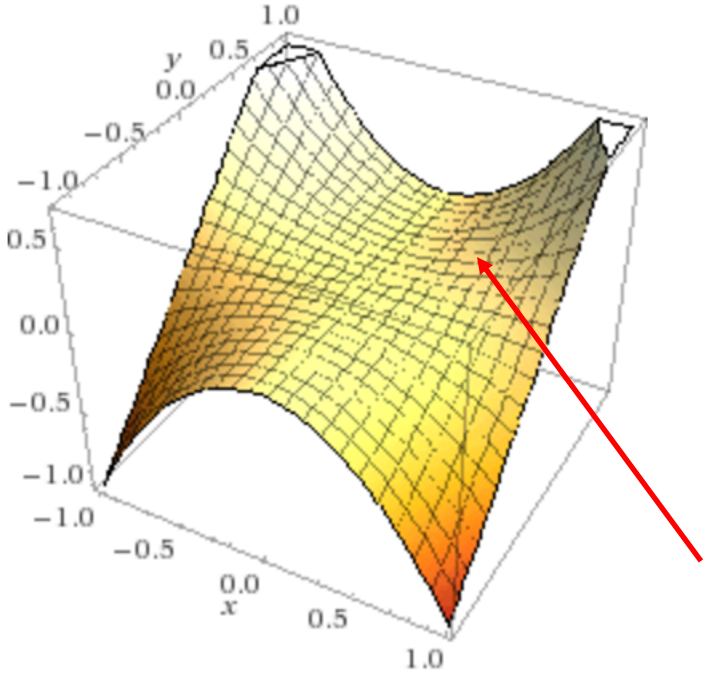
Instructors: Adam Janin & Josh Hug --- University of California, Berkeley

Review: Gradient

- What is the gradient ∇f of the function $f = x^2 y$ at x=0.5, y=0.5?

Last Time: Linear Classifiers

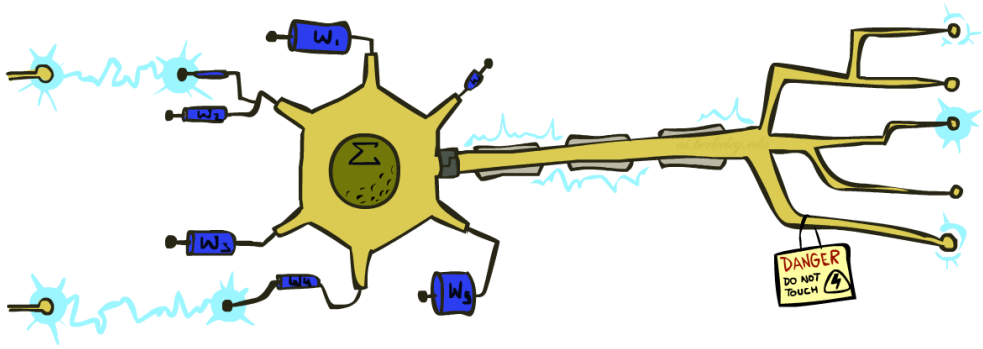
- What is the gradient ∇f of the function $f = x^2 y$ at x=0.5, y=0.5?
- $\nabla f = \frac{\partial f}{dx}\boldsymbol{i} + \frac{\partial f}{\partial y}\boldsymbol{j} = 2xy\boldsymbol{i} + x^2\boldsymbol{j} = 0.5\boldsymbol{i} + 0.25\boldsymbol{j}$



Slope of the graph in both directions.

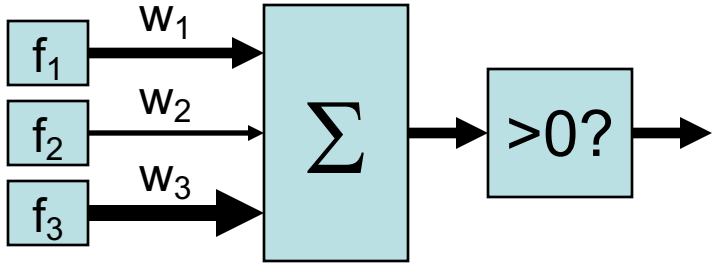
Last Time: Linear Classifiers

- Inputs are feature values
- Each feature has a weight
- Sum is the activation

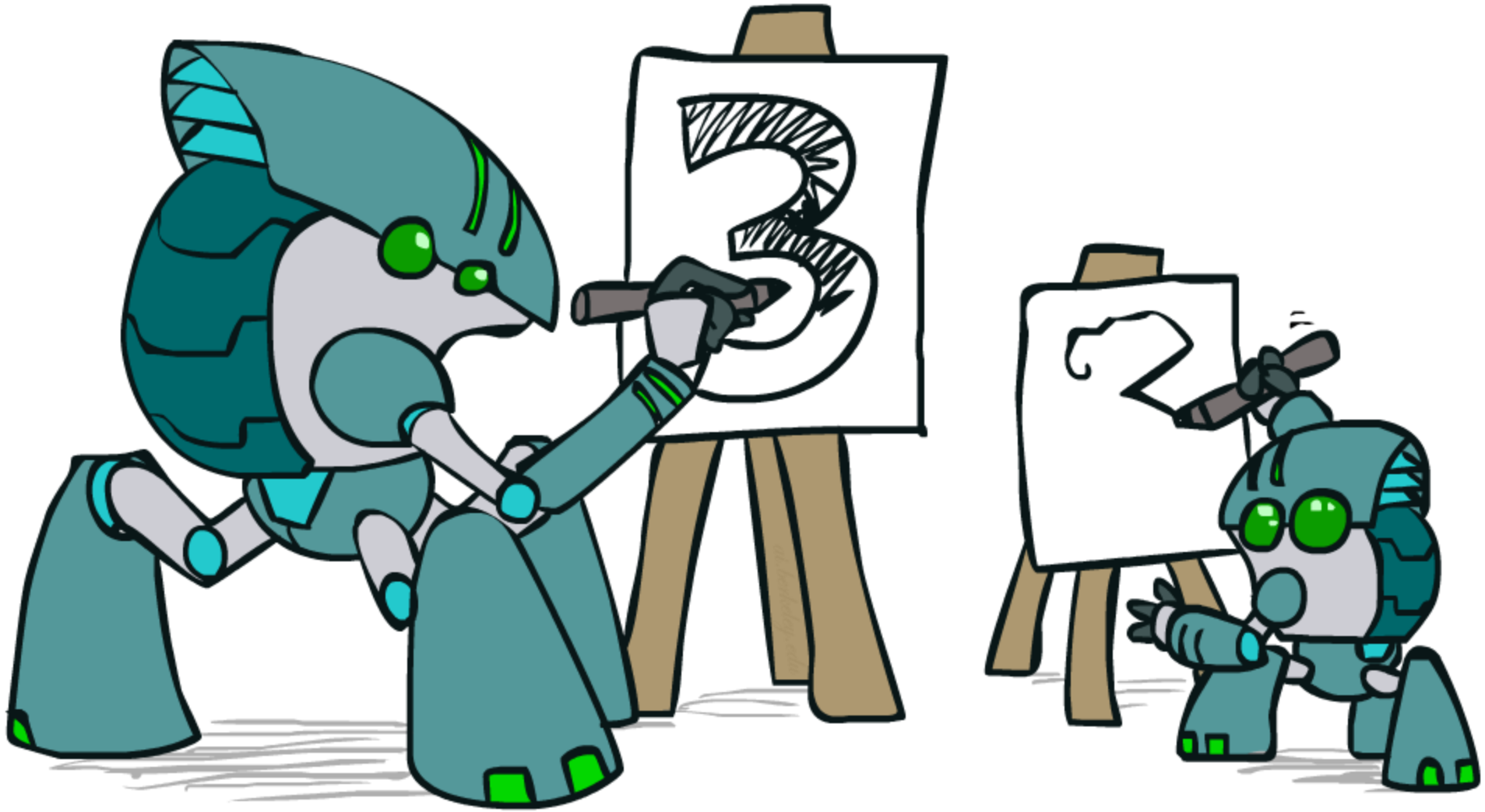


$$\text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x)$$

- If the activation is:
  - Positive, output +1
  - Negative, output -1

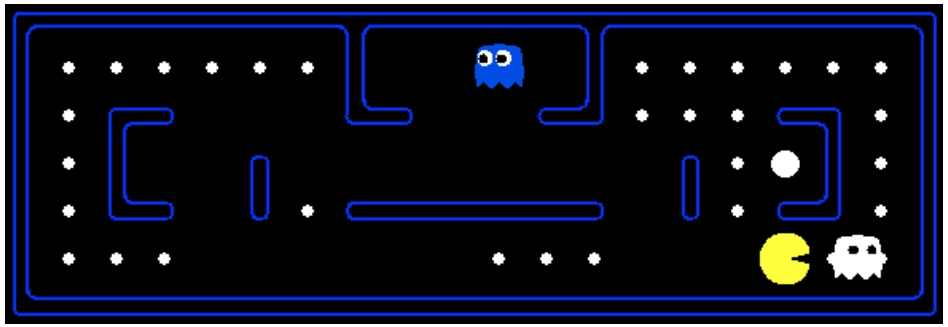


Apprenticeship



Pacman Apprenticeship!

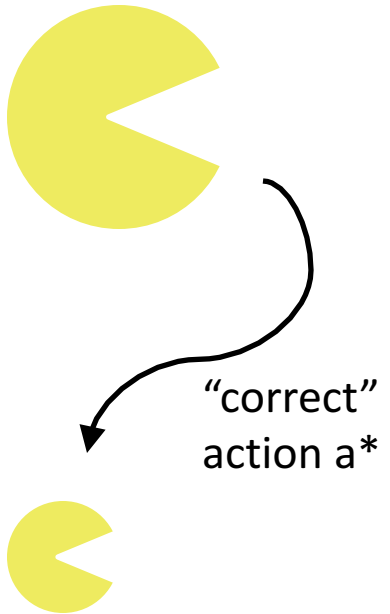
- Examples are states s



- Candidates are pairs (s,a)
- "Correct" actions: those taken by expert
- Features defined over (s,a) pairs: f(s,a)
  - f(s, a) = {closeToFood, closeToScaredGhost, score, …}
- Score of a q-state (s,a) given by:

$$w \cdot f(s,a)$$

- How is this VERY different from reinforcement learning?



$$\forall a \neq a^*,$$
$$w \cdot f(s,a^*) > w \cdot f(s,a)$$

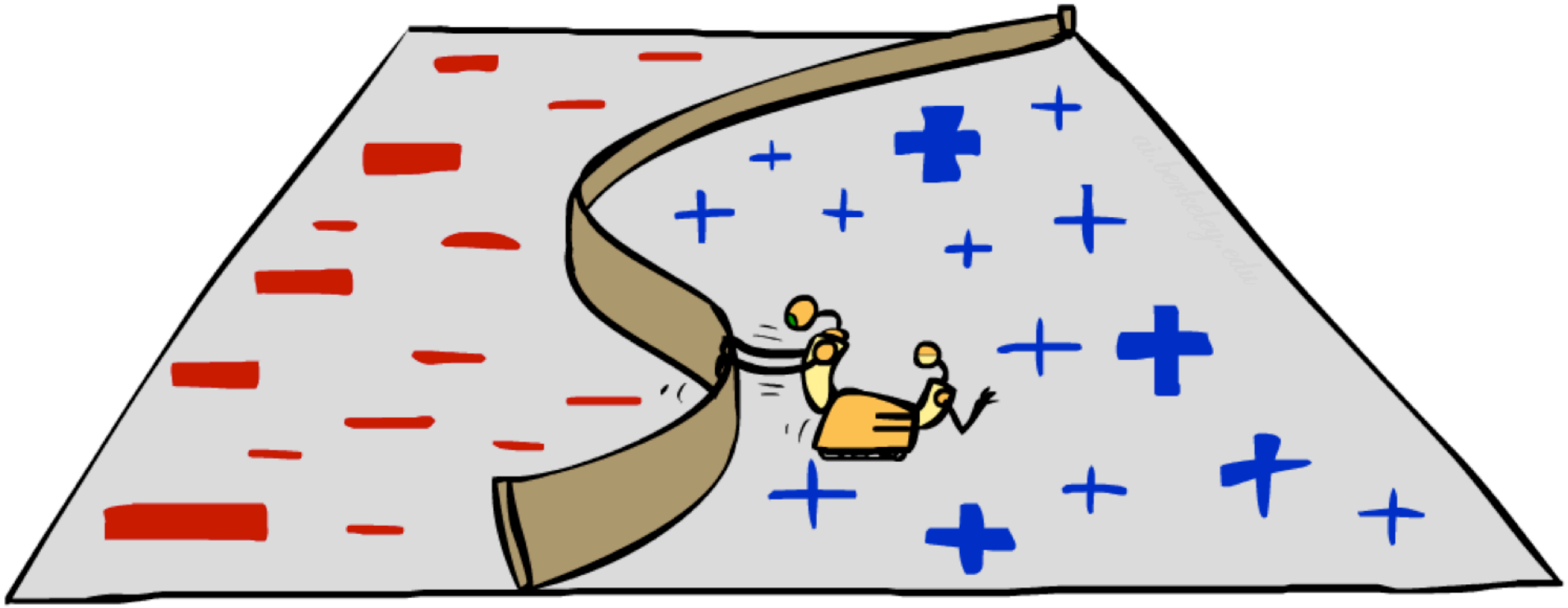
Pick largest of:

$$w \cdot f(s,N)$$
$$w \cdot f(s,E)$$
$$w \cdot f(s,W)$$
$$w \cdot f(s,S)$$

[Demo: Pacman Apprentice (L22D1,2,3)]

