CPSC 340: Machine Learning and Data Mining

Deep Learning

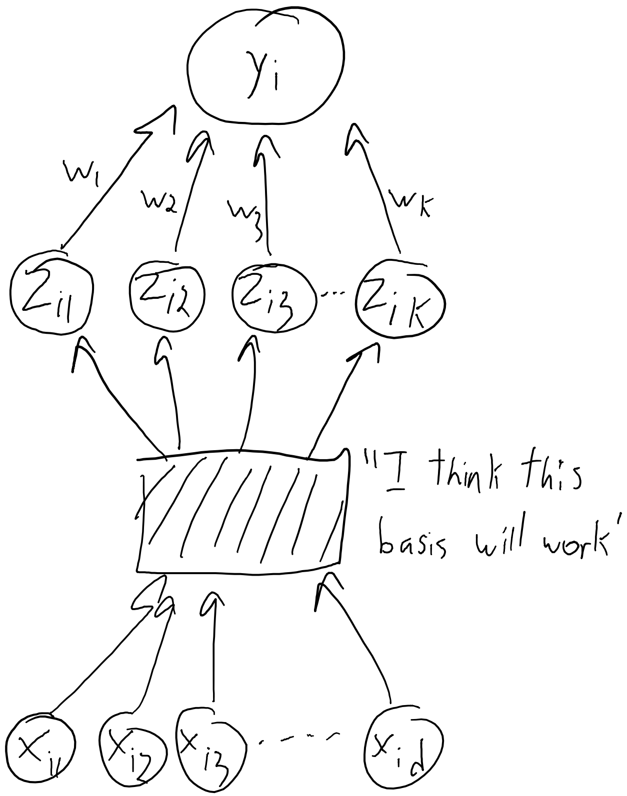
Fall 2015

Admin

- Assignment 4 due now.
- Midterm
  - After class pick up remaining/remarked midterms.
  - Missing cheat sheet: did someone grab one when returning midterms?
- Office hours on Tuesday of next week will be in ICICS 146.
- Assignment 5:
  - First two questions put on Piazza Saturday, full assignment on Monday.
  - Material to review for Monday tutorials:
    - NMF for Eigenfaces with alternating minimization.
    - Collaborative filtering for recommender systems with regularized SVD.
  - The TAs will put together a 'tutorial summary' document.

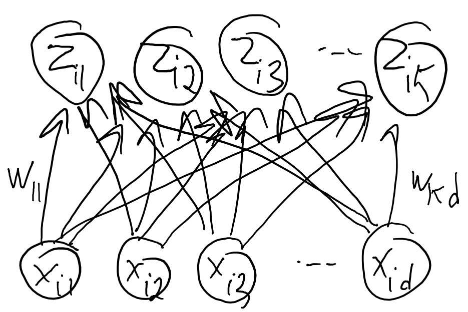
Last Time: Neural Networks

Hand-engineered features:

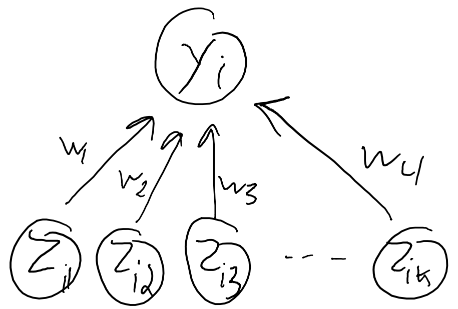


Requires domain knowledge
Time-consuming.

Learn latent-factor model:

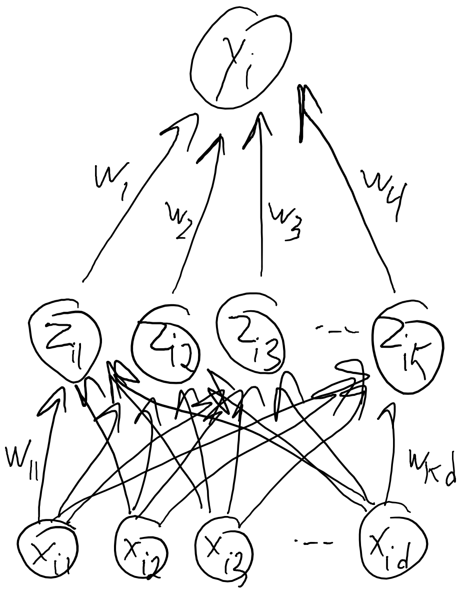


Use latent representation as features:



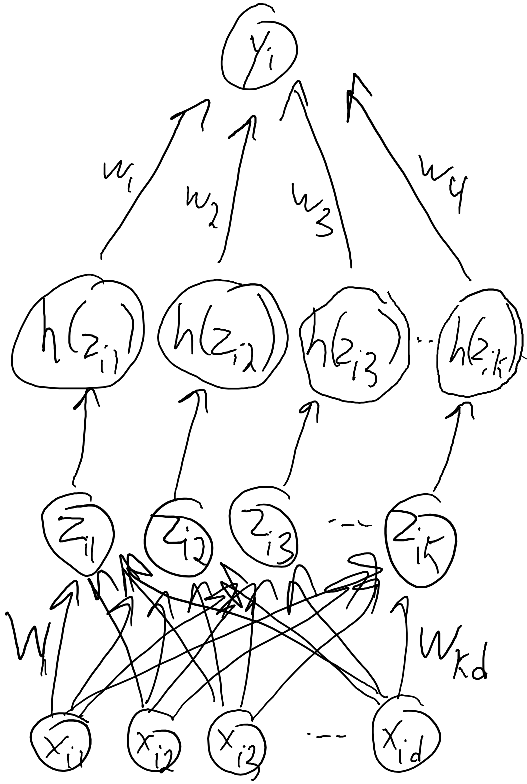
Good representation of x_i might be bad for predicting y_i

Learn 'w' and 'W' together:



Still a linear model.

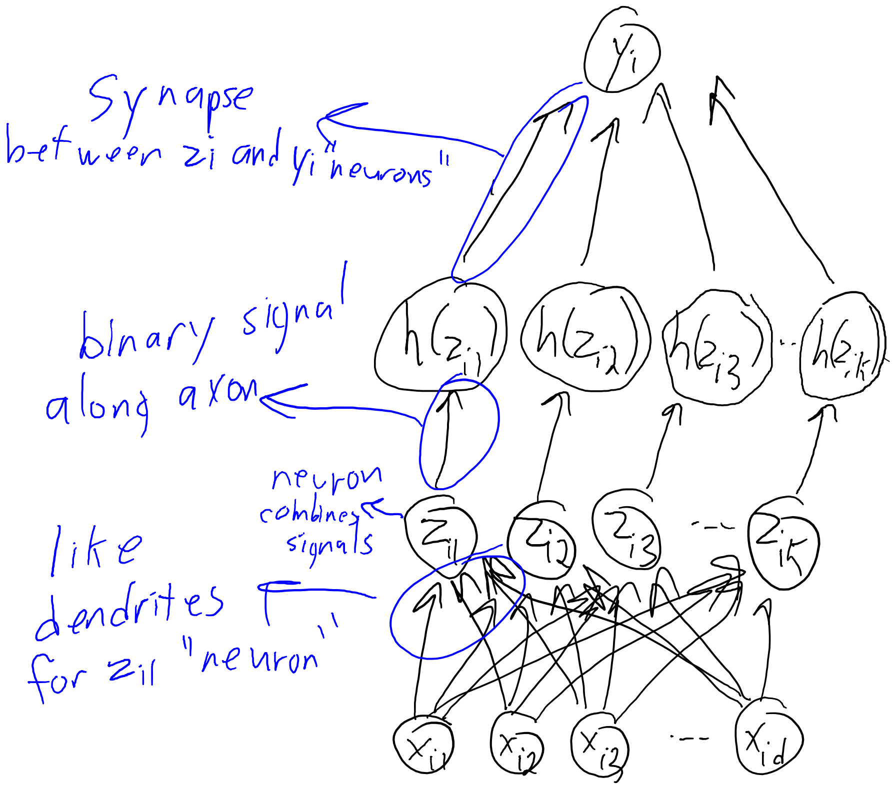
Neural networks:



Extra non-linear transformation of z_i values.

