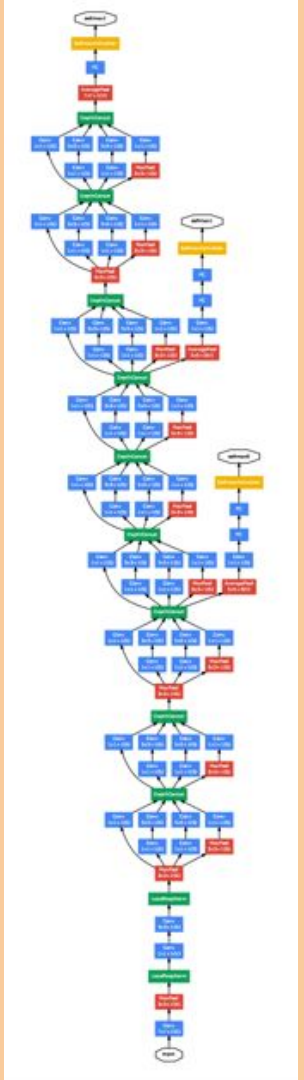
Deep Learning & Neural Networks

Iain Dunning

Software Tools for OR, IAP 2016



What will we achieve today?

- Understand the fundamentals of neural networks
- Understand some of the techniques and applications
- Learn basics of a software tool for implementation
- Learn about what is coming next in deep learning & AI
- Disclaimer!

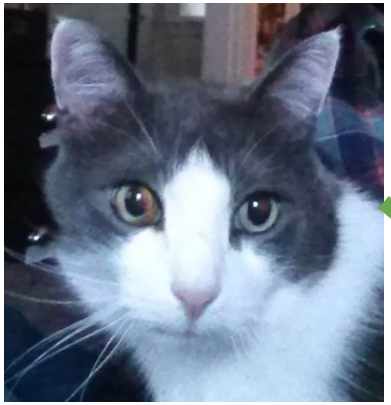
Structure

1. Introduction to function approximation & machine learning
2. What is an artificial neural network?
3. Implementing ANNs with TensorFlow (Project 1)
4. Deep learning = learning representations (Project 2)
5. Convolutional neural networks (Project 3)
6. Novel applications and new research (time permitting)

Expectation Setting

- Probably won’t use in your research
- **Good** for rich and/or unstructured (image, sound, text)
- **Not good** for small data, or for understanding data
- **Good** for approximating complicated functions
- Being used in the world today, and is growing

An **introduction** to **function approximation** & **machine learning**



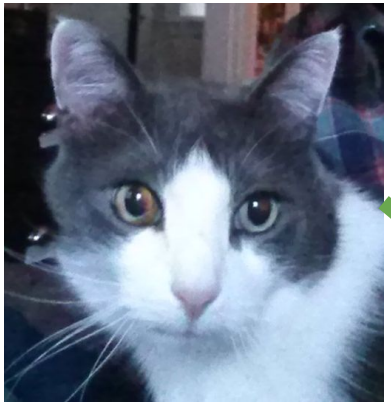
Is Cat?

CAT

NOT CAT

Wish we knew this

f(x)



Parameters β

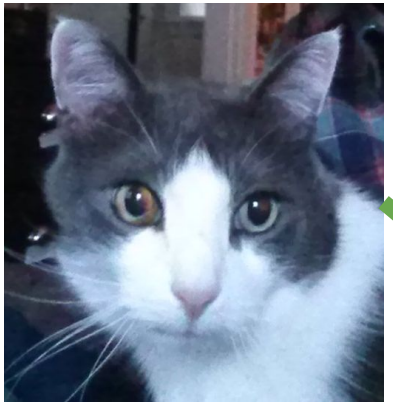
0.9
CAT

Logistic Regression

0.1
CAT

Approximates ***f***

$g(x,\beta)$



Hyperparameters λ

Parameters β

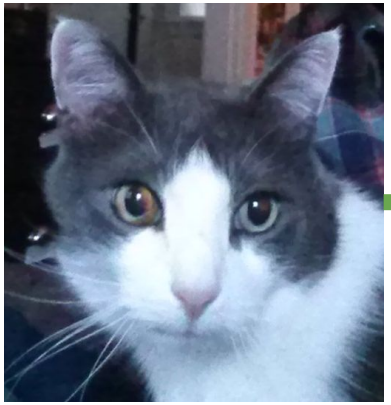
Reg. Log. Regression

0.9
CAT

0.1
CAT

Family of approximations

$g(x,\lambda,\beta)$



ϕ

Hyperparameters λ

Parameters β

Reg. Log. Regression

0.9
CAT

ϕ

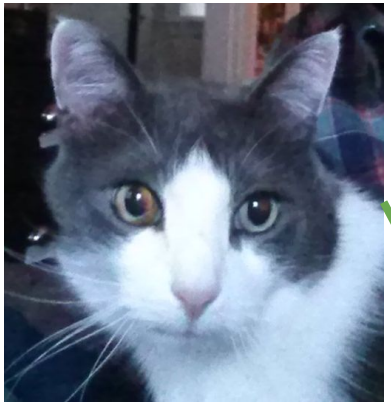
0.1
CAT

Feature engineering

$h(\phi(x),\lambda,\beta)$

What is this neural network stuff then?

- Just another **function approximation** technique
  - weighted combination of basis functions
  - some theory about “universal approximators”
- Take a **layered**/compositional approach
  - not a single-shot function - nested functions!
  - no/minimal feature engineering
  - train all layers simultaneously
- Still need for **hyperparameters = network structure**



Parameters β

LAYER

LAYER

LAYER

Learned features?

Learned features?

