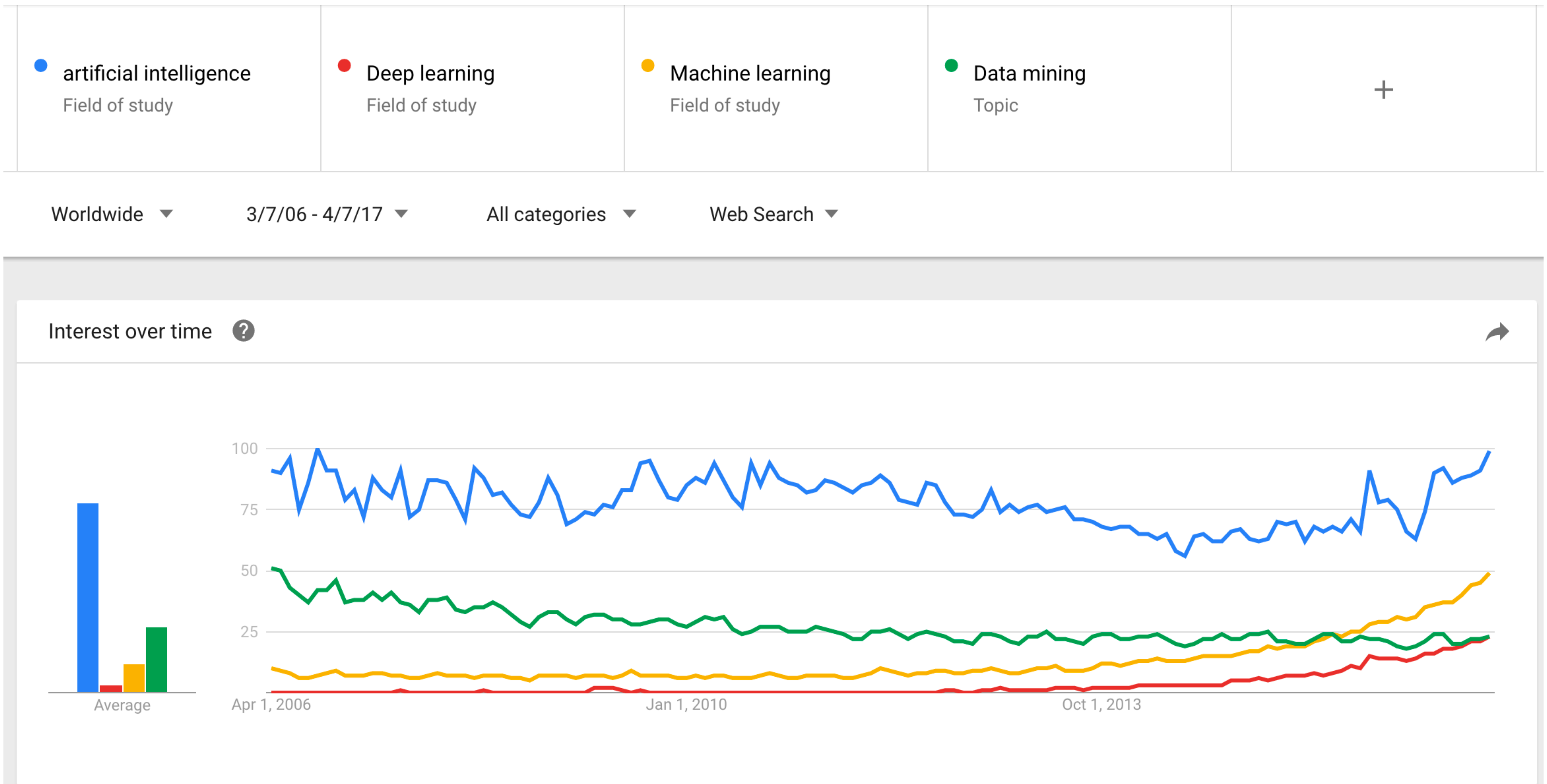
Introduction to Deep Learning

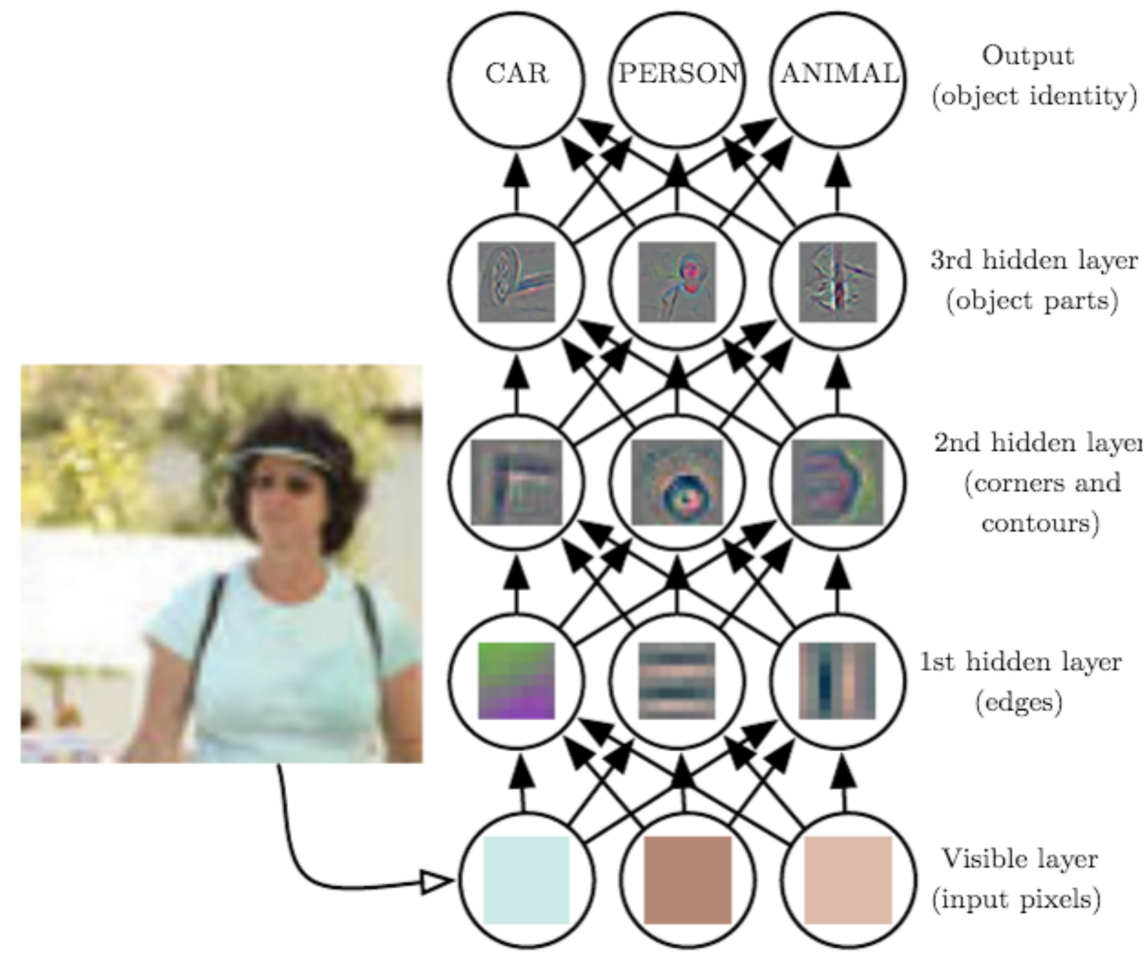
CS 534: Machine Learning

Deep Learning: “The New Cool”



Deep Learning: Overview

- Form of representation learning
- Aimed at learning feature hierarchies
- Features from higher levels of the hierarchy are formed by lower level features
- Each hidden layer allows for more complex features of input

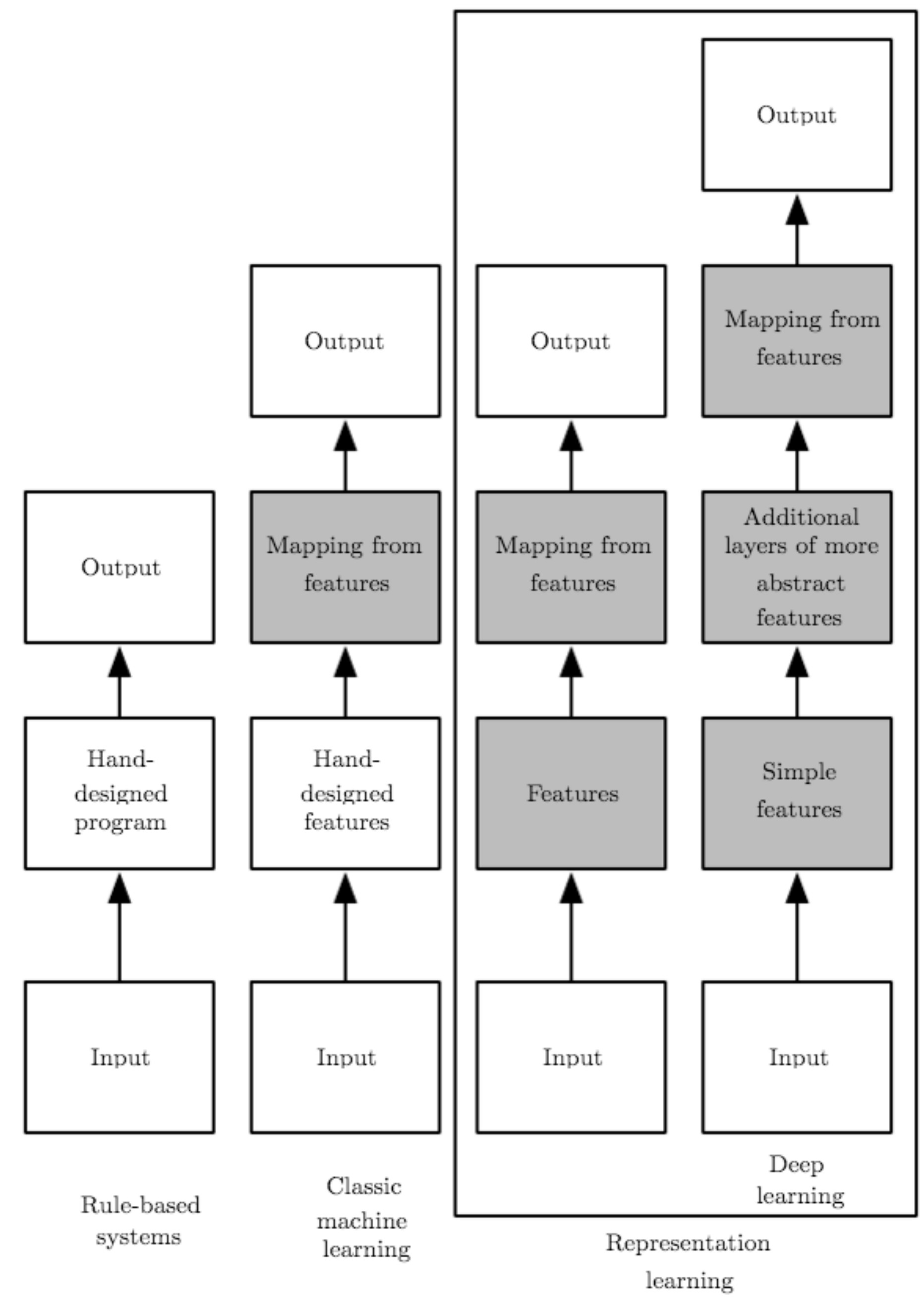


http://www.deeplearningbook.org/contents/intro.html

Deep Learning: The Promised Land

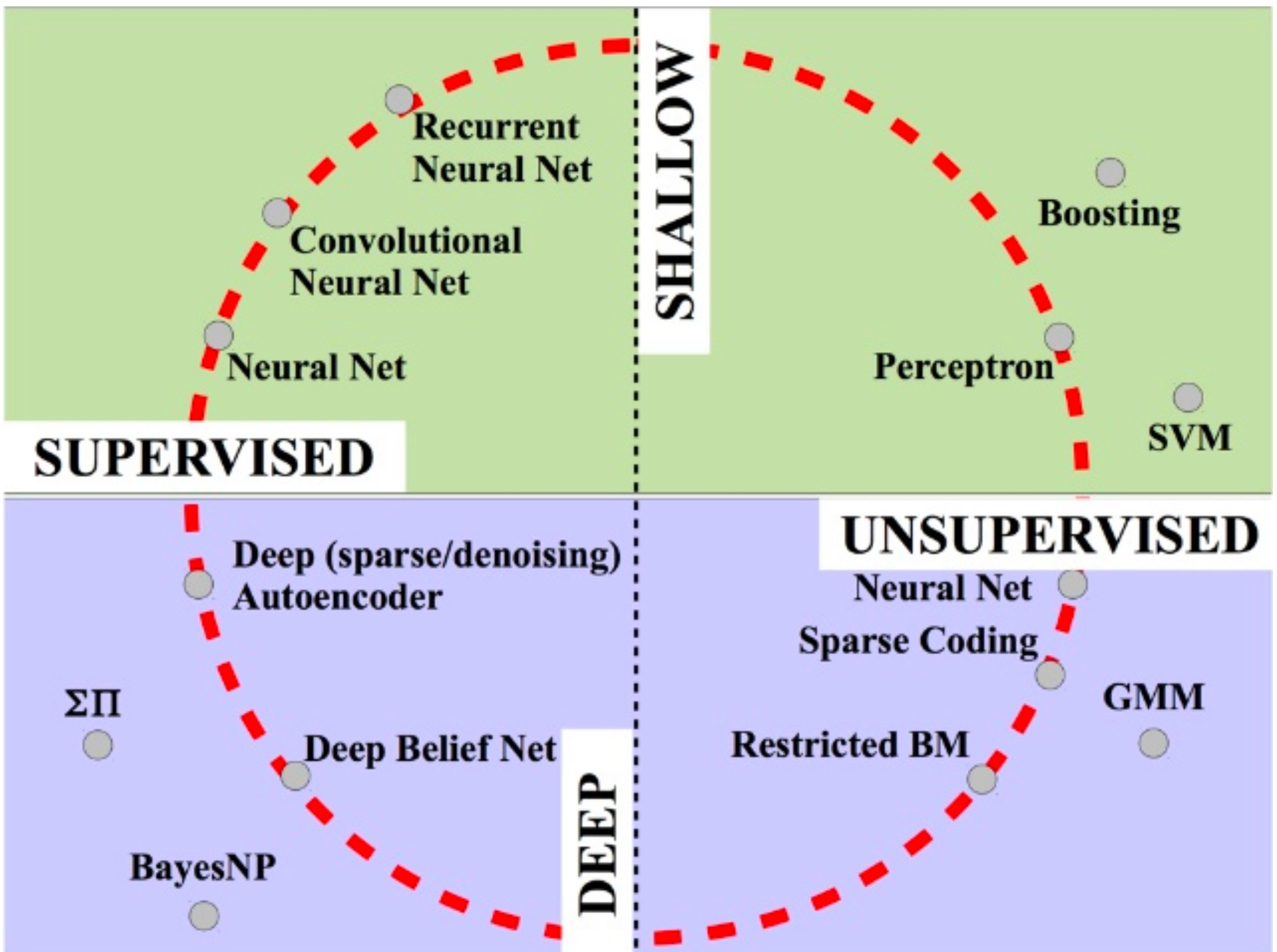
Automatic feature discovery

- Hidden layers discover semantically meaningful concepts
- Features learned without need for seeing exponentially large number of configuration of other features
- Expressiveness of deep networks