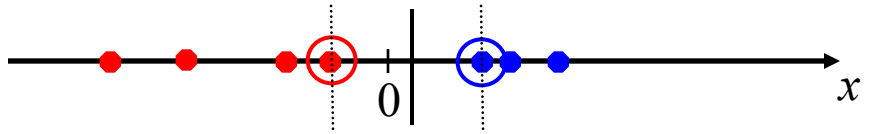
Video of Demo Pacman Apprentice

Non-Linearity

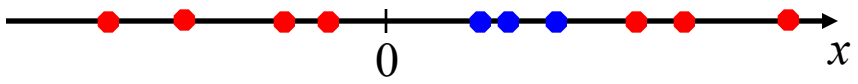


Non-Linear Separators

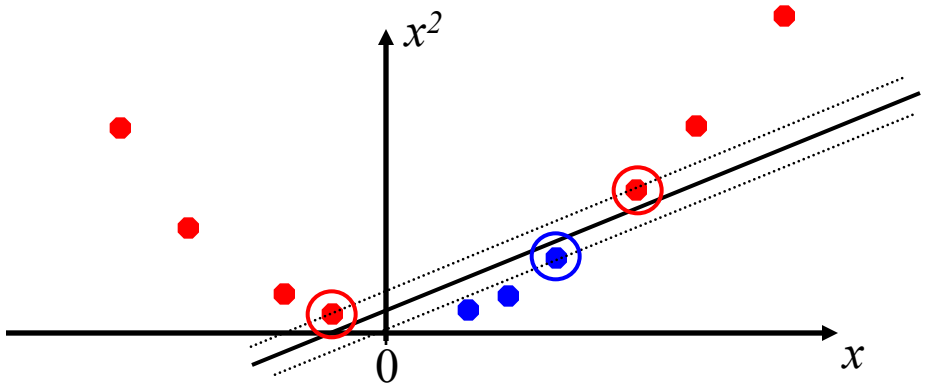
- Data that is linearly separable works out great for linear decision rules:



- But what are we going to do if the dataset is just too hard?



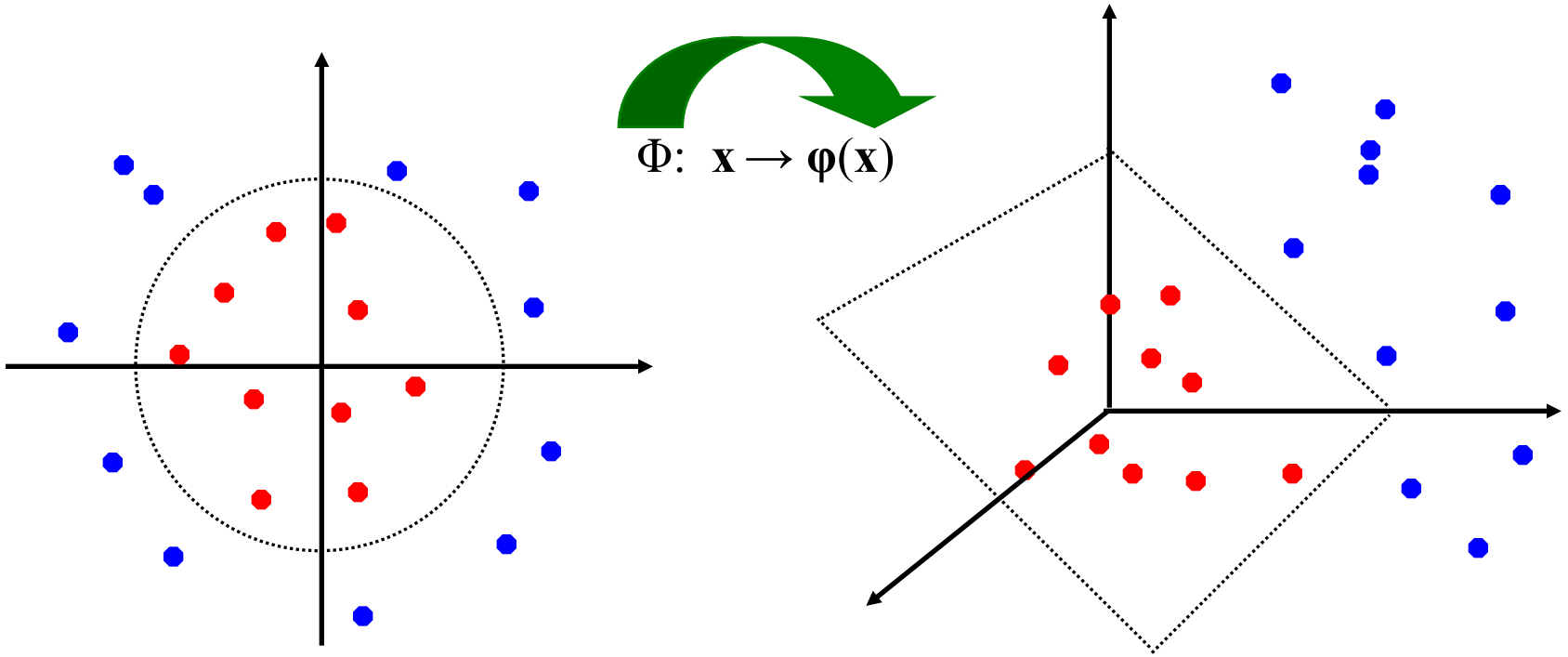
- How about… mapping data to a higher-dimensional space:



This and next slide adapted from Ray Mooney, UT

Non-Linear Separators

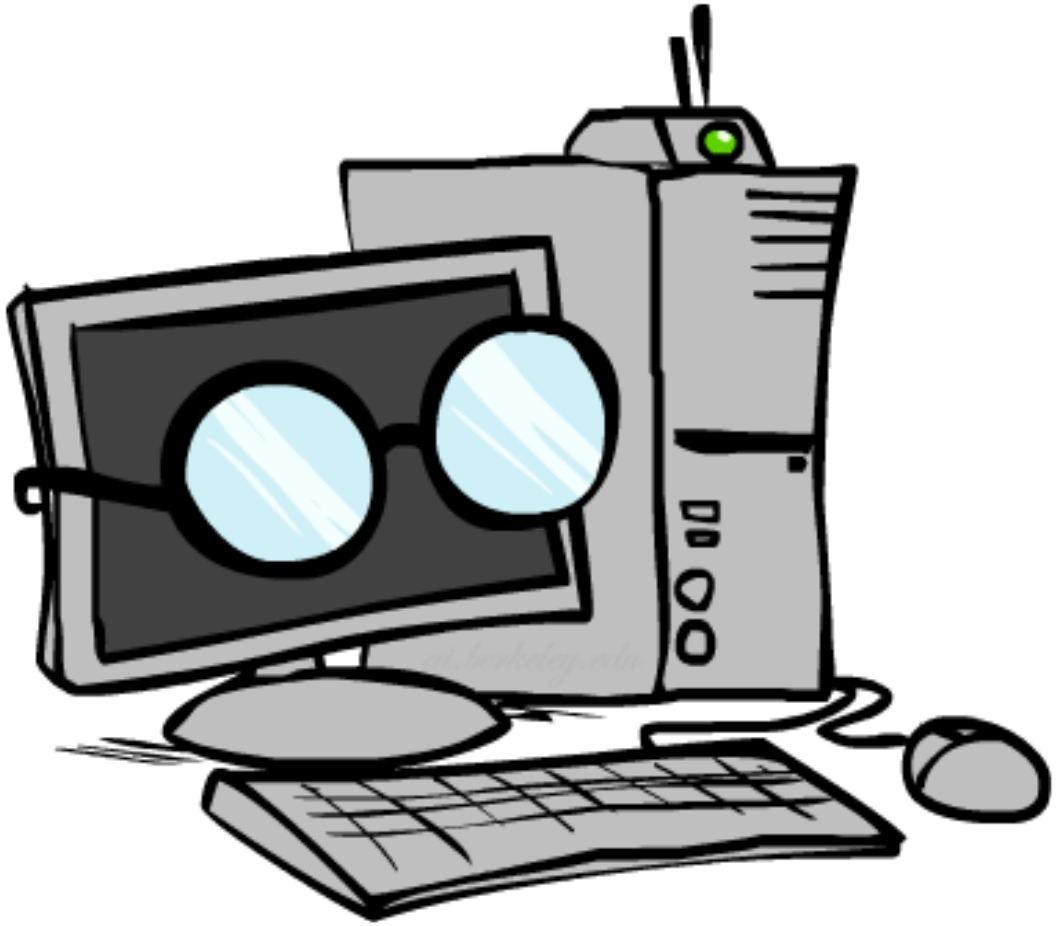
- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



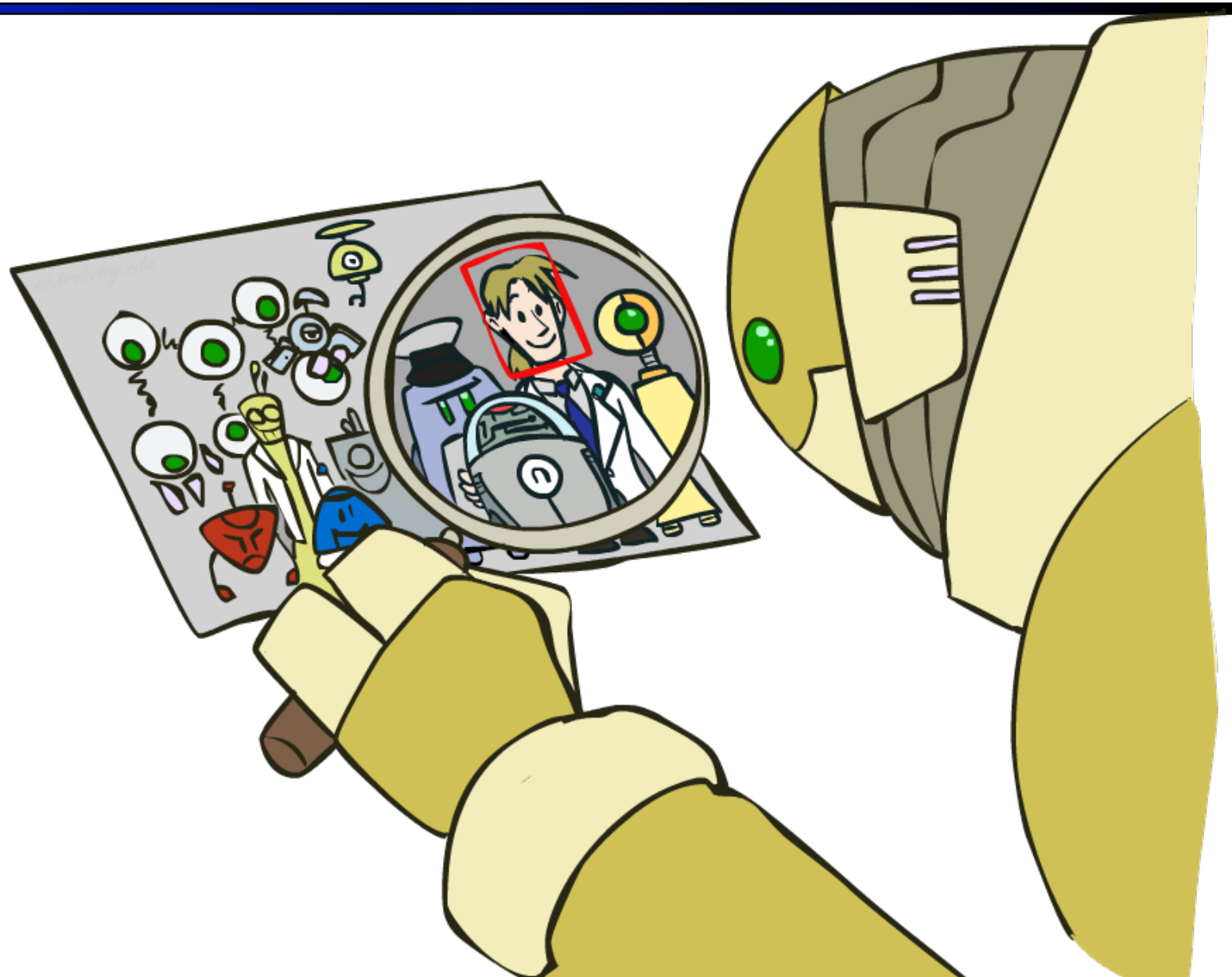
Feature Selection

- To choose between two feature sets:
  - For feature set 1: train perceptron on training data -> Classifier 1
  - For feature set 2: train perceptron on training data -> Classifier 2
  - Evaluate performance of Classifier 1 and Classifier 2 on hold-out data
    - Select the one performing best on the hold-out data

