Deep Learning

Getting Started

Jeremy Pinto

jeremy@jerpint.io

What this talk is about

- Overview of deep learning
- Examples of how it can be used
  - Image processing
  - Text analysis
- Tips on getting started

Machine Learning

- Teach machines to learn abstract concepts.
- Example: what is a chair?

Machine Learning

> What about these?

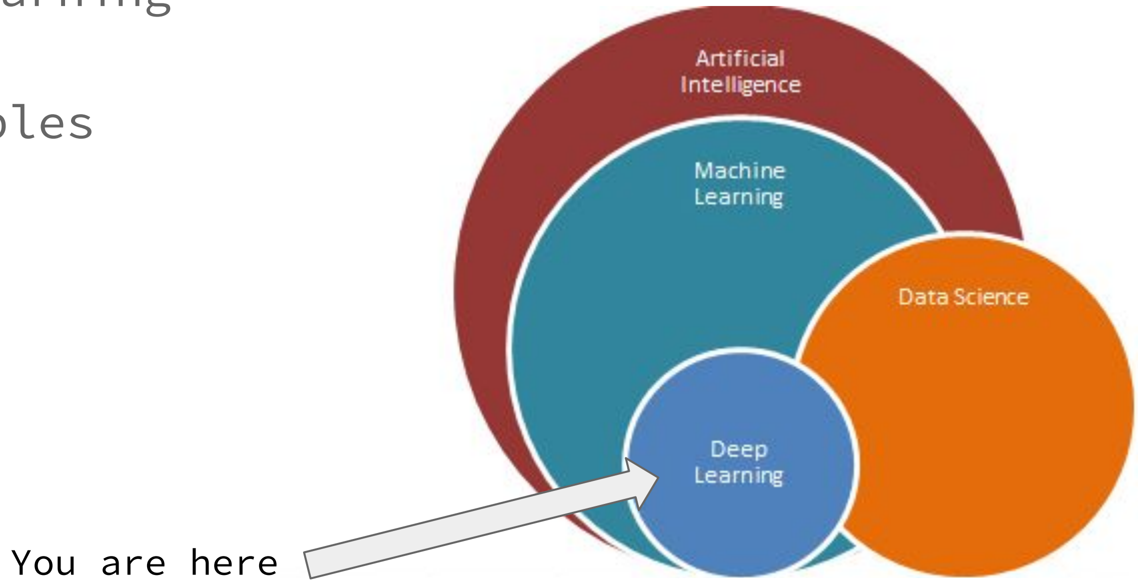




Semantics

> Typical machine learning problem:
  - **X**: set of examples
  - **y**: target

Find f(**X**) ≈ **y**



Deep Learning

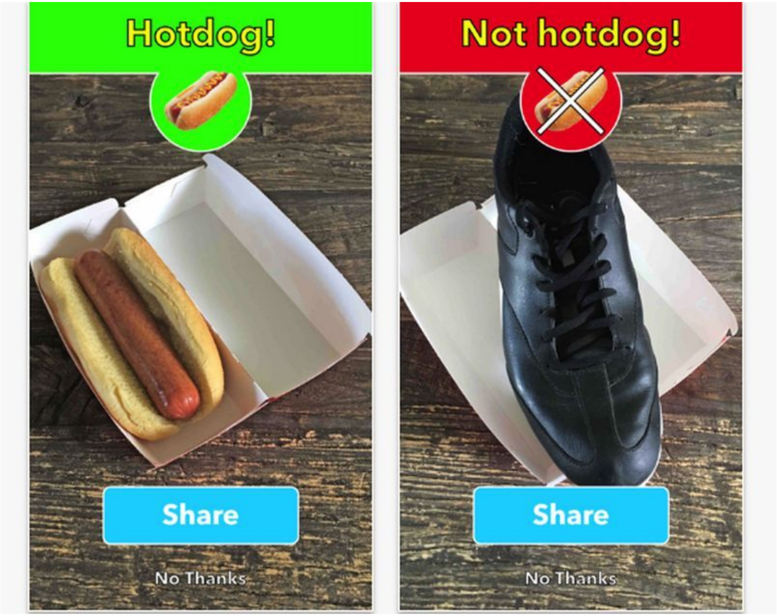
Example

X: Input (image)

y: Target (hot dog | not hot dog)

f(**X**) = **y’** = Prediction

L(**y’**,**y**): Loss (error function)