http://www.deeplearningbook.org/contents/intro.html

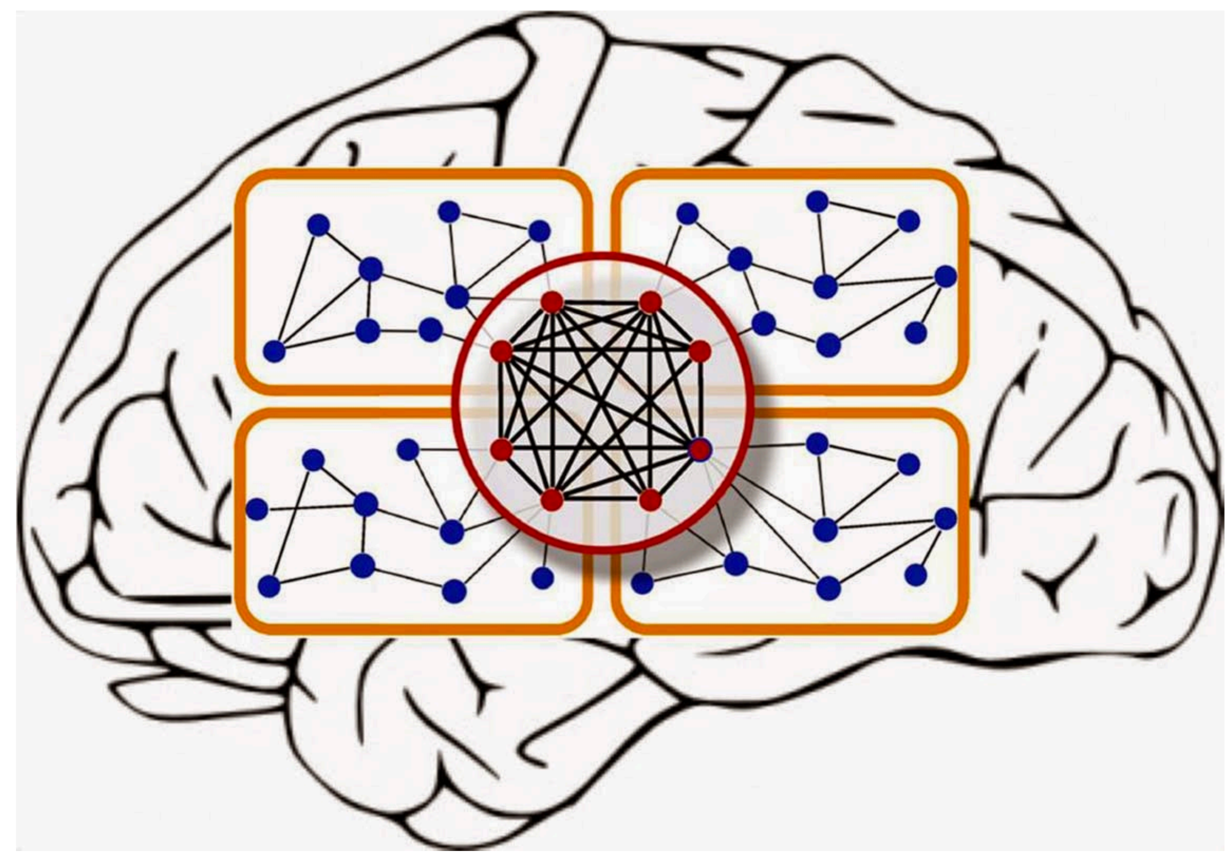
Deep vs Shallow Architectures



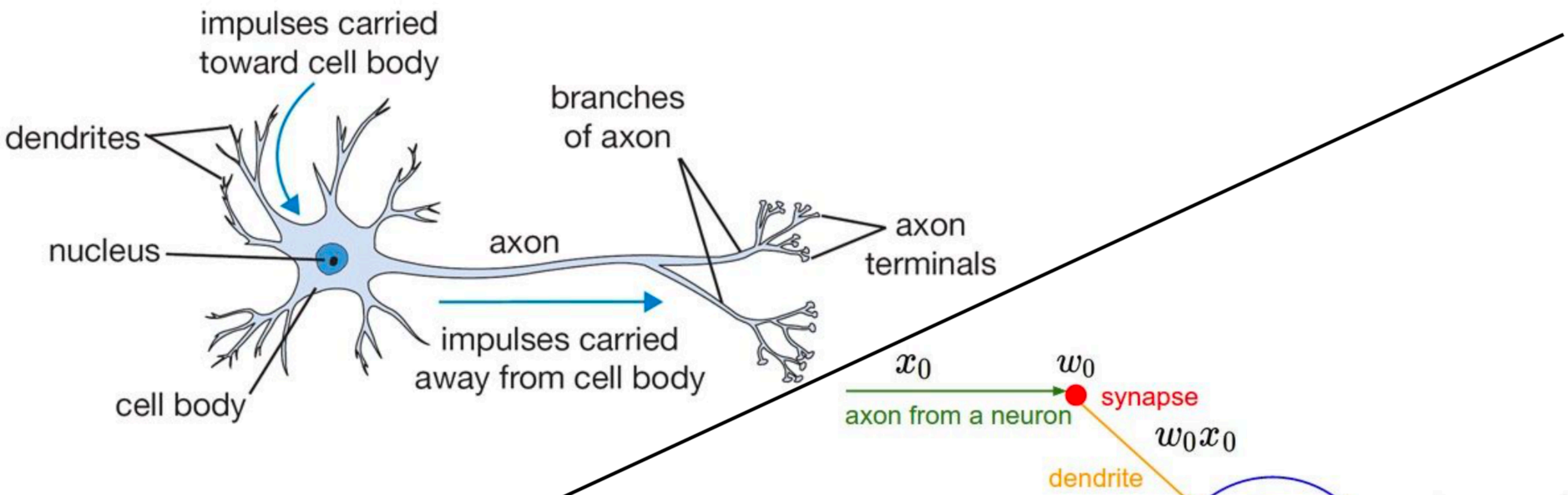
http://www.slideshare.net/roelofp/041114-dl-nlpwordembeddings

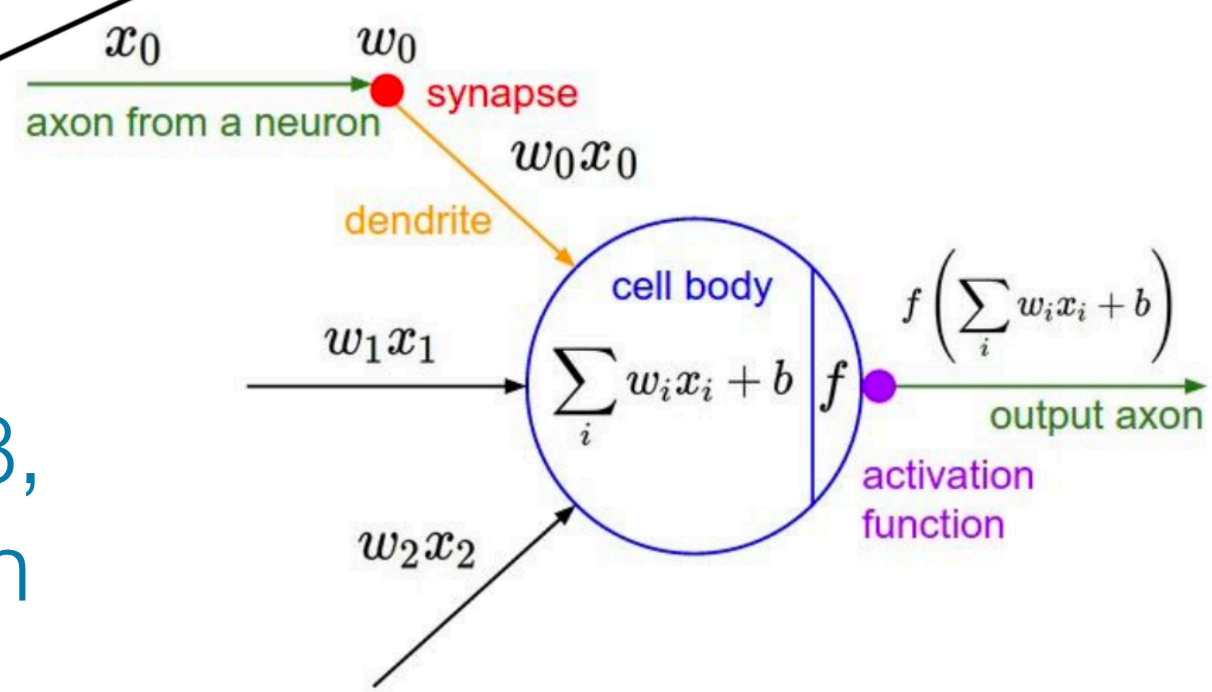
Review: Motivation by Human Brain

- Contains 10^{11} neurons, each with up to 10^5 connections
- Each neuron is fairly slow with switching time of 1 ms
- Computers at least 10^6 times faster in raw switching speed
- Brain is fast, reliable, and fault-tolerant



Review: Neuron —> Perceptron

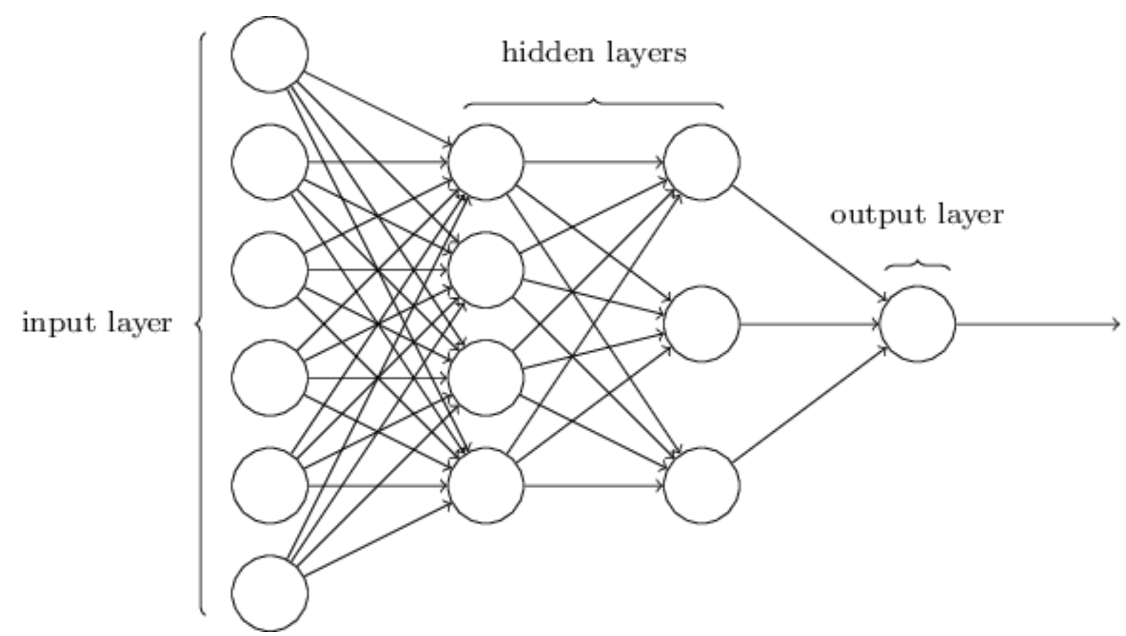




Birth of neural networks in 1958,
but linearly separability limitation
led to the first AI winter

