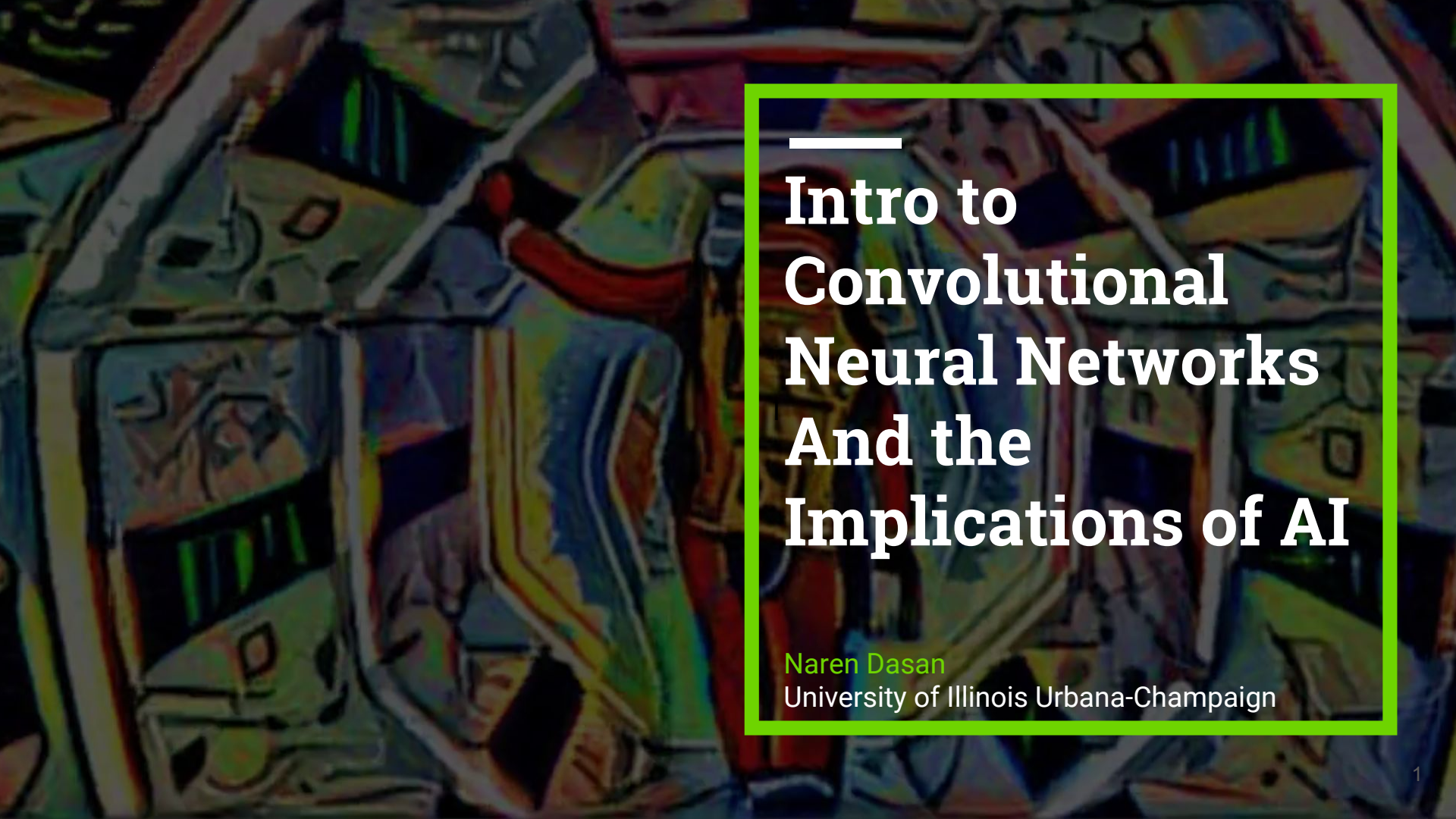


Intro to Convolutional Neural Networks And the Implications of AI

Naren Dasan
University of Illinois Urbana-Champaign

What do people mean by AI?

The field of study that concerns developing systems that can act intelligently, a subfield of which is Machine Learning, trying to develop algorithms that can learn from experience or teaching.

What is Machine Learning?