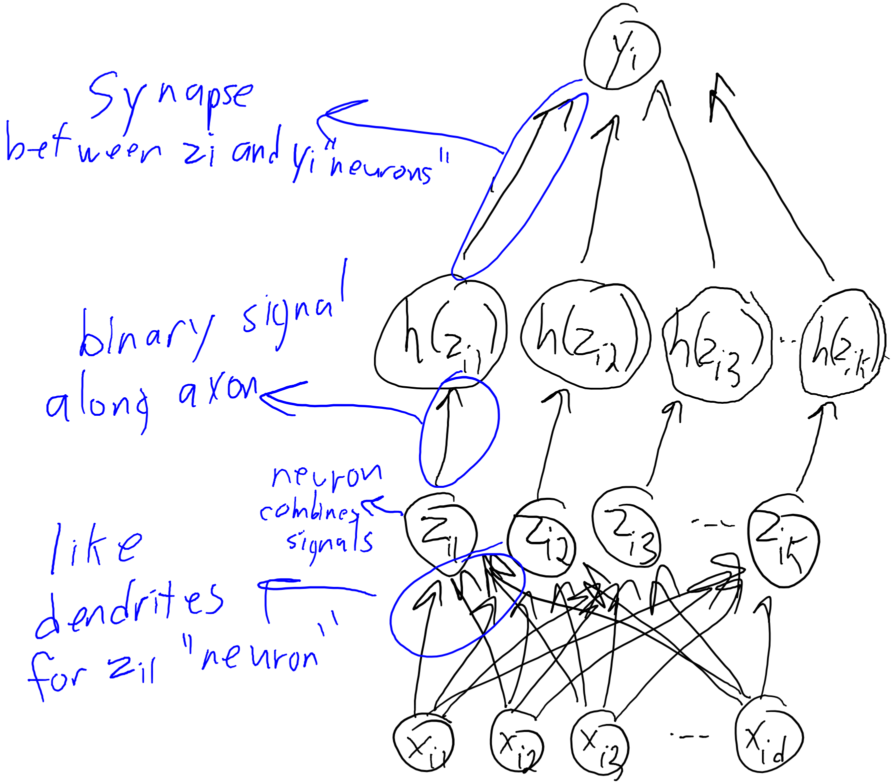
Neural Networks and Deep Learning

Neural networks:

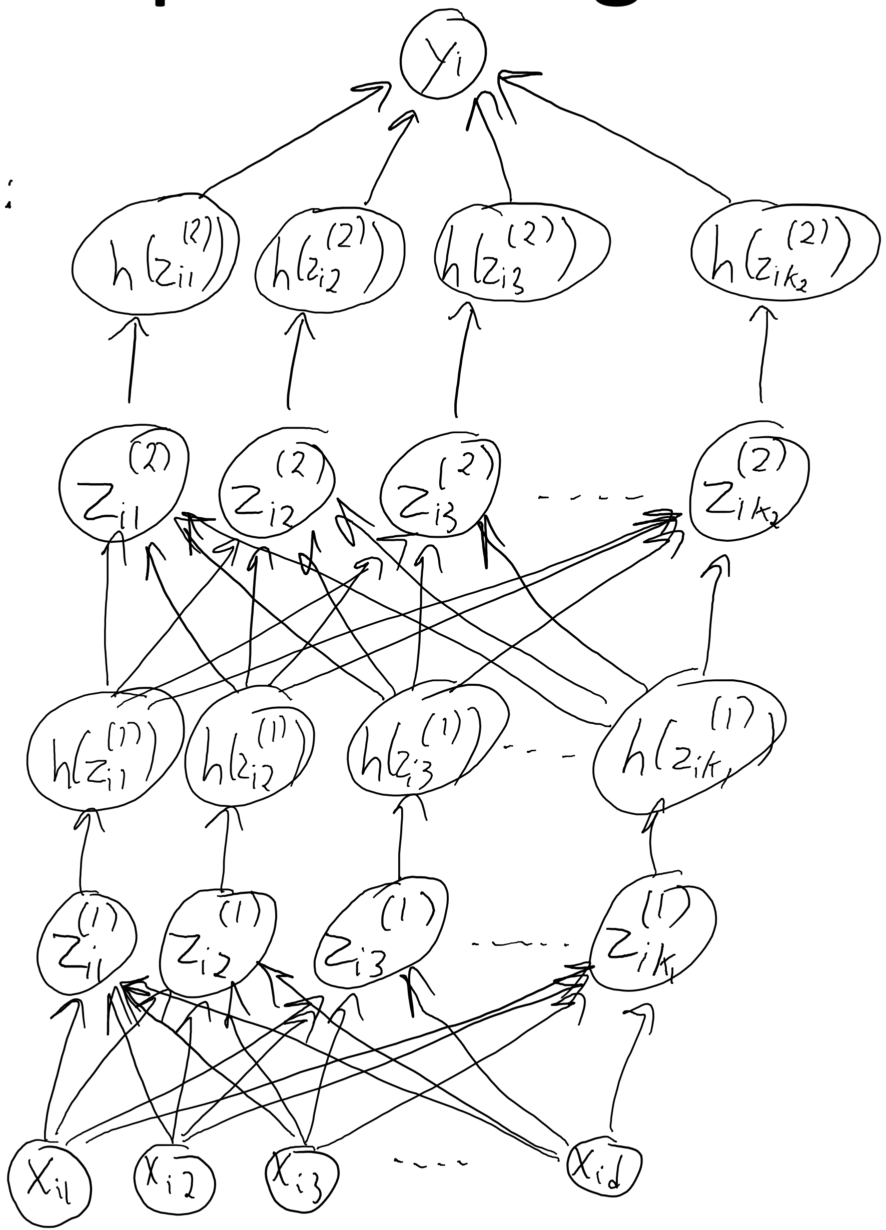


Neural Networks and Deep Learning

Neural networks:



Deep learning:



Neural Networks and Deep Learning

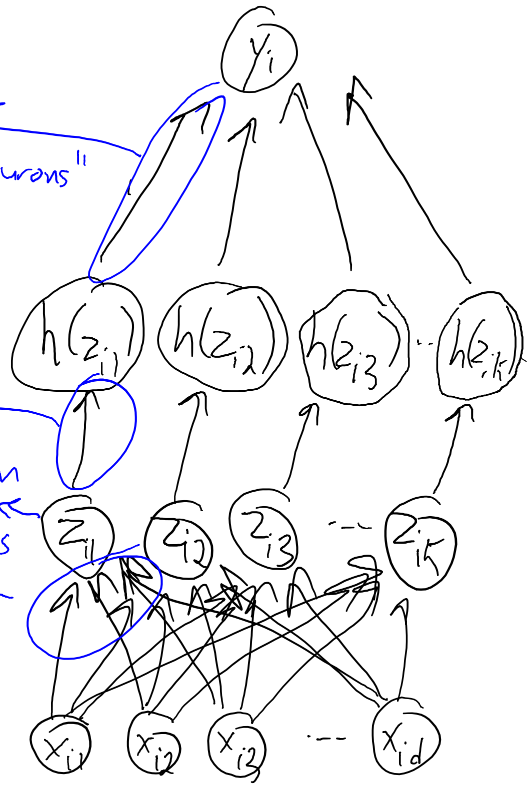
Neural networks:

Synapse between zi and yi "neurons"

binary signal along axon

neuron combines signals

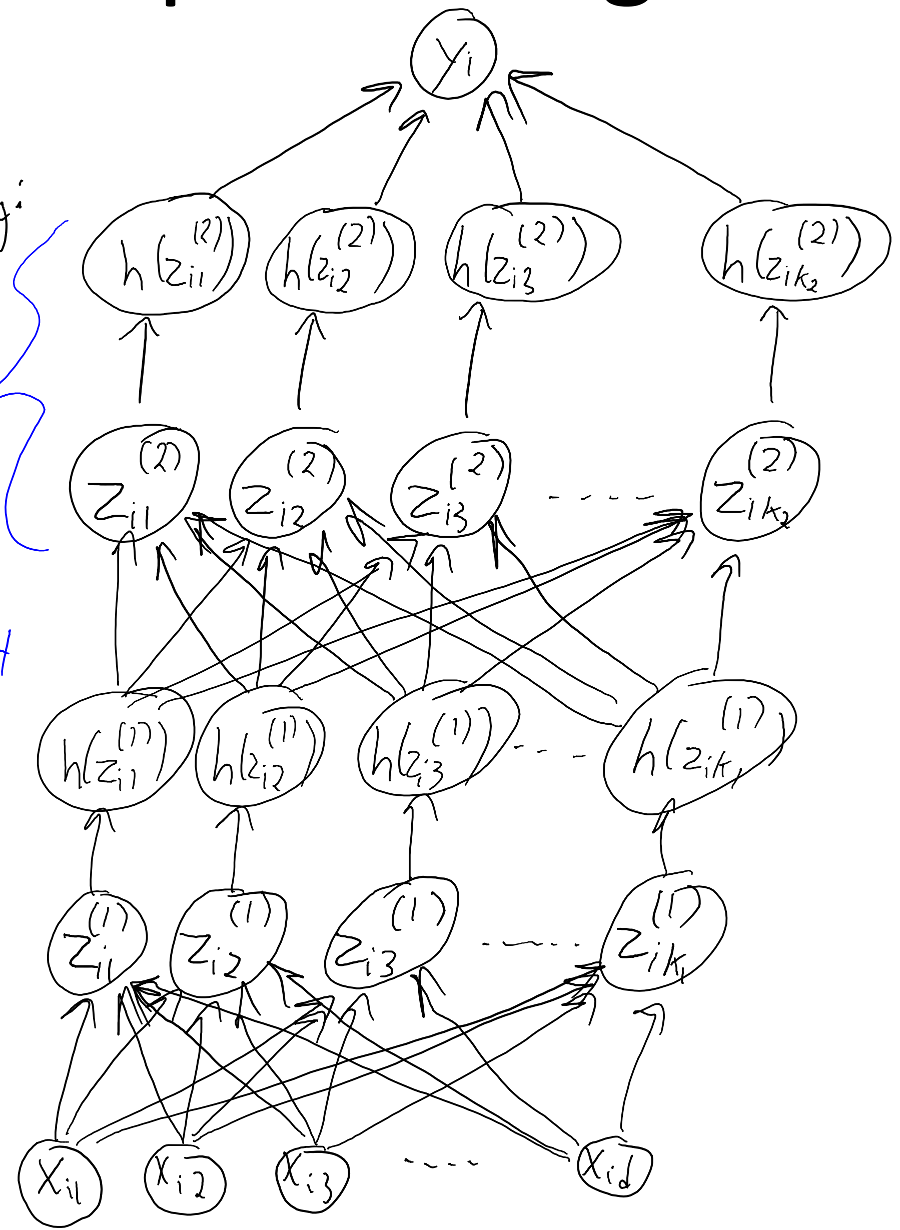
like dendrites for zi1 "neuron"



Deep learning:

we can repeat this many times.

compute gradient using backpropagation (i.e., chain rule).



Neural Networks and Deep Learning

Linear model:

$$\hat{y}_i = w^T x_i$$

Single-layer neural network:

$$\hat{y}_i = w^T h(W x_i)$$

"Deeper" neural network:

$$\hat{y}_i = w^T h(W_{(2)} h(W_{(1)} x_i))$$

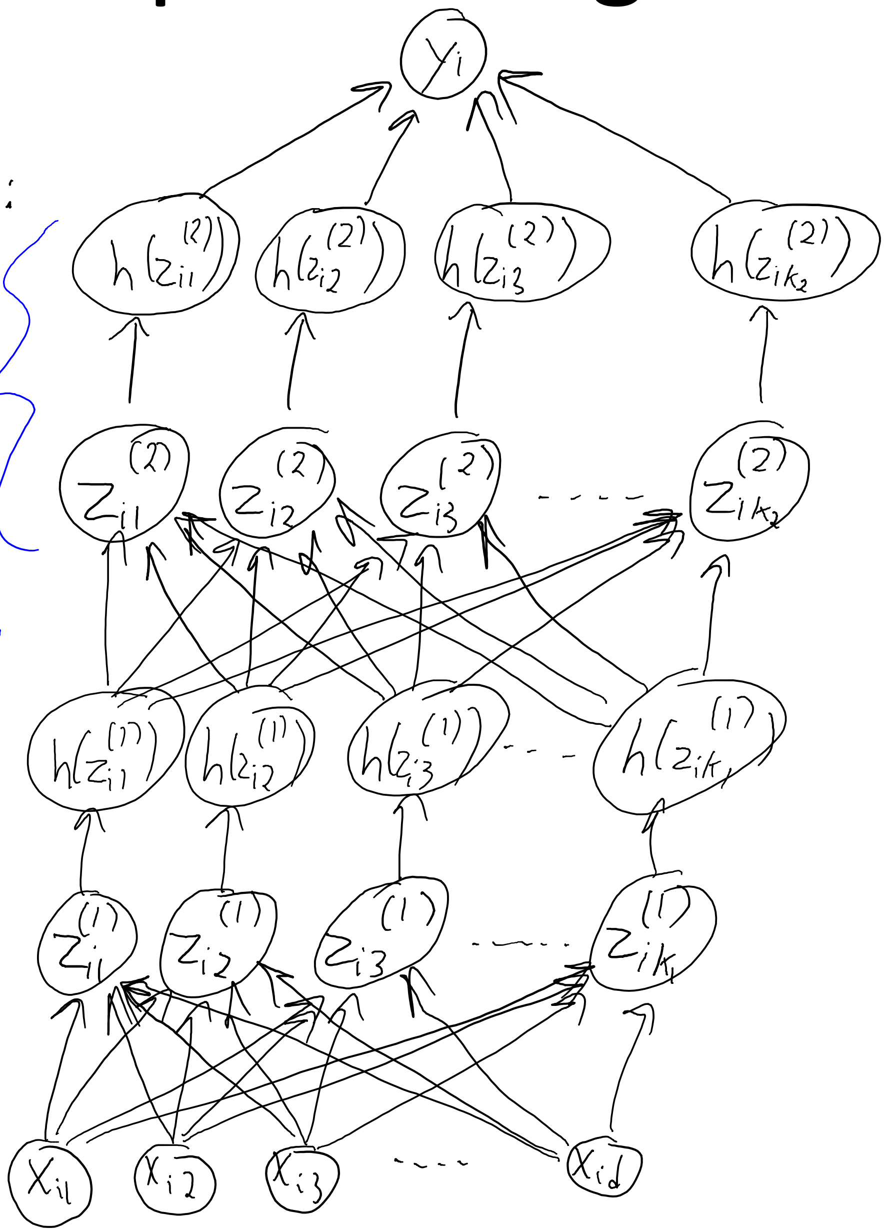
Even "deeper":