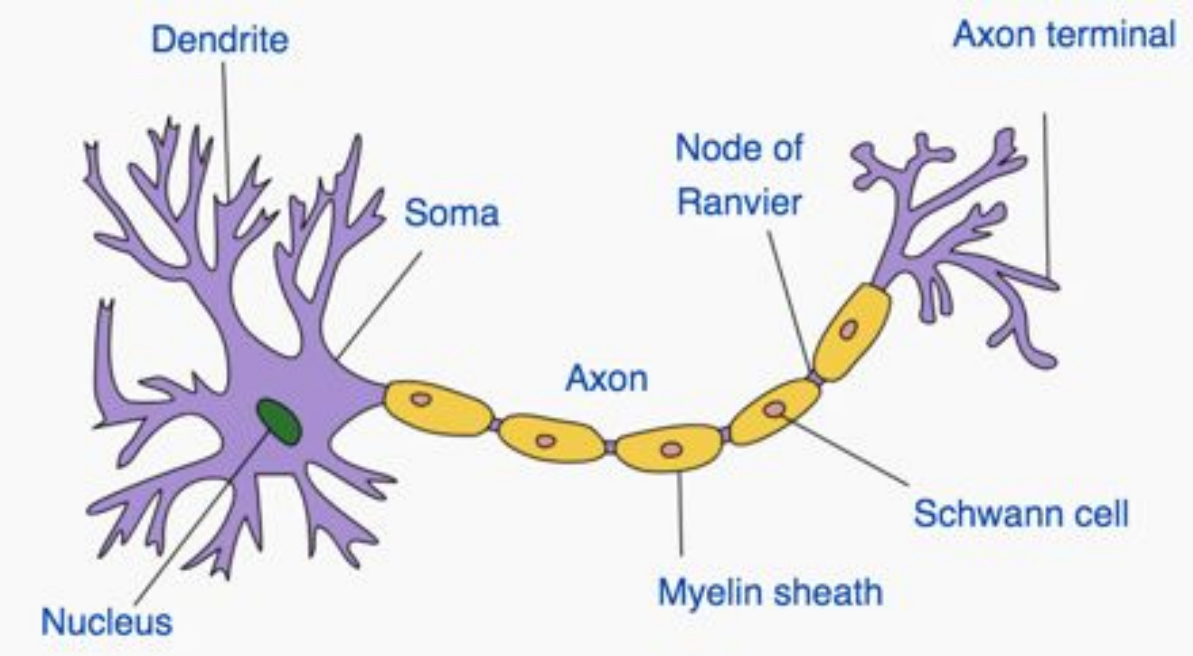
0.9 CAT

0.1 CAT

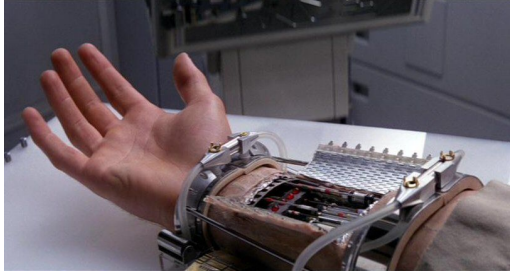
What is an

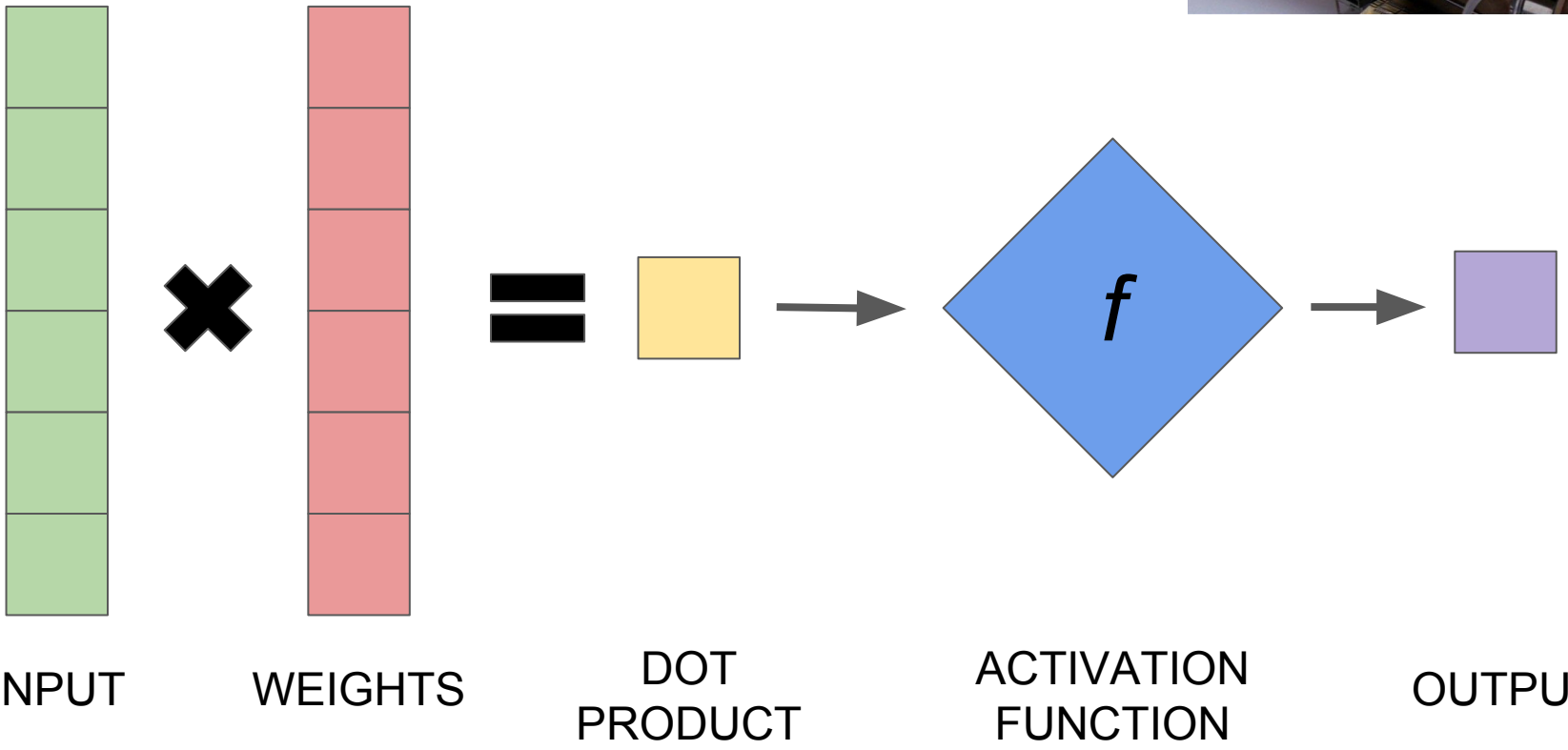
artificial neural network?

Biological Neuron?

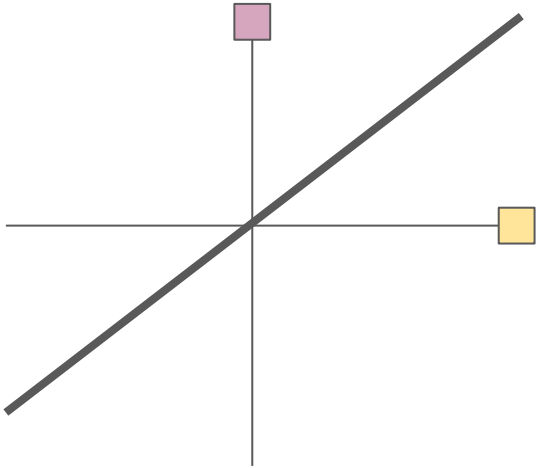


Artificial Neuron?

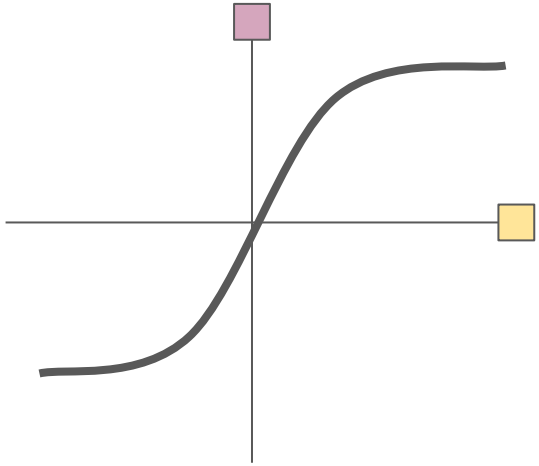




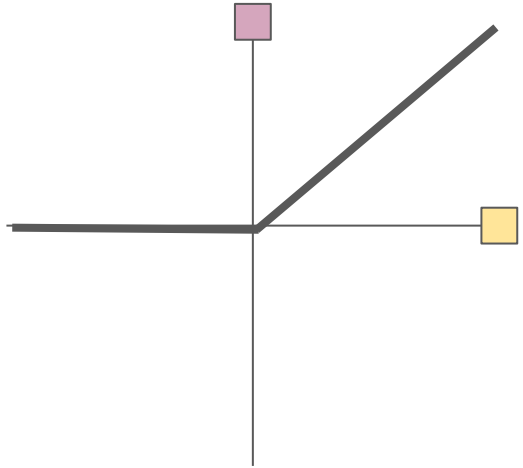
Activation Functions



LINEAR
like linear regression
(only used for final layer)



LOGISTIC / SIGMOIDAL / TANH
Smooth, differentiable, saturating functions



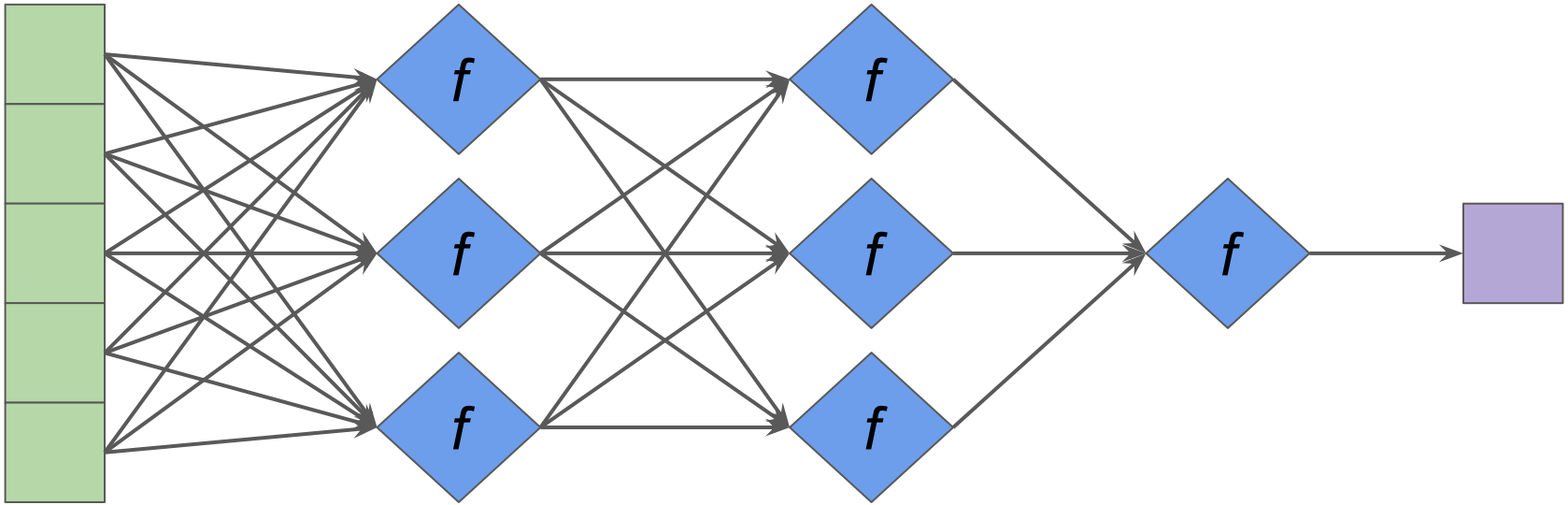
RECTIFIED LINEAR (ReLU)
Cheap to compute, popular lately

Training an Artificial Neuron

- Neuron is function $f(x,w) \rightarrow y$ data x, y weights w
- Calculate **loss/error** $L(f(x,w), y)$
- Calculate **derivative** of L with respect to w
- Use derivative to **optimize w**
- Stochastic gradient descent, momentum, RMSprop…

  many techniques!

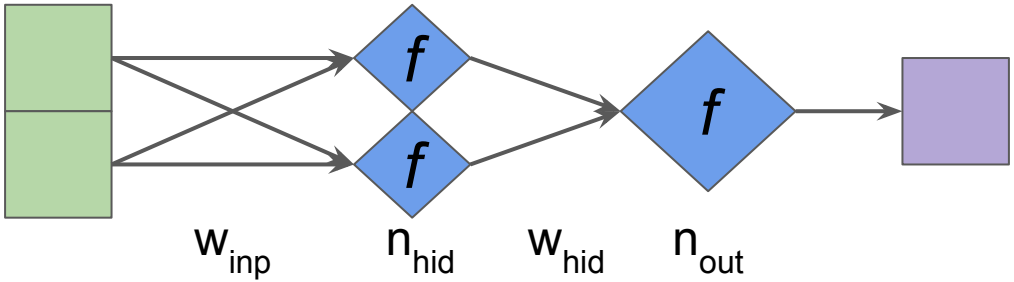
Neural Networks = Layers of Neurons



Neural Networks = Layers of Neurons

- More parameters, more complicated function
- Increased risk of overfitting - **can address**
- Outer layers learn features (**not theoretical claim**)
- Now we have ***F(x,W)***, how to get train?
- Solution: **backpropagation** (= **chain rule!**)

Backpropagation in Neural Networks



$$L(n_{out}, y) = 0.5(n_{out} - y)^2$$

$$\frac{\partial L}{\partial w_{hid}} = \frac{\partial L}{\partial n_{out}} \cdot \frac{\partial n_{out}}{\partial w_{hid}}$$

$$\frac{\partial L}{\partial n_{out}} = n_{out} - y$$

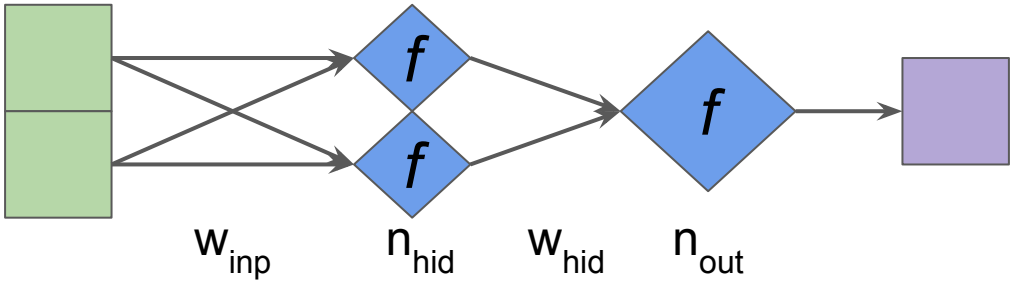
$$n_{out} = w_{hid} x + \dots$$

$$\frac{\partial n_{out}}{\partial w_{hid}} = x$$

$$\frac{\partial L}{\partial w_{hid}} = \frac{\partial L}{\partial n_{out}} \cdot \frac{\partial n_{out}}{\partial w_{hid}} = (n_{out} - y) \cdot x$$

Chain rule!