Example

f() =

p = 0.59

p = 0.41

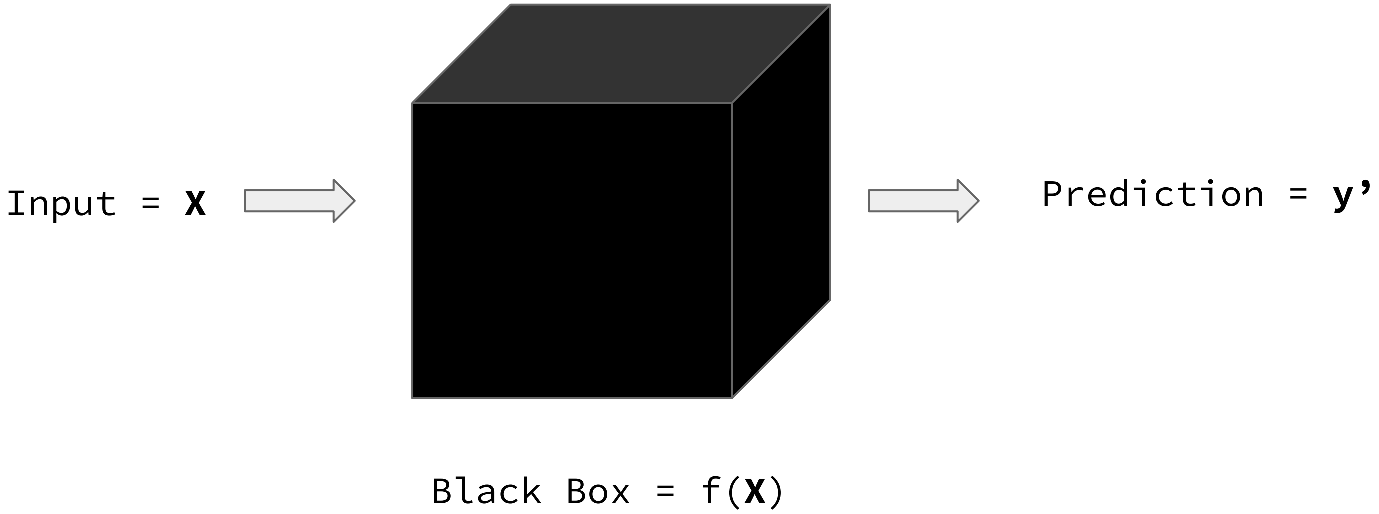
Train on examples

f() =

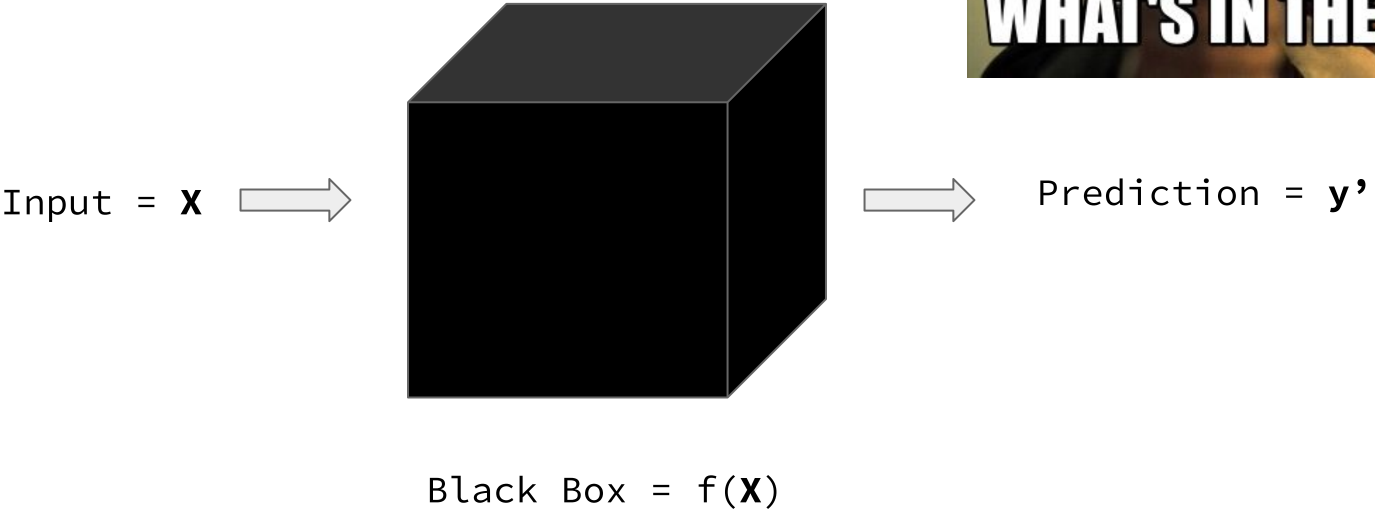
p = 0.99

p = 0.01

Deep learning system

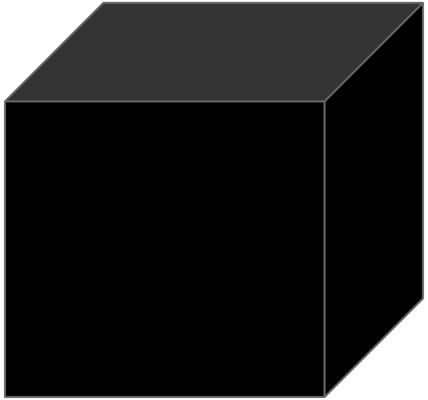


Deep learning system



Deep learning system

> Linear Algebra
> Calculus
> Non-Linearities



Black Box = f(**X**)

“Linear algebra is one of the fundamental mathematical disciplines necessary to understanding deep learning.”
– The Deep Learning Book

http://www.deeplearningbook.org

Deep Learning – Optimization

Goal : Minimize the error of our black box

Given: f(**X**) = **y’** = prediction

Error: L(**y’**,**y**) = how good is our prediction?