$$\hat{y}_i = w^T h(W_{(3)} h(W_{(2)} h(W_{(1)} x_i)))$$

Deep learning:

we can repeat this many times.

compute gradient using backpropagation (i.e., chain rule).



Digression: Bias Variables

Linear model:

$$\hat{y}_i = w^T \underline{x_i}$$

don't need bias if x_i includes a variable that is always 1.

Single-layer neural network:

$$\hat{y}_i = w^T h(W x_i) \underline{+ \beta}$$

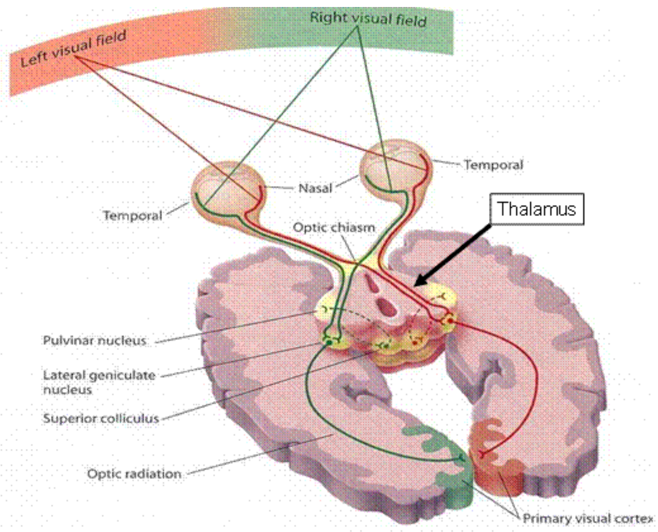
you can have an explicit bias β, or if h is sigmoid then fix one column of W to zeroes.

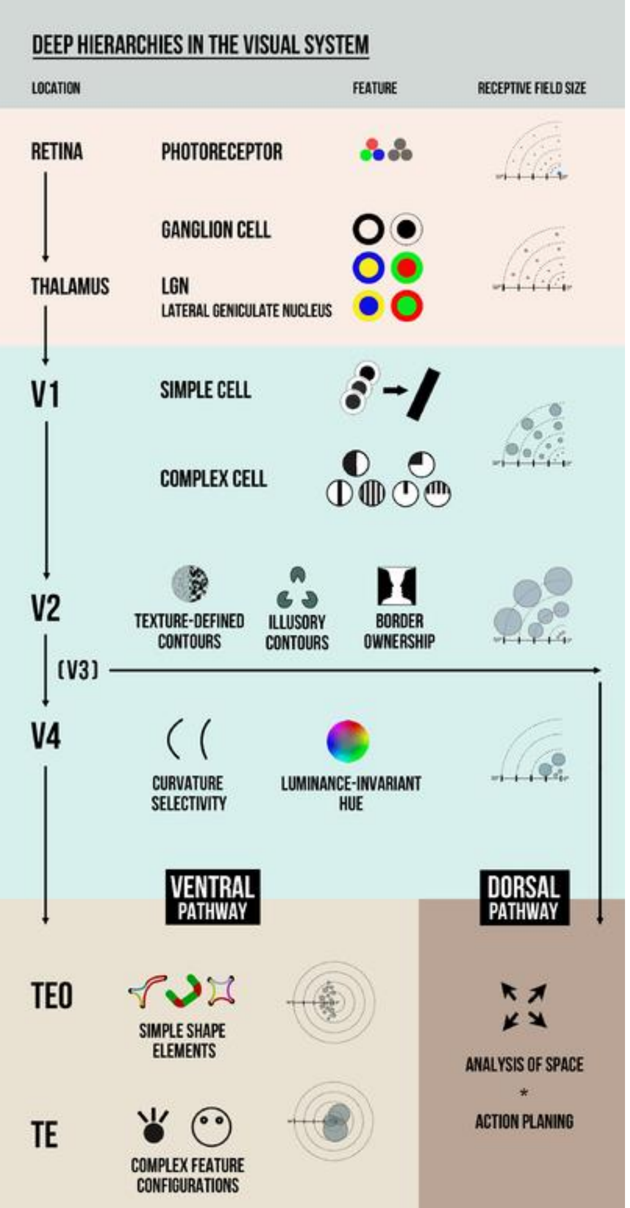
"Deeper" neural network:

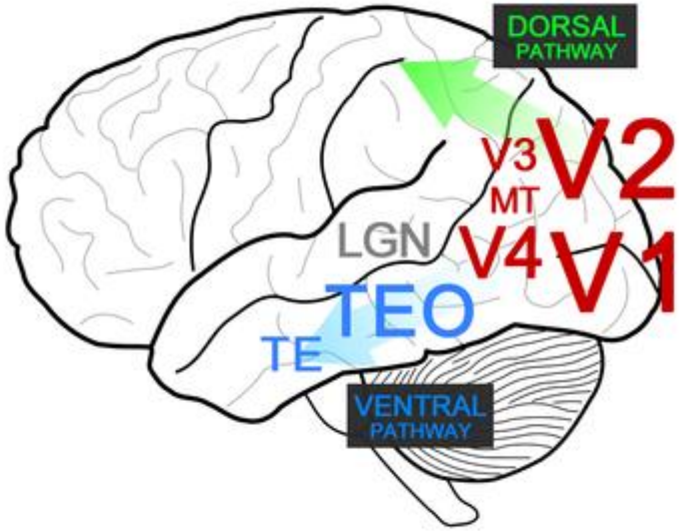
$$\hat{y}_i = w^T h(W_{(2)} h(W_{(1)} x_i) + b_{(2)}) + \beta$$