Backpropagation in Neural Networks



$$n_{out} = w_{hid}x + \dots$$
$$= w_{hid}n_{hid} + \dots$$

$$\frac{\partial L}{\partial w_{inp}} = \frac{\partial L}{\partial n_{out}} \cdot \frac{\partial n_{out}}{\partial n_{hid}} \cdot \frac{\partial n_{hid}}{\partial w_{inp}}$$

$$\frac{\partial n_{out}}{\partial n_{hid}} = w_{hid}$$

$$n_{hid} = w_{inp}x + \dots$$

$$\frac{\partial n_{hid}}{\partial w_{inp}} = x$$

$$\frac{\partial L}{\partial w_{inp}} = \frac{\partial L}{\partial n_{out}} \cdot \frac{\partial n_{out}}{\partial n_{hid}} \cdot \frac{\partial n_{hid}}{\partial w_{inp}}$$
$$= (n_{out} - y) \cdot w_{hid} \cdot x$$

Training Neural Networks

- Can calculate derivatives **automatically**
- Opportunities for **parallelism**
- Re-using intermediate values and other efficiencies
- Painful do DIY - if only there was a framework...
- Aside: evidence that this happens in brains? https://en.wikipedia.org/wiki/Neural_backpropagation

Implementing ANNs with **TensorFlow**

Frameworks for Implementing Neural Networks

theano

dmlc
mxnet

Caffe

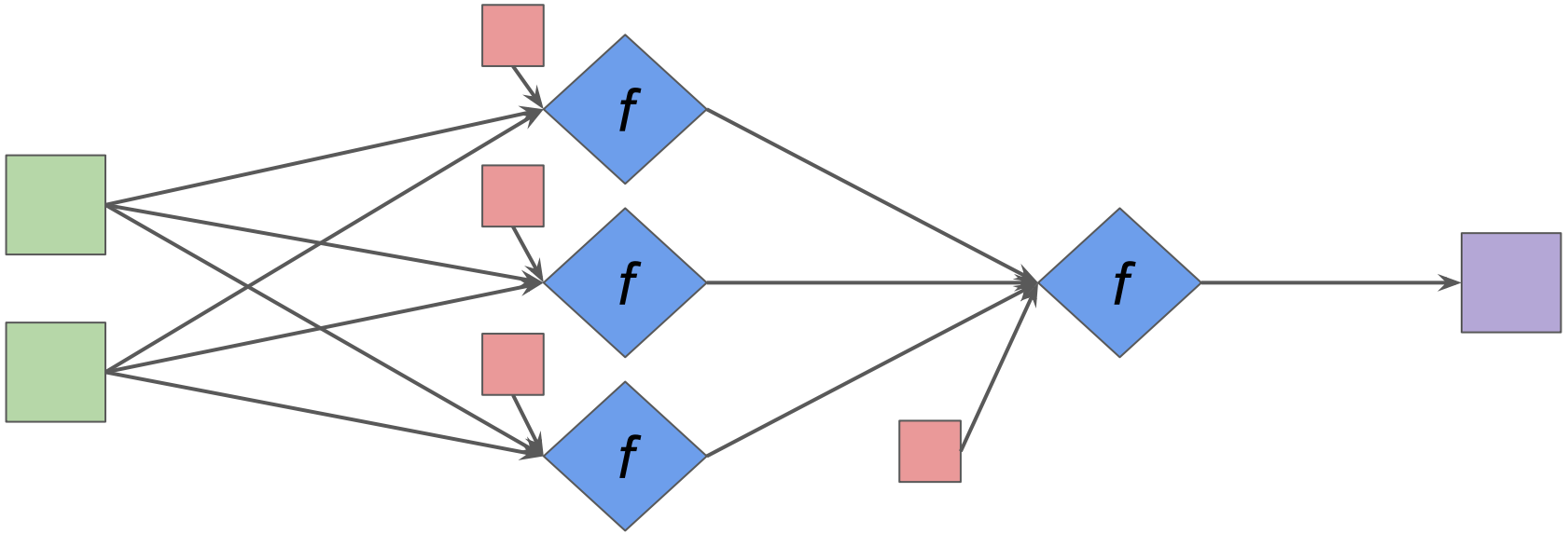
Today: Google TensorFlow (via Python)

- Use TF to describe computations as a **graph**
- TF **schedules** computations on devices - CPU, GPU...
- Performs **automatic differentiation** (like JuMP!)
- Every framework unique, but similar ideas
- Python is easy to use

Project 1 - Nonlinear Regression

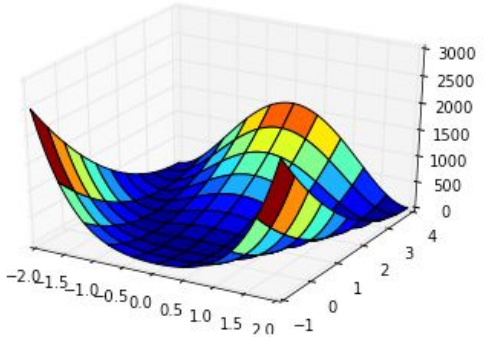
- **Goal**: approximate an unknown function from data
- **Plan**: generate data, create ANN, fit it
- **Live coding** in IPython - follow along!
- **Play** with extending the code

Project 1 - Nonlinear Regression



Project 1 - Assignments - Try...

- … changing the activation function
- … changing the number of hidden neurons
- … changing the learning rate
- … changing the training data
- … to induce overfitting
- … adding an additional hidden layer
- … understanding how ReLU performs with more layers?



Deep Learning

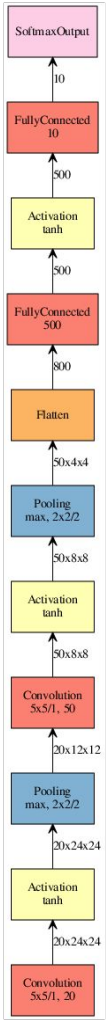
=

Learning Representations

What is this Deep Learning stuff?

- Could just have 1 or 2 layers and make wider...
- Experimentation revealed **deeper** architectures better
- Not homogenous: different nonlinearities throughout
- One explanation: outer layers learn features, getting more and more abstract (e.g. ear, fur -> furry ear -> cat)

“LeNet” 1998:



“AlexNet” 2012:

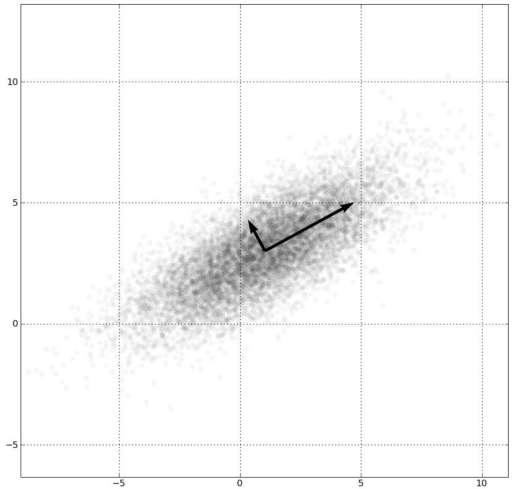
“GoogLeNet” 2014:

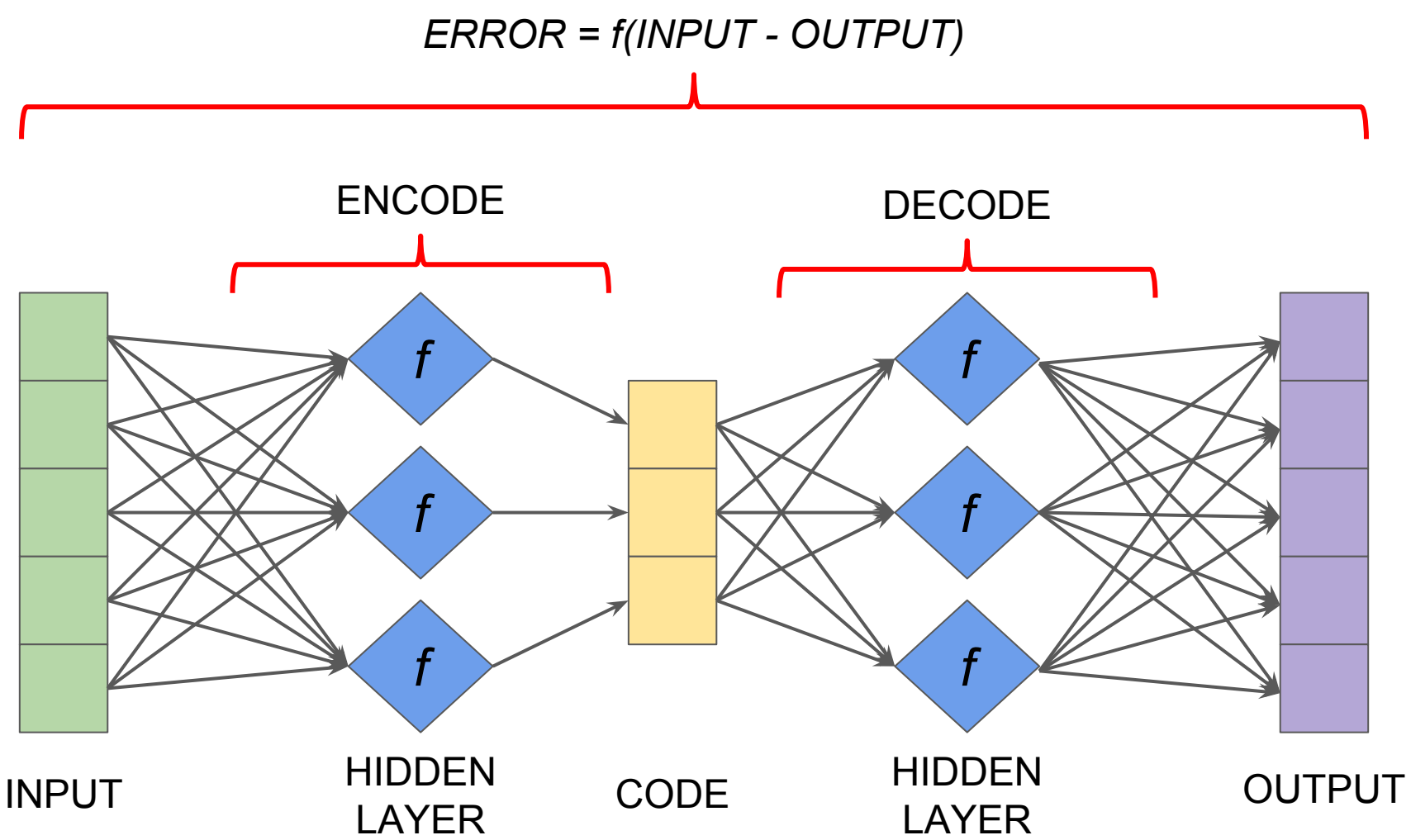
“Inception v3” 2015:

Source: http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/

Learning representations - autoencoders

- Images etc. can be viewed as **high-dimensional vectors**
- **Hypothesis**: can **project** high-dim. vectors into low-dim. without losing much
- **PCA** is a classic way of doing this
- Use ANN to **encode** and **decode**!