$$\text{Let } \mathbb{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$$

$$\text{where } x^{(i)} \text{ is an image}$$

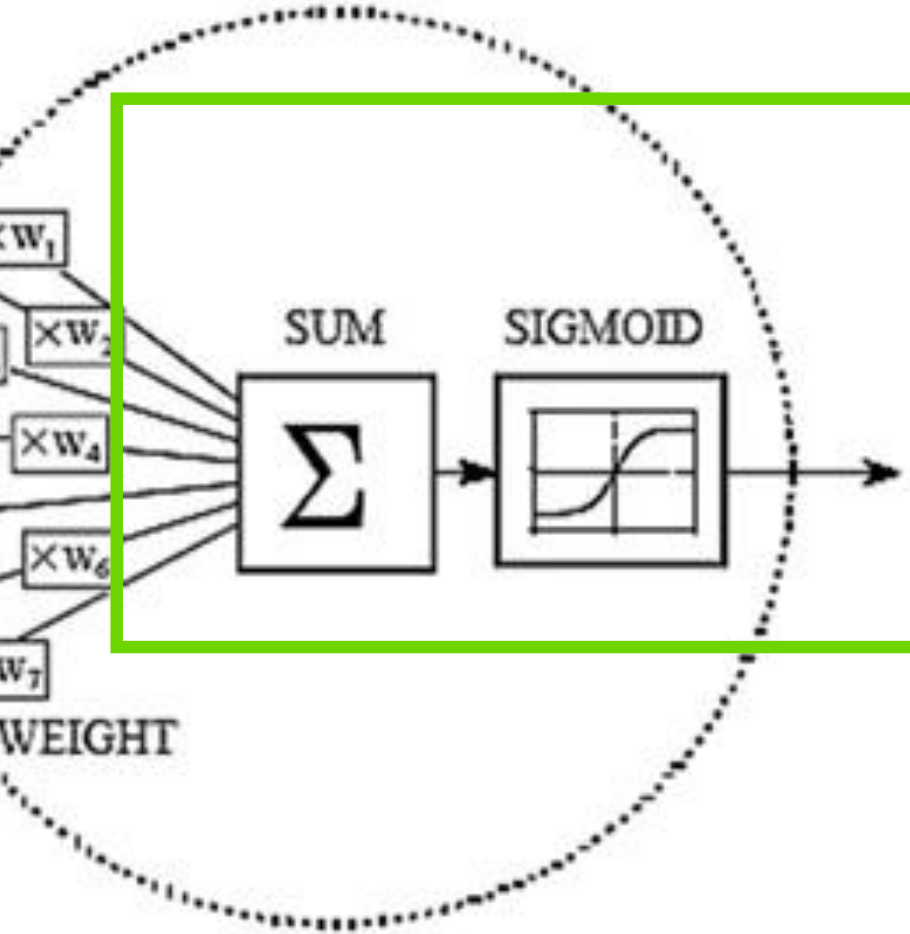
$$y^{(i)} \text{ is a label}$$

$$f : X \to Y$$

How can we get a Computer to do complex tasks?

We can formulate some tasks as a function associating an observation to an explanation and those are the tasks we are going to talk about today, but what function to use?

What if we had the computer learn it instead of programming it?

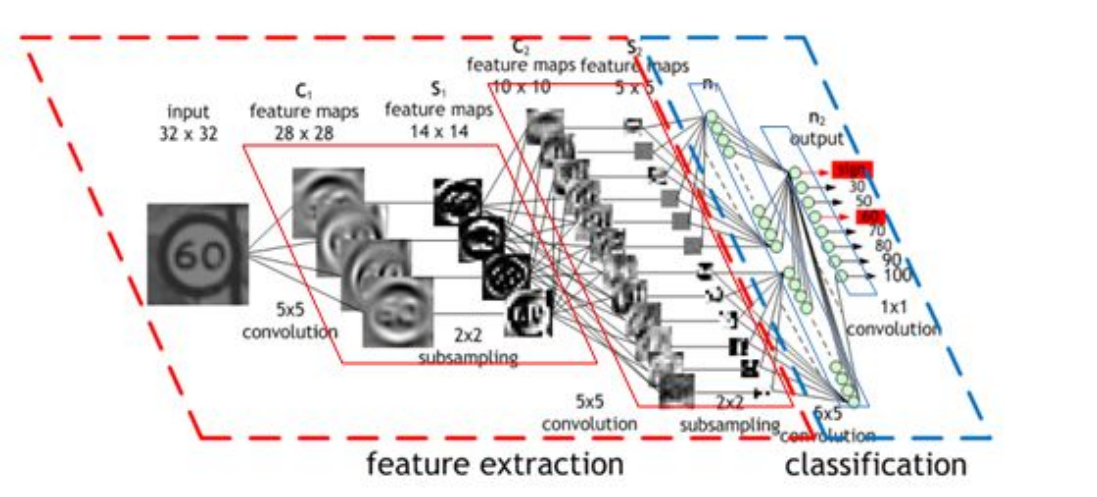


Then what is Deep Learning?

Roughly replicating the structure of the brain, layering sets of “preceptrons” to create a hierarchy of representations of the input data, until you can go from pixels to a vector that has semantic meaning

This formulation can learn an arbitrary continuous function fairly efficiently. Exactly what we need.