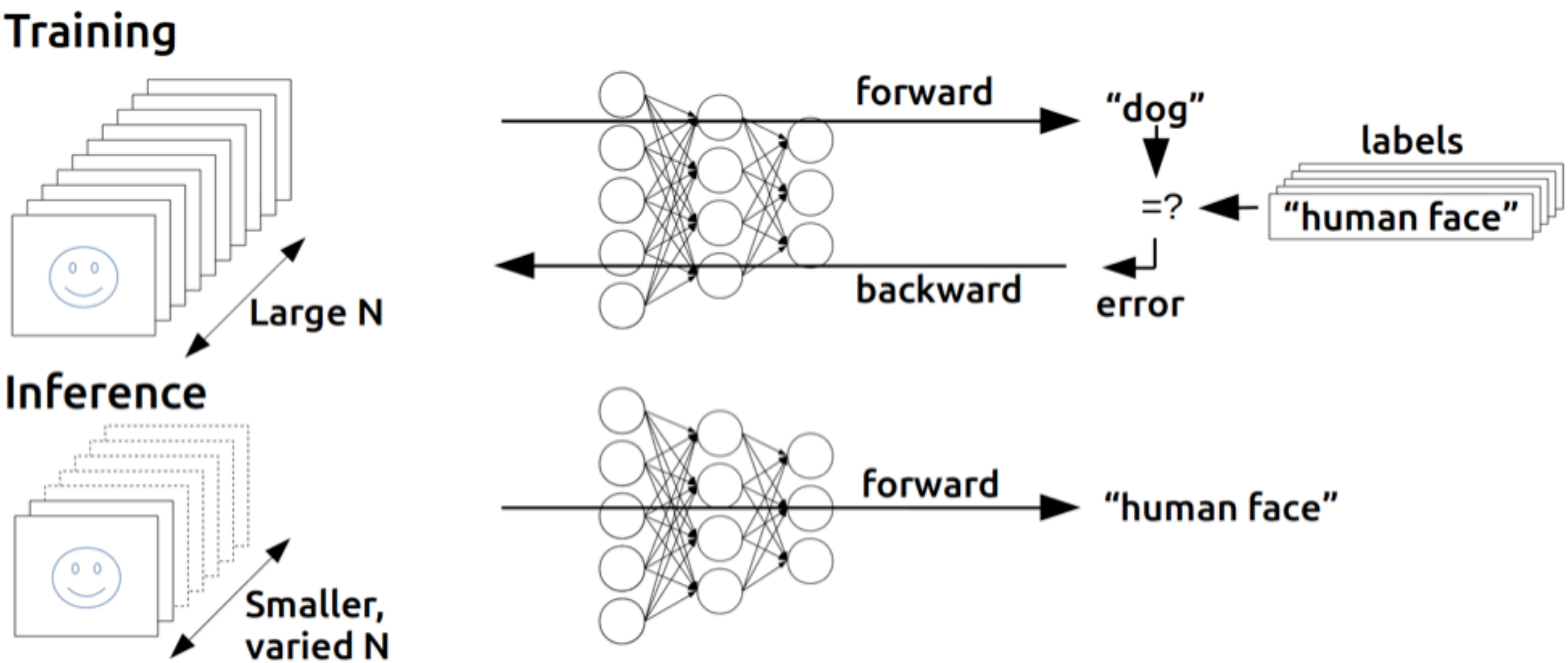
Review: MLP

- Composition of neurons with an activation function
- Typically, each unit of layer t is connected to every unit of the previous layer t - 1 only
- No cross-connections between units in the same layer



http://neuralnetworksanddeeplearning.com/chap1.html

Review: Backpropogation

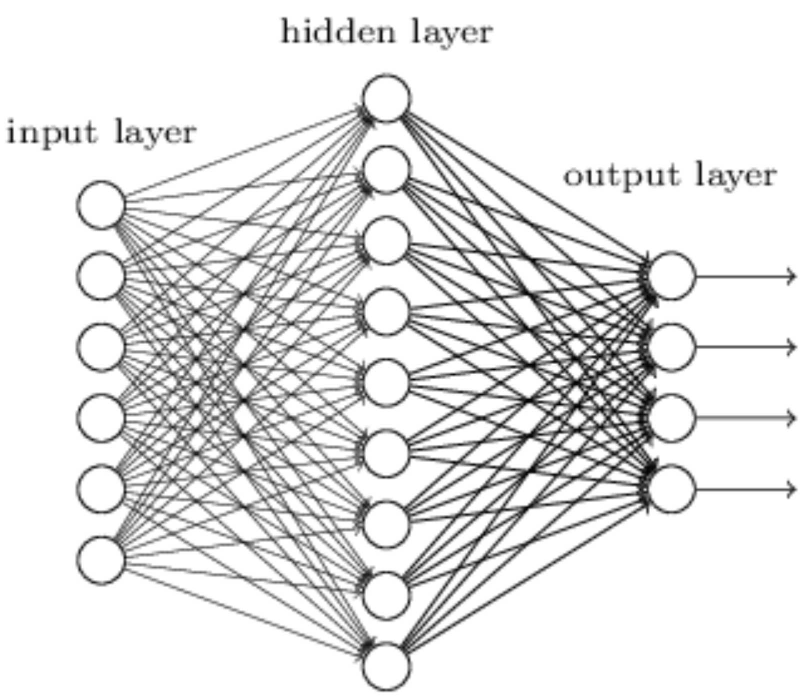


Backpropogation introduced in the early 1970s but Rumelhart, Hinton, and Williams formulated for MLPs — rise of neural networks again!

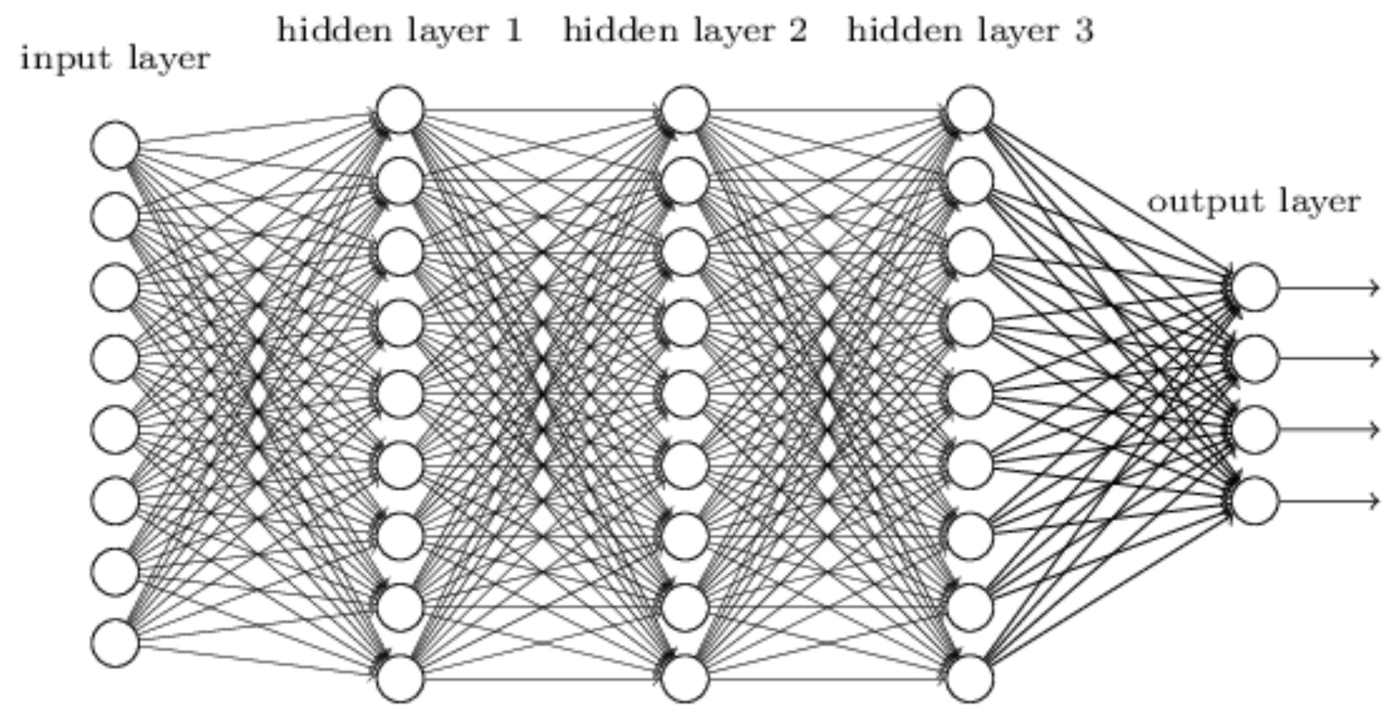
https://devblogs.nvidia.com/parallelforall/inference-next-step-gpu-accelerated-deep-learning/

Deep Neural Networks

"Non-deep" feedforward
neural network



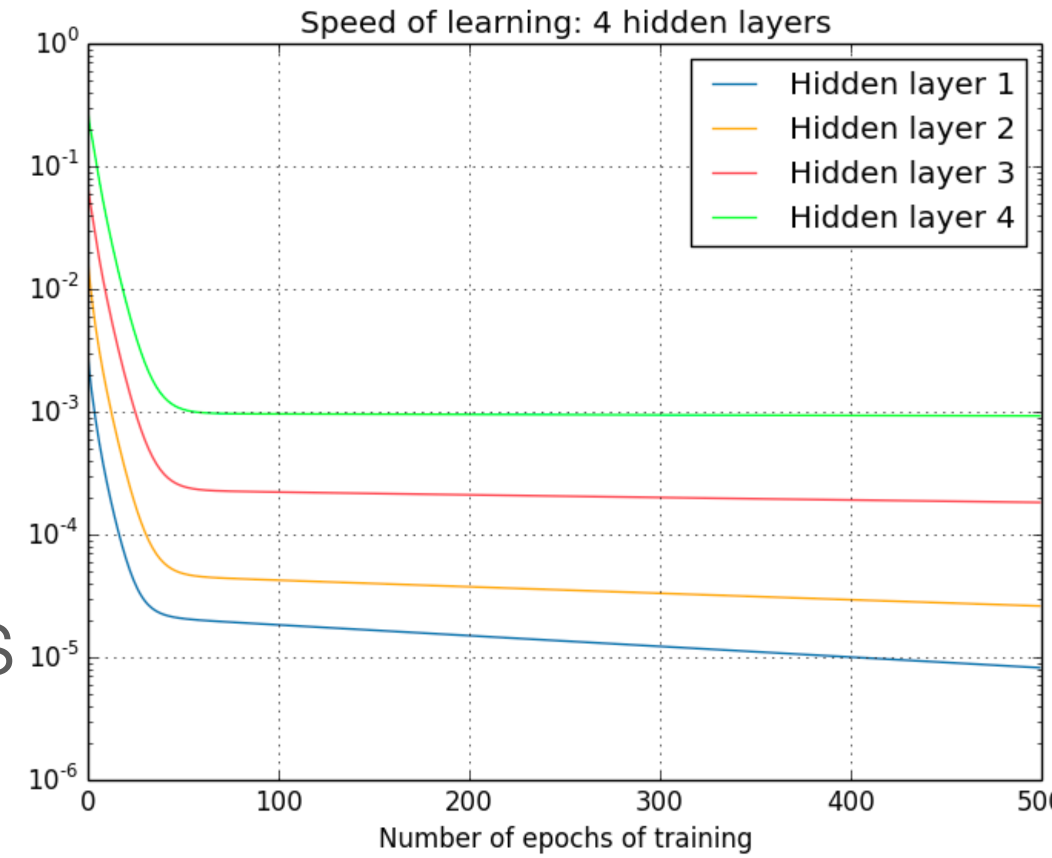
Deep neural network



http://www.coldvision.io/2016/07/29/image-classification-deep-learning-cnn-caffe-opencv-3-x-cuda/

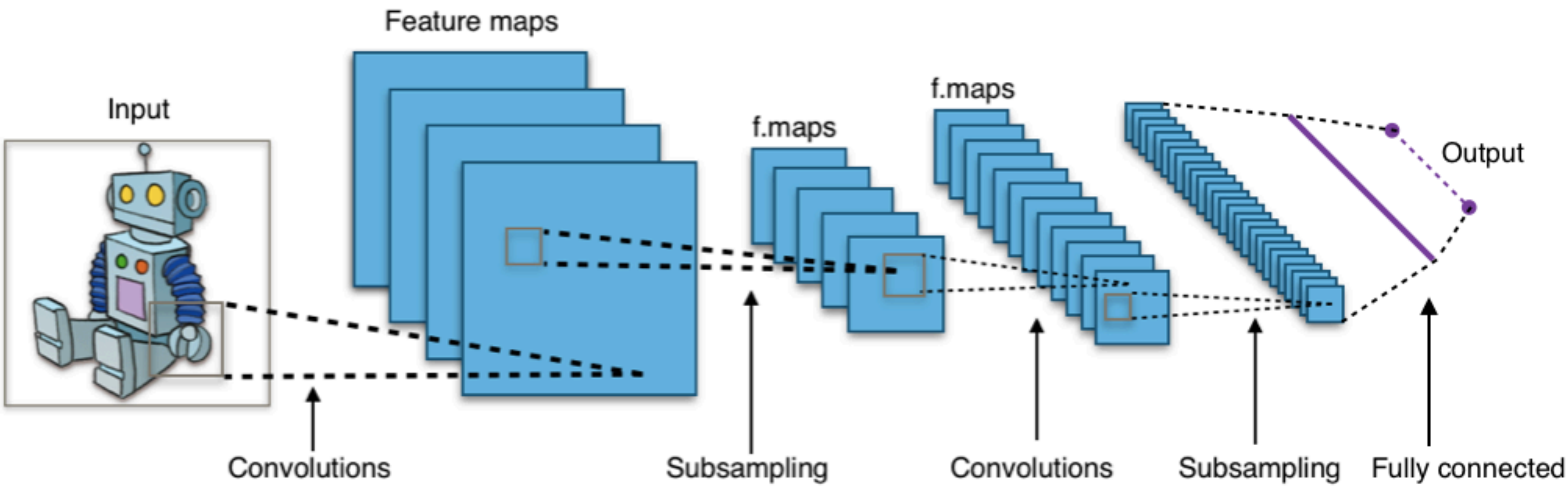
Review: Obstacles to Deep MLPs

- Requires lots of labeled training data
- Computationally extremely expensive
    - Vanishing & unstable gradients
- Difficult to tune



http://neuralnetworksanddeeplearning.com/chap5.html

Review: CNN



CNN was only successful deep network up to 2006, as anything past 3 layers was impossible to train