ERROR = f(INPUT - OUTPUT)
ENCODE
DECODE
f
f
f
f
f
f
INPUT
HIDDEN LAYER
CODE
HIDDEN LAYER
OUTPUT

Project 2 - Can robots learn binary?

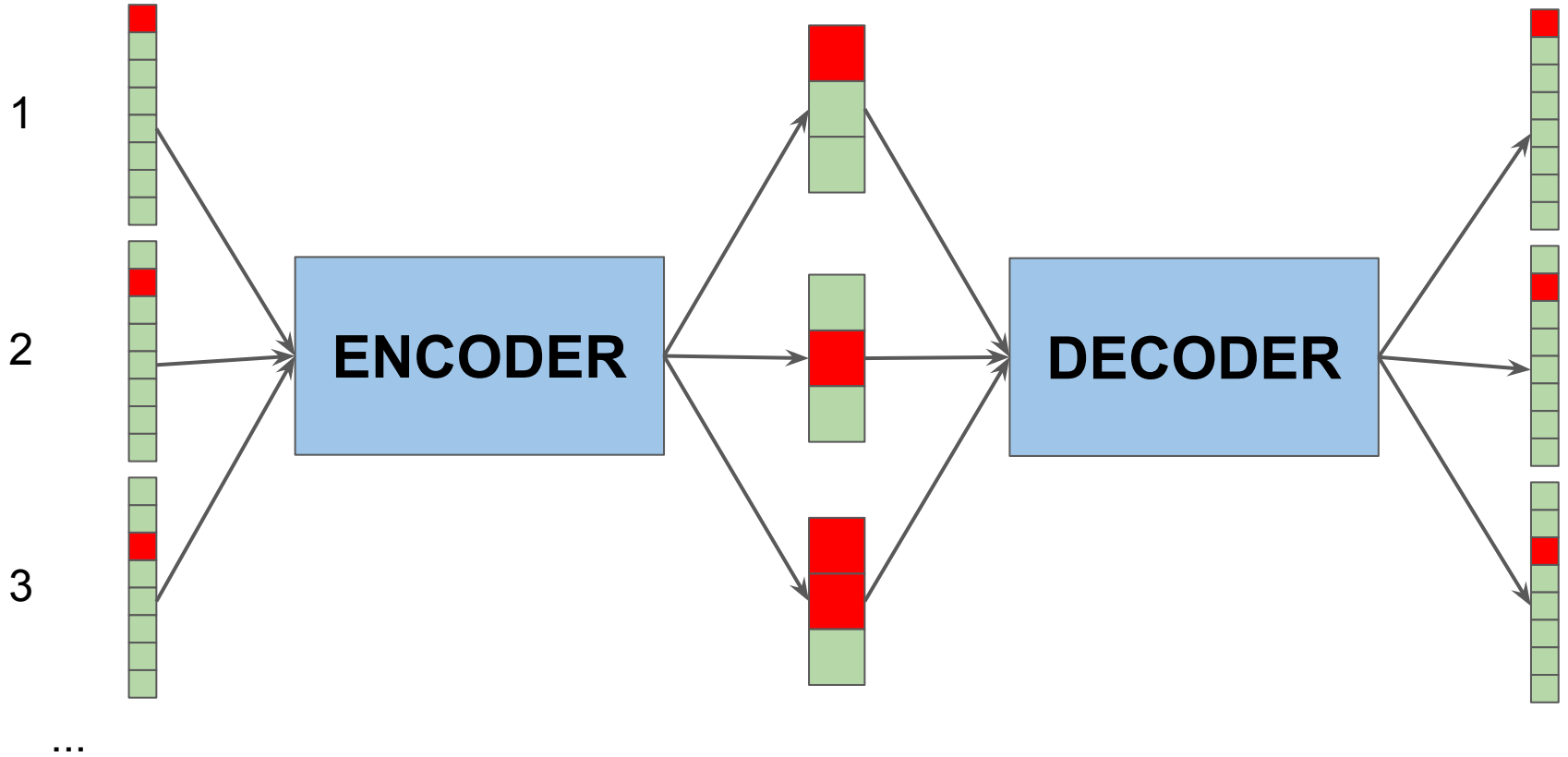
- Consider “**one-hot” vectors** for the numbers 1 to 8
  - [1,0,0,0,0,0,0,0] [0,1,0,0,0,0,0,0] [0,0,1,0,0,0,0,0] …
- Can represent these vectors in 3D, e.g.
  - [0,0,0] [0,0,1] [0,1,0] [0,1,1]

    [1,0,0] [1,0,1] [1,1,0] [1,1,1]
- Can a neural network learn that?

Concept from: https://blog.wtf.sg/2014/05/10/recursive-auto-encoders-an-introduction/

Concept from: https://blog.wtf.sg/2014/05/10/recursive-auto-encoders-an-introduction/

Project 2 - Can robots learn binary?



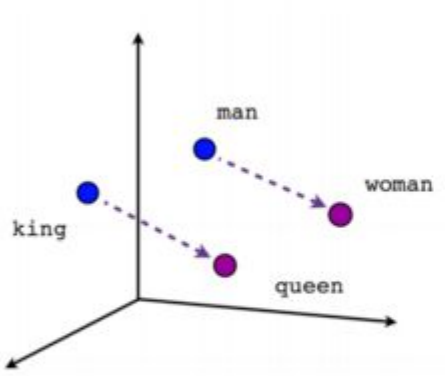
Project 2 - Assignments - Try...

- … changing the activation functions
- ... using fewer hidden units - what can you recover?
- ... using multiple layers
- … using same weight for encoding and decoding

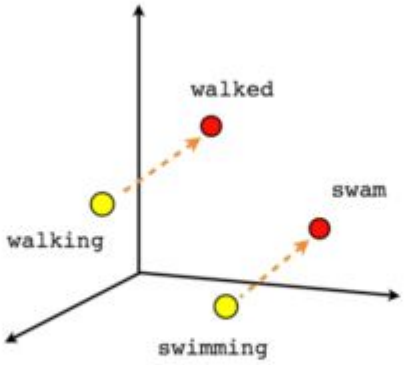
Recent work on representations: *word2vec*

- Given a text corpus, embed words in n-dimensional space
- “Similar” words should be “close” in this space
- Use a (fairly simple) neural network to encode
- Results are…

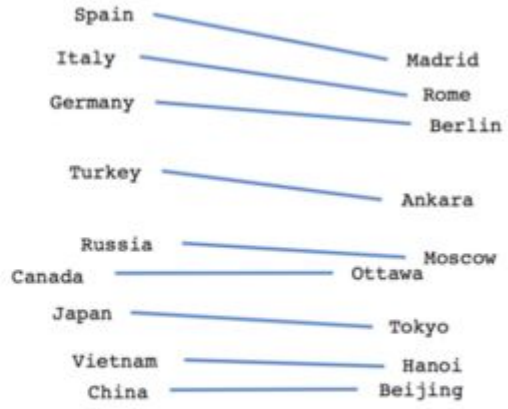
Paper: http://arxiv.org/pdf/1301.3781.pdf
TensorFlow example: https://www.tensorflow.org/versions/master/tutorials/word2vec/index.html