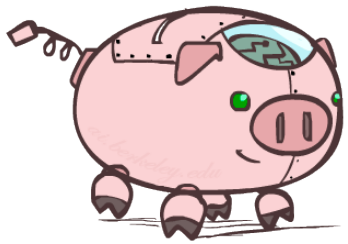
Computer Vision

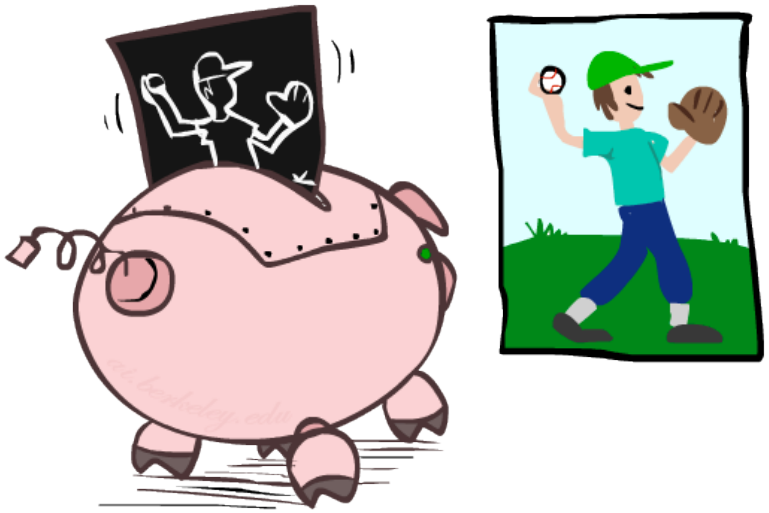


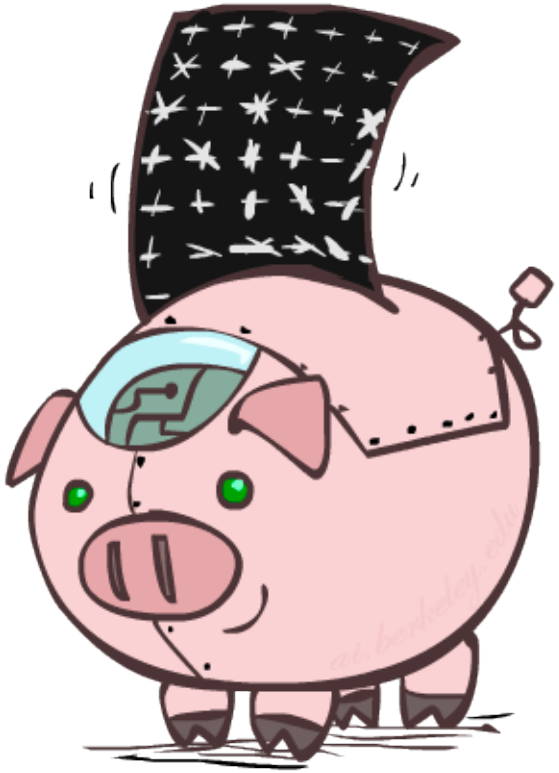
Object Detection



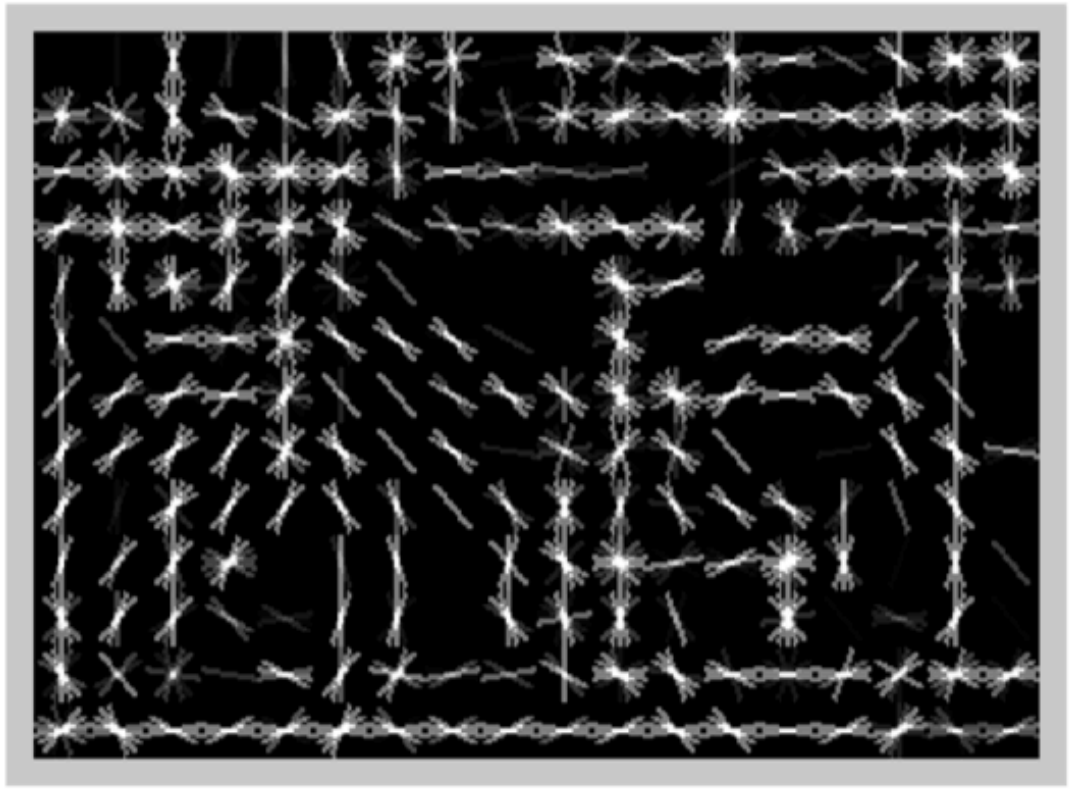
Manual Feature Design







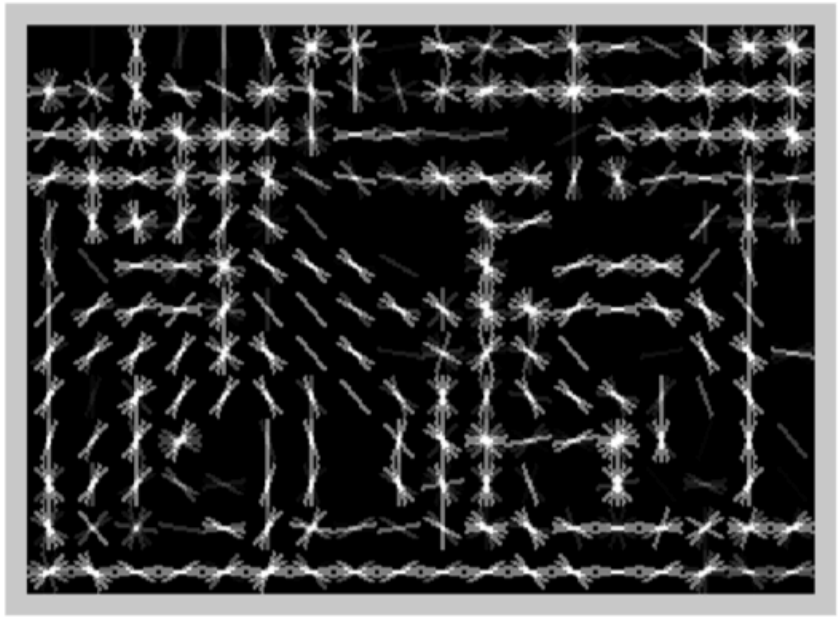
Features and Generalization



[Dalal and Triggs, 2005]

Features and Generalization

Image



HoG

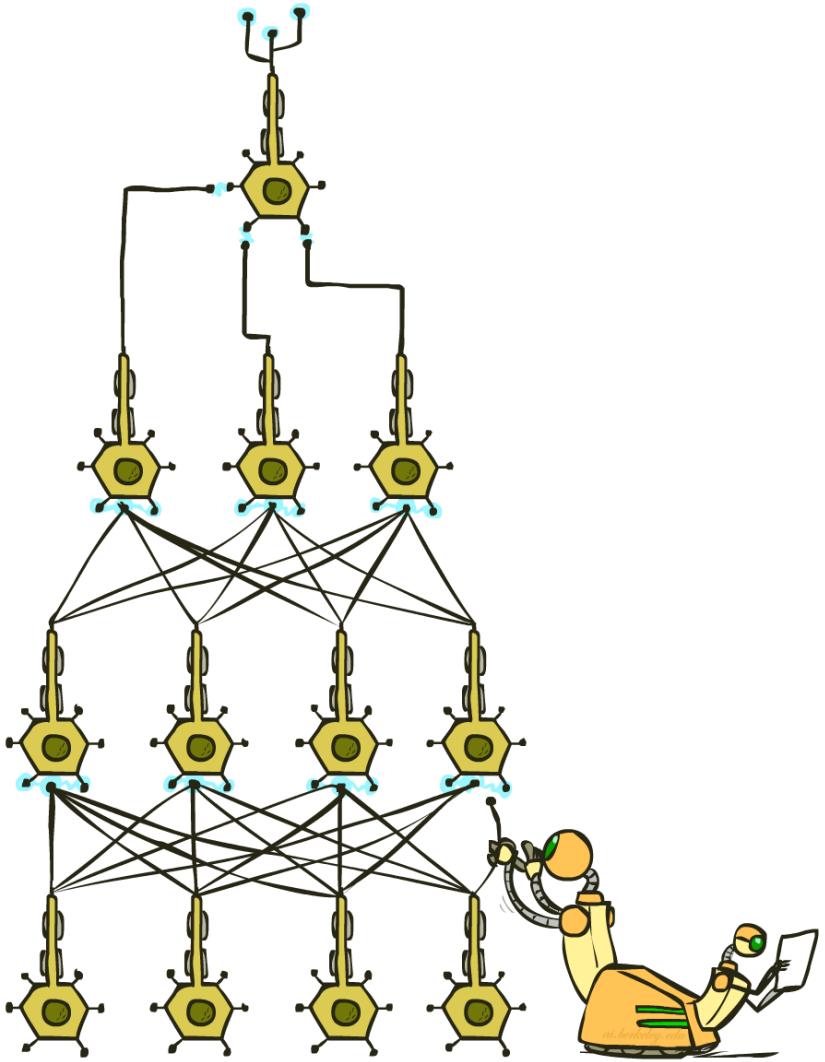
Training

- Round 1
  - Training set =
    - Positive examples: from labeling
    - Negative examples: random patches

  → preliminary SVM
- Round 2 (“bootstrapping” or “mining hard negatives”)
  - Training set =
    - Positive examples: from labeling
    - Negative examples: patches that have score >= -1

  → final SVM

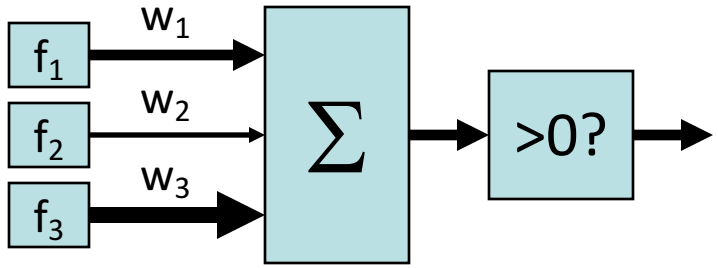
Manual Feature Design → Deep Learning



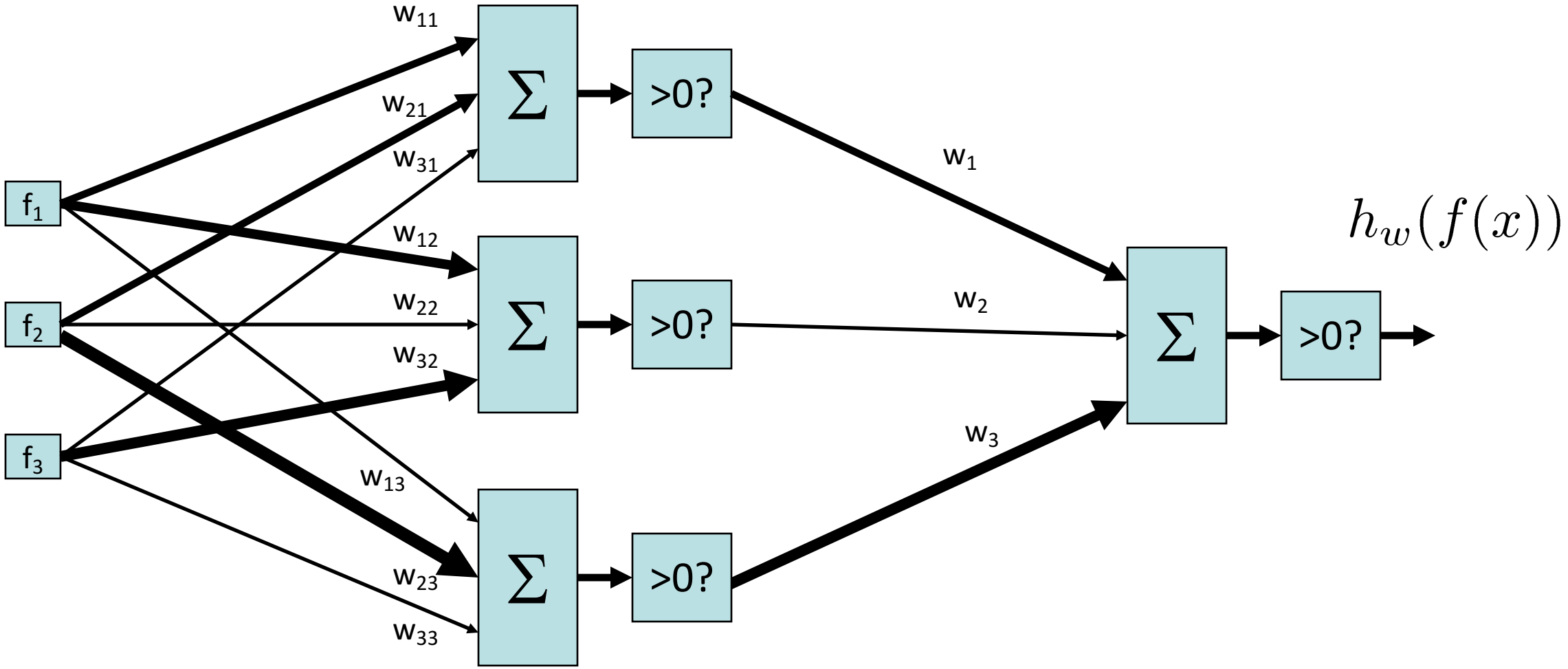
- Manual feature design requires:
  - Domain-specific expertise
  - Domain-specific effort