Convolutional Neural Nets

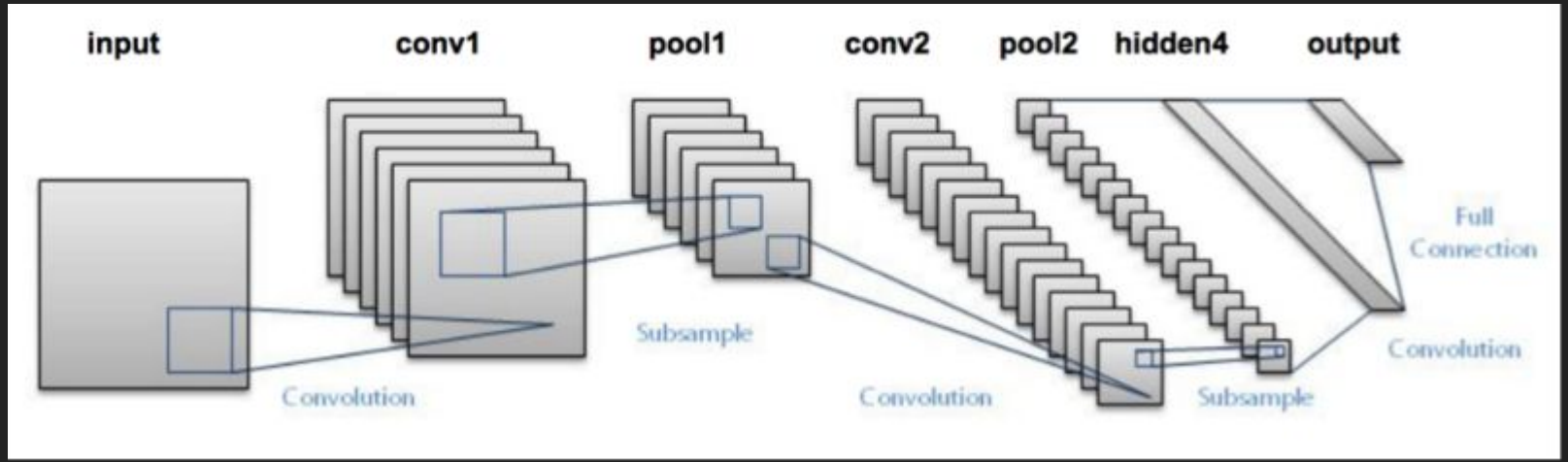


Use a standard technique in Computer Vision (Convolution Kernels) to extract many different types of “features”.

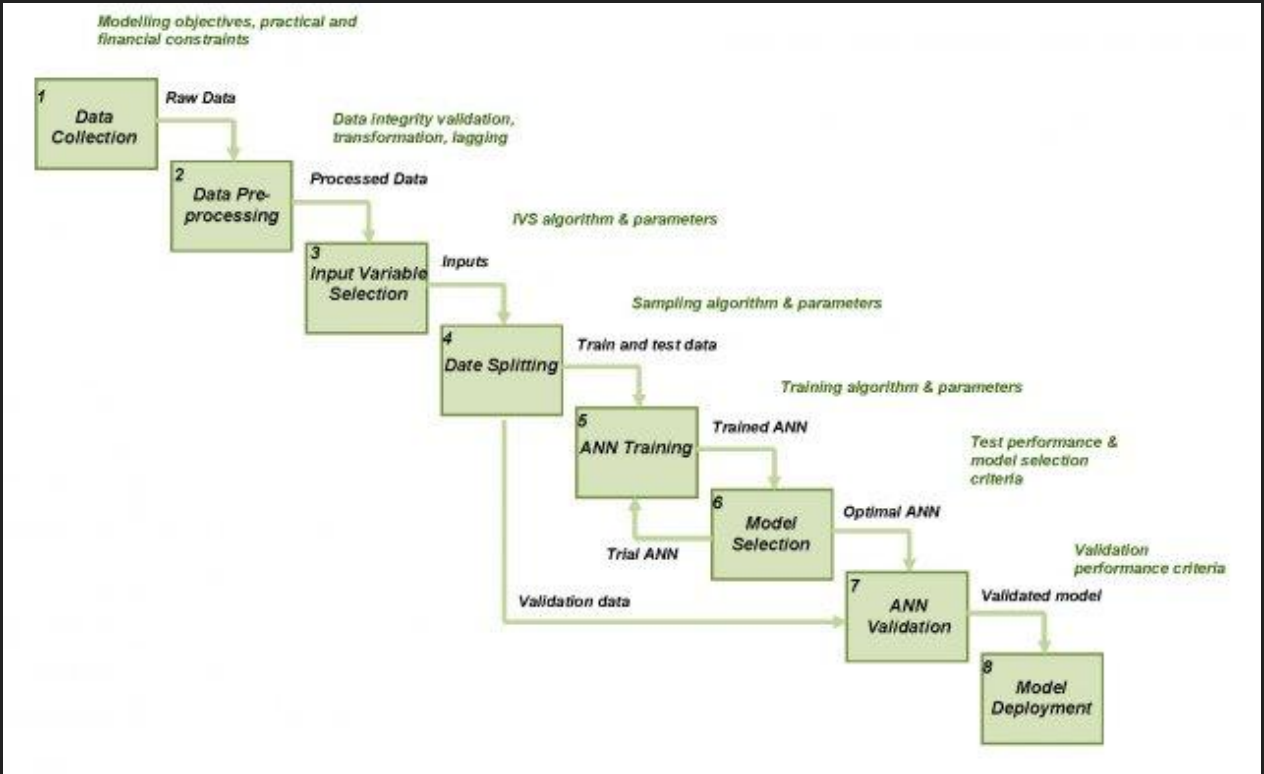
Then using AI techniques to learn which features to extract and pay attention to.