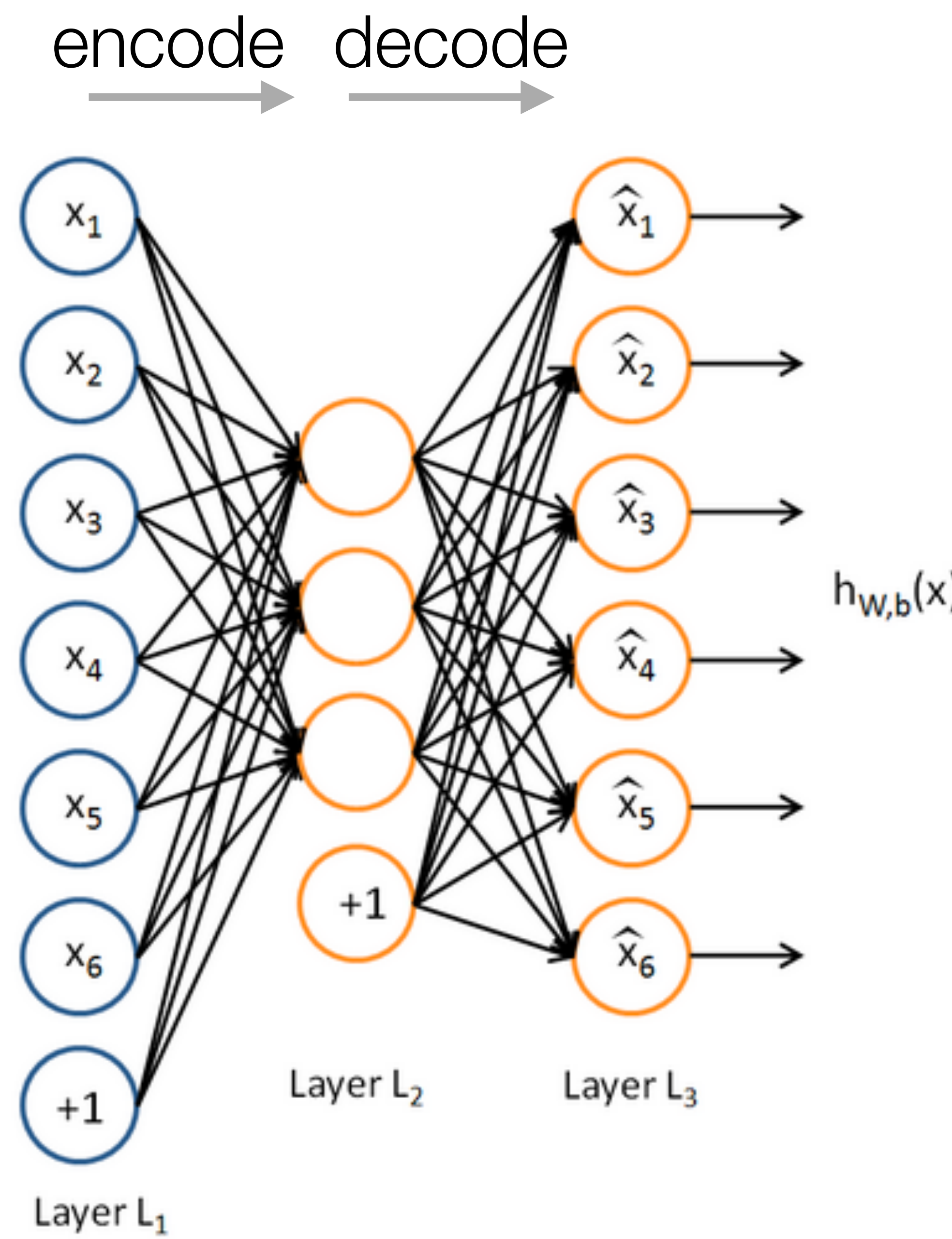
https://en.wikipedia.org/wiki/Convolutional_neural_network

Neural Nets Go Unsupervised

- CNN and MLPs used to automate rote tasks
  - Example: Reading checks
- What about smaller representation (i.e., compression) of the data?
- Can we think about only using the training data to efficiently translate / encode data to a compact format?

Autoencoder

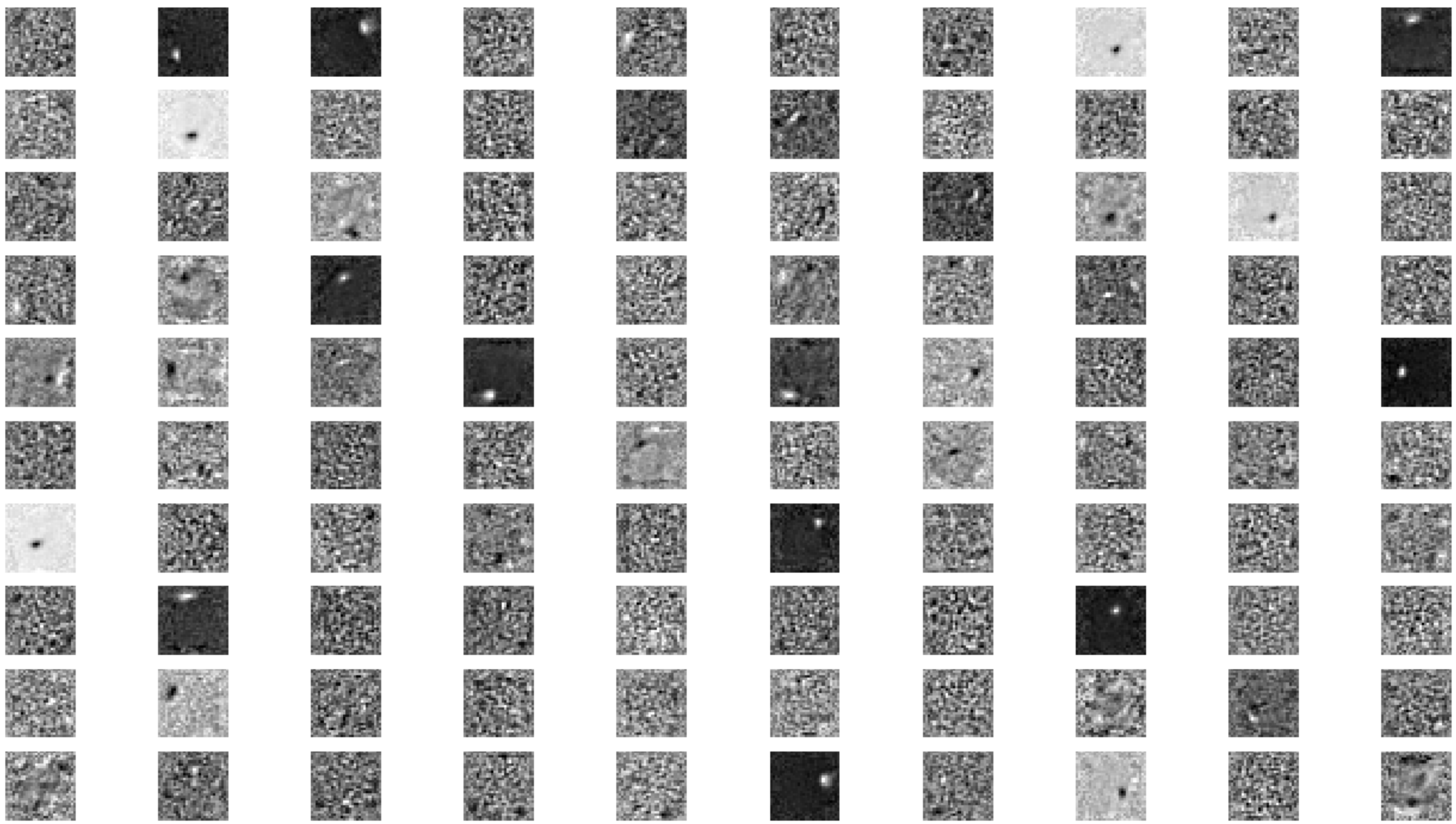
- MLP only works with labeled training examples
- Autoencoder learns compressed, distributed representation (encoding) of the dataset
- Aim to "recreate" the input
- Introduced in 1986



http://ufldl.stanford.edu/wiki/index.php/Autoencoders_and_Sparsity

Autoencoder: MNIST Results

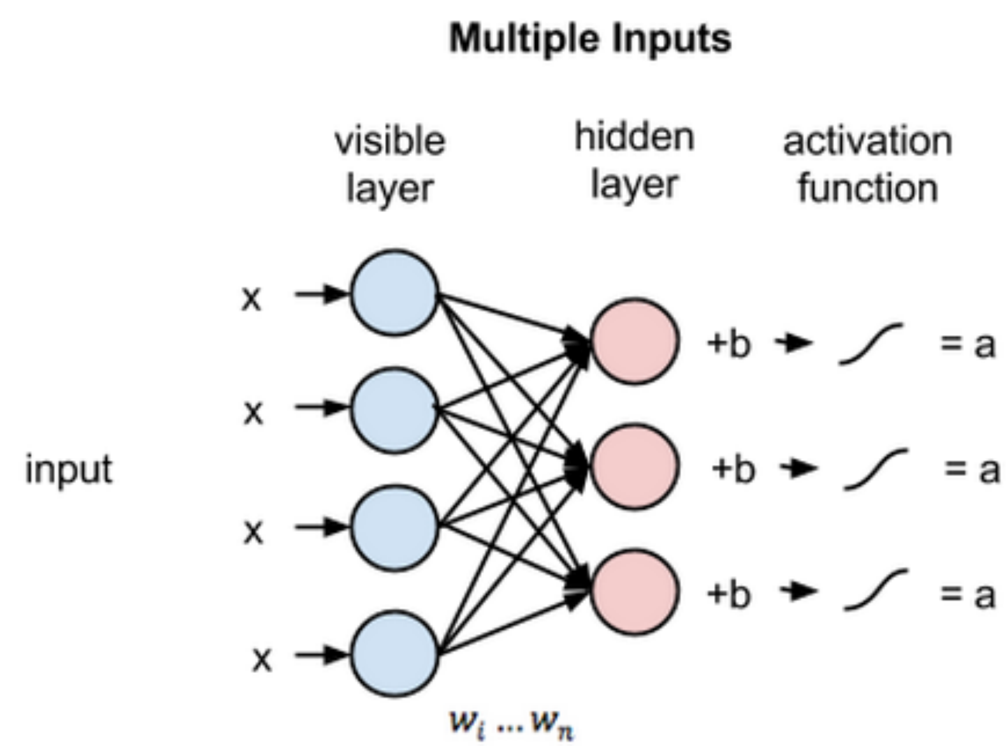
500 hidden units with 20 epochs and mini batch size of 20



https://triangleinequality.wordpress.com/2014/08/12/theano-autoencoders-and-mnist/

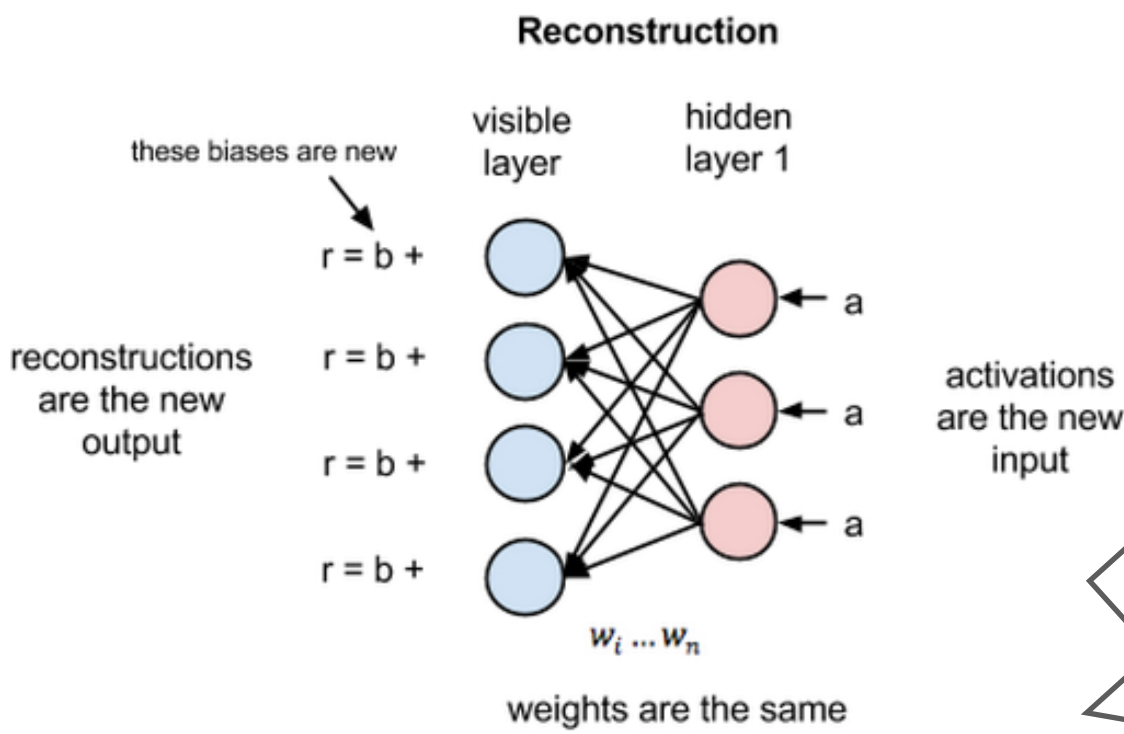
Restricted Boltzmann Machines (RBM)

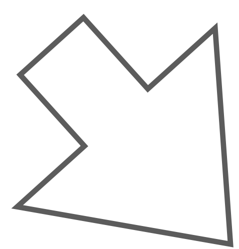
- Generative stochastic neural network that can learn a probability distribution over its set of inputs
- Restrict connectivity to make learning easier
  - One layer of hidden units
  - No connections between hidden units



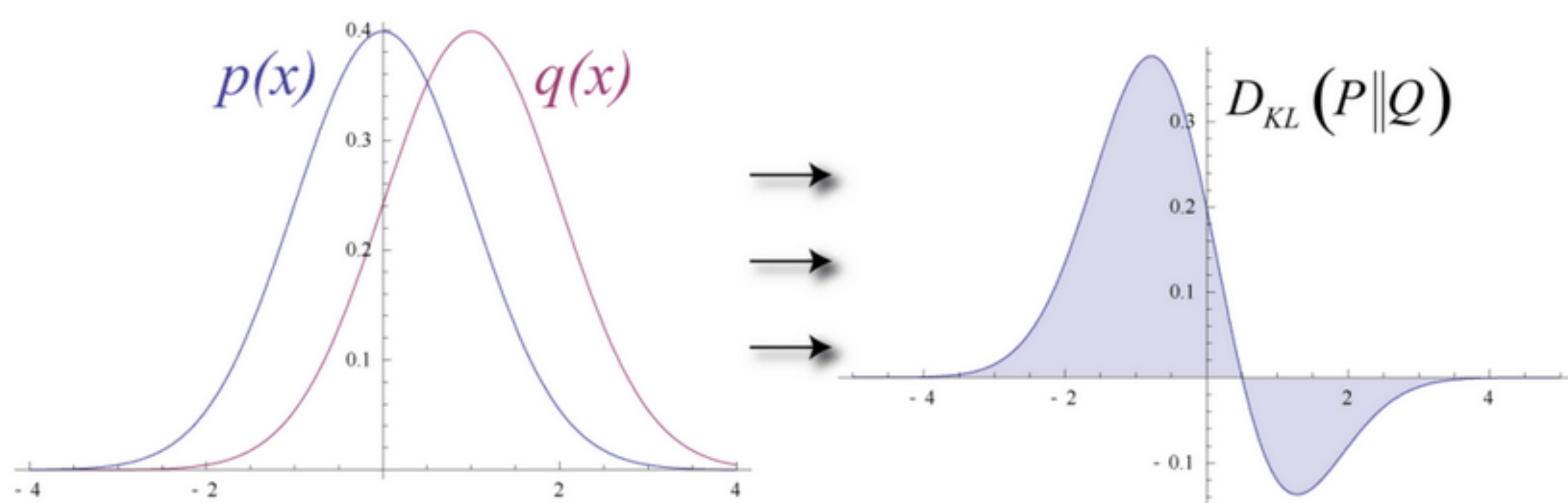
http://deeplearning4j.org/restrictedboltzmannmachine.html