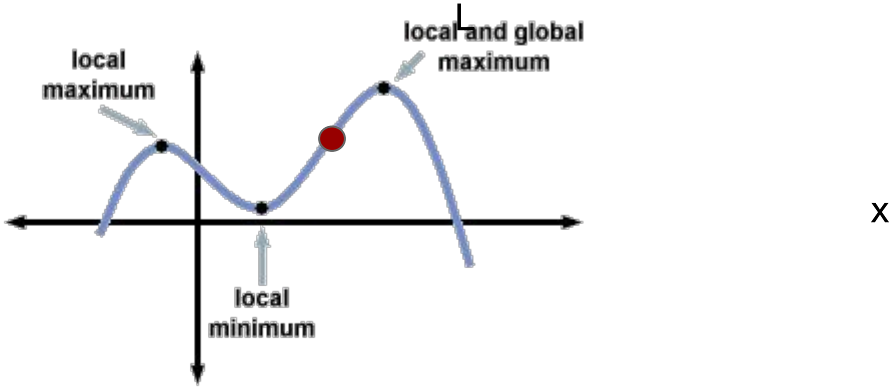
Given our error, tune the parameters in the model

Recall: Minimization

Calculus: given a function L(x), find its minimum

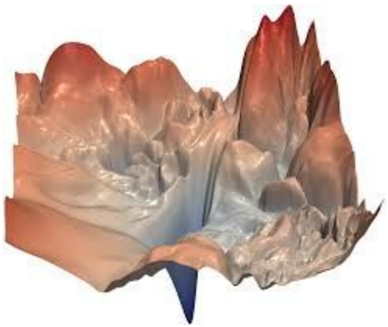
$$\frac{\partial L}{\partial x} = 0$$

$$\frac{\partial^2 L}{\partial x^2} > 0$$

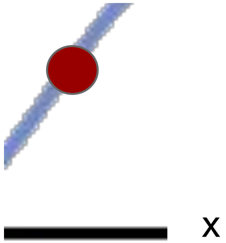


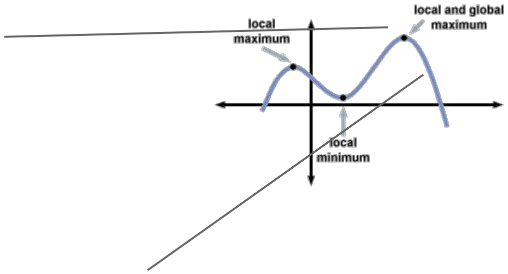
Recall: Minimization

Deep Learning - The loss function and its derivative is only approximated locally.



Control x

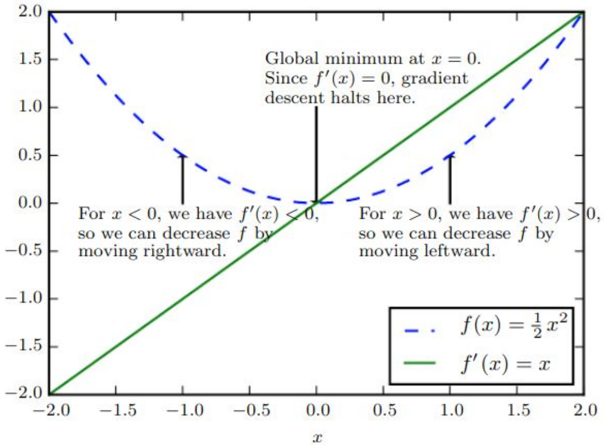




Recall: Minimization

Deep Learning - The loss function and its derivative is only approximated locally.

Control x



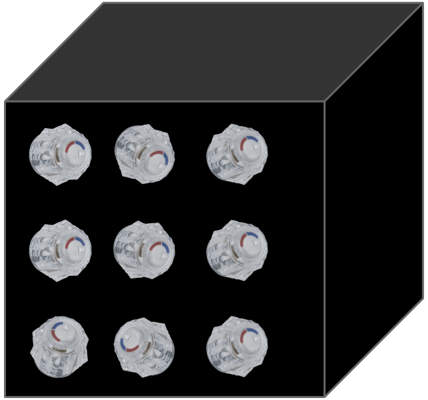
Gradient Descent

$$\boldsymbol{x}' = \boldsymbol{x} - \epsilon \nabla_{\boldsymbol{x}} f(\boldsymbol{x})$$

http://www.deeplearningbook.org

Deep learning system

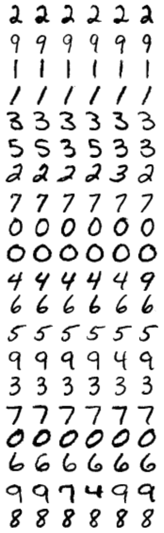
- Linear Algebra
- Calculus
- Non-Linearities



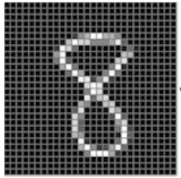
Black Box = f()