Convert the features into a small vector representing the object

Hello World for CNNs

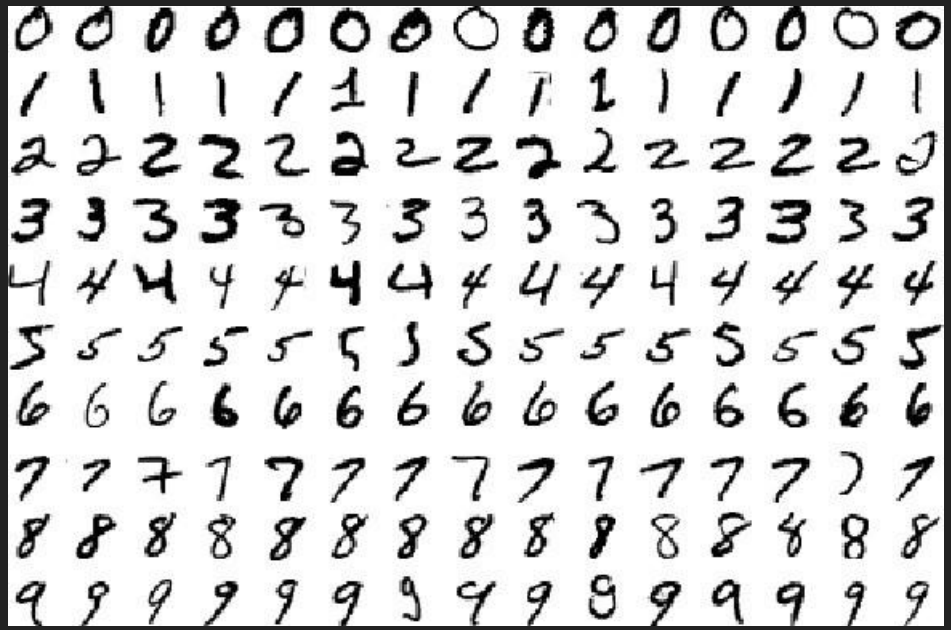


How do you train a CNN?



MNIST

How are images encoded?



Logistic/Softmax Regression

What if we assign each pixel a “importance “ value?

- We can encode these parameters as **W** size (height x width)x10
- We can then given the “optimal parameters” w^* we can calculate the probability an image matches a label using the function

$$P(y^{'} = 1) = \sigma(F(w, x))$$

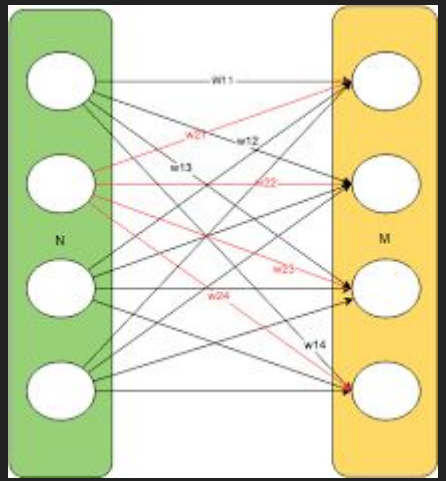
$$\text{where } \sigma(\theta) = \frac{1}{1 + e^{\theta}}$$

$$F(w, b, x) = \boldsymbol{w}^T x + b$$

Fully Connected Layer

- If we think of the function (which is referred to as an inference function) as a graph (or network) of functions, the “first layer” is

$$F(w, b, x) = \boldsymbol{w}^T x + b$$



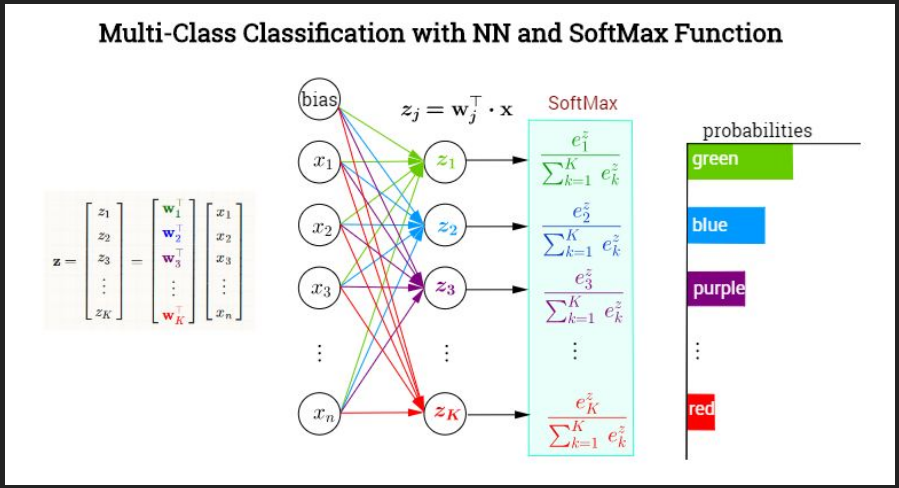
(Credit: Peng's Blog)

Softmax

- A system of Linear equations can only approximate a linear function, we need something non linear most of the time

$$\text{where } \sigma(\theta) = \frac{1}{1 + e^{\theta}}$$

- Logistic Regression is a binary classification so we augment the sigmoid function to handle multiple classes



Objective Function

Loss Function

$$\min_{w} \sum_{(x^{(i)},y^{(i)})\in\mathbb{D}} log(1 + exp(-y^{(i)}\boldsymbol{w}^T\phi(x^{(i)})))$$

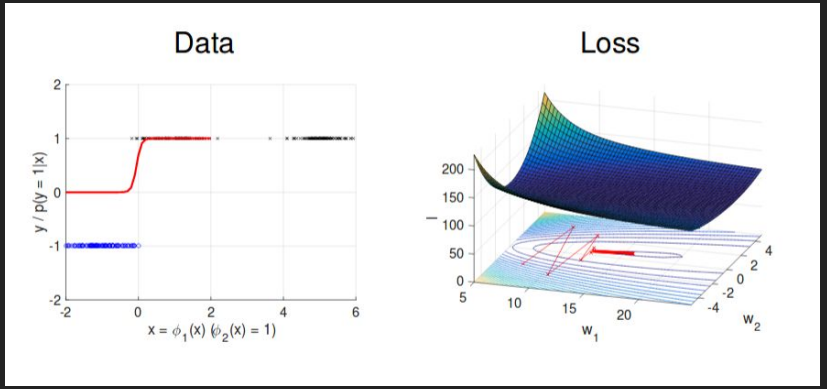
Where $\boldsymbol{w}$ are the parameters (weights) of the model and $\phi(x^{(i)})$ is what is a called a feature transform