RBM: Reconstruction via Backpropogation





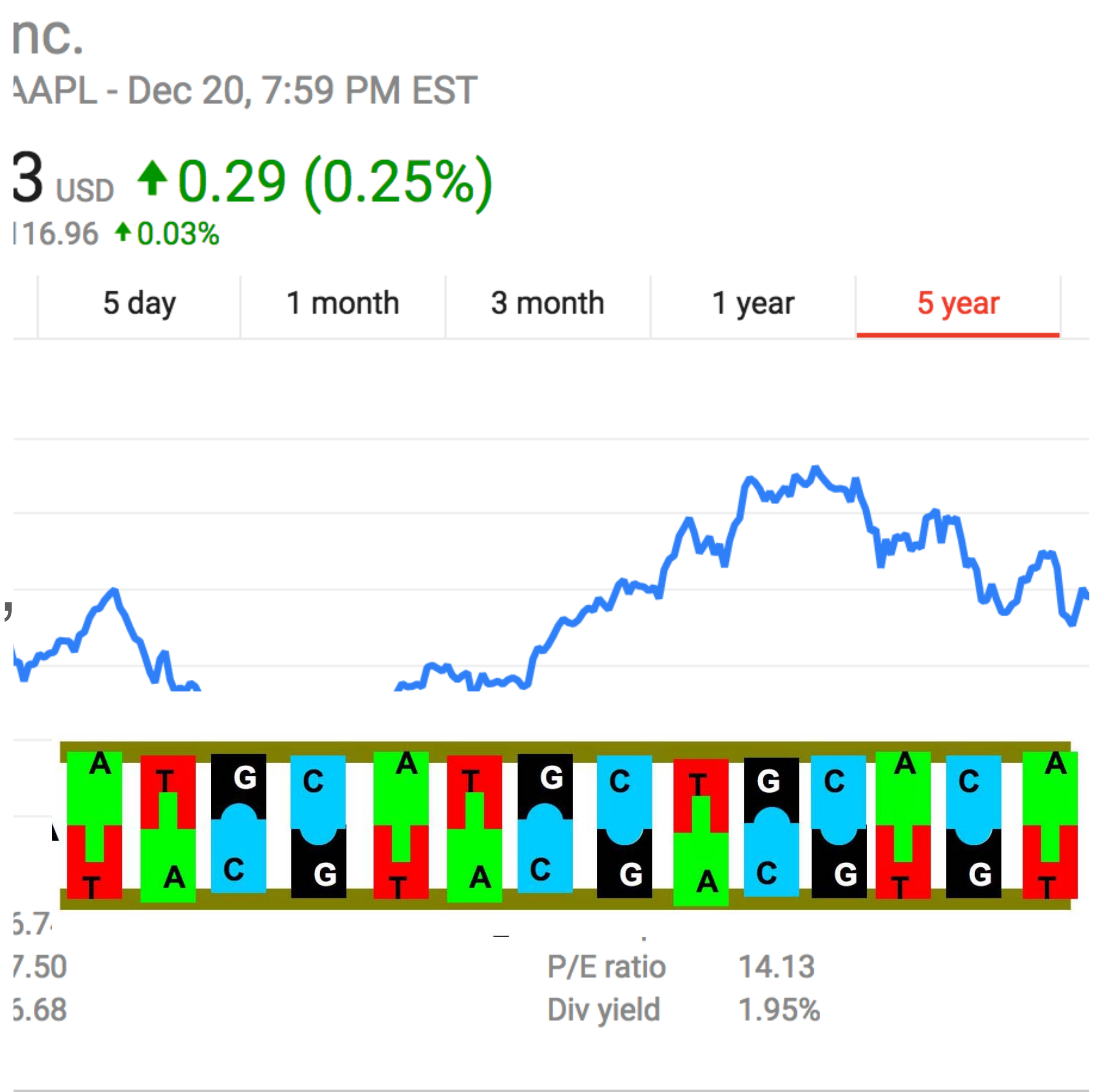
minimize KL divergence between probability distribution and ground-truth distribution



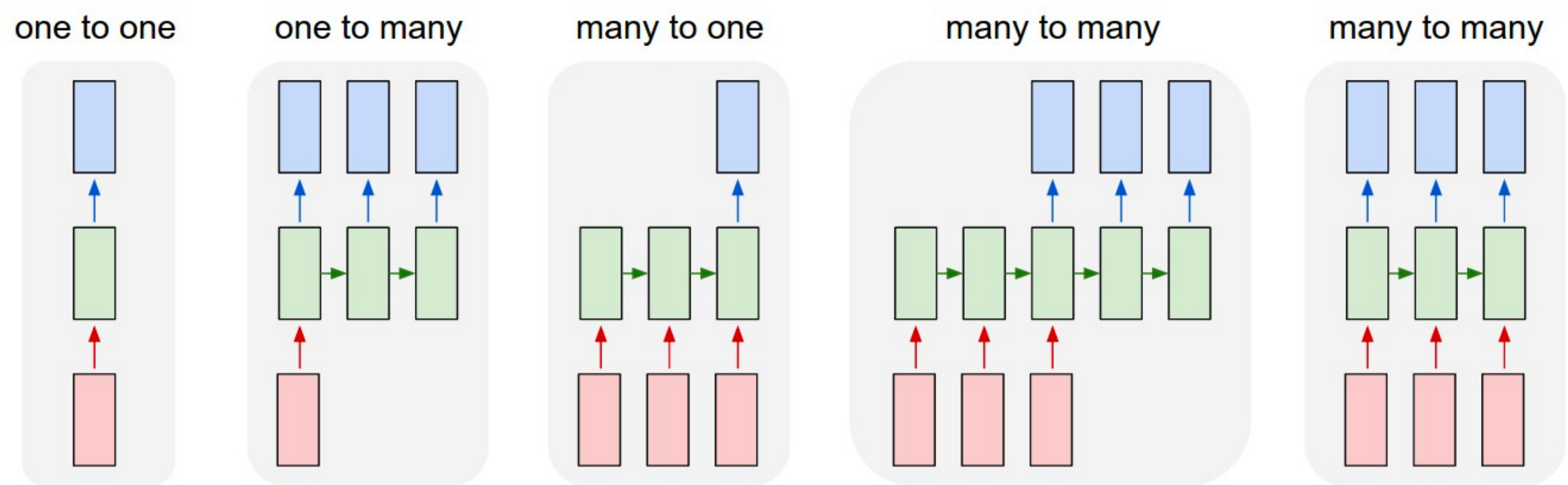
http://deeplearning4j.org/restrictedboltzmannmachine.html

Review: Sequential Data

- What about sequential data?
    - Time-series: Stock market, weather, speech, video
    - Ordered: Text, genes



The Need for Sequences



Sequence output (e.g., image captioning task where image becomes sequence of words)

Sequence I/O (e.g., machine translation)

Synced sequence I/O (e.g., video classification on frame level)

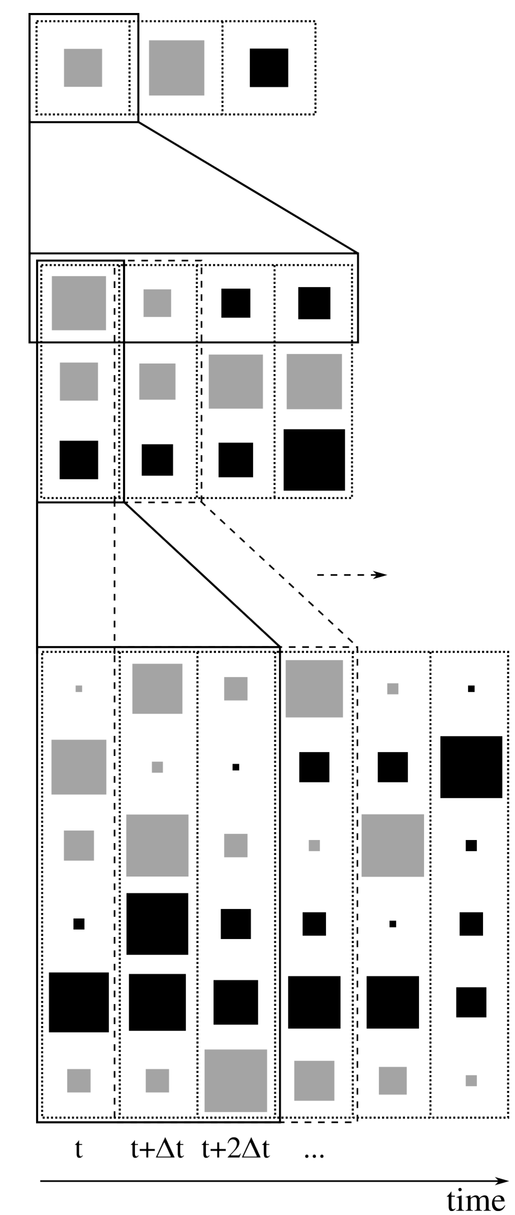
"Vanilla" NN: fixed-sized input to fixed-size output

Sequence input (e.g., sentiment analysis)

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Time-Delay Neural Networks (TDNN)

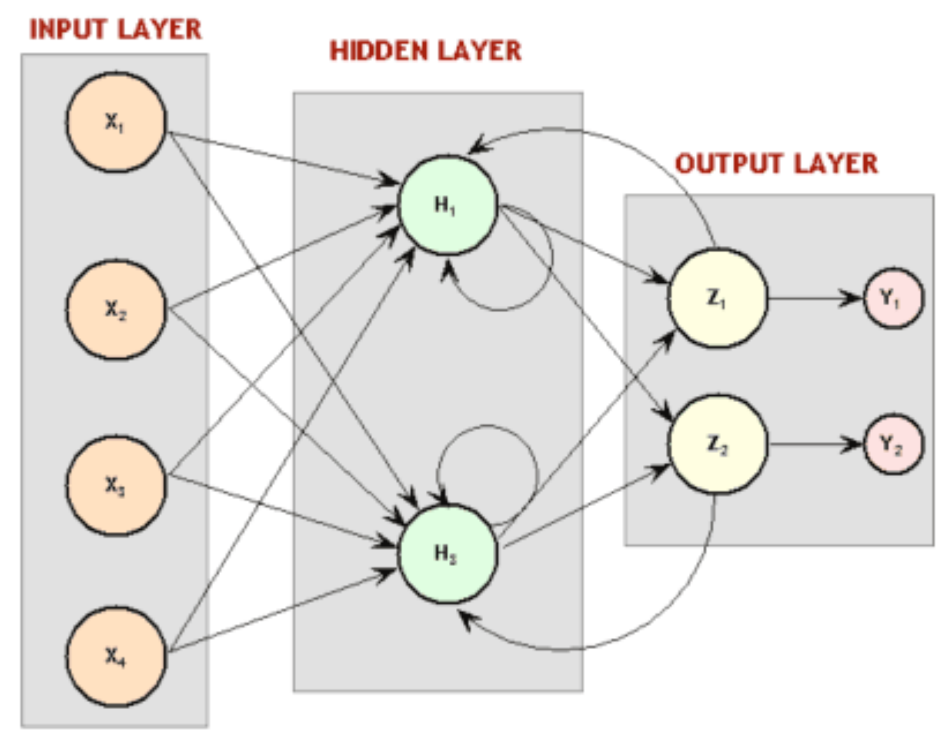
- Each neuron processes subset of input
- Different weights for different delays of input data
- Similar to CNNs since it looks at subset of input at a time



https://en.wikipedia.org/wiki/Time_delay_neural_network

Recurrent Neural Networks (RNN)

- Family of neural networks for processing sequential data
- Output of the layer can connect back to the neuron itself or a layer before it
- Share same weights across several time steps



RNN: Recurrence

We can process a sequence of vectors **x** by applying a recurrence formula at every time step:

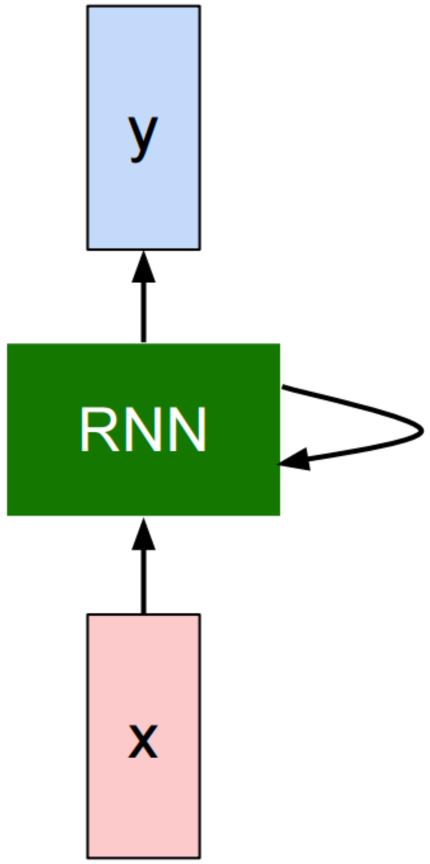
$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