Image Processing

MNIST – Hello (Deep Learning) World



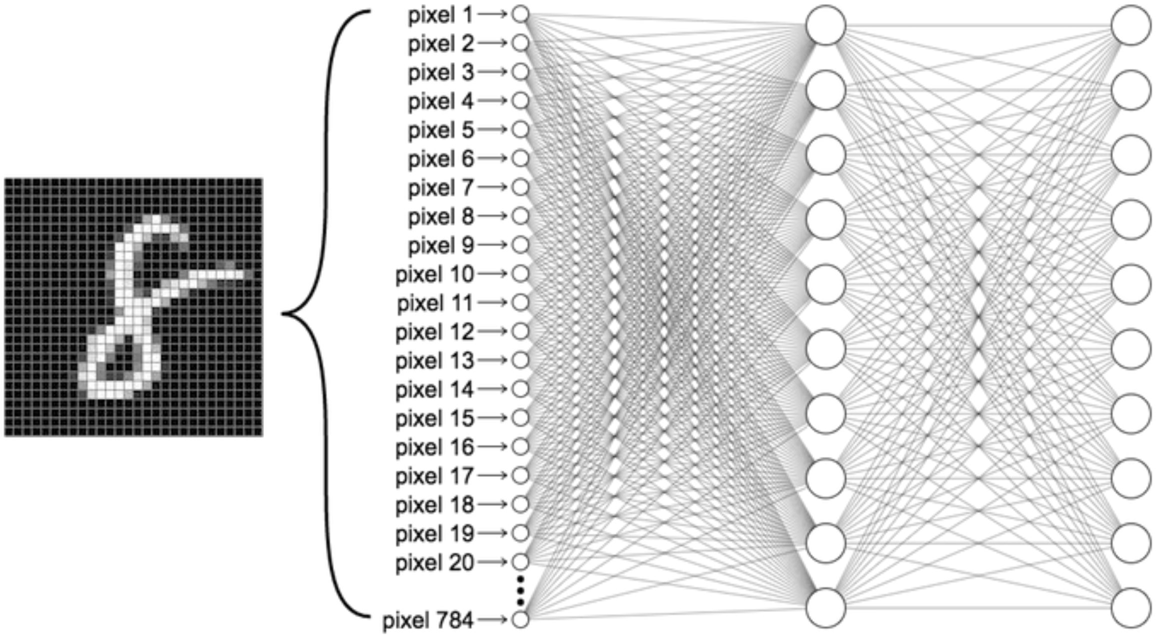
Dataset



28 x 28
784 pixels

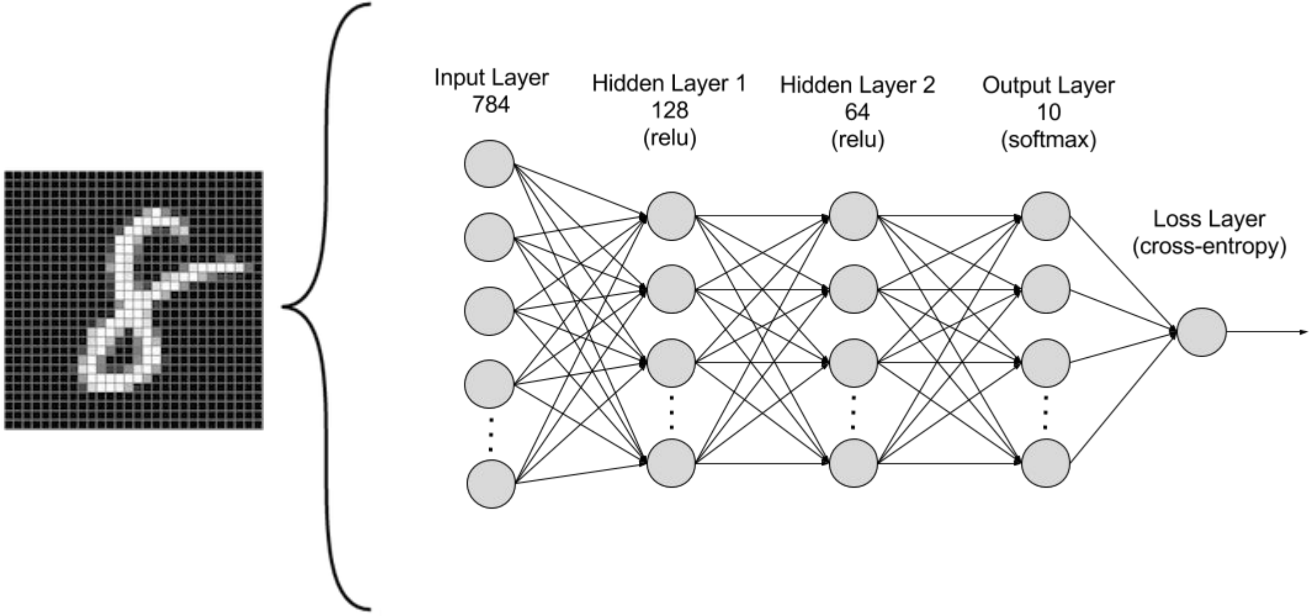
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	4	62	146	182	254	254	181	176	139	15	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	34	186	253	217	208	136	136	136	166	232	99	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	61	242	208	111	3	0	0	0	0	0	18	32	107	43	0	0	0	0	0	0
0	0	0	0	0	0	0	0	156	242	23	0	0	0	0	0	0	0	13	191	181	6	0	0	0	0	0	0
0	0	0	0	0	0	0	0	121	255	98	3	0	0	0	0	0	8	194	225	12	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	169	253	120	3	0	0	0	0	128	247	51	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	3	111	244	169	19	0	14	131	249	117	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	59	241	235	72	142	229	66	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	25	218	254	231	36	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	133	253	221	33	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	19	237	111	196	217	19	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	174	138	0	23	193	204	18	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	96	224	0	0	0	25	218	169	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	138	0	0	0	0	86	253	99	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	3	162	214	11	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	0	118	253	68	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	185	157	0	0	0	0	0	40	254	98	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	50	244	61	0	0	0	0	112	244	58	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	174	251	142	59	83	167	244	111	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	6	133	253	253	253	169	61	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Multi Layer Perceptron (Fully Connected)