(Stochastic) Gradient Descent

- Finding the local optimum of an optimization problem has no **Closed Form** solution most times
- There is also no efficient algorithm (the problem is NP-Hard)
- What if we calculated how good we are using the objective function and just tried to get better?

$$\nabla_{\boldsymbol{w}} = \sum_{(x^{(i)}, y^{(i)}) \in \mathbb{D}} \frac{-y^{(i)} \phi(x) exp(-y^{(i)} \boldsymbol{w}^T \phi(x^{(i)})}{1 + exp(-y^{(i)} \boldsymbol{w}^T \phi(x^{(i)})}$$

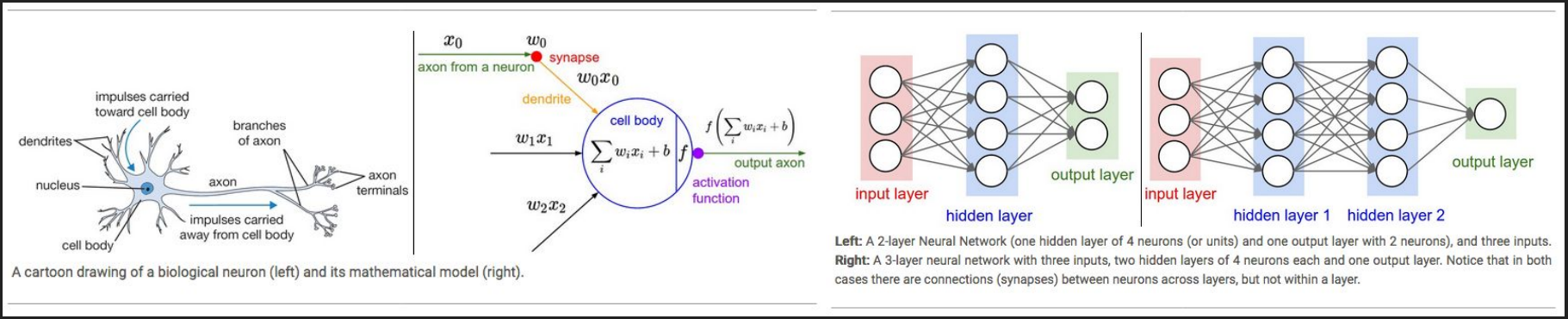
$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha \nabla_{\boldsymbol{w}_t}$$



(Credit: CS 446 - UIUC)

- If we update our model in the same order every time what could happen?
- How long would it take if we needed to go through the entire dataset for every update?

Multilayer Perceptrons



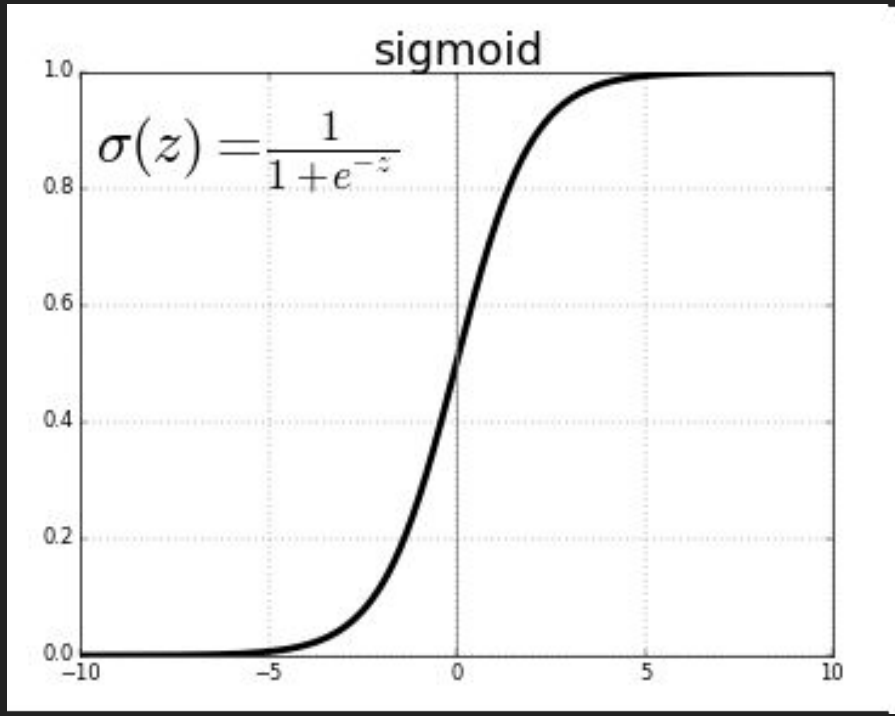
A cartoon drawing of a biological neuron (left) and its mathematical model (right).