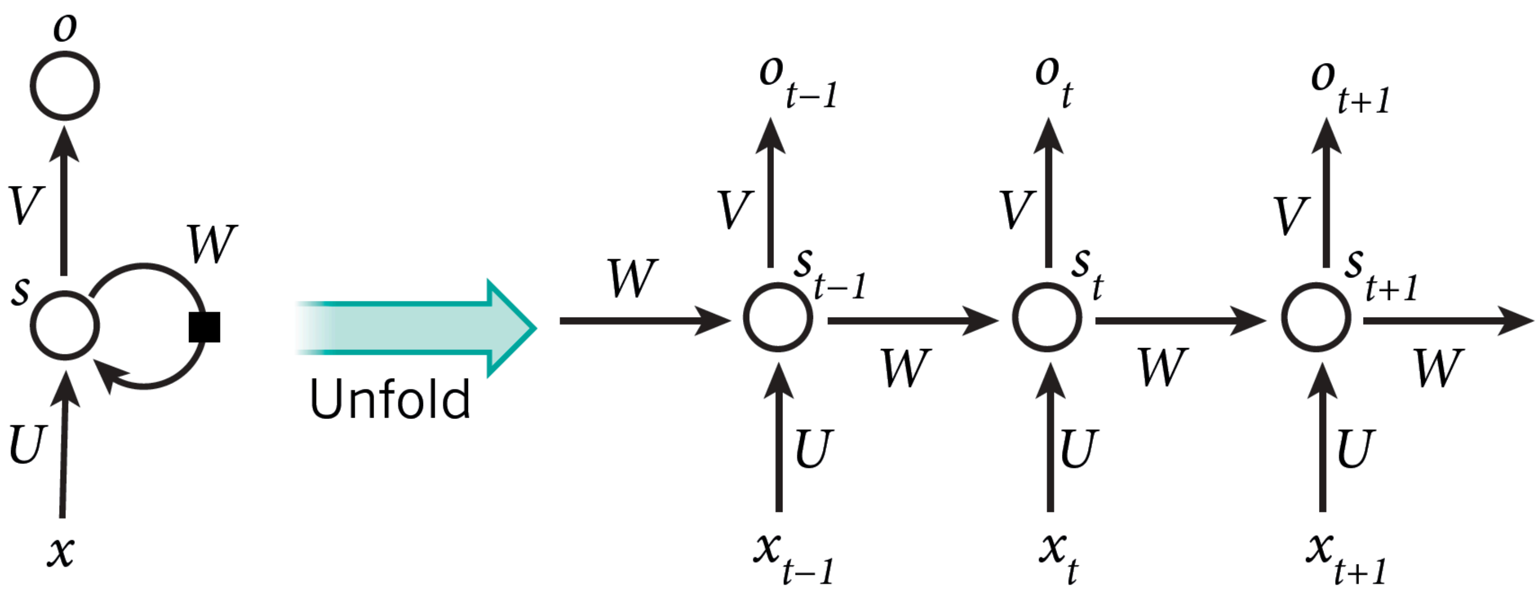
old state

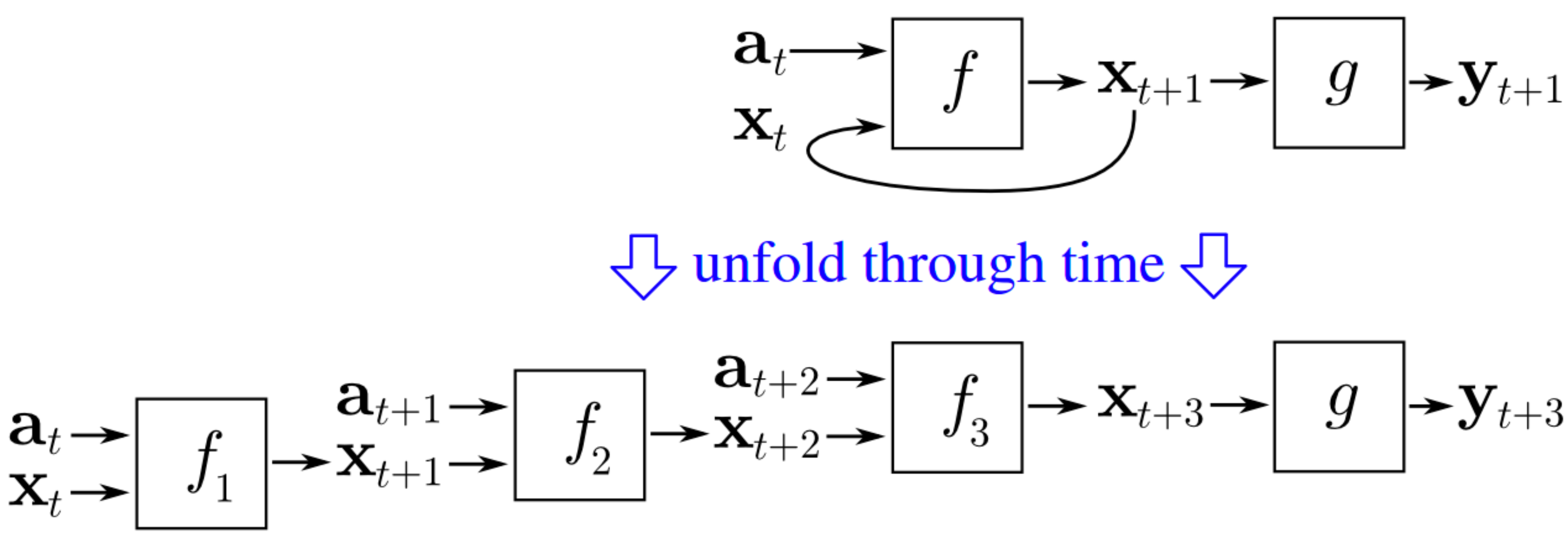
input vector at some time step

some function with parameters W



RNN: Unfolding for Backpropogation



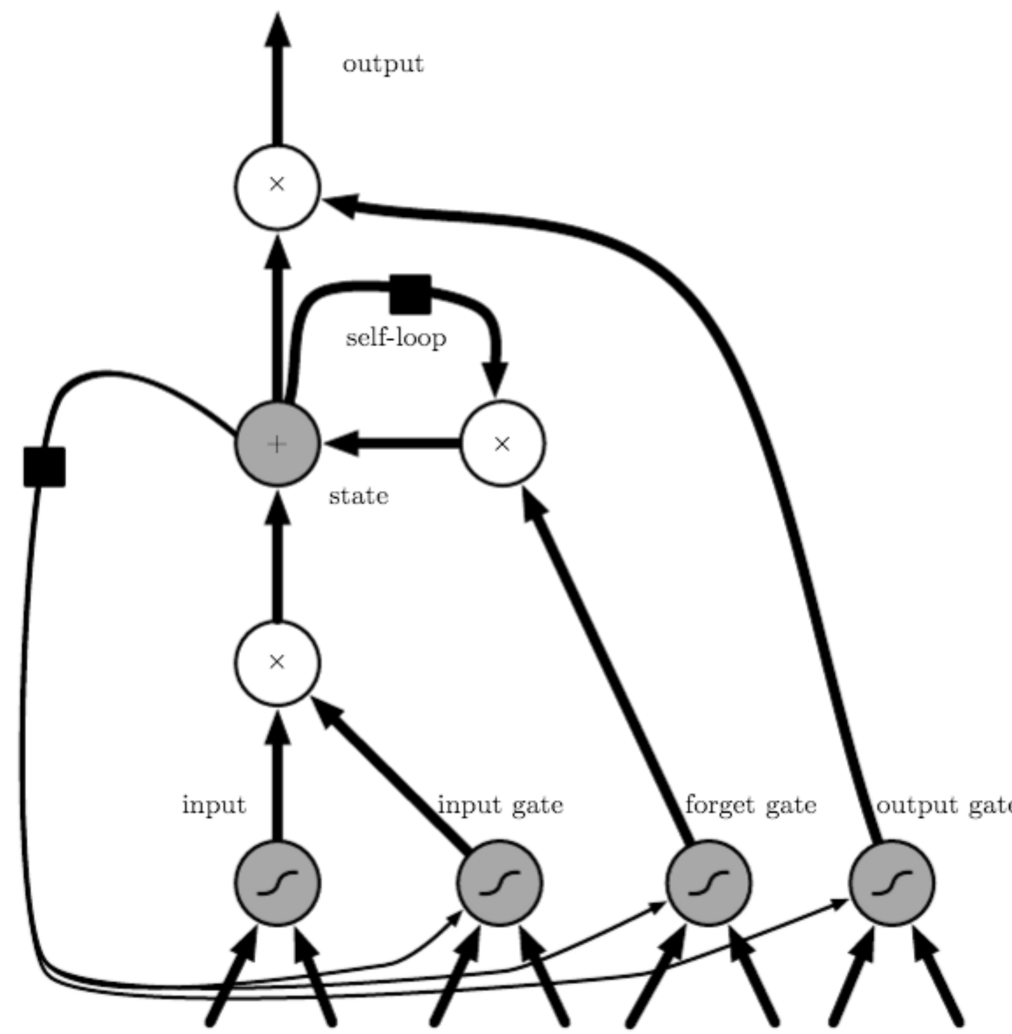


Long-Term Dependency Problems

- Appeal of RNN is to connect previous information to present task
- Gap between relevant information and point of needing it can be large (e.g., word prediction for a sentence like I grew up in France … I speak fluent ___)
- Long-range dependencies are difficult to learn because of vanishing gradient or exploding gradient problem (depending on the activation function)

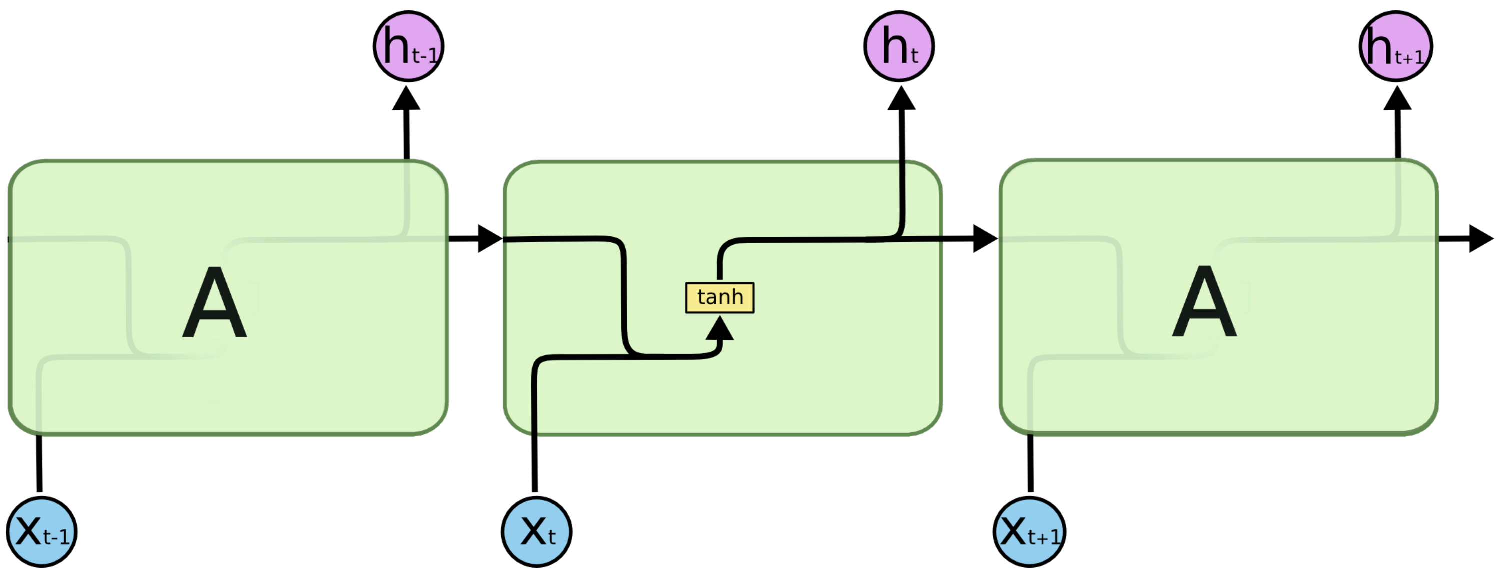
Long Short-Term Memory Units (LSTM)

- Introduction of a new structure called memory cell
- 4 components: input gate, a neuron with a self-recurrent connection, a forget gate, and an output gate
- Ability to remove or add information to the cell state through the gates