- What if we could learn the features, too?
  - **-> Deep Learning**

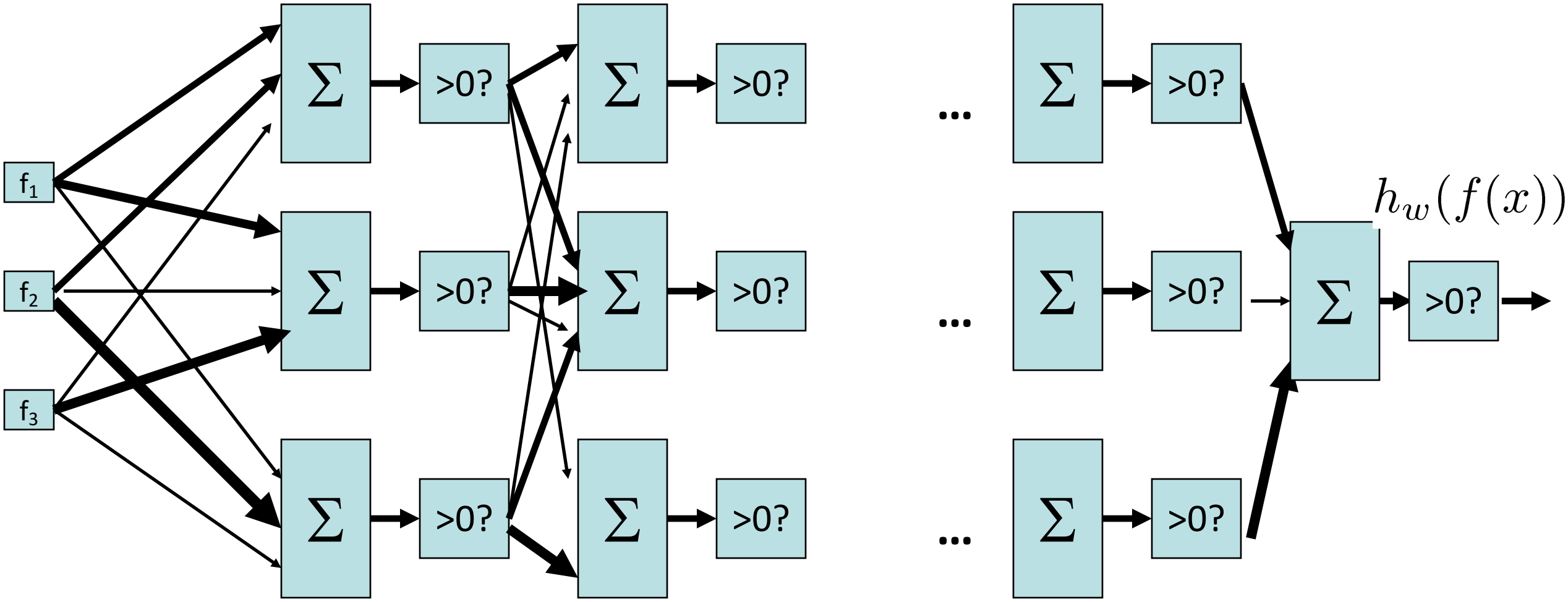
Perceptron



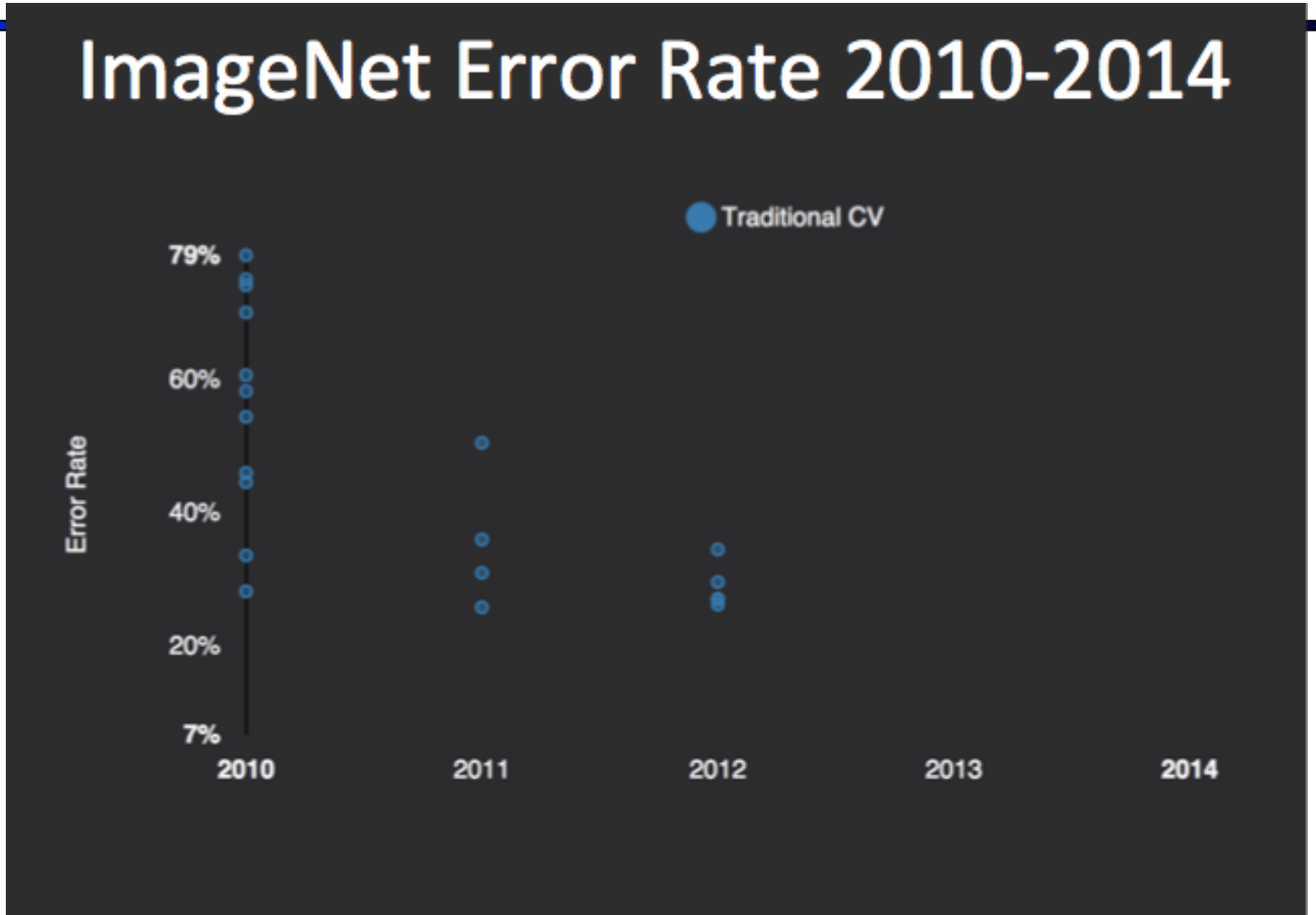
Two-Layer Perceptron Network



N-Layer Perceptron Network

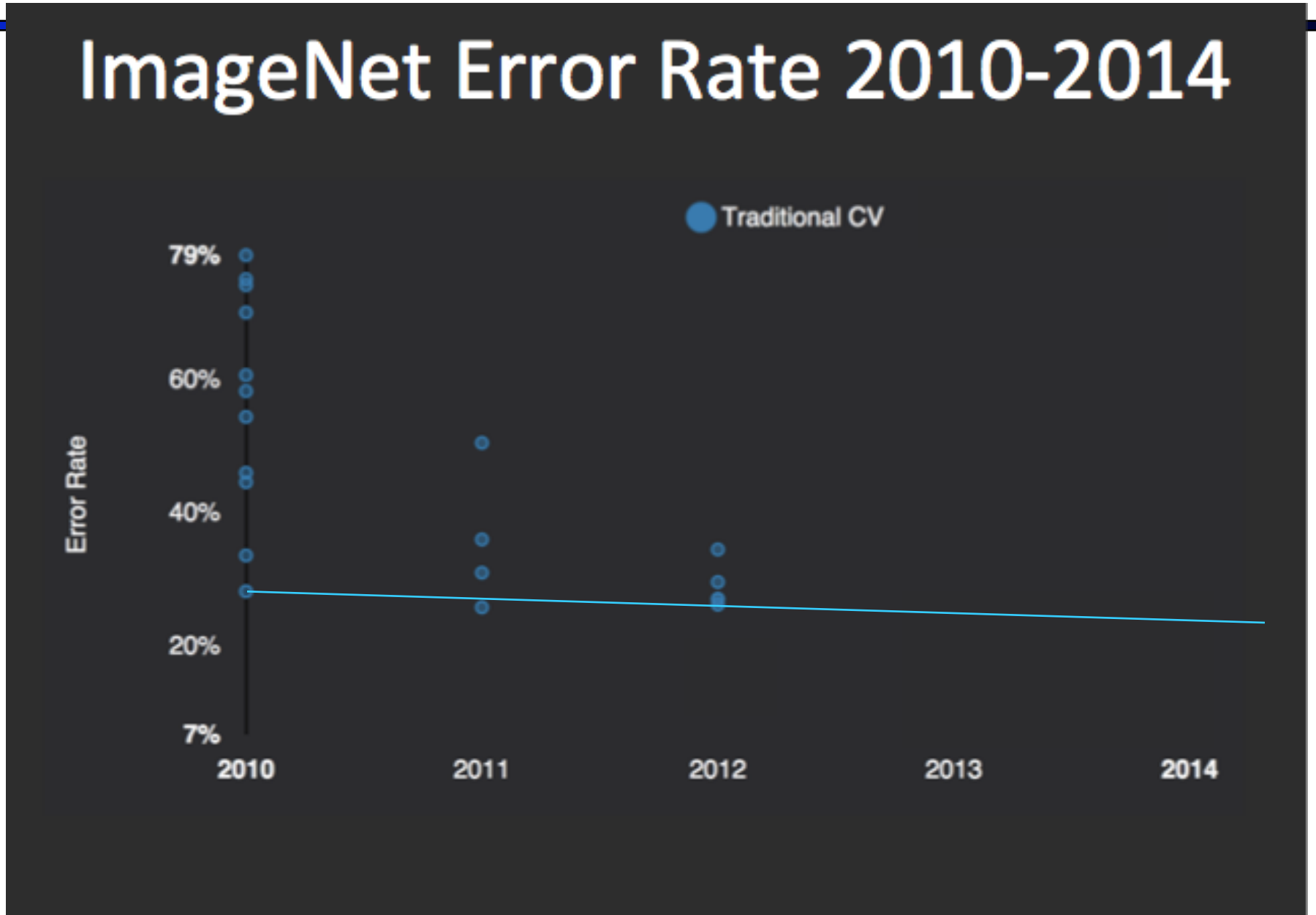


Performance



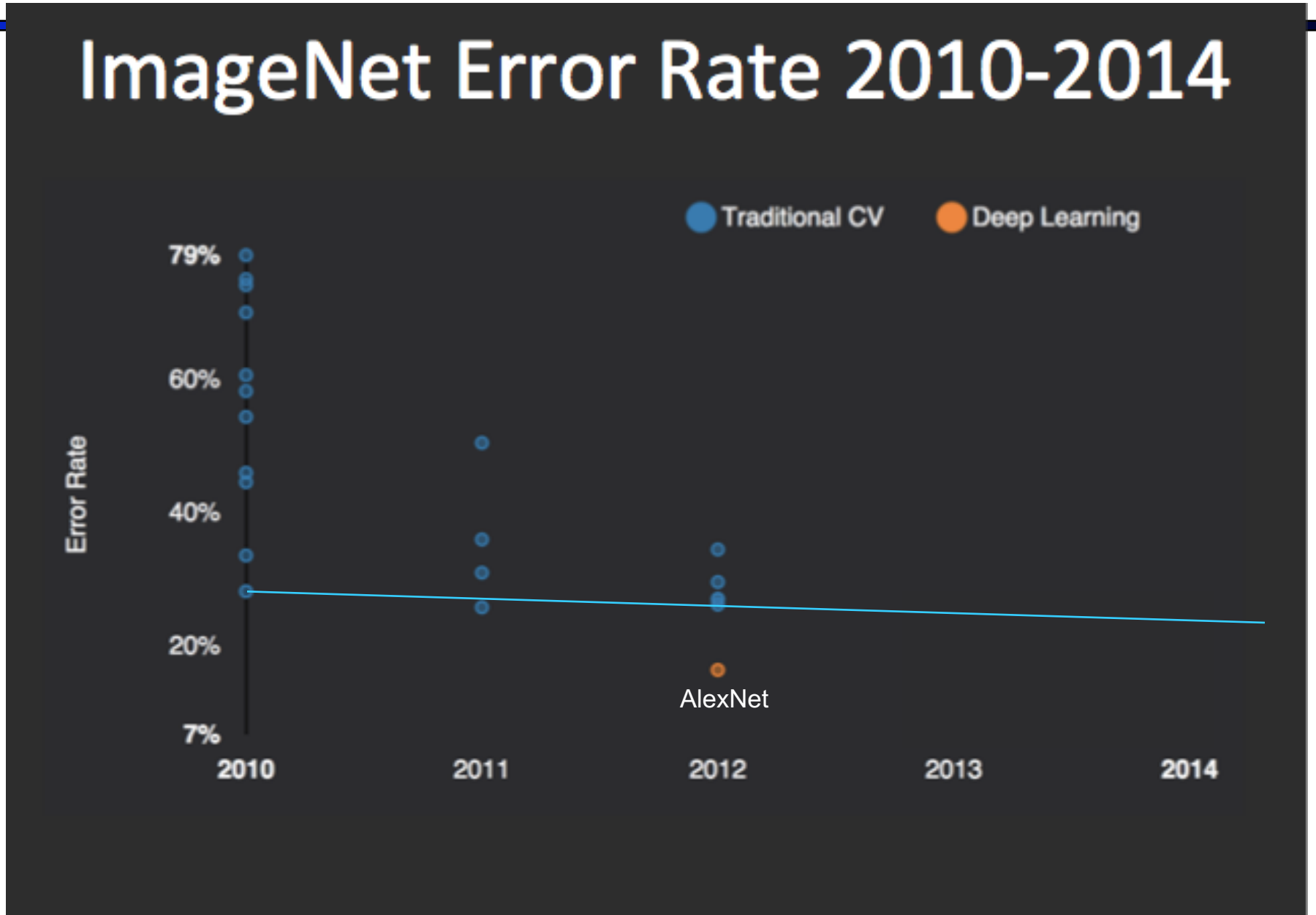
graph credit Matt Zeiler, Clarifai

Performance



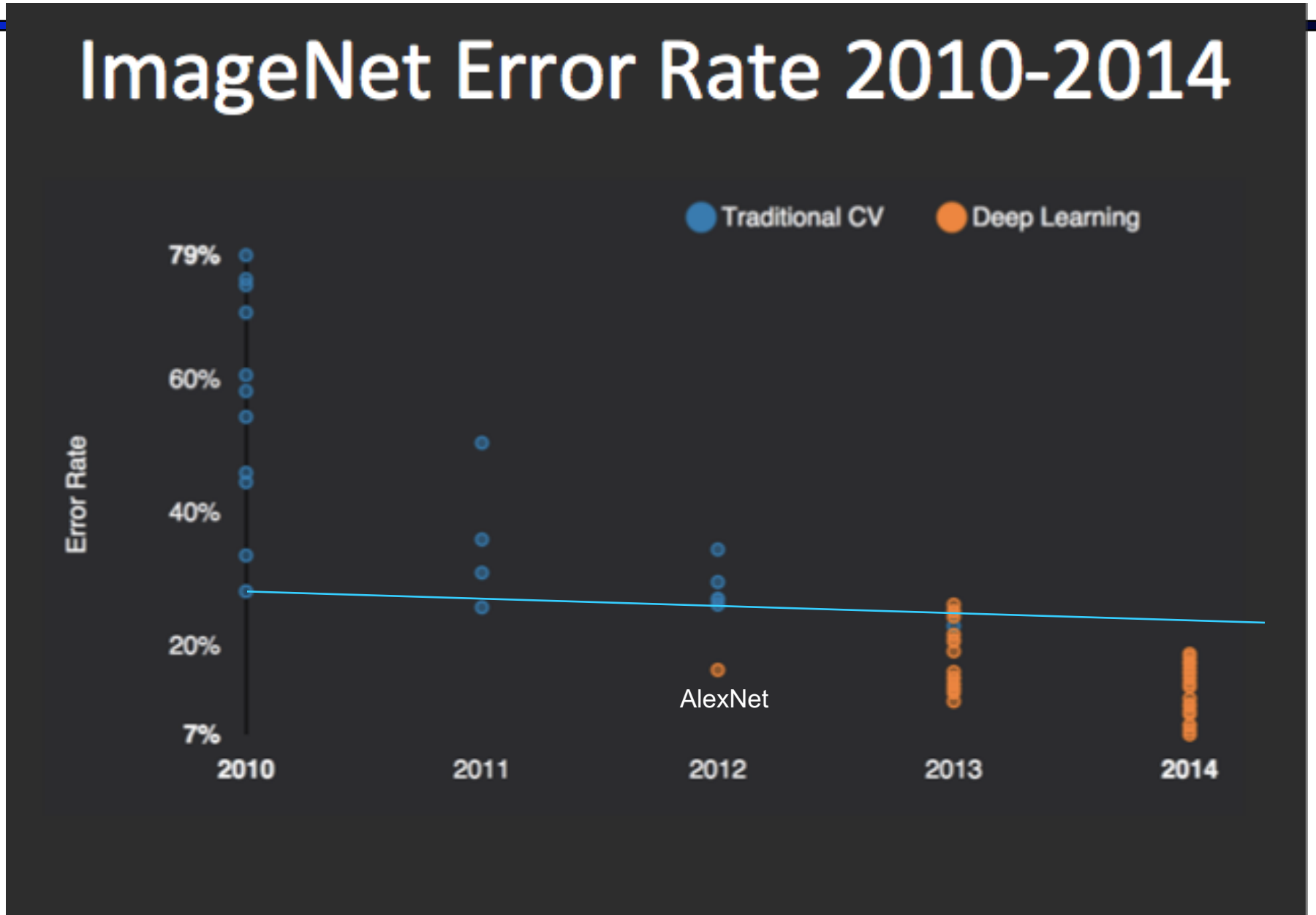
graph credit Matt Zeiler, Clarifai

Performance



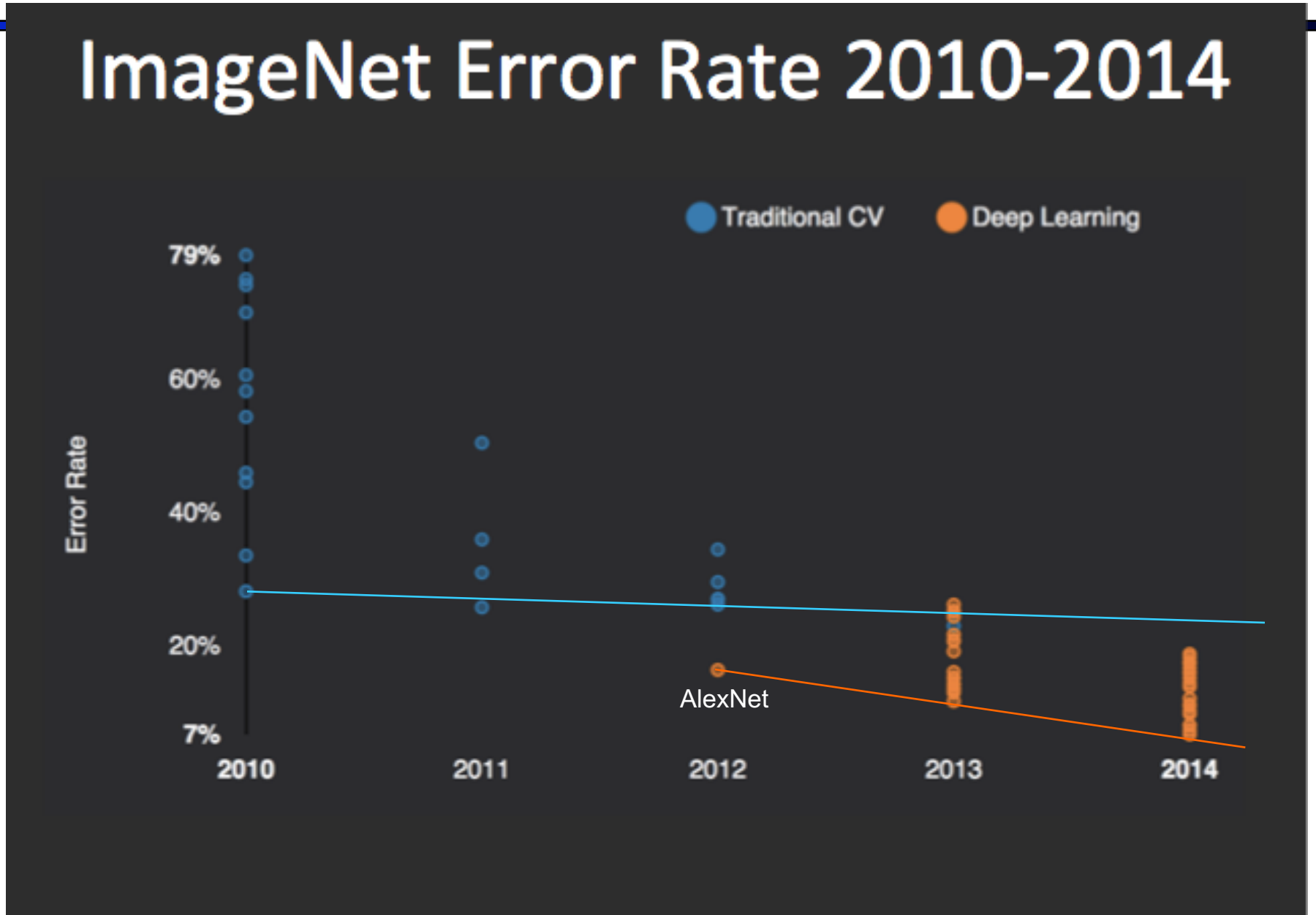
graph credit Matt Zeiler, Clarifai

Performance



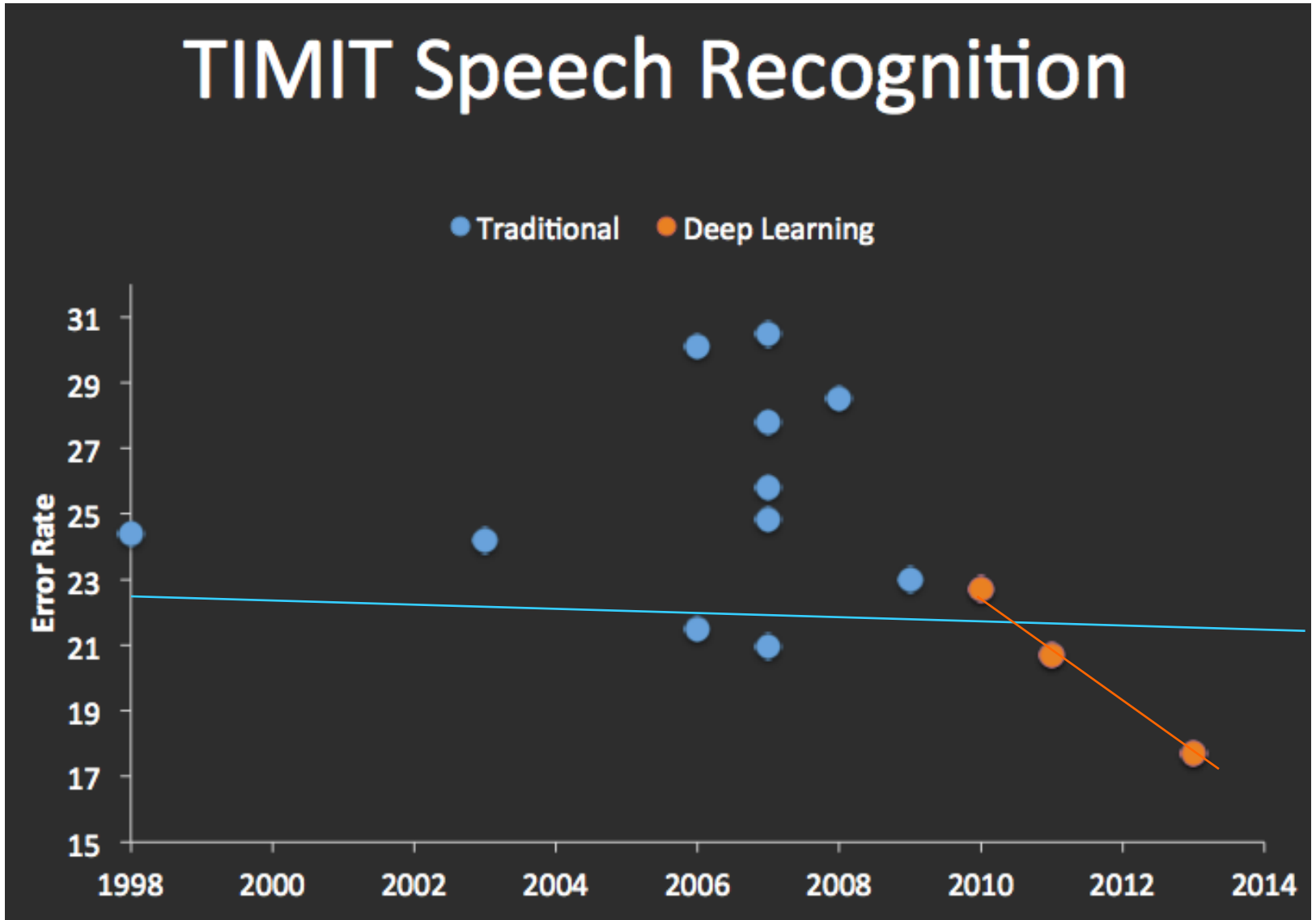
graph credit Matt Zeiler, Clarifai

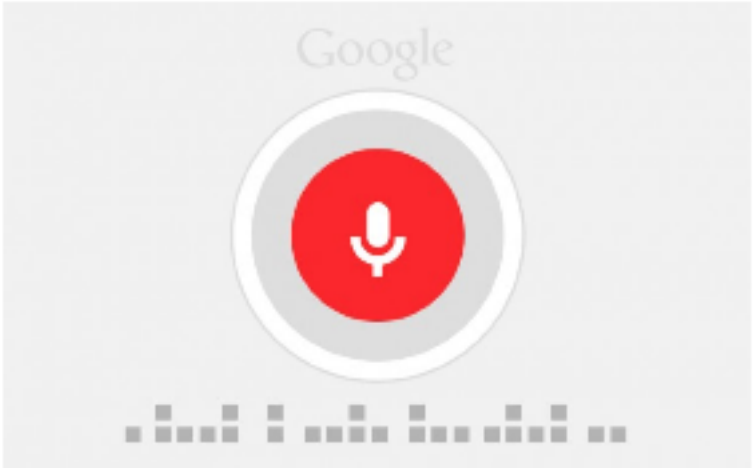
Performance



graph credit Matt Zeiler, Clarifai

Speech Recognition





graph credit Matt Zeiler, Clarifai

Speech Synthesis: WaveNet

- WaveNet (2016)
  - Approach for the “Text to Speech” problem (TTS).
  - Concatenative TTS: Short recorded speech fragments concatenated.
  - Parametric TTS: High level model based speech generation (from scratch).
  - WaveNet: “Dumb” neural network based model for generating human speech at the level of individual audio samples.