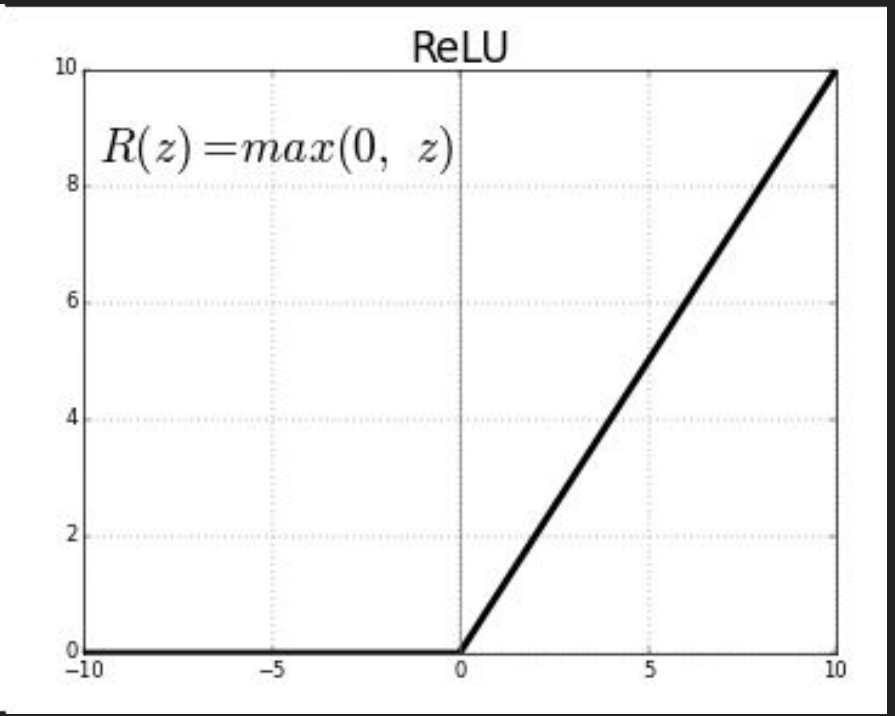
Left: A 2-layer Neural Network (one hidden layer of 4 neurons (or units) and one output layer with 2 neurons), and three inputs.
Right: A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. Notice that in both cases there are connections (synapses) between neurons across layers, but not within a layer.

$$P(y = \{1...10\}) = \sigma(F(w_3, b_3, F(w_2, b_2, F(w_1, b_1, x))))$$
$$\text{where } F(w, b, x) = relu(w^T x + b)$$

(Credit: CS 231n - Stanford University)

Activation Functions





Objective Function

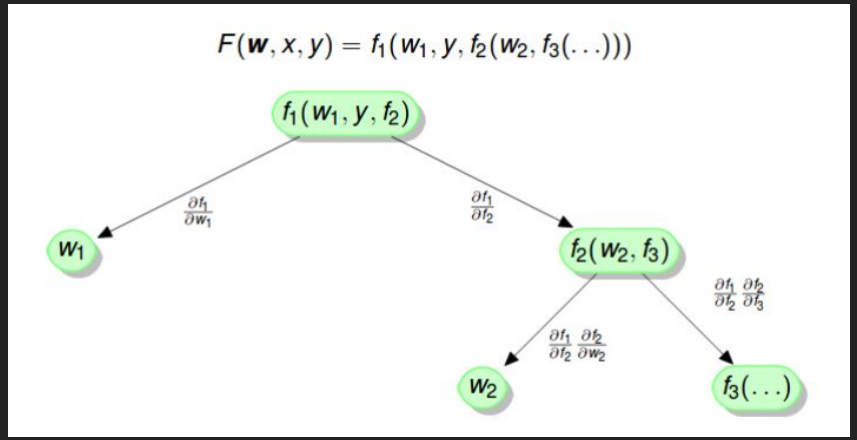
Loss Function

$$\min_{w} \frac{\lambda}{2}||\boldsymbol{w}||_2^2 + \sum_{i \in D} \left(\ln \sum_{\hat{y}} \exp F_{nn}(\boldsymbol{w}, x^{(i)}, \hat{y}) - F_{nn}(\boldsymbol{w}, x^{(i)}, y^{(i)}) \right)$$