bias within layer

Biological Motivation for Deep Learning

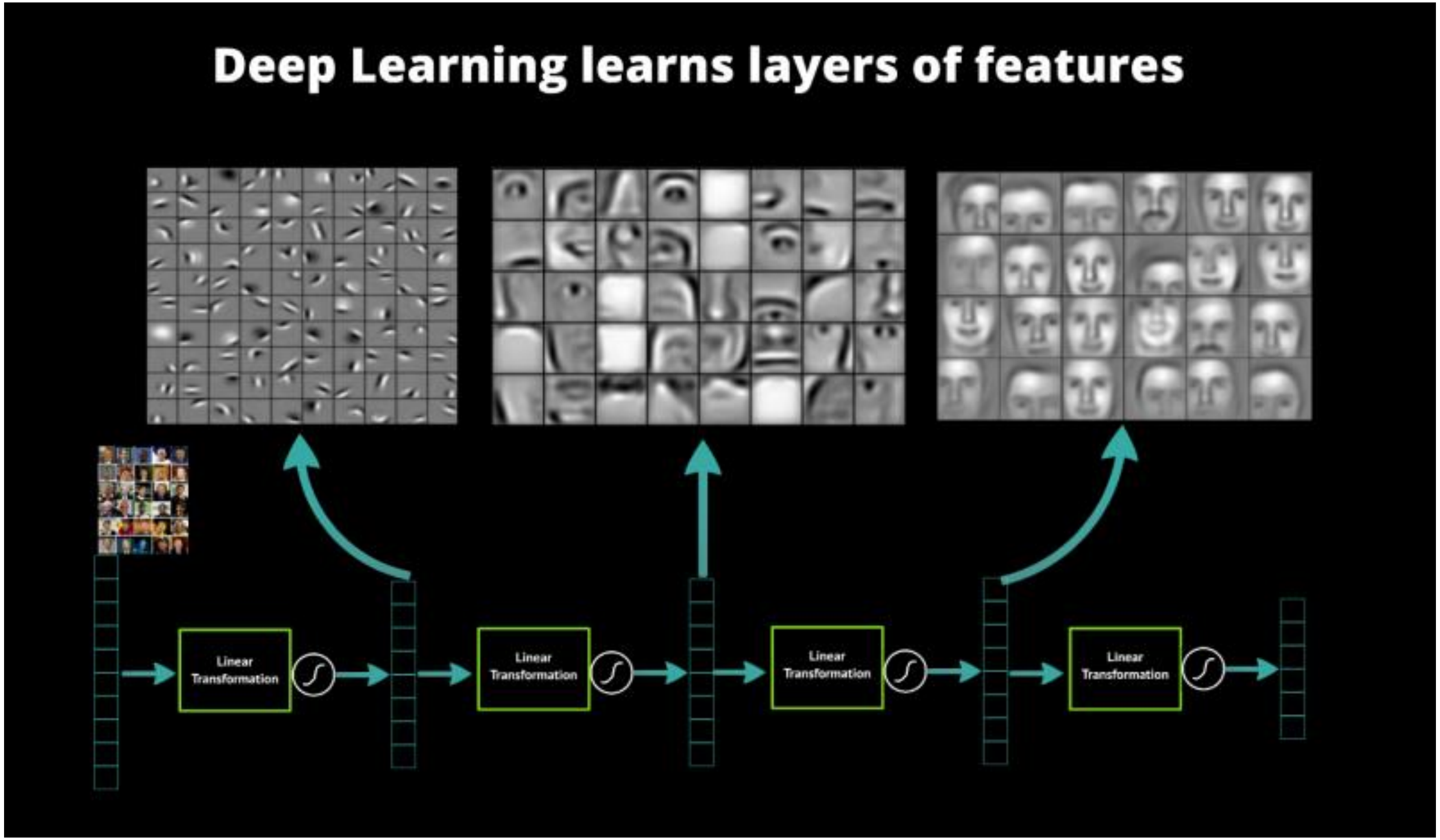






http://www.strokenetwork.org/newsletter/articles/vision.htm
https://en.wikibooks.org/wiki/Sensory_Systems/Visual_Signal_Processing

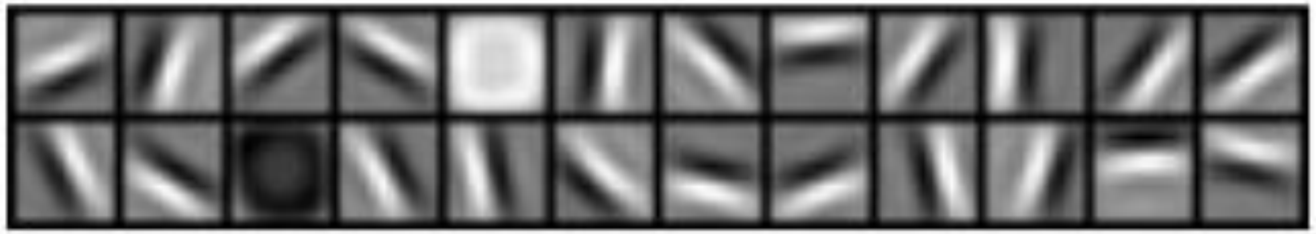
Cool Picture Motivation for Deep Learning



http://www.datarobot.com/blog/a-primer-on-deep-learning/

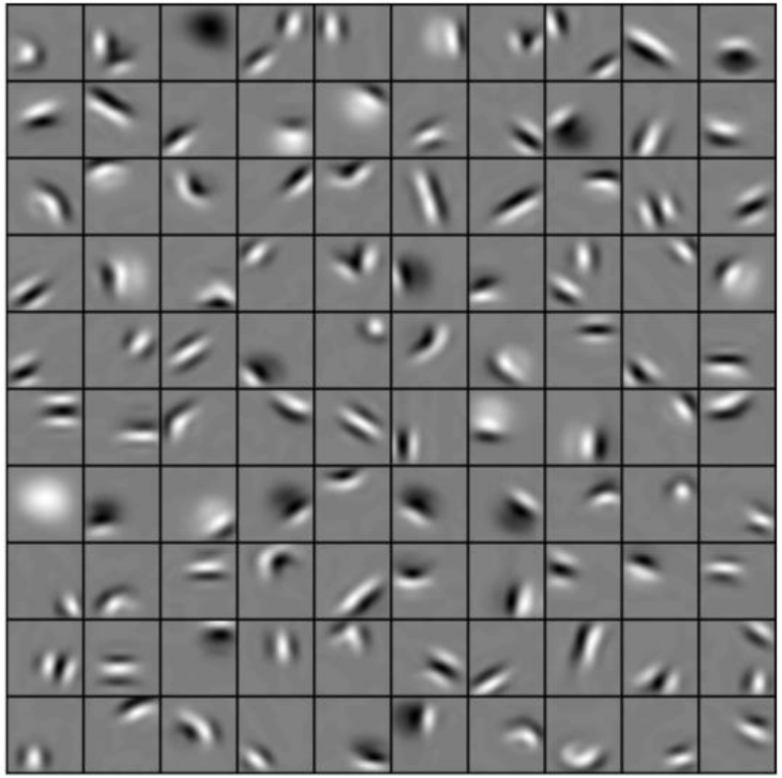
Cool Picture Motivation for Deep Learning

- First layer of z_i trained on 10 by 10 image patches:



- Attempt to visualize second layer:
  - Corners, angles, surface boundaries?