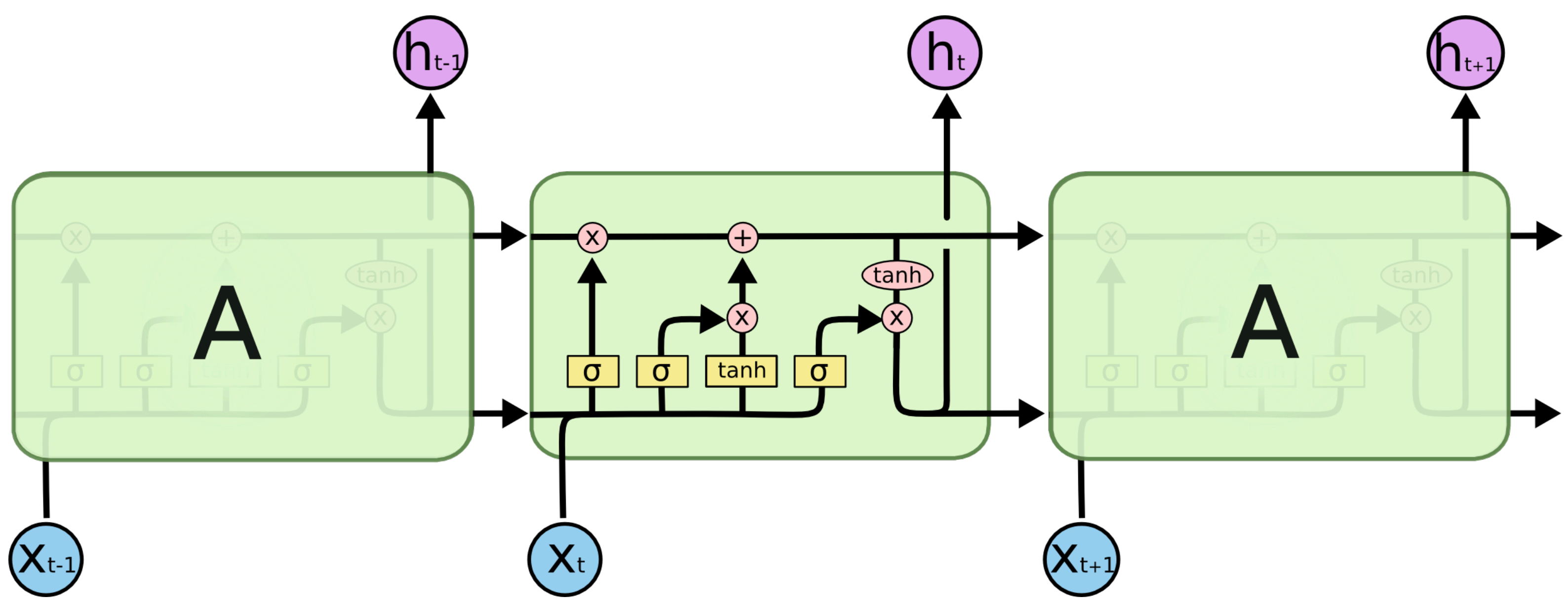


http://www.deeplearningbook.org/contents/rnn.html

Simple RNN vs LSTM

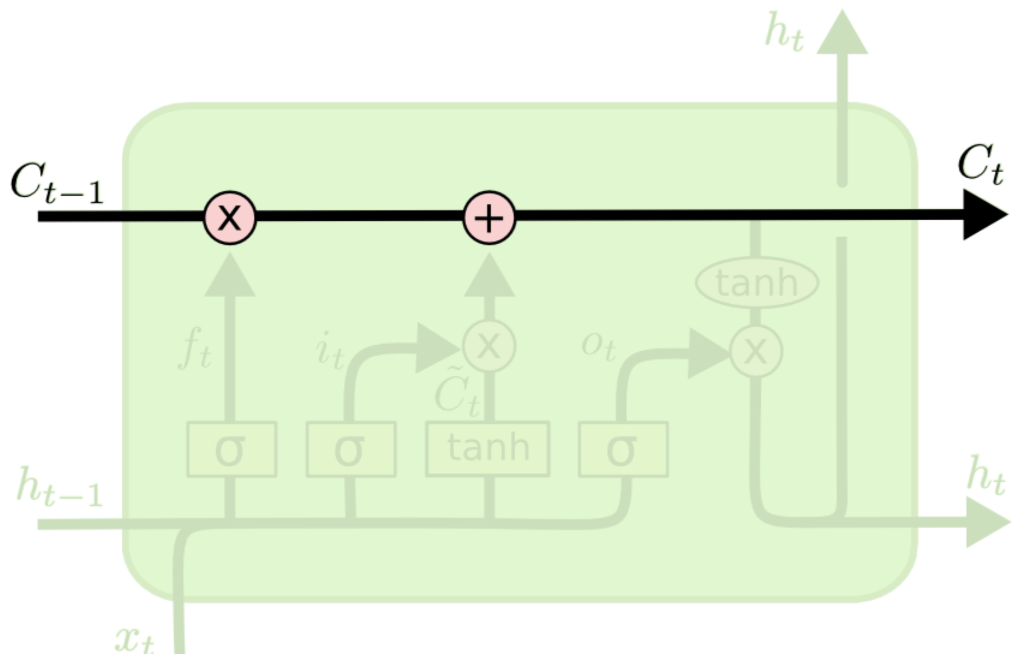


vs



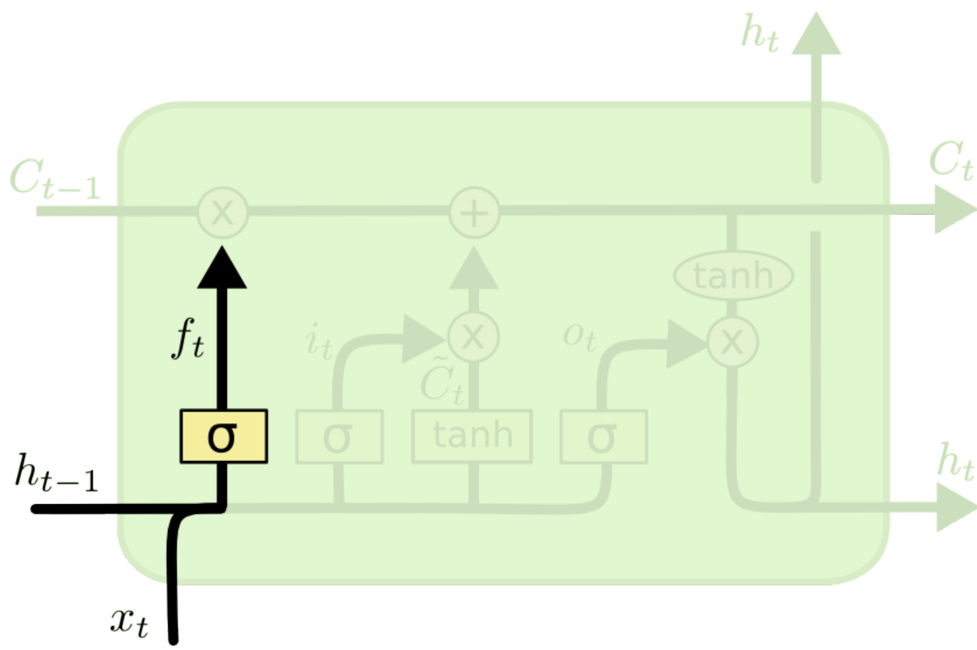
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM: Cell State



- Key idea: cell state runs through the entire chain
- Easy for information to just flow along unchanged
- Add/remove information via gates

LSTM: Forget Gate Layer

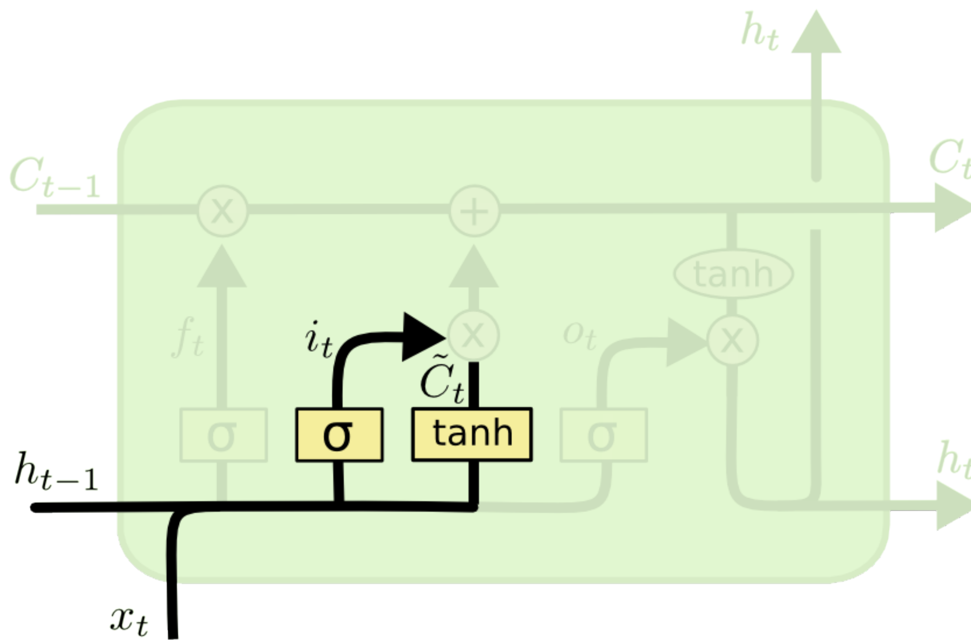


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

- Looks at the current value x_t and the previous state (h_{t-1}) and outputs a number between 0 and 1 for each number in cell state

- 1 = completely keep, 0 = completely forget

LSTM: Input Layer

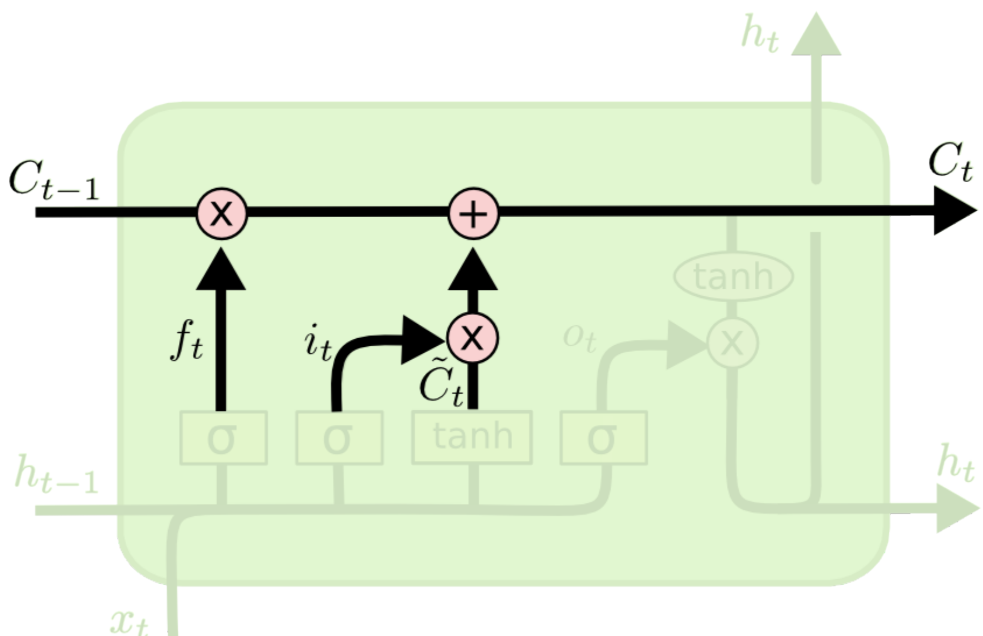


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

- Input gate layer (sigmoid layer) decides which values to update
- Tanh layer creates a new vector of candidates to be added to the state

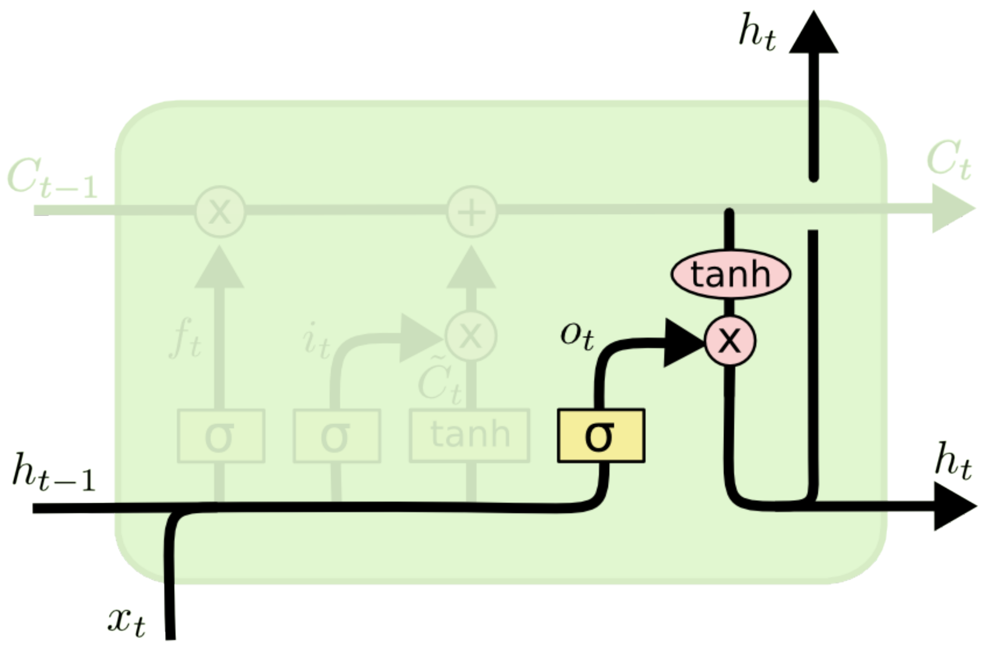
LSTM: Update Layer



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- Drop the information from forget gate layer and add information from input layer

LSTM: Output Layer



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

- Output based on cell state (filtered version)
- Sigmoid layer determines which parts of cell states to output
- Tanh pushes values between -1 and 1