Regularization/Weight Decay

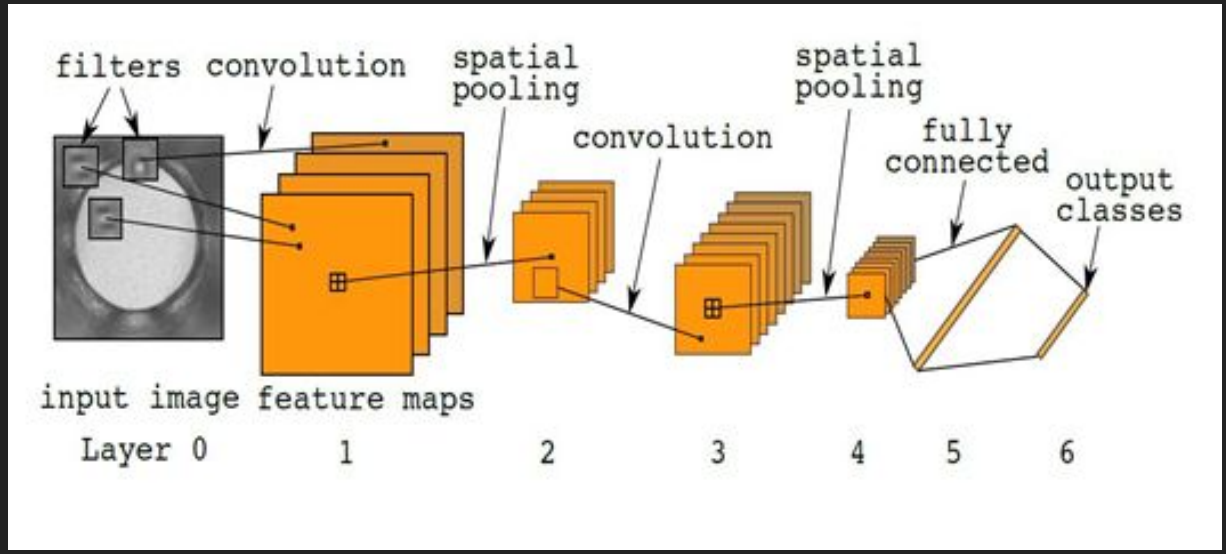
Backpropagation

- Because MLPs are just nested functions, the Gradient at each layer can be calculated with the layer previous' gradient instead of having to use the full loss function.
- Hence the loss "propagated" backwards during gradient descent



(Credit: CS 446 - UIUC)

LeNet



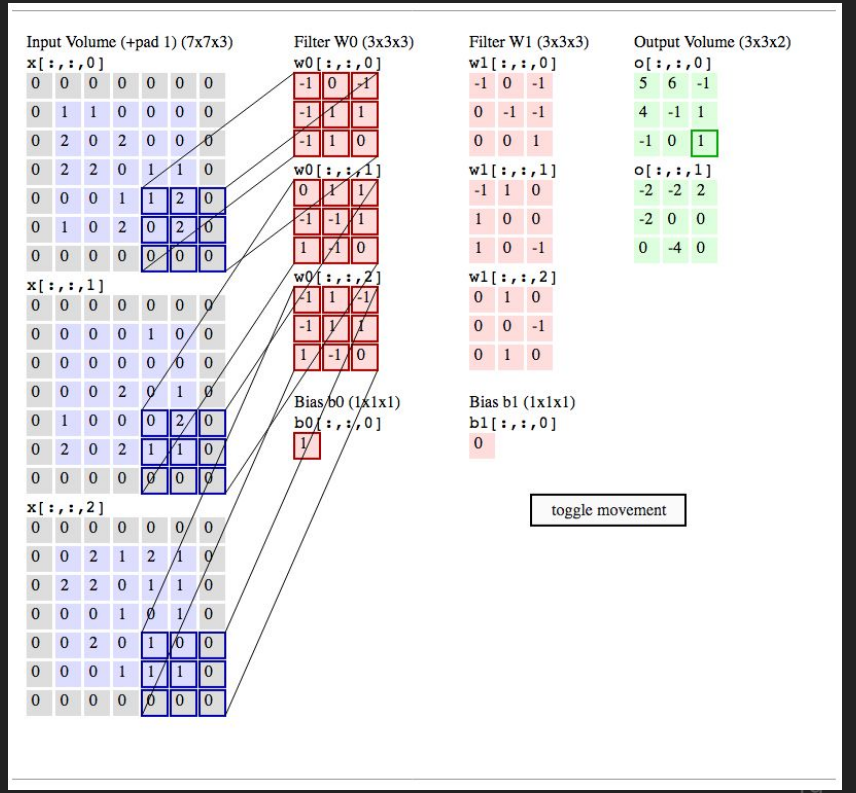
$$P(y = \{1...10\}) = \sigma(FC(w_4, b_4, FC(w_3, b_3, relu(MaxPool(Conv(w_2, b_2, MaxPool(Conv(w_1, b_1, x))))))$$