Male-Female



Verb tense



Country-Capital

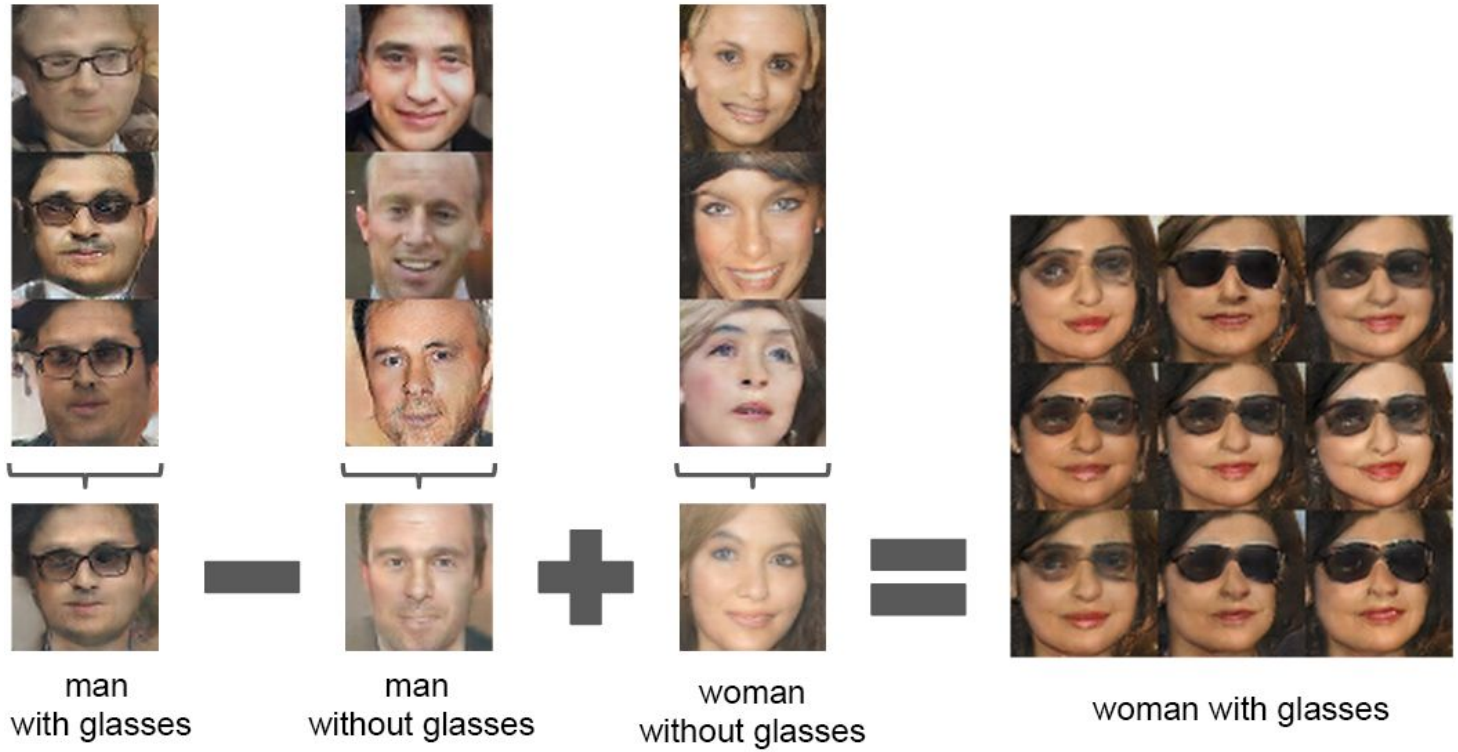
PARIS - FRANCE + ITALY = ROME

JAPAN - SUSHI + GERMANY = BRATWURST

BIG - BIGGER + COLD = COLDER

https://github.com/Newmu/dcgan_code

“Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”



Convolutional

Neural Networks

How can we classify images?

- **Challenge:**

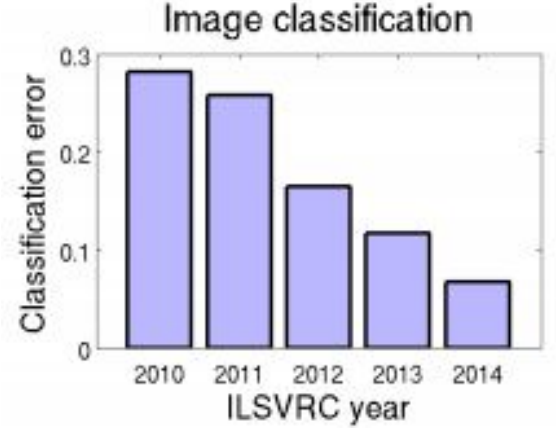
  Classify a 256-by-256 RGB image as either a cat or a dog

- Input dimension: roughly **200k** (256*256*3)
- Each neuron in first layer would need 200k weights…
- Even if could train it, does it make sense?

More info: http://arxiv.org/abs/1409.0575

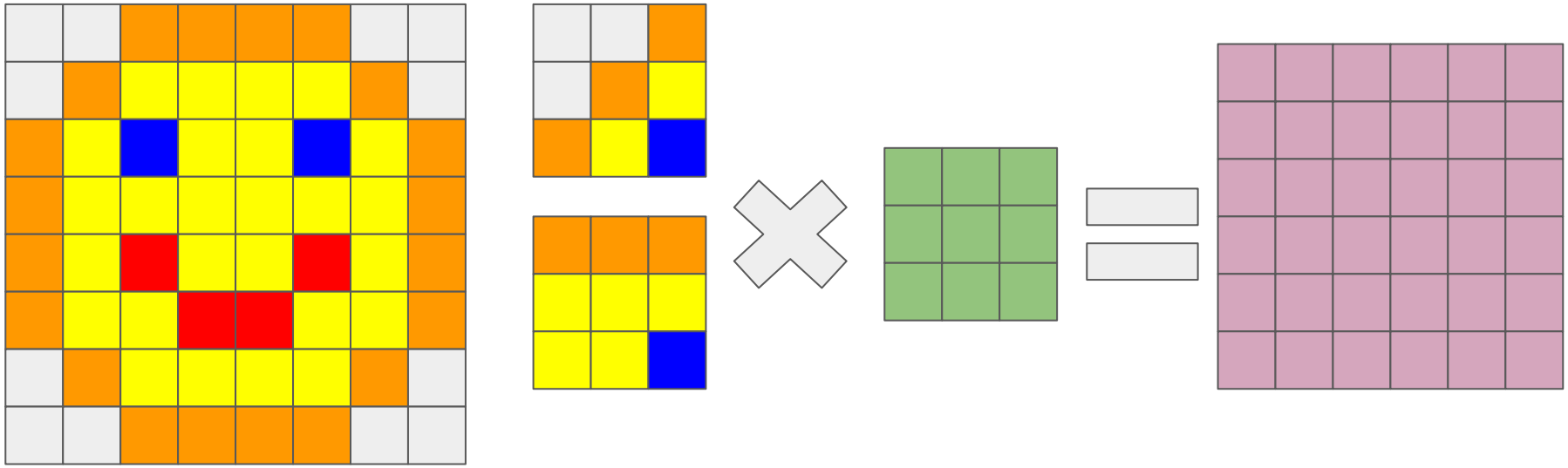
ImageNet Classification Challenge (annual)

- 1000 object classes, Approximately 1.2 million training, 50k validation, and 100k test images.
- Big improvements: 28.2% error rate to 6.7% error rate - human level! How? CNNs!

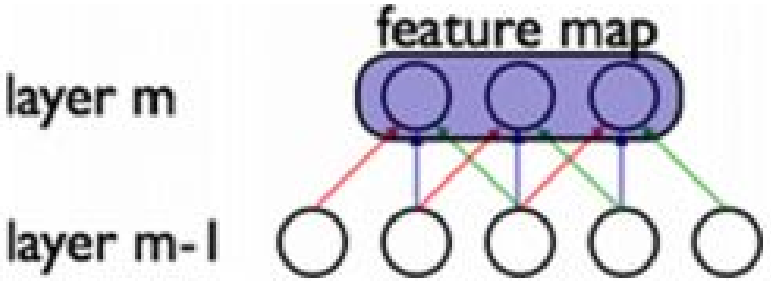


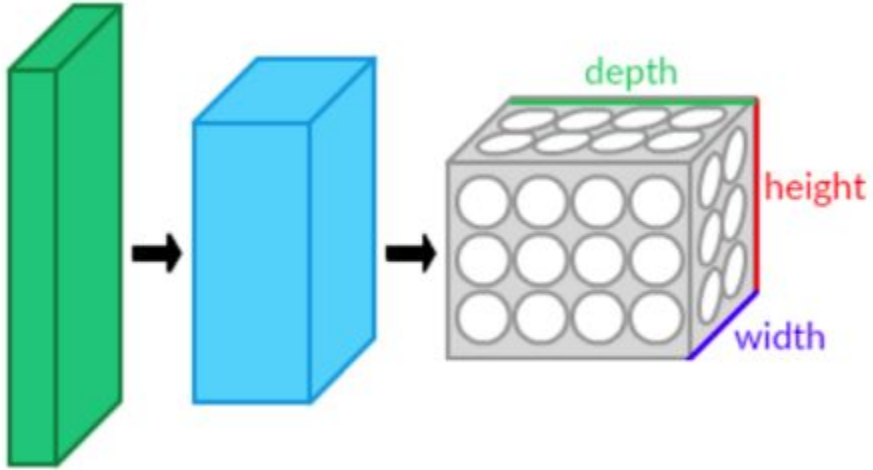
Convolutional neural networks

- **Share weights** by convolving weights with the input



Other visualizations of ConvNets

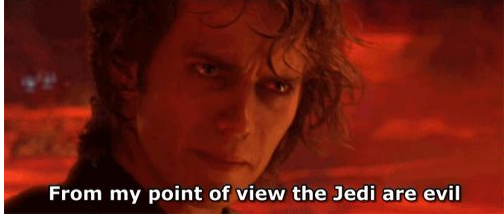




http://deeplearning.net/tutorial/lenet.html

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

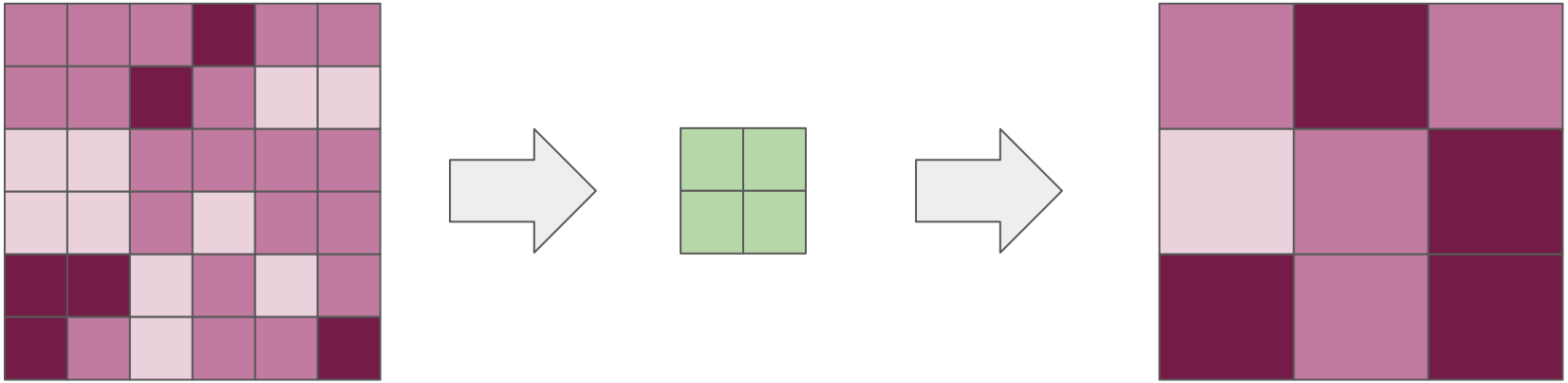
Convolutional neural network



- Shared weights - a filter/kernel - “slid” across input
- Can learn multiple filters simultaneously
- Can vary size, stride, activation function
- Filters might learn, e.g. edges detection, colour patterns - previously hardcoded features!
- Far fewer parameters, suitable for parallel computation

Pooling layers in CNN, deep structures

- Maybe a filter learns to detect **eyes**
- Precise location of eye is maybe unimportant
- Use a **pooling** layer to downsample