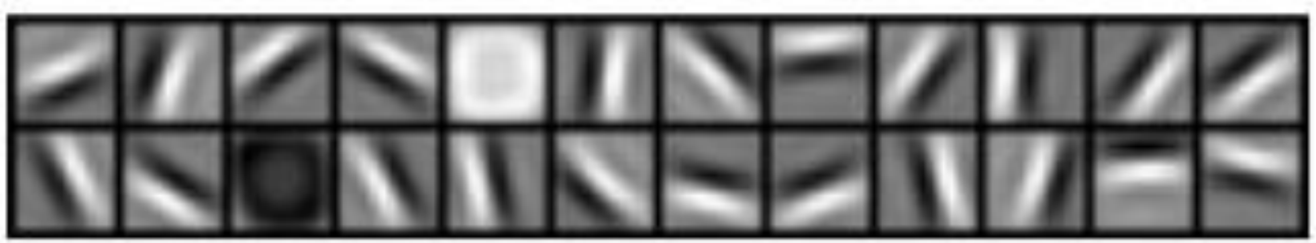
- Models require many tricks to work.



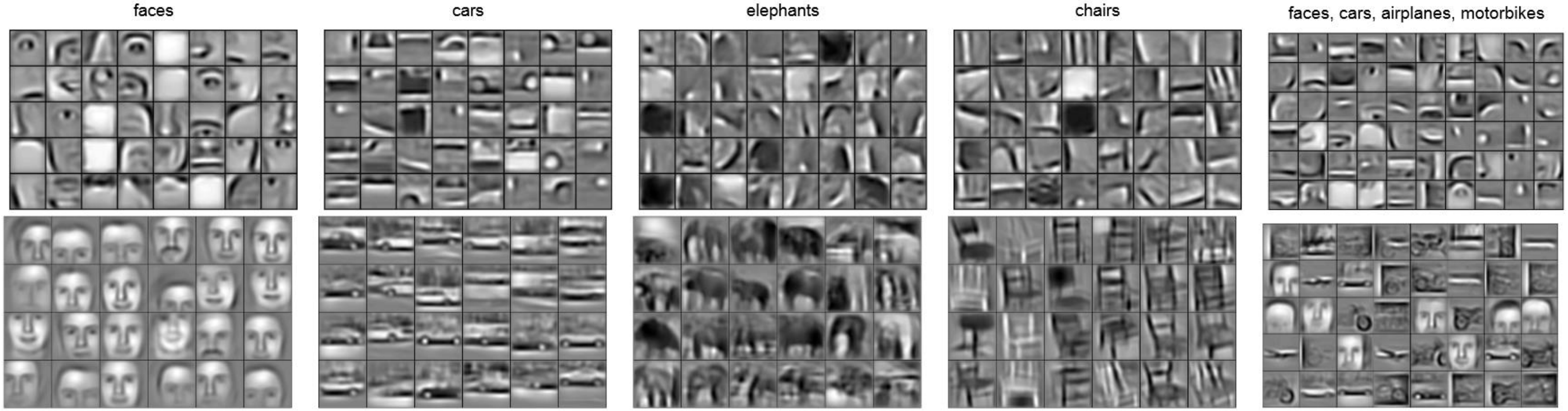
http://www.cs.toronto.edu/~rgrosse/icml09-cdbn.pdf

Cool Picture Motivation for Deep Learning

- First layer of z_i trained on 10 by 10 image patches:



- Visualization of second and third layers trained on specific objects:



http://www.cs.toronto.edu/~rgrosse/icml09-cdbn.pdf

Historical Notes

- 1950 and 1960s: Perceptrons!
  - Roughly: a linear classifier trained with stochastic gradient.
  - “the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.” New York Times (1958).
  - Quickly realized limitations of linear models.
- 1970 and 1980s: Connectionism and backpropagation!
  - Connected networks of simple units.
    - Use parallel computation and distributed representations.
  - Adding hidden layers (z_i) increases expressive power.
    - With 1 layer and enough sigmoid units, it is a universal approximator.
  - Success in optical character recognition (next lecture).

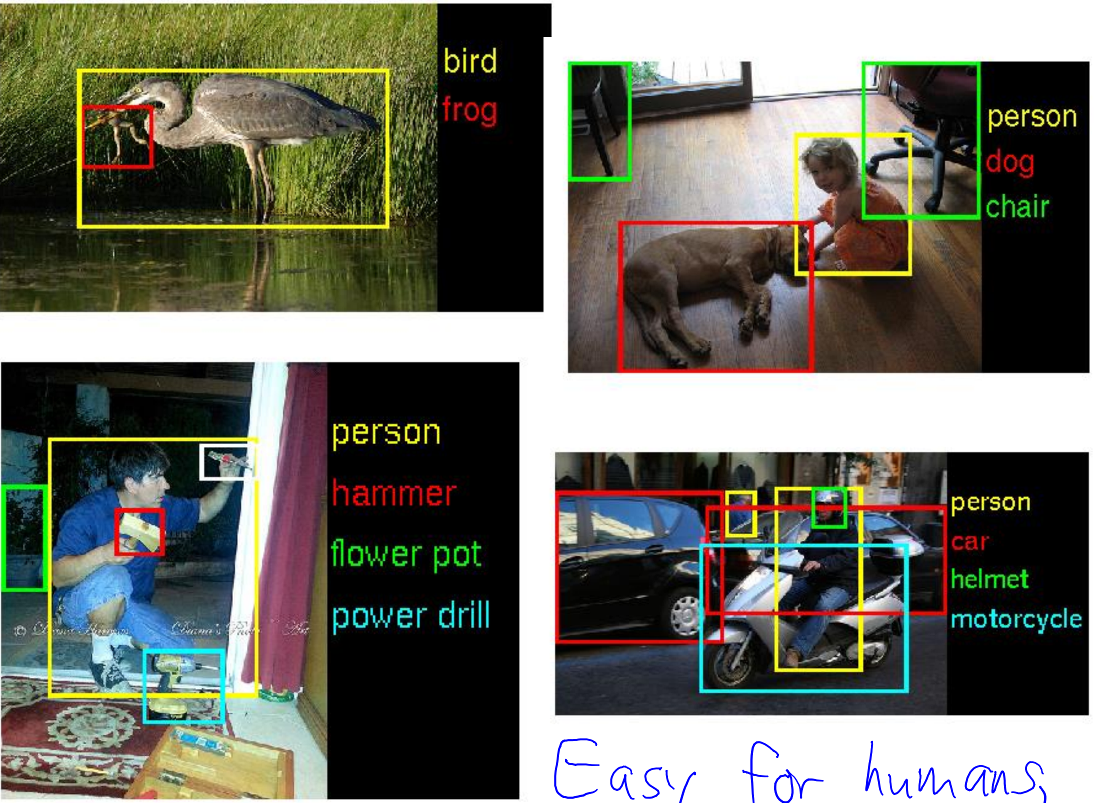
Historical Notes

- 1990s and early-2000s: drop in popularity.
  - It proved really difficult to get multi-layer models working robustly.
  - We obtained similar performance with simpler models:
    - Rise in popularity of logistic regression and SVMs with regularization and kernels.
- Late 2000s: rise in popularity of deep learning.
  - Canadian Institute For Advanced Research (CIFAR) NCAP program:
    - "Neural Computation and Adaptive Perception".
    - Led by Geoff Hinton, Yann LeCun, and Yoshua Bengio ("Canadian mafia").
  - Unsupervised successes: deep belief networks and autoencoders.
    - Could be used to initialize deep neural networks.