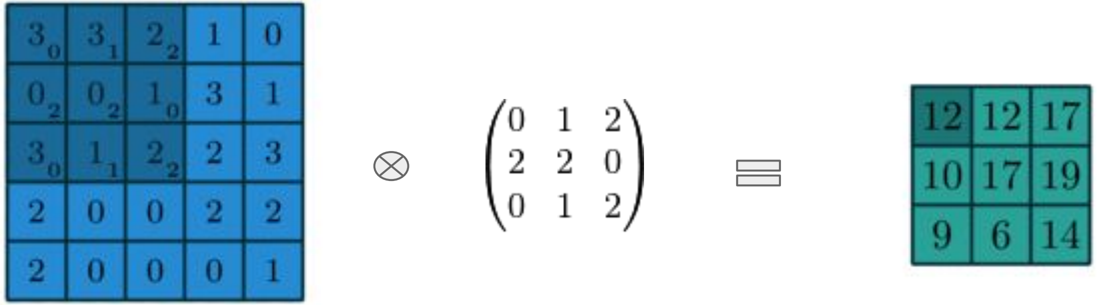


Digit probability

Multi Layer Perceptron (Fully Connected)

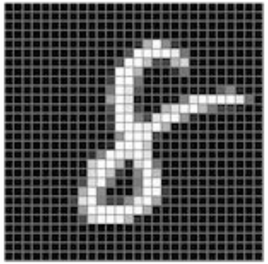


Convolution

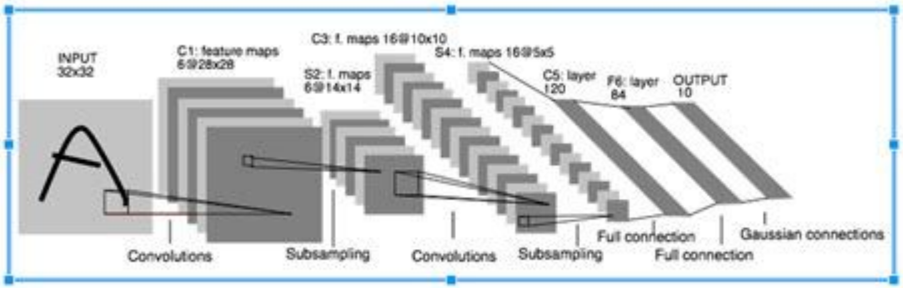


http://deeplearning.net/software/theano/tutorial/conv_arithmetic.html

LeNet (convolutional neural network)

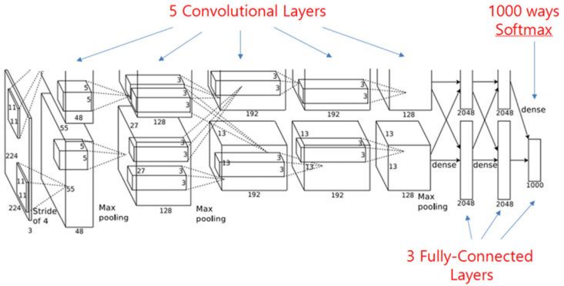


Convolutional Neural Nets (CNNs): 1989



LeNet: a layered model composed of convolution and subsampling operations followed by a holistic representation and ultimately a classifier for handwritten digits. [LeNet]

AlexNet – ImageNet Winner 2012

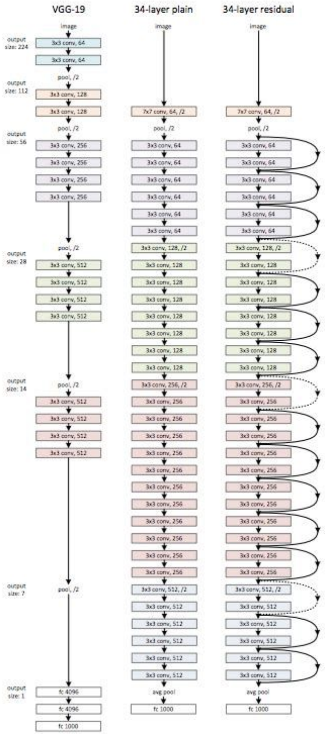


ImageNet Dataset

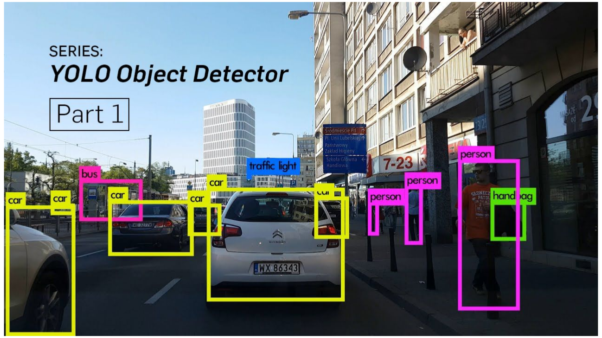
Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*. [web]