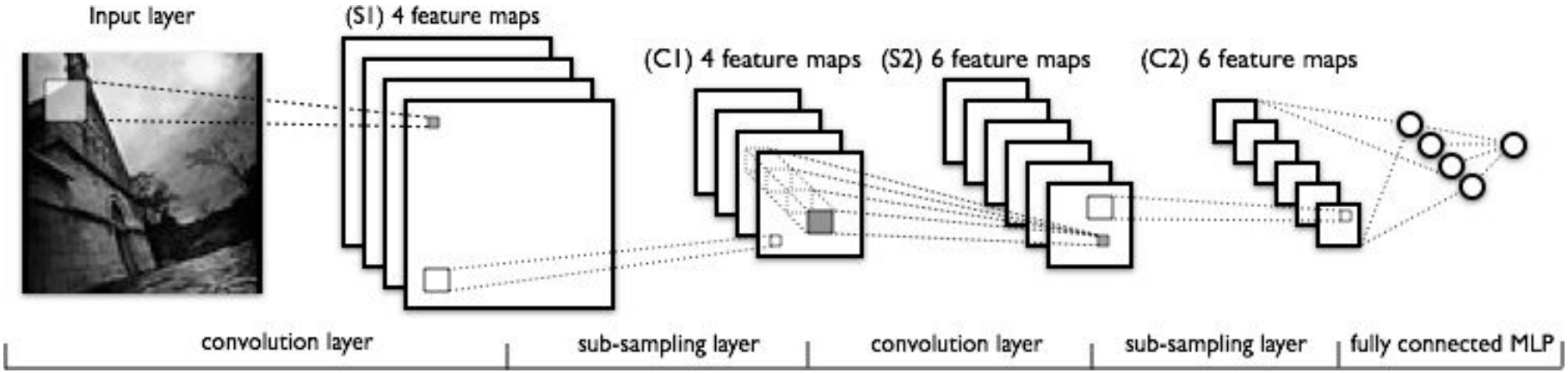


CONVOLUTION OUTPUT

2x2 MAX POOL

POOLING OUTPUT

Early ConvNet: “LeNet-5”

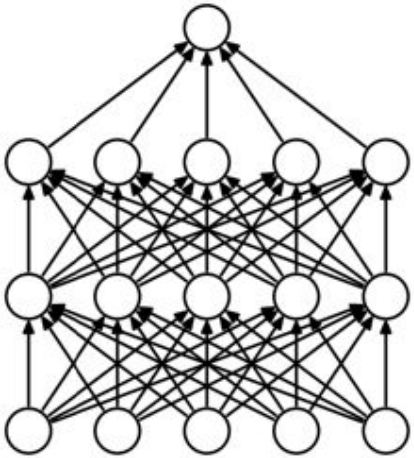


http://deeplearning.net/tutorial/lenet.html

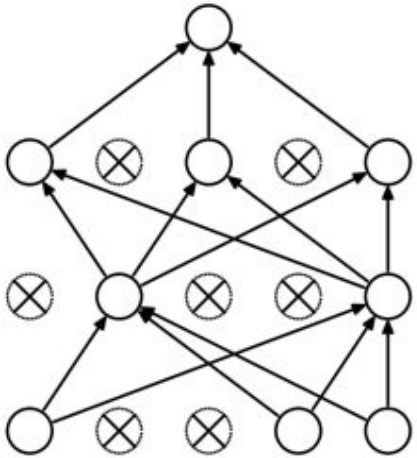
Project 3 - MNIST Digit Recognition

- Classify hand-written images of digits 0 to 9
- Classic “MNIST” data set used widely in ML research
- Will use convolutional neural networks!

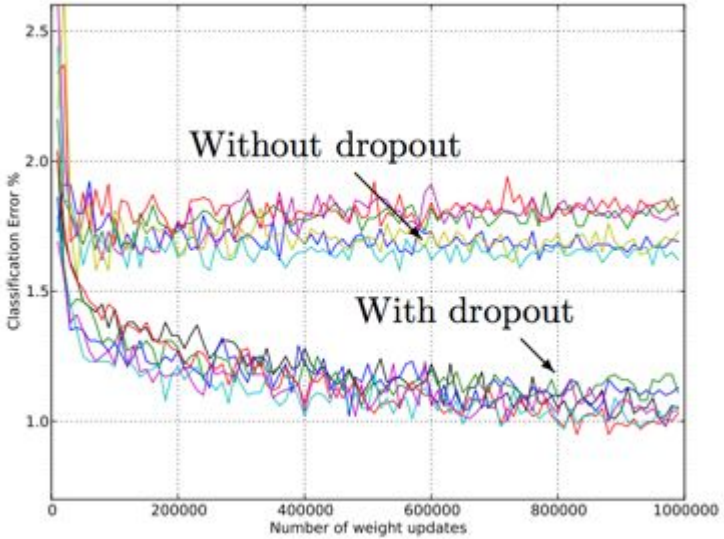
Dropout - cheap regularization for ANN



(a) Standard Neural Net



(b) After applying dropout.



http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Dropout for Linear Regression = Ridge Reg.

$$R \in \{0,1\}^{N \times D}$$

$$\underset{\mathbf{w}}{\text{minimize}} \quad \mathbb{E}_{R \sim \text{Bernoulli(p)}} \left[||\mathbf{y} - (R * X)\mathbf{w}||^2 \right]$$

=

$$\underset{\mathbf{w}}{\text{minimize}} \quad ||\mathbf{y} - pX\mathbf{w}||^2 + p(1-p)||\Gamma\mathbf{w}||^2$$

Project 3 - Assignments - Try...

- … changing the filter sizes, max pool size
- ... adding or removing layers
- ... using SGD
- … varying the amount of dropout (with iteration?)
- … varying the batch size

Novel applications

and

new research

https://www.cs.toronto.edu/~vmnih/docs/dqn.pdf

Reinforcement Learning: Deep Q-Learning

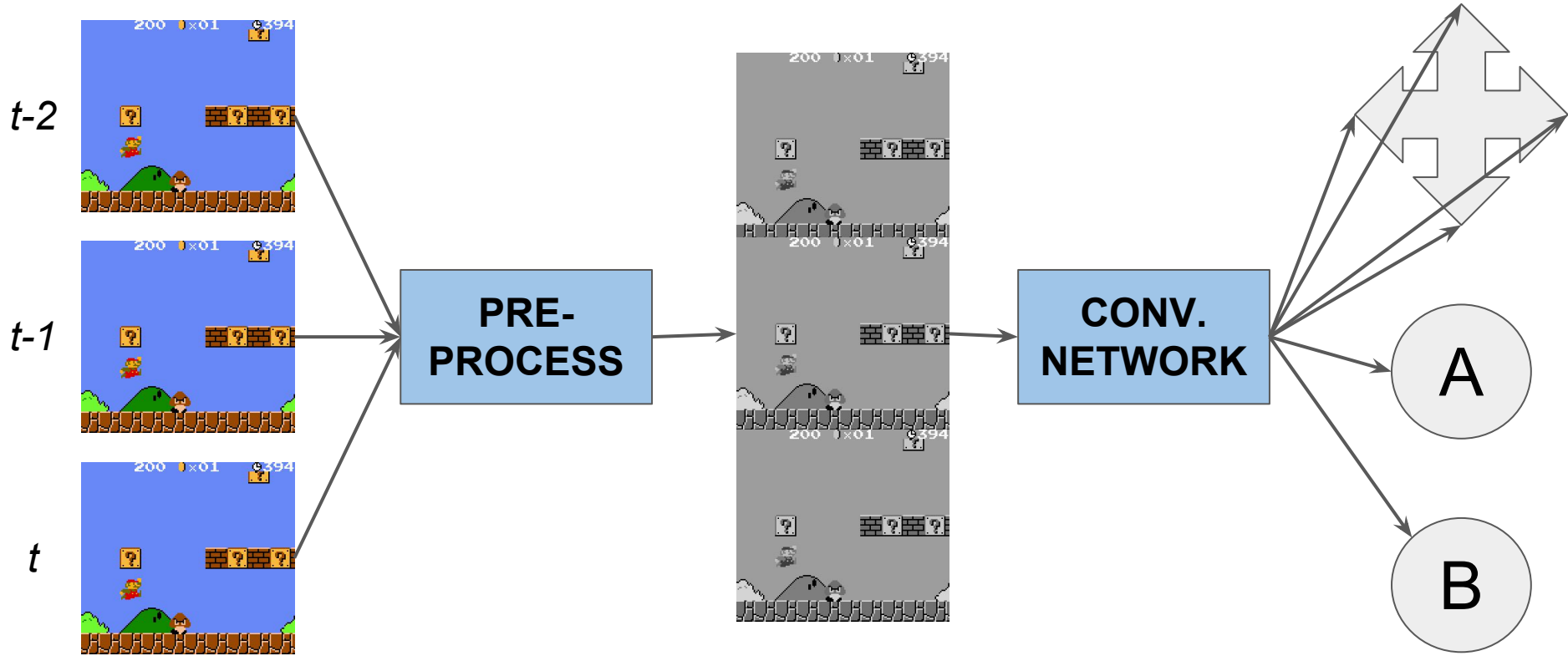
- Setting: an **agent** interacting with an environment
- At each time-step, take an **action** (from a set of actions)
- At each time-step, observe some **state**
- May be hidden state - all we have is sequence of observed states and actions taken = MDP

Reinforcement Learning: Deep Q-Learning

- Actions will give us some **reward** (may be 0)
- Agent's goal: **maximize total (discounted?) rewards**
- Need **something** that tells agent best action to take, so we can....

https://www.youtube.com/watch?v=Q70uIPJW3Gk

Reinforcement Learning: Deep Q-Learning



Training the network

- Net inputs = screen(s) - Net outputs = scores
- **Don’t know** “true” scores (just immediate reward)
- **Approximate** using the current network output