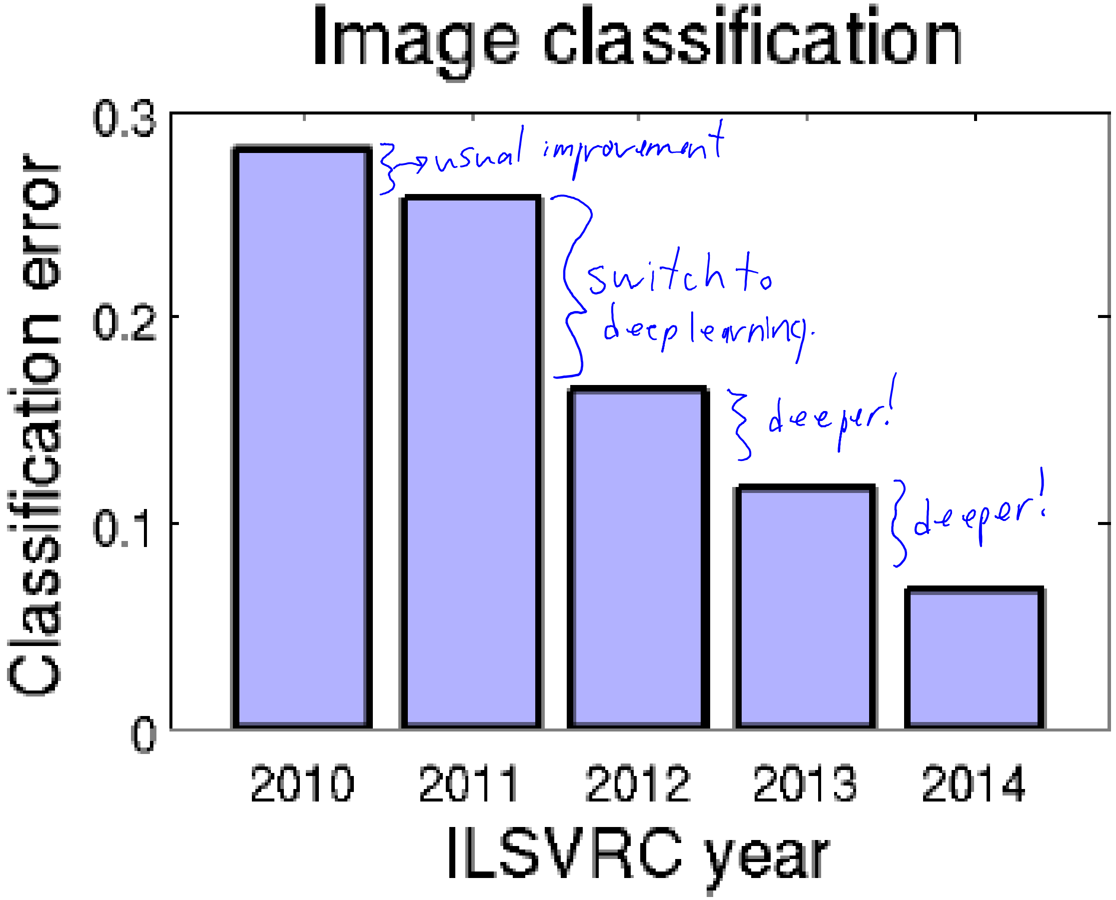
2010s: DEEP LEARNING!!!

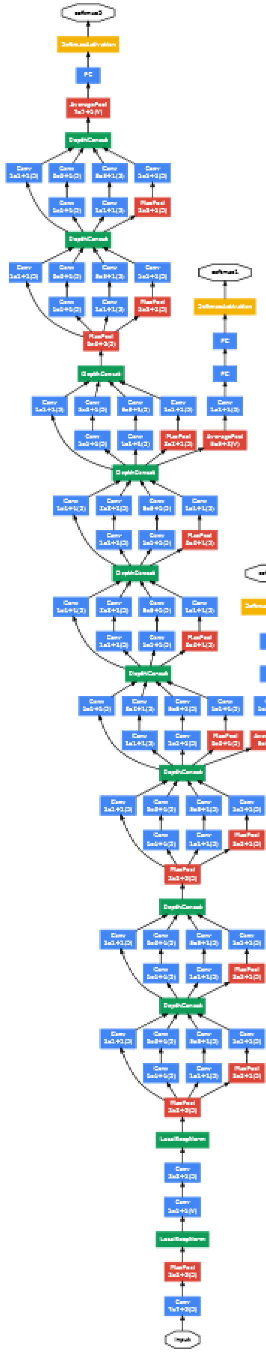
- Bigger datasets, bigger models, parallel computing (GPUs/clusters).
  - And some tweaks to the models from the 1980s.
- Huge improvements in automatic speech recognition (beginning 2009).
  - All phones now have deep learning.
- Huge improvements in computer vision (beginning 2012).
  - This is now finding its way into products.
- Natural language understanding is next?
- Media hype:
  - "How many computers to identify a cat? 16,000", New York Times (2012).
  - "Why Facebook is teaching its machines to think like humans", Wired (2013).
  - "What Is 'deep learning' and why should businesses care?", Forbes (2013).
  - "Computer eyesight gets a lot more accurate" New York Times (2014).

ImageNet Challenge



Easy for humans, very hard for computers.





http://arxiv.org/pdf/1409.0575v3.pdf
http://www.image-net.org/challenges/LSVRC/2014/
http://arxiv.org/pdf/1409.4842v1.pdf

ImageNet Challenge

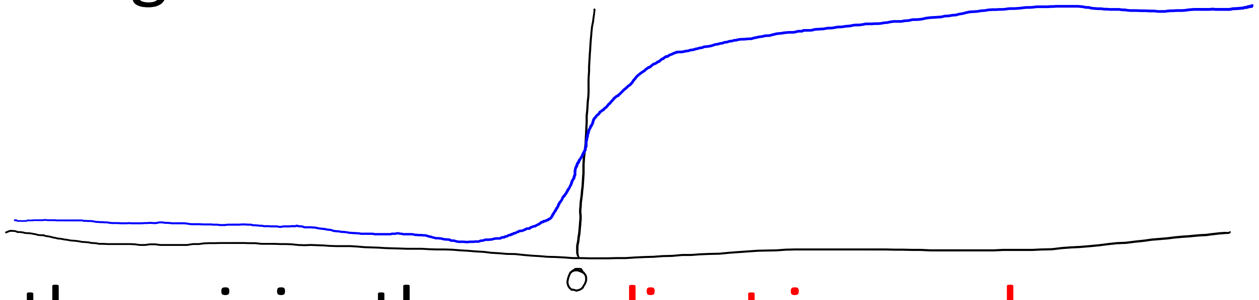
- ImageNet organizer visited UBC this summer.
- "Besides huge dataset/model/cluster, what is the most important?"
  1. Image transformations (translation, rotation, scaling, lighting, etc.).
  2. Optimization.
- Why would optimization be so important?
  - Neural network objectives are highly non-convex (and worse with depth).
  - Optimization has huge influence on quality of model.