  - (Note: This is not a classifier! Not a direct application of today’s ideas, but instead an example of the generality of deep neural networks)
- https://deepmind.com/blog/wavenet-generative-model-raw-audio/

- Insert AI in the news here

How Many Computers to Identify a Cat?

HOME PAGE | TODAY'S PAPER | VIDEO | MOST POPULAR | U.S. Edition

Subscribe: Digital / Home Delivery | Log In | Register Now | Help

The New York Times — Business Day Technology

Search All NYTimes.com Go

Orange Savings Account℠

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION | ARTS | STYLE | TRAVEL | JOBS | REAL ESTATE | AUTOS

How Many Computers to Identify a Cat? 16,000

Jim Wilson/The New York Times

An image of a cat that a neural network taught itself to recognize.

By JOHN MARKOFF
Published: June 25, 2012

MOUNTAIN VIEW, Calif. — Inside Google's secretive X laboratory, known for inventing self-driving cars and augmented reality glasses, a small group of researchers began working several years ago on a simulation of the human brain.

There Google scientists created one of the largest neural networks for machine learning by connecting 16,000 computer processors, which they turned loose on the Internet to learn on its own.

Presented with 10 million digital images found in YouTube videos, what did Google's brain do? What millions of humans do with YouTube: looked for cats.

FACEBOOK
TWITTER
GOOGLE+
E-MAIL
SHARE
PRINT
REPRINTS

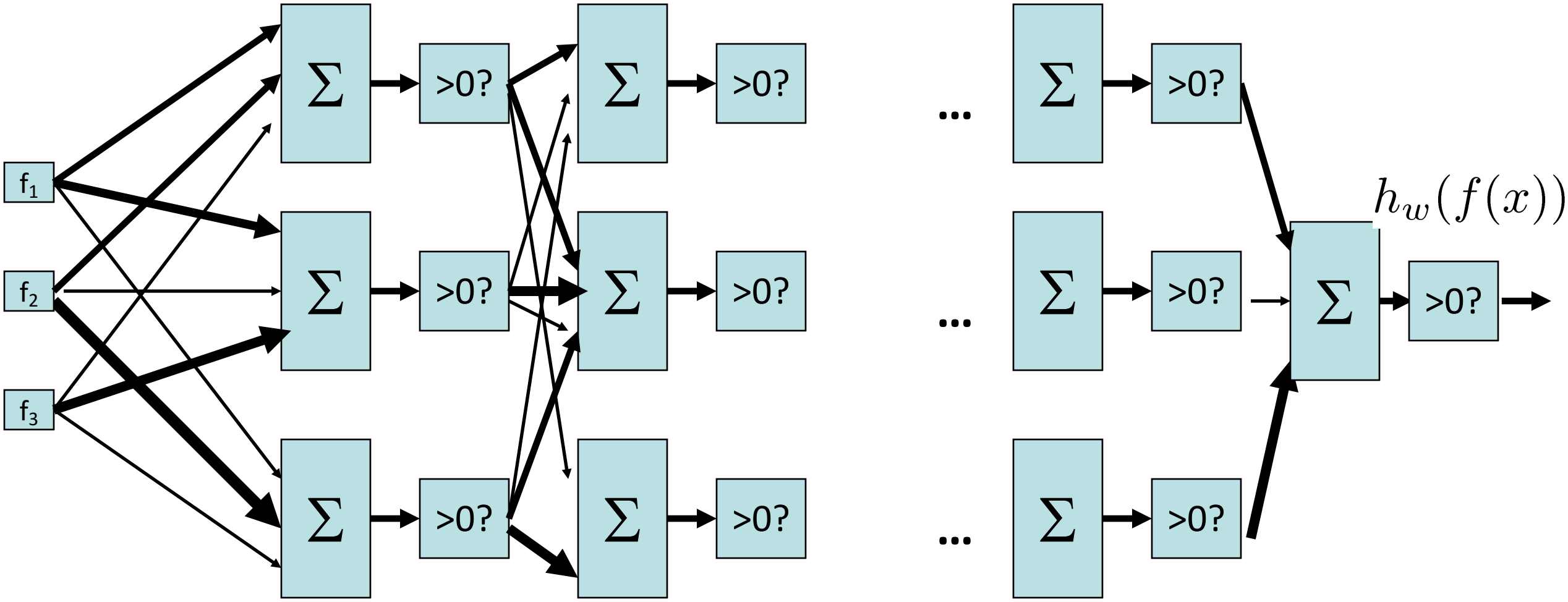
Multimedia

Business Day Live | Google's Brain

"Google Brain"

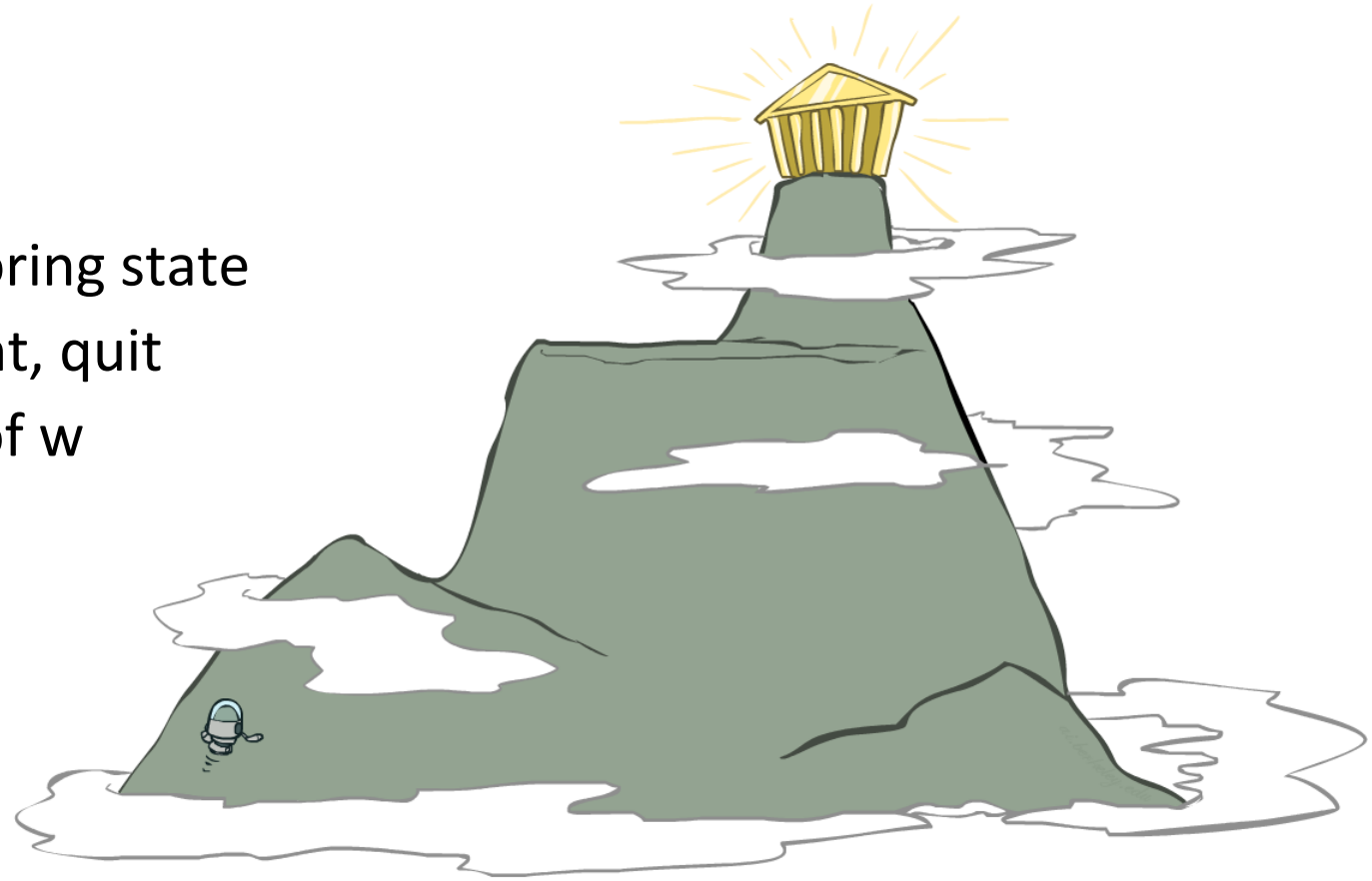
[Le, Ng, Dean, et al, 2012]

N-Layer Perceptron Network



Local Search

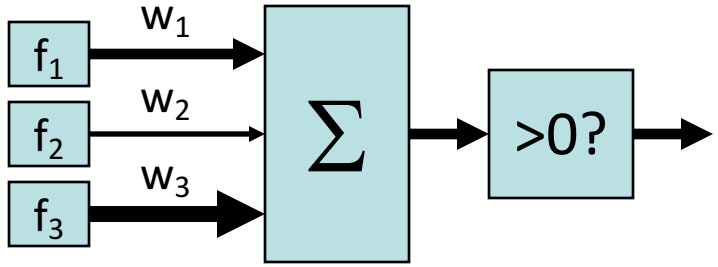
- Simple, general idea:
  - Start wherever
  - Repeat: move to the best neighboring state
  - If no neighbors better than current, quit
  - Neighbors = small perturbations of w
- Properties
  - Plateaus and local optima



→How to escape plateaus and find a good local optimum?

→How to deal with very large parameter vectors? E.g., $w \in \mathbb{R}^{1billion}$

Perceptron: Accuracy via Zero-One Loss



- Objective: Classification Accuracy

$$l^{\text{acc}}(w) = \frac{1}{m} \sum_{i=1}^{m} \left(\text{sign}(w^\top f(x^{(i)})) == y^{(i)} \right)$$

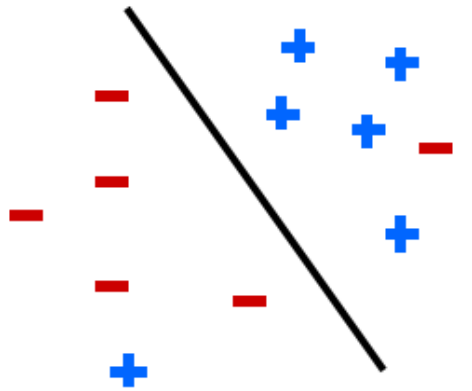
Note: $w^T f(x) = w \cdot f(x)$, both notations are common.