Deep Q-Learning Algorithm

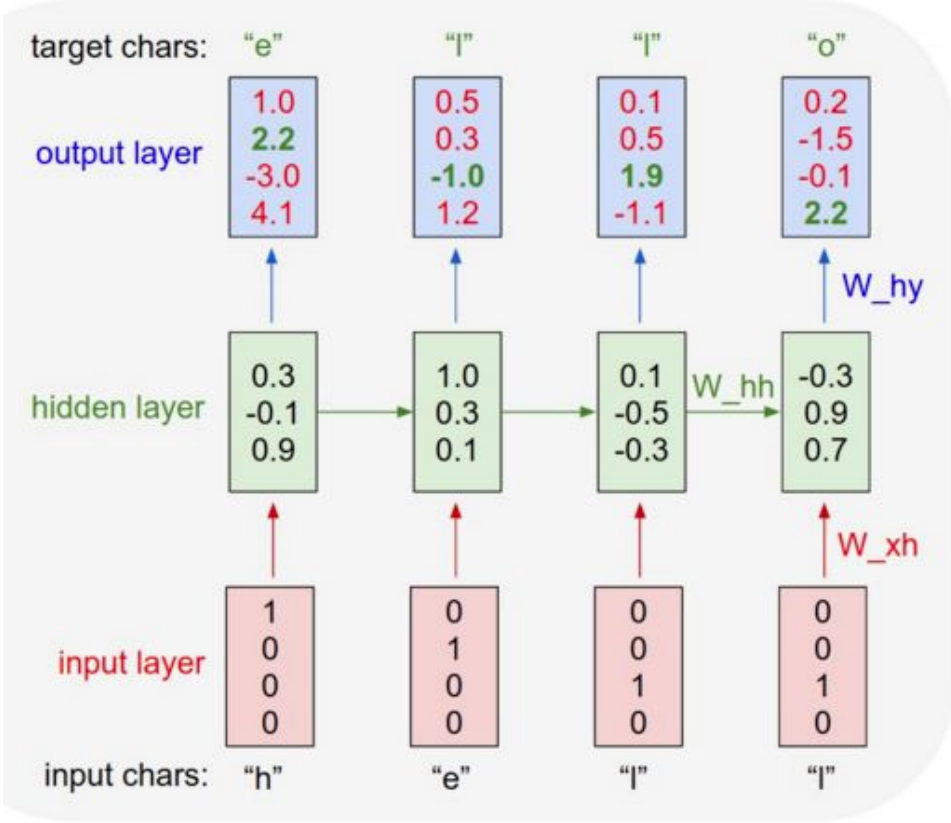
Initialize **replay memory** and **network**
For each **game**, and for each **time-step**

- Select best action according to net now (***before***)
- Execute action, receive **reward**, move to next state (***after***)
- Store the **experience *(before, action, reward, after)***
- Select a random subset of past experiences
- For each experience, the “true” output is
  reward + discount * (max output of NETWORK(after))
- Update network (i.e. gradient descent) using error

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

Recurrent Neural Networks

- Many inputs, output depends on all
- Very successful lately: translation, question answering, “reasoning memories, attention”
- “Neural Turing Machines”

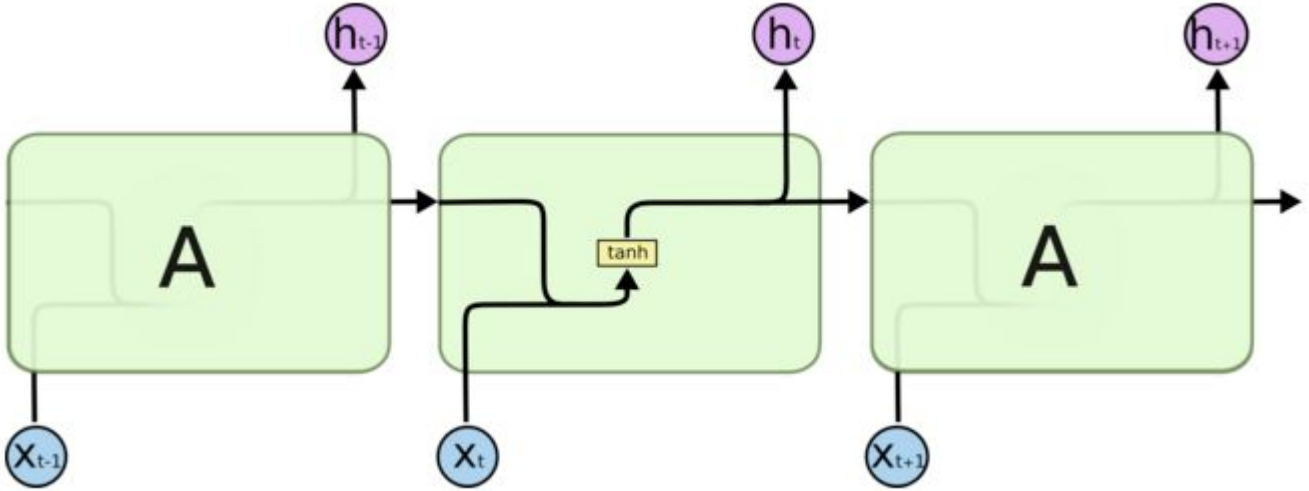


https://soundcloud.com/seaandsailor/sets/char-rnn-composes-irish-folk-music

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

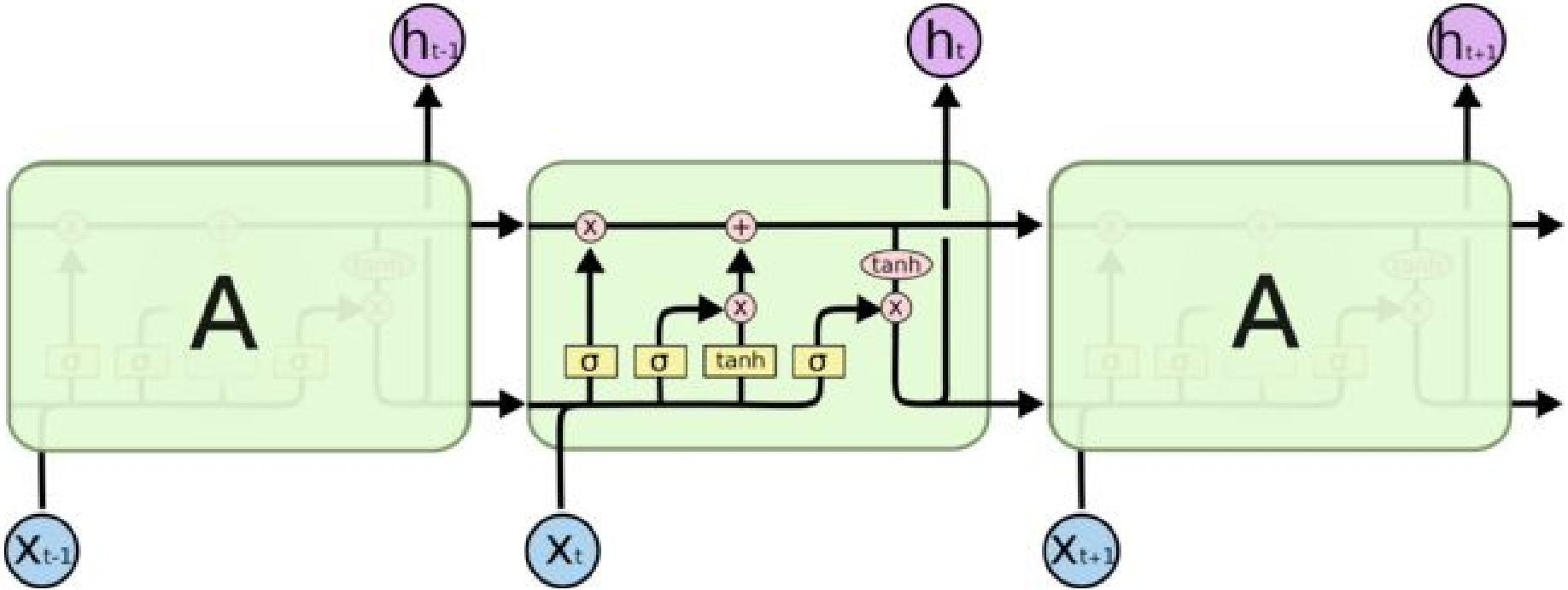
LSTM - Long Short Term Memory

- Flaw in RNNs: output can be far from important input
- Gradients decay, noise accumulates…
- Solution: let the network learn what to remember, to forget
- RNN:



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

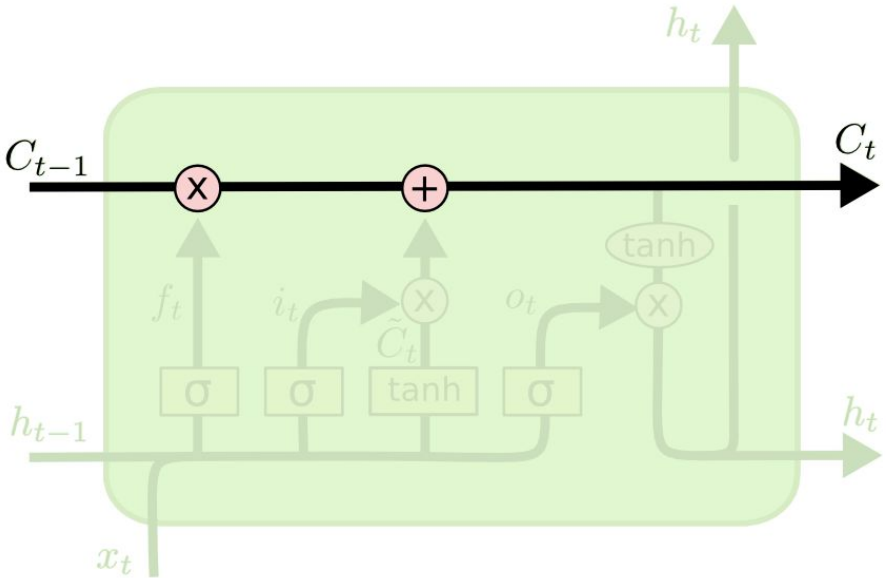
LSTM



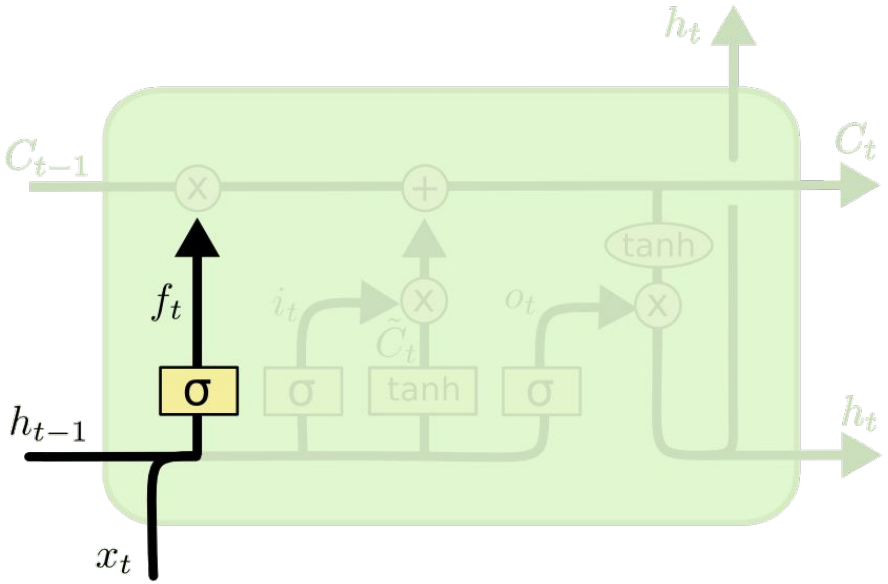
Taken from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM - “Cell State”