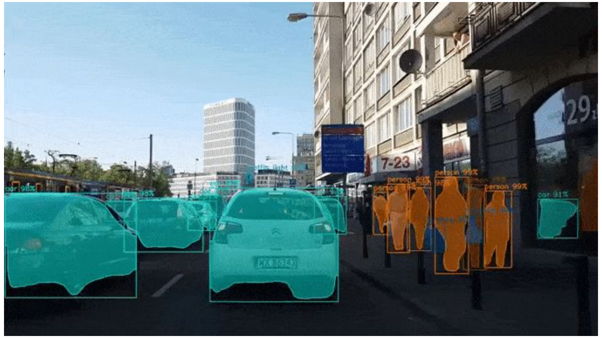
3

Deep Learning Revolution

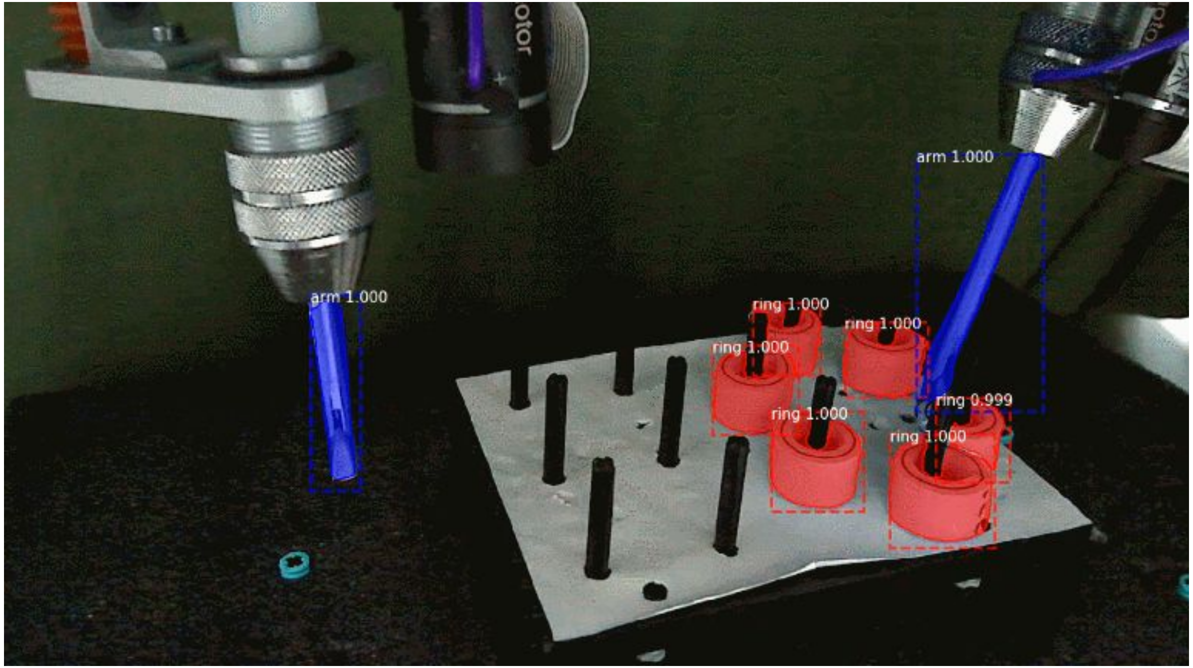


Deep Learning Applications (images)

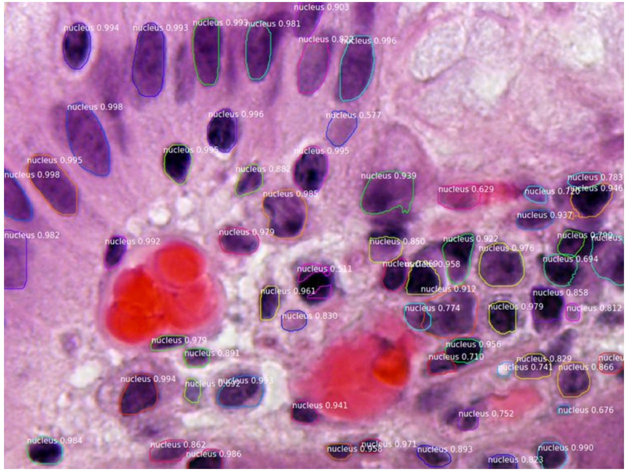




Mask R-CNN



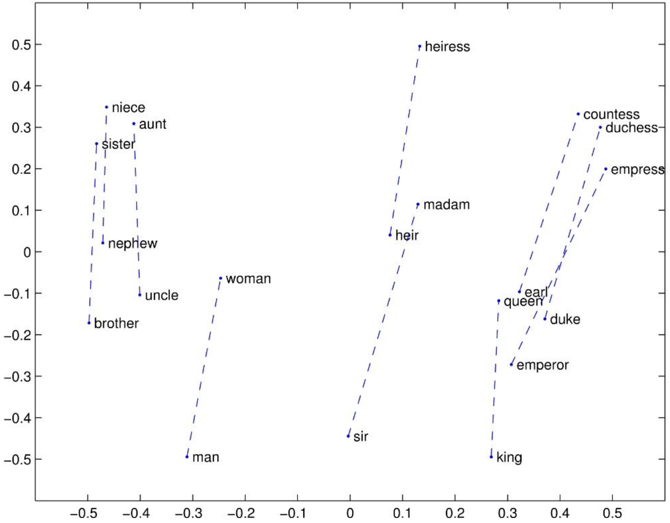
Deep Learning Applications (images)



Natural Language Processing

Word Embeddings (word2vec, Glove etc.)