Perceptron: Accuracy via Zero-One Loss

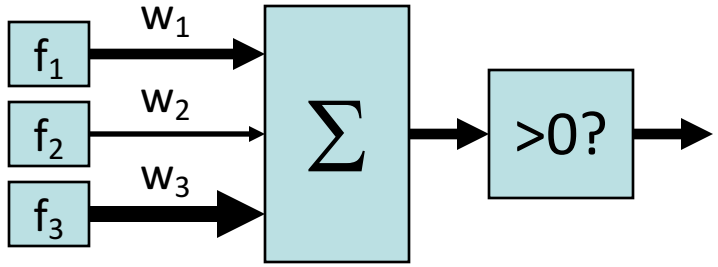
- Objective: Classification Accuracy

$$l^{\text{acc}}(w) = \frac{1}{m} \sum_{i=1}^{m} \left(\text{sign}(w^\top f(x^{(i)})) == y^{(i)} \right)$$

- For the weights, data, and correct labels below, what is l^{acc}?



Perceptron: Accuracy via Zero-One Loss



- Objective: Classification Accuracy

$$l^{\text{acc}}(w) = \frac{1}{m} \sum_{i=1}^{m} \left(\text{sign}(w^\top f(x^{(i)})) == y^{(i)} \right)$$

  - Issue: many plateaus → how to measure incremental progress?

Soft-Max

- Score for y=1: $w^\top f(x)$ Score for y=-1: $-w^\top f(x)$
- Probability of label:

$$p(y=1|f(x);w) = \frac{e^{w^\top f(x^{(i)})}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

$$p(y=-1|f(x);w) = \frac{e^{-w^\top f(x)}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

Soft-Max

- Score for y=1: $w^\top f(x)$ Score for y=-1: $-w^\top f(x)$
- Probability of label:

$$p(y=1|f(x);w) = \frac{e^{w^\top f(x^{(i)})}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

$$p(y=-1|f(x);w) = \frac{e^{-w^\top f(x)}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

- Suppose w = [-3, 4, 2] and x = [1, 1, 0]
  - What label will be selected if we score as usual?
  - What are the probabilities of the possible label assignments if we score using this new approach?

Soft-Max

- Score for y=1: $w^\top f(x)$ Score for y=-1: $-w^\top f(x)$
- Probability of label:

$$p(y=1|f(x);w) = \frac{e^{w^\top f(x^{(i)})}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

$$p(y=-1|f(x);w) = \frac{e^{-w^\top f(x)}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

- Suppose w = [-3, 4, 2] and x = [1, 2, 0]
  - What label will be selected if we score as usual?
    - Label “1”, since -3 * 1 + 4 * 2 = 5
  - What are the probabilities of the possible label assignments if we score using this new approach?
    - Score for “1” = 5, Score for “-1”: -5
    - $p(y=1) = e^5/(e^5+e^{-5})$, $p(y=-1) = e^{-5}/(e^5+e^{-5})$
    - 0.9999546 and 0.0000453, respectively

Soft-Max

- Score for y=1: $w^\top f(x)$ Score for y=-1: $-w^\top f(x)$
- Probability of label:

$$p(y=1|f(x);w) = \frac{e^{w^\top f(x^{(i)})}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

$$p(y=-1|f(x);w) = \frac{e^{-w^\top f(x)}}{e^{w^\top f(x)} + e^{-w^\top f(x)}}$$

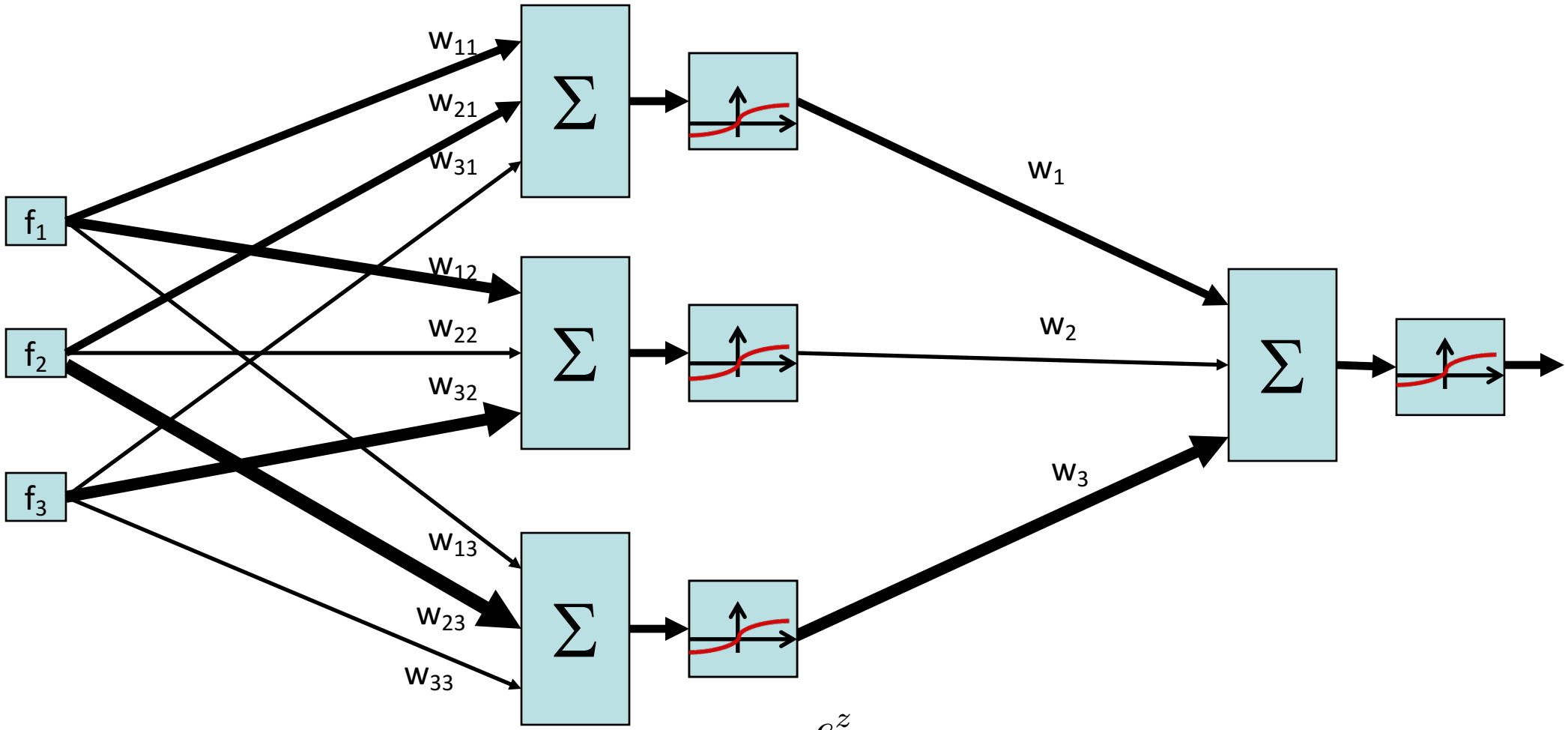
- Objective:

$$l(w) = \prod_{i=1}^{m} p(y=y^{(i)}|f(x^{(i)});w)$$

- Log:

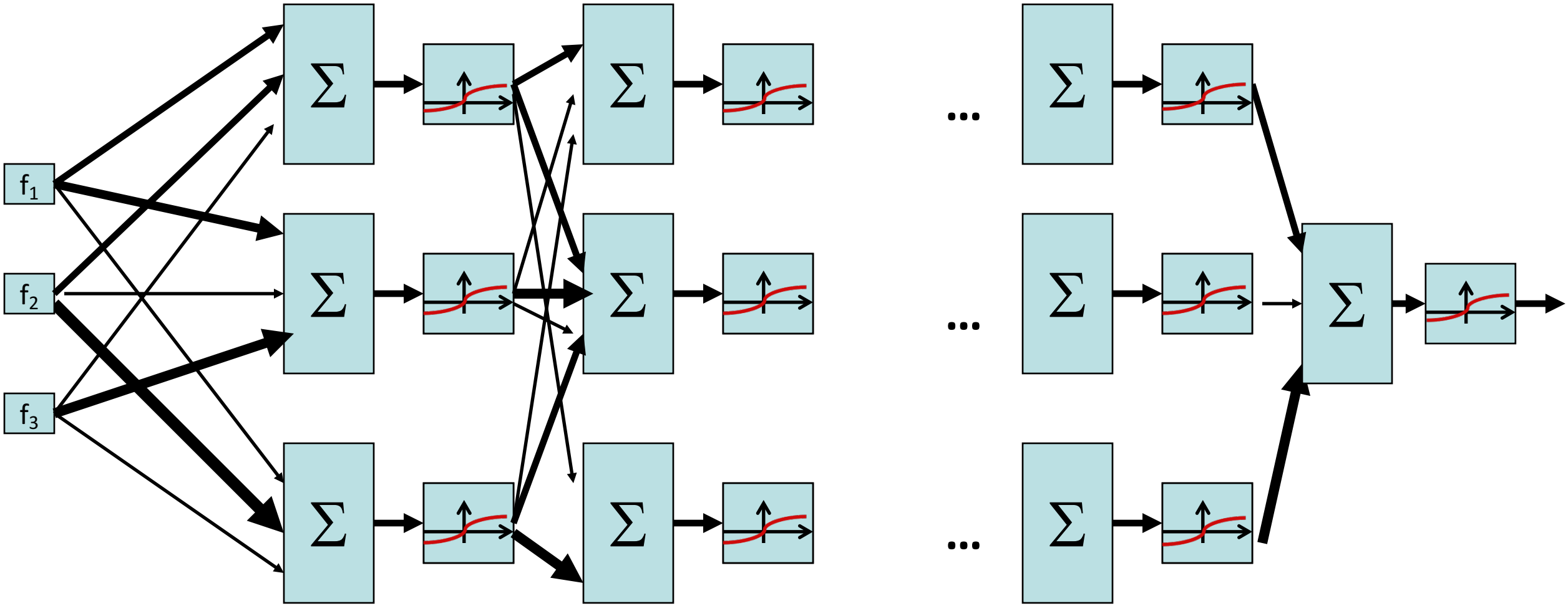
$$ll(w) = \sum_{i=1}^{m} \log p(y=y^{(i)}|f(x^{(i)});w)$$

Two-Layer Neural Network



$$z \rightarrow \frac{e^z}{e^z + e^{-z}}$$

N-Layer Neural Network



Our Status

- Our objective $ll(w)$
  - Changes smoothly with changes in w
  - Doesn't suffer from the same plateaus as the perceptron network

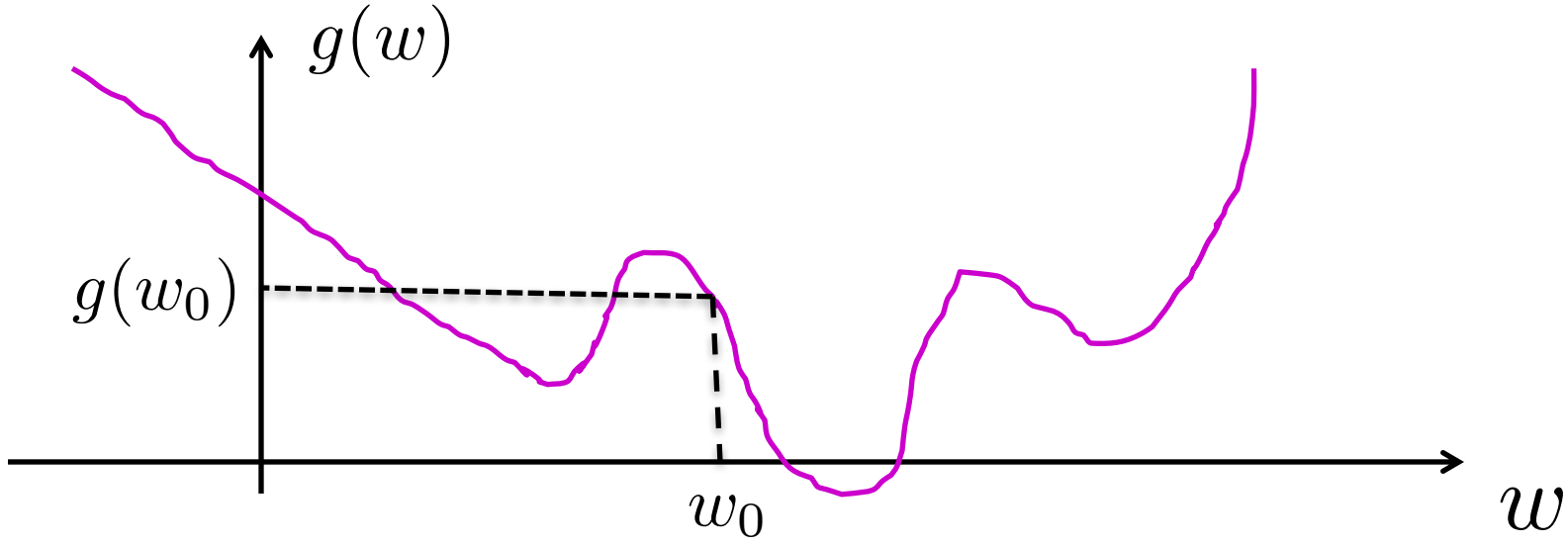
- Challenge: how to find a good w ?