- Passes through all the hidden cells - **memory**

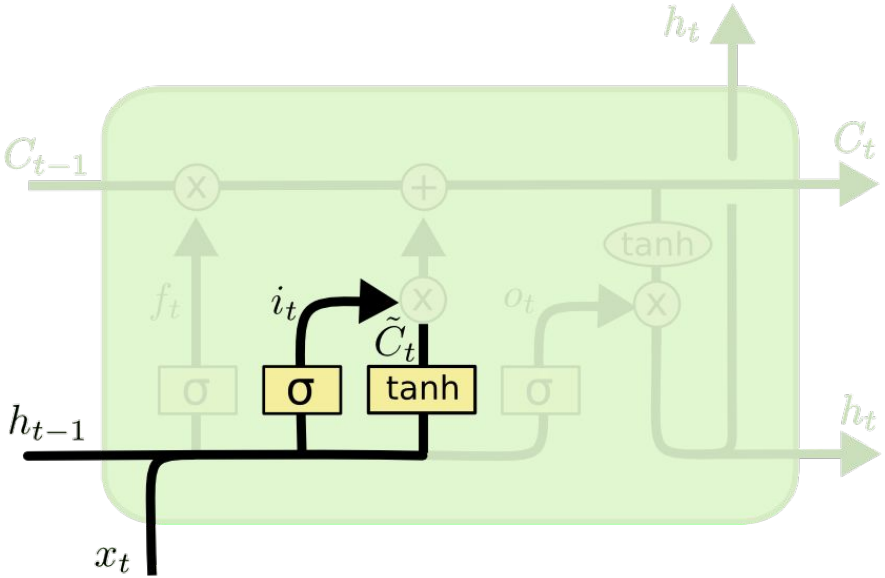


LSTM - “Forget”



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

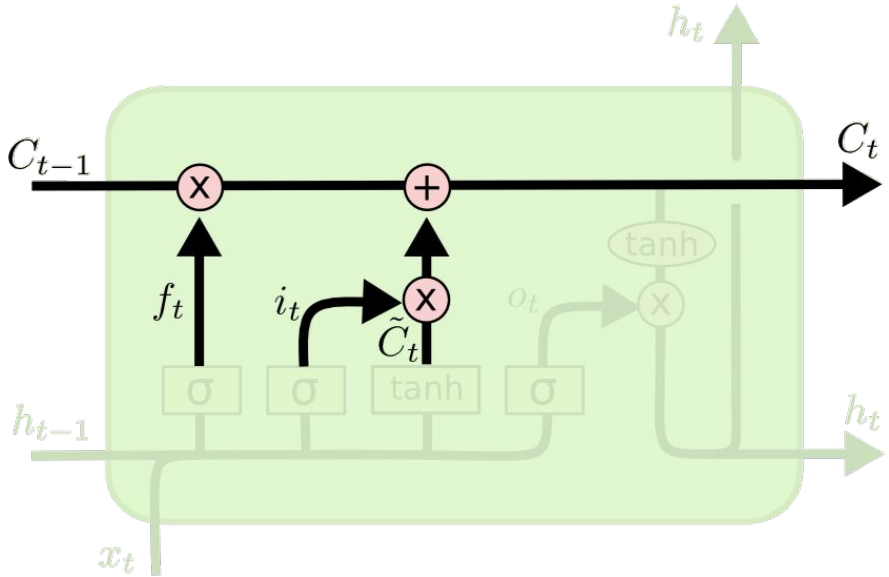
LSTM - “Remember”



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

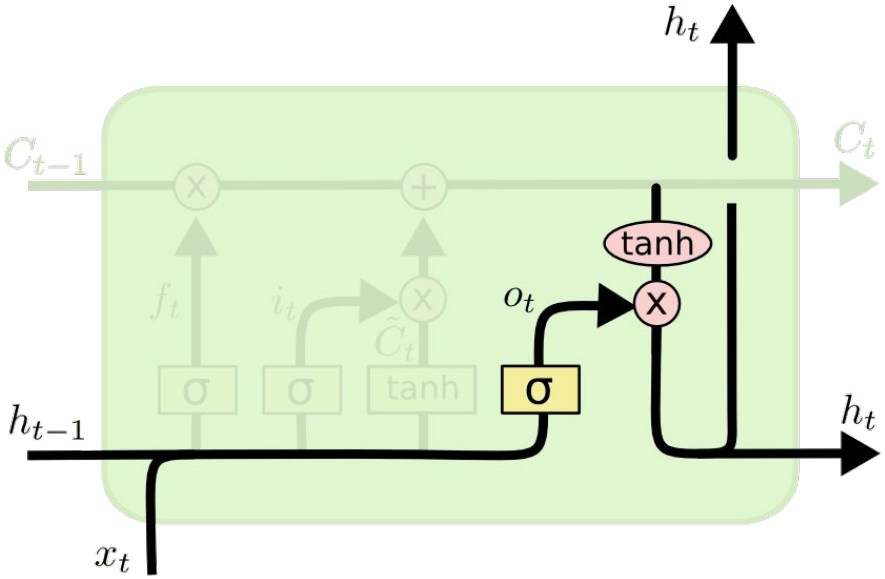
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

LSTM - “Update”



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

LSTM - “Output”



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

$$h_t = o_t * \tanh\left(C_t\right)$$

http://arxiv.org/pdf/1506.03340v1.pdf

“Teaching Machines to Read and Comprehend”

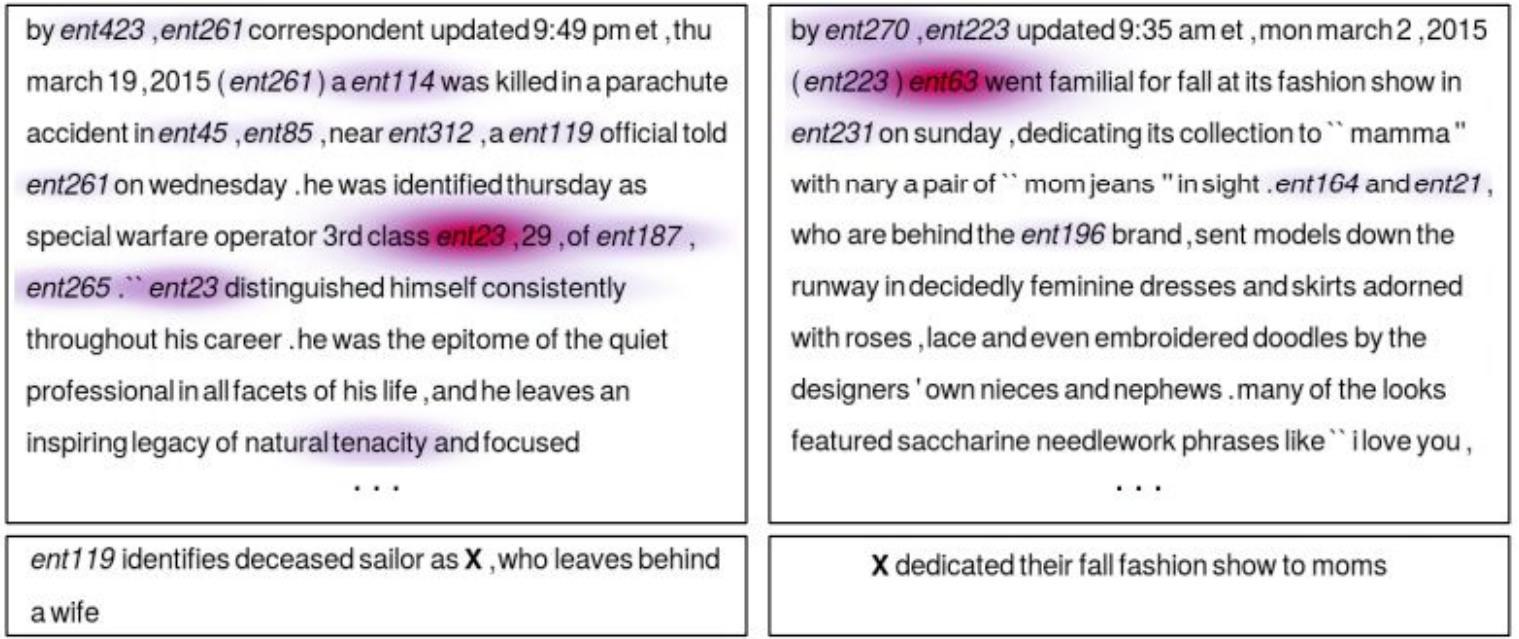
(Google DeepMind + Oxford) @ NIPS2015

Original Version	Anonymised Version
Context	
The BBC producer allegedly struck by Jeremy Clarkson will not press charges against the “Top Gear” host, his lawyer said Friday. Clarkson, who hosted one of the most-watched television shows in the world, was dropped by the BBC Wednesday after an internal investigation by the British broadcaster found he had subjected producer Oisin Tymon “to an unprovoked physical and verbal attack.” . . .	the *ent381* producer allegedly struck by *ent212* will not press charges against the “ *ent153* ” host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* “ to an unprovoked physical and verbal attack . ” . . .
Query	
Producer **X** will not press charges against Jeremy Clarkson, his lawyer says.	Producer **X** will not press charges against *ent212*, his lawyer says.
Answer	
Oisin Tymon	*ent193*

http://arxiv.org/pdf/1506.03340v1.pdf

“Teaching Machines to Read and Comprehend”

- “We feed our documents one word at a time into a Deep LSTM encoder, after a delimiter we then also feed the query into the encoder” (!!!)



The end?


STAR WARS
THE FORCE AWAKENS
DECEMBER 18
IN 3D, REAL D 3D AND IMAX 3D