> Learn a high-dimensional vector representation for words



https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

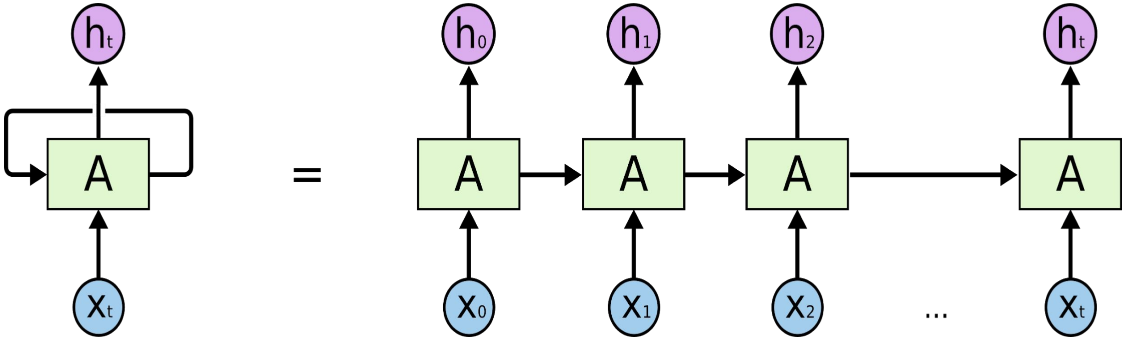
NLP – Char-RNN

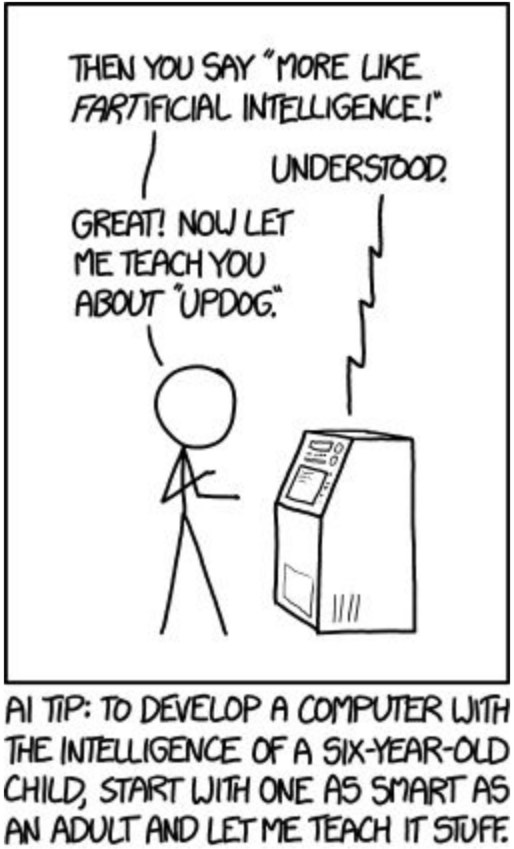
```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We 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.
```

```
CINNA
No, by no means.

METELLUS CIMBER
O, let us have him, for his silver hairs
Will purchase us a good opinion
And buy men's voices to commend our deeds:
It shall be said, his judgment ruled our
hands;
Our youths and wildness shall no whit
appear,
But all be buried in his gravity.
```

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

Recurrent Neural Networks





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

Getting Started

Python

Deep Learning Libraries

- Start with tutorials
- Focus on inputs and outputs to common architectures

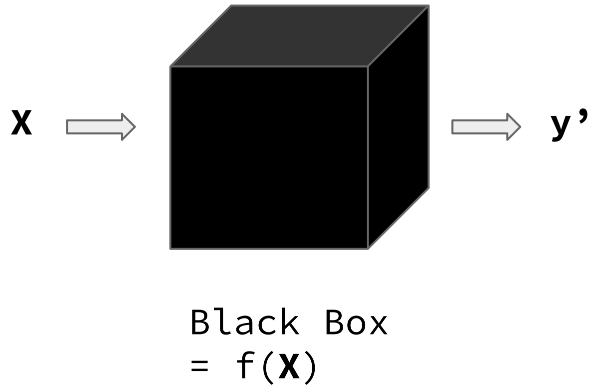
Explicitly define your problem

X: What is your input?

y: What is your target?

Define proper evaluation metrics

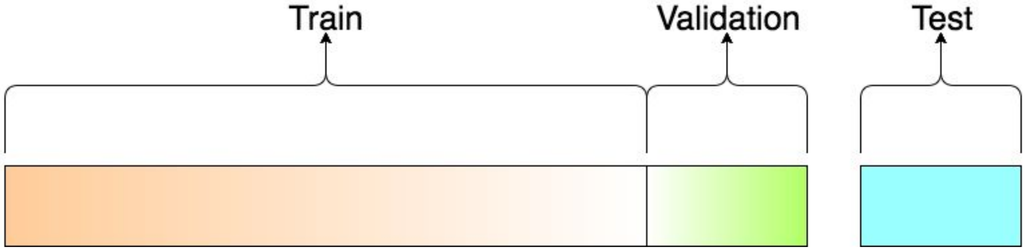
What architecture should you use?