Experiment: Shakespearean Writing

- Download all works of Shakespeare into single file
- Train 3-layer RNN with 512 hidden nodes on each layer
- Create samples for both speaker's names and the contents

```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

New Winter Dawns

Failure of backpropogation and ascent of SVMs, random forests led to a slump in the early 2000s

Deep Learning: The Dark Ages

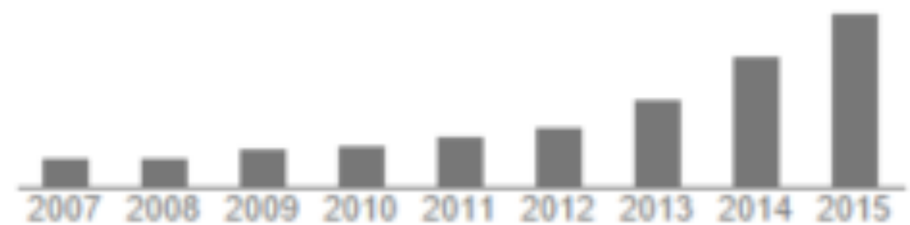
- Hinton & Bengio hatched plan to “rebrand” neural networks with deep learning
- Resurgence with “A fast learning algorithm for deep belief nets” [Hinton et al., 2006]
  - Clever way to initialize neural networks rather than randomly
- Followed by “Greedy layer-wise training of deep networks” [Bengio et al., 2007]

Deep Learning Rises Again

Citation indices	All	Since 2010
Citations	117128	47516
h-index	113	86
i10-index	273	200

Geoffrey Hinton

Citation indices	All	Since 2010
Citations	29582	17815
h-index	77	59
i10-index	179	141

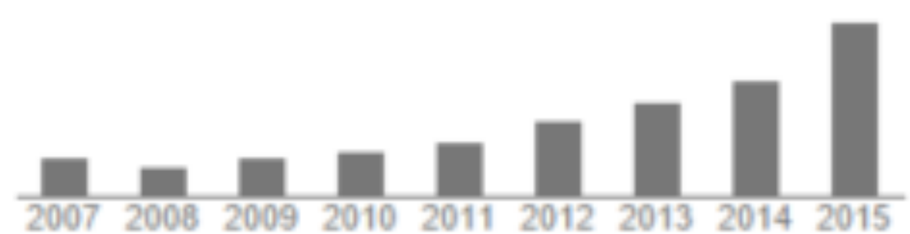


Yann LeCun

Citation indices	All	Since 2010
Citations	32736	25285
h-index	73	65
i10-index	245	200

Yoshua Bengio

Citation indices	All	Since 2010
Citations	15412	10292
h-index	64	48
i10-index	242	178



Juergen Schmidhuber

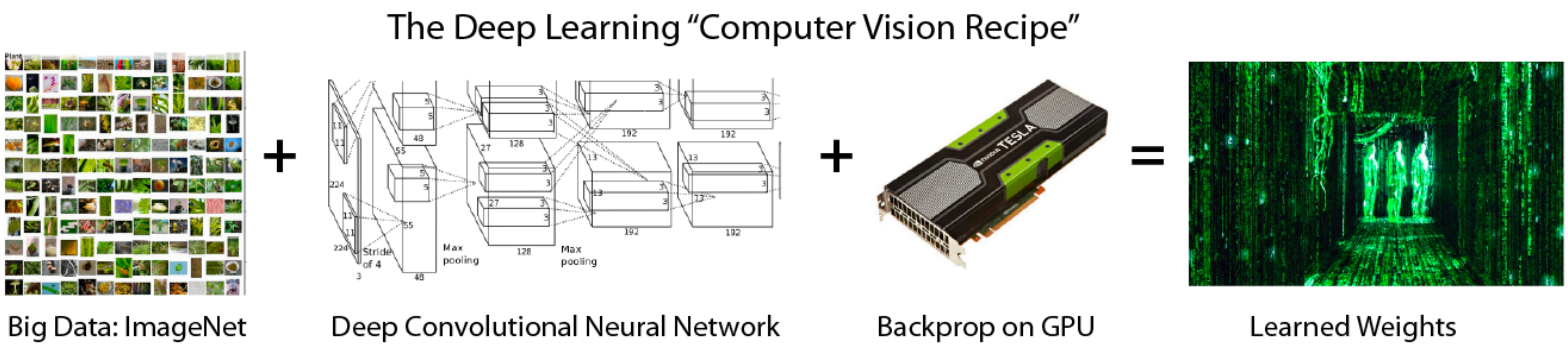
http://www.andreykurenkov.com/writing/a-brief-history-of-neural-nets-and-deep-learning-part-4/

Deep Learning Rises Again

- Labeled datasets were thousands of times too small
    - Unsupervised pre-training could help mitigate bad initialization
- Computers were millions of times too slow
- Weights were initialized in a stupid way
- Used wrong type of non-linearity

http://www.andreykurenkov.com/writing/a-brief-history-of-neural-nets-and-deep-learning-part-4/

Deep Learning Rises Again



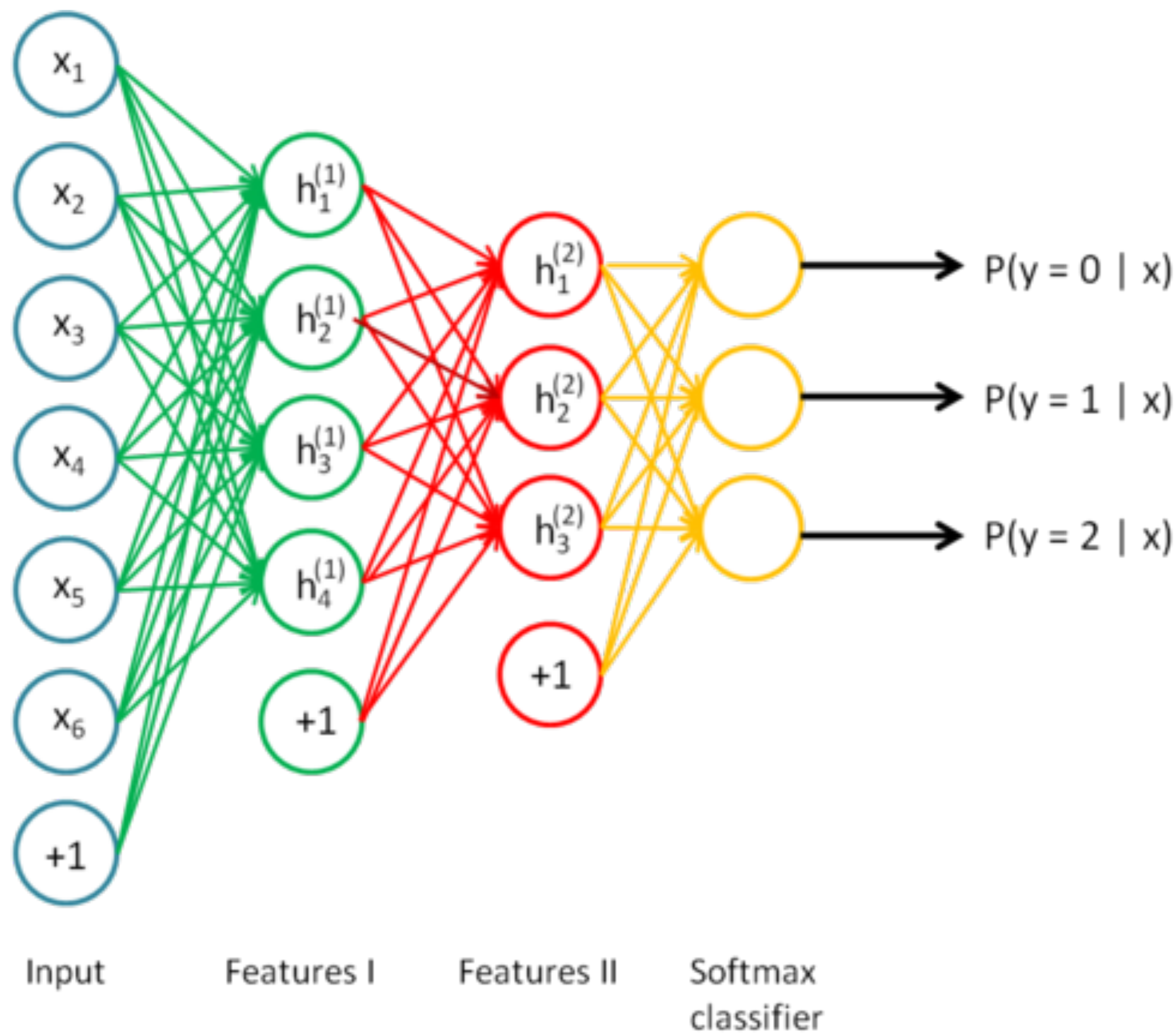
Deep learning = lots of training data + parallel computation + scalable, smart algorithms

http://www.andreykurenkov.com/writing/a-brief-history-of-neural-nets-and-deep-learning-part-4/

Stacked Autoencoders

- Network of multiple stacked auto encoders
- Can capture “hierarchical grouping” or “part-whole decomposition” of input
- Greedy training algorithm
  - Train first autoencoder using backpropogation (to learn raw inputs)
  - Train second layer autoencoder using output of first layer to learn these secondary features

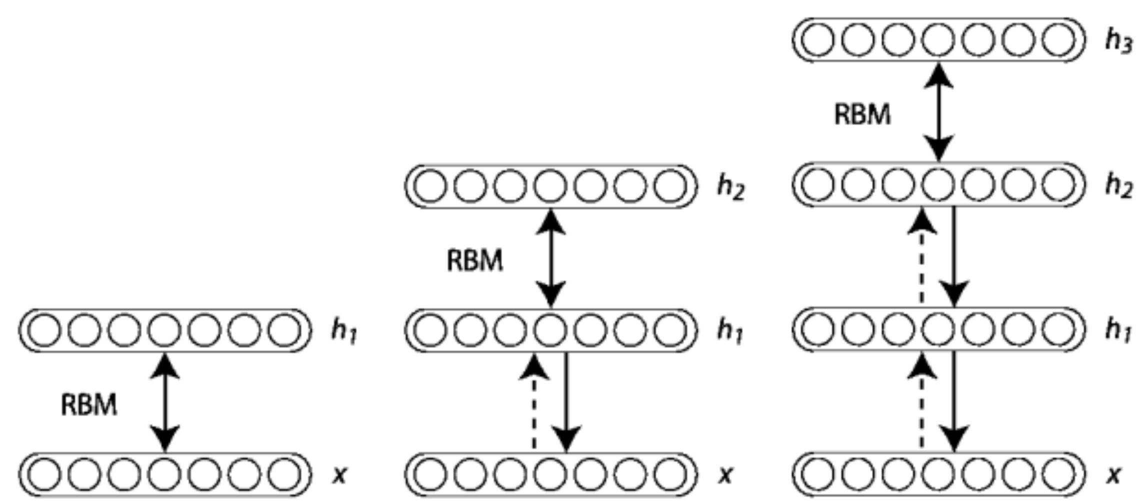
Stacked Autoencoders: Classification



http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders

Deep Belief Network (DBN)

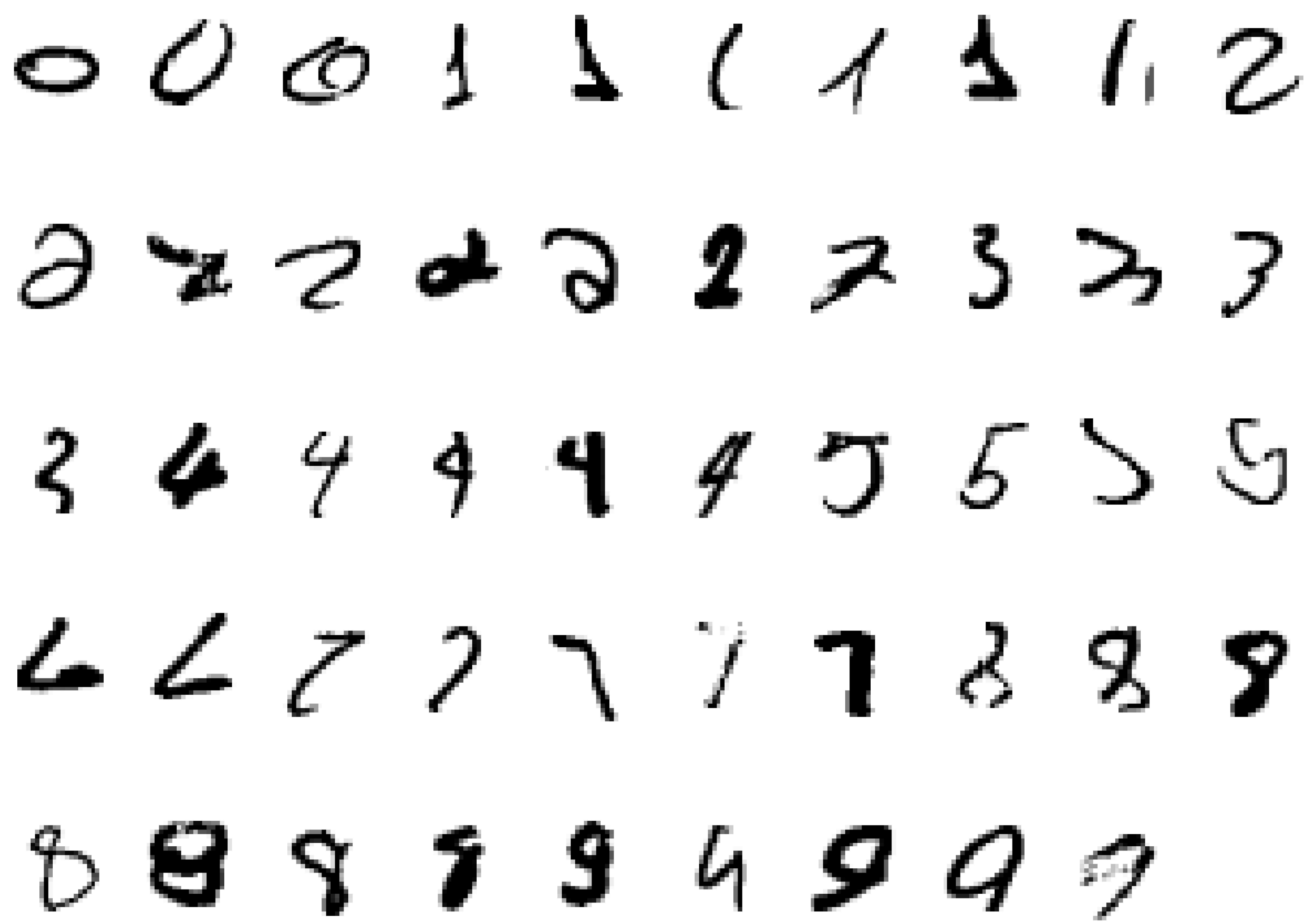
- Probabilistic generative models
- Deep architecture — multiple layers
    - Each layer contains high-order correlations between the activities of hidden features in the layer below
    - Stack RBM to get layers



http://www.pyimagesearch.com/wp-content/uploads/2014/09/deep_belief_network_example.png

DBN: MNIST Dataset Results

Examples of correctly recognized handwritten digits that the network hadn't seen before



DBN: MNIST Dataset Results

Model	Test Error
Generative model via RBM	1.25%
SVM [Decoste et al.]	1.4%
Backpropogation with 1000 hidden units [Platt]	1.6%
Backpropogation with 500 —> 300 hidden units	1.6%
K-nearest neighbor	~3.3%

https://www.cs.toronto.edu/~hinton/nipstutorial/nipstut3.pdf

Deep Learning Resources

- Website with variety of resources and pointers at deeplearning.net
- Deep Learning Tutorial by Stanford (http://ufldl.stanford.edu/tutorial/)
- Neural Networks and Deep Learning online book (http://neuralnetworksanddeeplearning.com/)
- Deep Learning book by Goodfellow, Bengio, and Courville (http://www.deeplearningbook.org/)

Deep Learning Resources

- NIPS 2015 Tutorial by Hinton, Bengio & LeCun (http://www.iro.umontreal.ca/~bengioy/talks/DL-Tutorial-NIPS2015.pdf)
- Deep Learning for Java (http://deeplearning4j.org/)
- Andrej Karpathy's Blog on Neural Networks (http://karpathy.github.io/)
- Colah's Blog on Neural Networks (https://colah.github.io/)

Deep Learning Toolkits

- TensorFlow (by Google)
- Theano (developed by academics)
- Torch (written by Lua)
- Caffe

For a reasonable comparison of the frameworks, see https://github.com/zer0n/deepframeworks/blob/master/README.md