Deep Learning Tricks

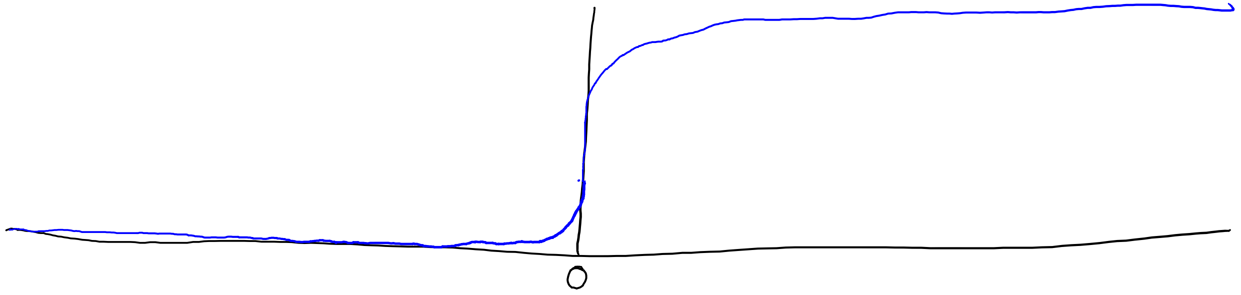
- Standard training method is stochastic gradient (SG):
  - Getting SG to work for convex problems is tricky.
  - For deep neural networks, naïve methods do not work well.
- Are local mimima the problem?
  - There is some empirical/theoretical evidence that local minima are good.
  - But naïve stochastic gradient often does not even find local mimima.
    - Most time is spent near saddle points.
- We've discovered 'tricks' to train deep models:
  1. Different non-linear transformations.
  2. Step-size strategies.
  3. Regularization.
  4. Initialization.
  5. Special network structures.

Vanishing Gradient Problem

- Consider the sigmoid function:



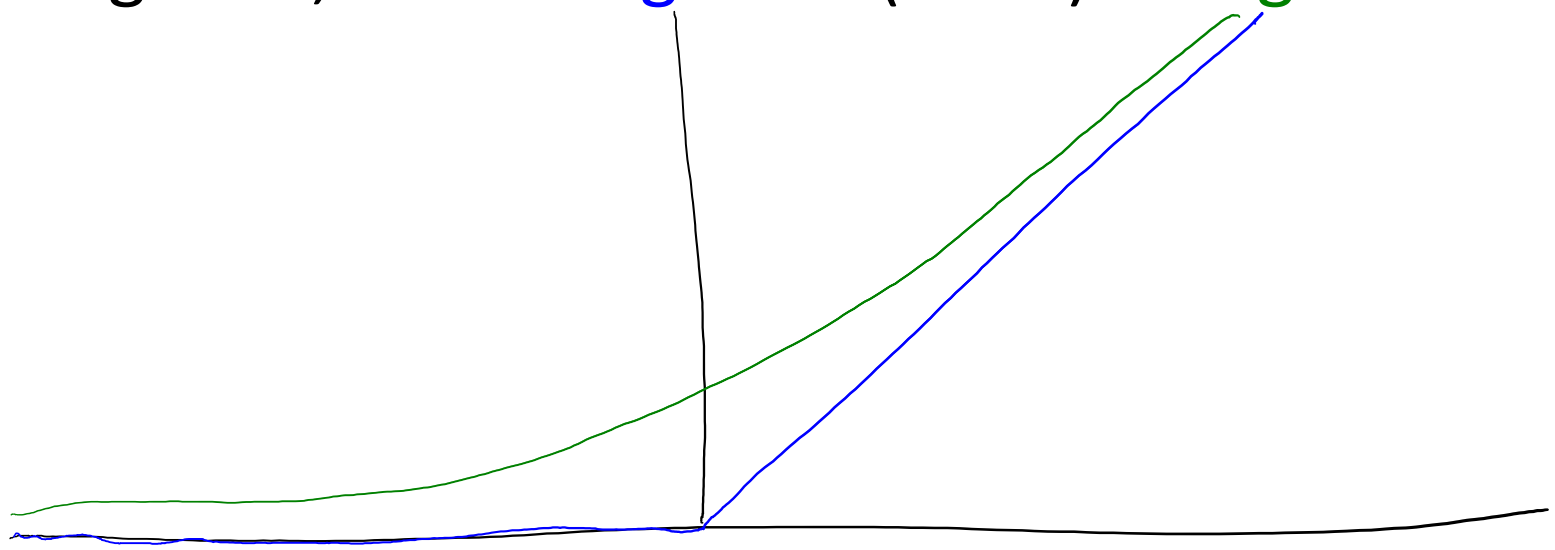
- Away from the origin, the gradient is nearly zero.
- The problem gets worse when you take the sigmoid of a sigmoid:



- In deep networks, many parameters will be ‘stuck’.

Rectified Linear Units (ReLU)

- Instead of sigmoid, use a hinge loss (ReLU) or logistic loss:

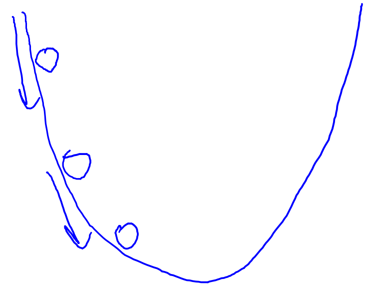


- The gradient approaches zero or one, depending on the sign.
  – Gives sparse of activations.
  – Not really simulating binary signal, but could be simulating rate coding.

Setting the Step-Size

- Stochastic gradient is very sensitive to the step size in deep models.
- Bottou trick:
  1. Grab a small set of training examples.
  2. Do a binary search for a step size that works well on them.
  3. Use this step size for a long time (or slowly decrease it from there).
- Also common: manual 'babysitting' of step size.
- Momentum:
  - Add term that moves in previous direction:

$$w^{t+1} = w^t - \alpha_t \nabla f_{i_t}(w^t) + \underbrace{\beta_t (w^t - w^{t-1})}_{\text{Keep going in the old direction}}$$



- Bias step-size multiplier: use bigger step-size for the bias variables.

Summary

- Deep learning considers neural networks with many hidden layers.
- Biological motivation for these representations.
- Unprecedented performance on difficult pattern recognition tasks.
- Optimization is key to good performance, many engineering tricks.

- Next time:
  - Deep learning tricks underlying speech/vision systems.