Split your data

You should have 3 sets of data

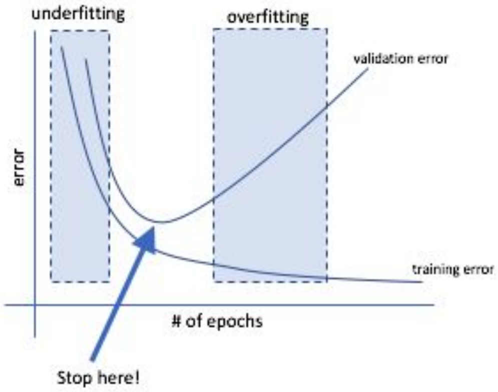
> Train (~80%)
> Validation (~10%)
> Testing (~10%)



This will avoid overfitting your data

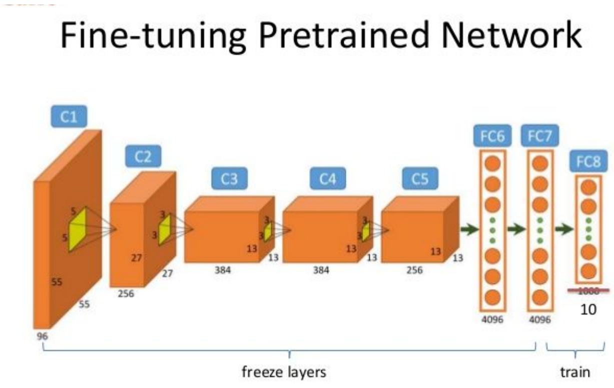
Overfitting

- Can happen when your data is noisy or if little training data is available
- Signs of overfit
  - 100% accuracy?
  - test set and train set too similar?
  - Data leak? Bias in the data?



Fine-Tuning

- Use an existing model as a starting point for your experiment
  - Great for smaller datasets
  - Works well in practice
  - Allows you to save a lot of implementation time!
- Example - Use a model trained on ImageNet to classify new classes
  - Filters learned are already generalized enough
  - Takes a lot less time to converge



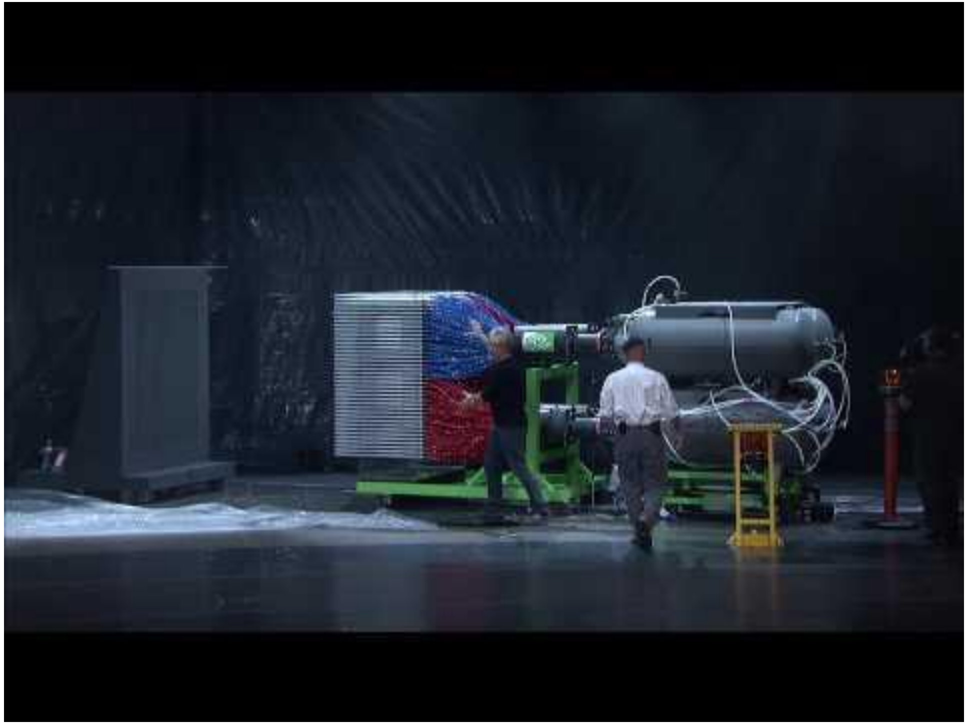
Do’s

- Look for pre-trained models
- Look for similar datatsets
- Adapt code to suit your needs
- Ask questions
- Read A LOT!
  - Lots of great blog posts!
  - Understand concepts first, understand details after

Dont’s

- Reimplement papers from scratch (Right away)
  - Great for learning
  - Time consuming
  - Lots of debugging
  - implementation details aren’t always clear
- Train from scratch
  - Learning from scratch is hard and resource intensive
  - Fine-tune instead!

GPU vs CPU



Computing Resources

- Most libraries can run on CPU
  - Inference (making a prediction) is fine on CPU
- GPU is ideal for training larger models
- GPUs are expensive and need the right drivers (CUDA, CUDNN)
- Cloud alternatives : Amazon, Azure, Google Cloud etc.

Learn the jargon

Loss, epoch, splits, learning rate, accuracy, dropout, early-stopping, layers, convolution, RNN, LSTM, MLP, Fully-connected etc.

Practical Considerations

- How large are your models?
- Can they work offline? (edge computing)
- How do they scale?
- Is it for real-time applications?
- How will you deploy?
- How does it compare to simpler models?

Use the right tools

GitHub

docker

CONDA

arXiv

stackoverflow

Questions?

Jeremy Pinto

jeremy@jerpint.io

http://