$$\max_w ll(w)$$

- Equivalently:

$$\min_w -ll(w)$$

1-d optimization

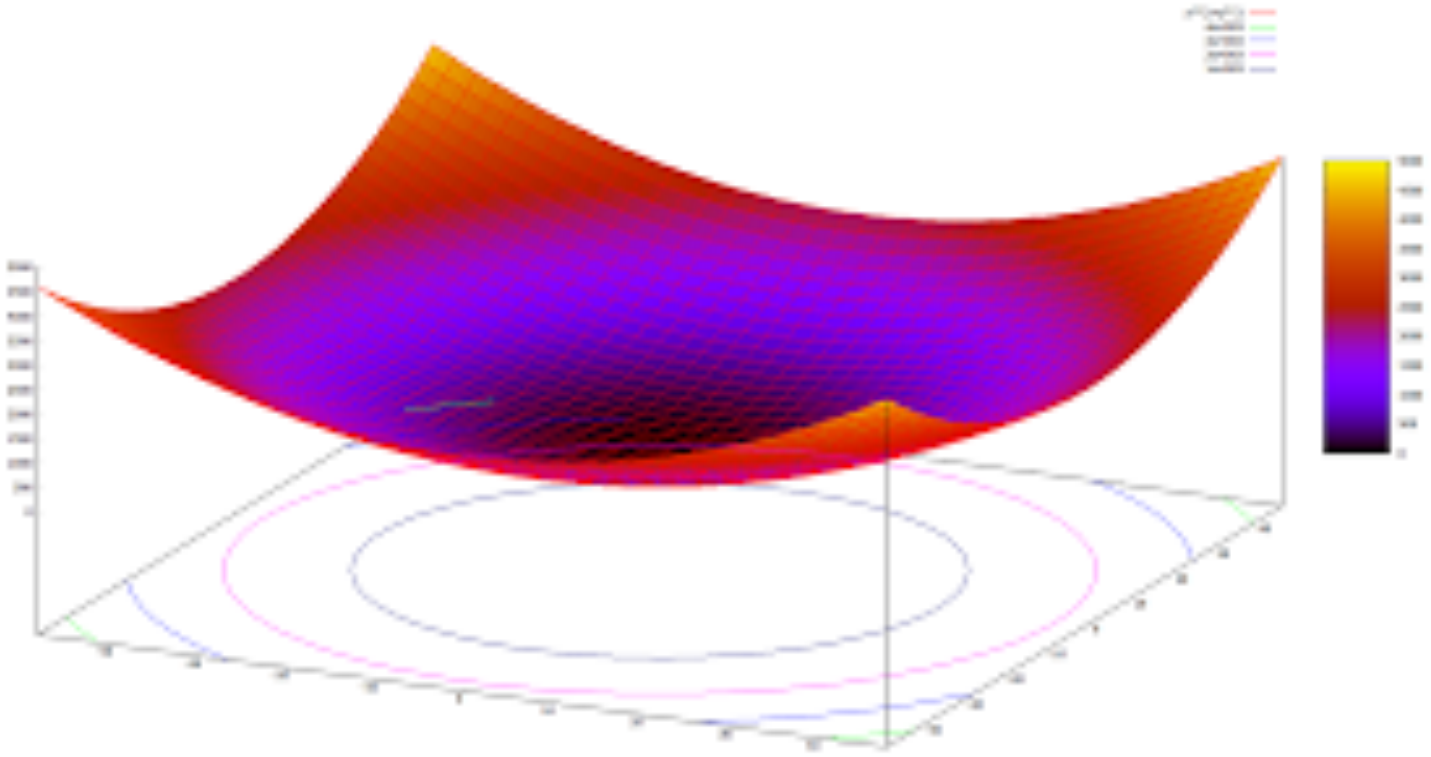


- Could evaluate $g(w_0 + h)$ and $g(w_0 - h)$
  - Then step in best direction
- Or, evaluate derivative:

$$\frac{\partial g(w_0)}{\partial w} = \lim_{h \to 0} \frac{g(w_0 + h) - g(w_0 - h)}{2h}$$

  - Which tells which direction to step into

2-D Optimization



Source: Thomas Jungblut's Blog

Steepest Descent

- Idea:
  - Start somewhere
  - Repeat: Take a step in the steepest descent direction

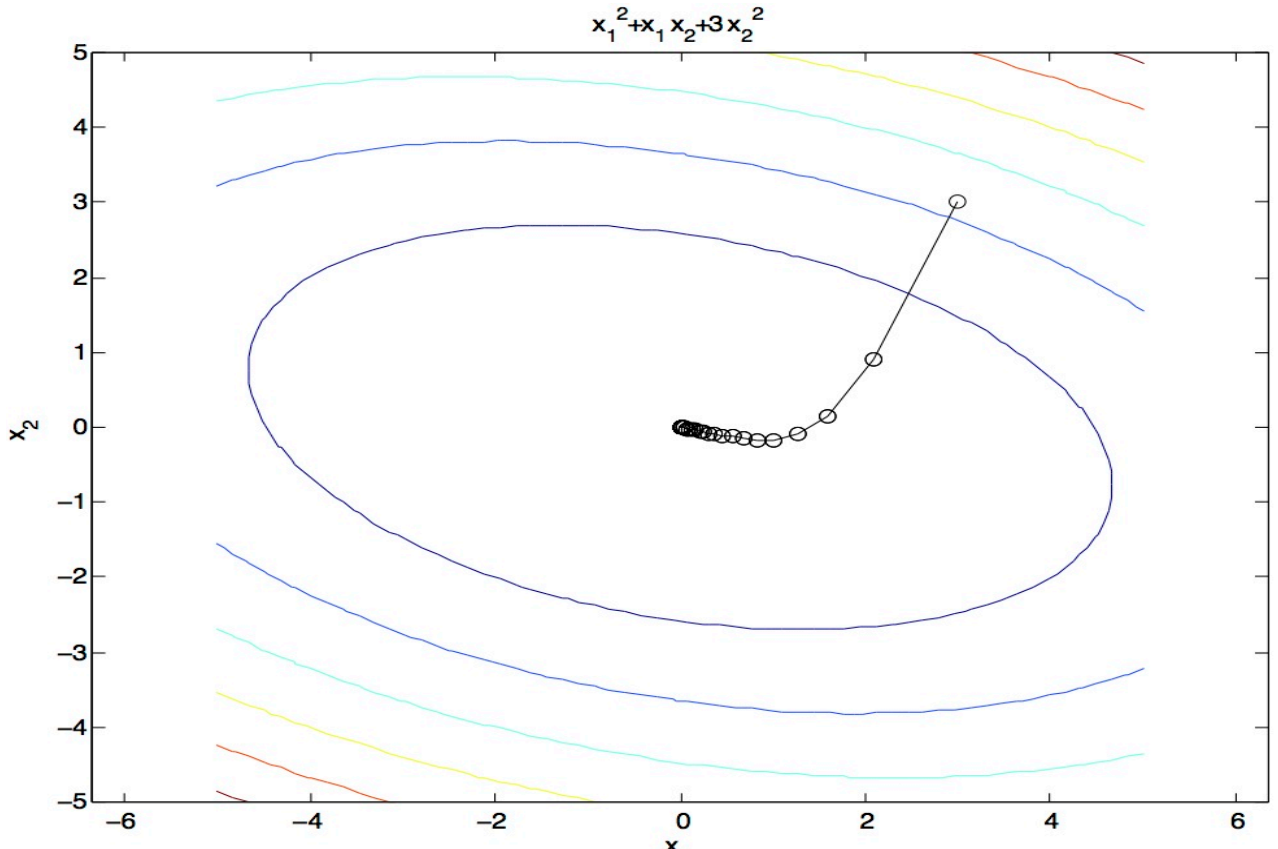


Figure source: Mathworks

What is the Steepest Descent Direction?

$$\min_{\Delta:\Delta_1^2+\Delta_2^2\leq\epsilon} g(w+\Delta)$$

- First-Order Taylor Expansion: $g(w+\Delta) \approx g(w) + \frac{\partial g}{\partial w_1}\Delta_1 + \frac{\partial g}{\partial w_2}\Delta_2$
- Steepest Descent Direction: $\min_{\Delta:\Delta_1^2+\Delta_2^2\leq\epsilon} \frac{\partial g}{\partial w_1}\Delta_1 + \frac{\partial g}{\partial w_2}\Delta_2$
- Recall: $\min_{a:\|a\|\leq\epsilon} a^\top b \quad \rightarrow \quad a = -b\frac{\epsilon}{\|b\|}$
- Hence, solution: $-\nabla g \frac{\epsilon}{\|\nabla g\|}$

$$\nabla g = \begin{bmatrix} \frac{\partial g}{\partial w_1} \\ \frac{\partial g}{\partial w_2} \end{bmatrix}$$

Generally, Steepest Direction

- Steepest Direction = direction of the gradient

$$\nabla g = \begin{bmatrix} \frac{\partial g}{\partial w_1} \\ \frac{\partial g}{\partial w_2} \\ \cdots \\ \frac{\partial g}{\partial w_n} \end{bmatrix}$$

Optimization Procedure 1: Gradient Descent

- Init: w
- For i = 1, 2, …

$$w \leftarrow w - \alpha * \nabla g(w)$$

$$g(w) = ll(w)$$

$$ll(w) = \sum_{i=1}^{m} \log p(y = y^{(i)} | f(x^{(i)}); w)$$

- α: learning rate --- tweaking parameter that needs to be chosen carefully
- How? Try multiple choices
  - Crude rule of thumb: update should change w by about 0.1 – 1 %
- Important detail we haven't discussed (yet): How do we compute $\nabla g(w)$?

Try Many Learning Rates

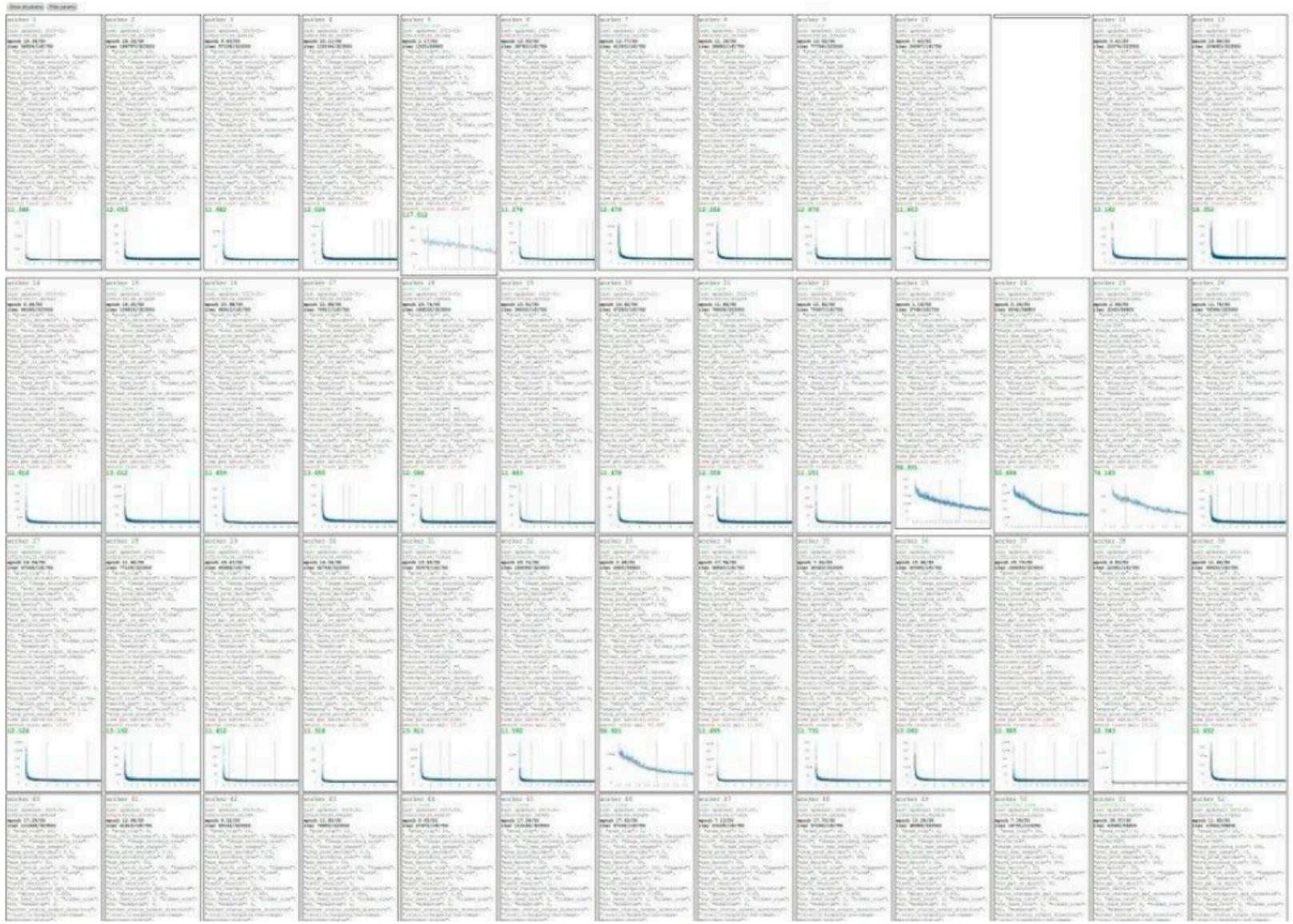


Figure source:
Andrej Karpathy

Neural Network Playground

- Let’s see some Neural Networks in action: (Link)
  - Note: ”Perceptrons” in this example use tanh activation function, not thresholded to zero or one.