where $FC(w, x) = w^T x + b$

and $Conv(w, x) = (w * x) + b$

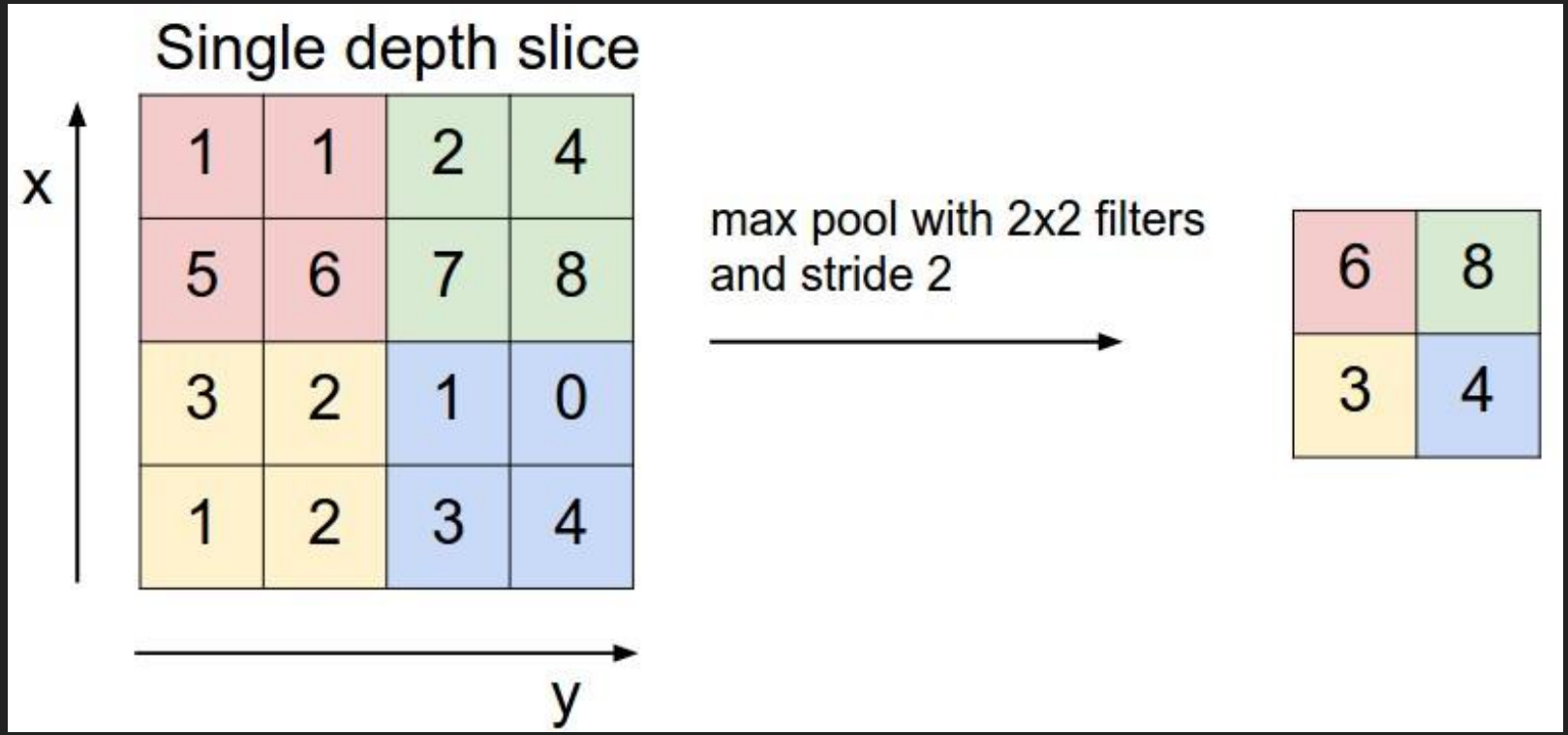
Convolution Layer

$$\mathrm{F}(x) = b + w * x$$



(Credit: CS 231n - Stanford University)

Max Pooling Layer



(Credit: CS 231n - Stanford University)

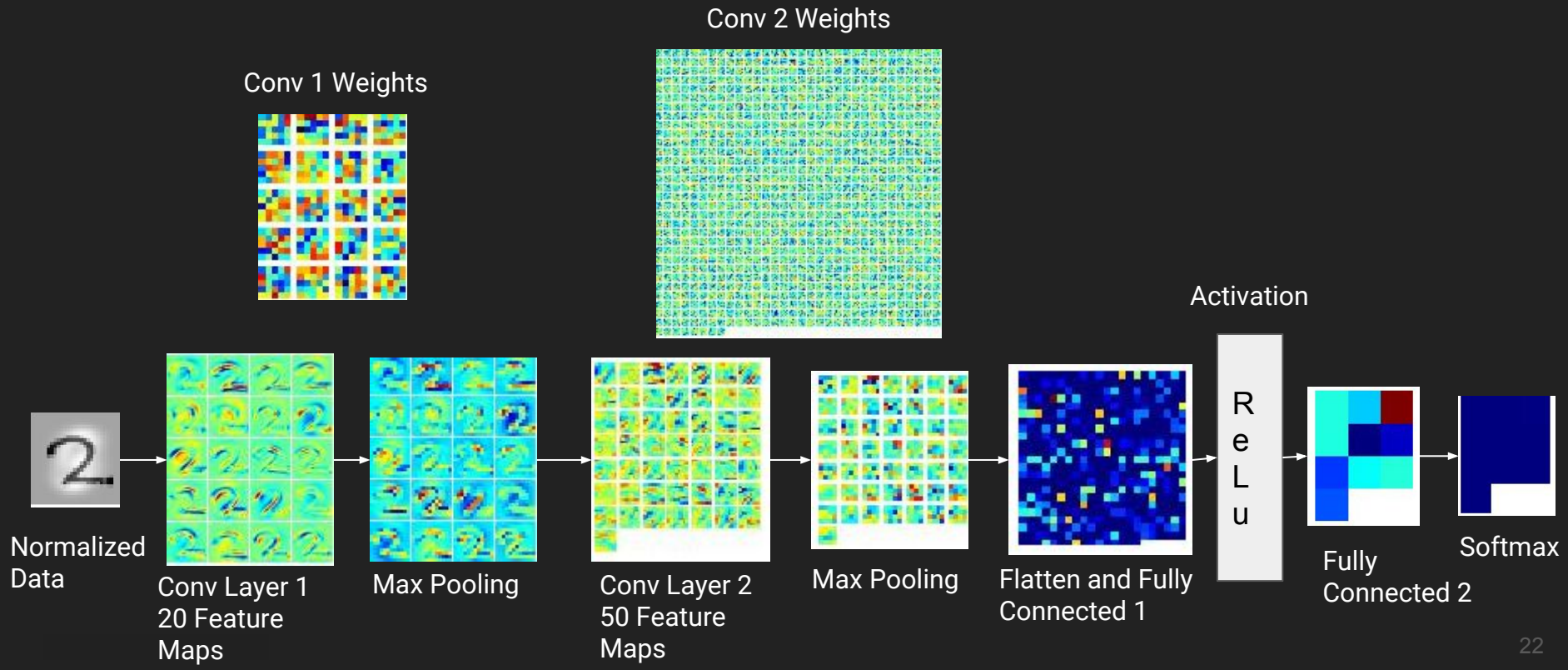
Objective Function

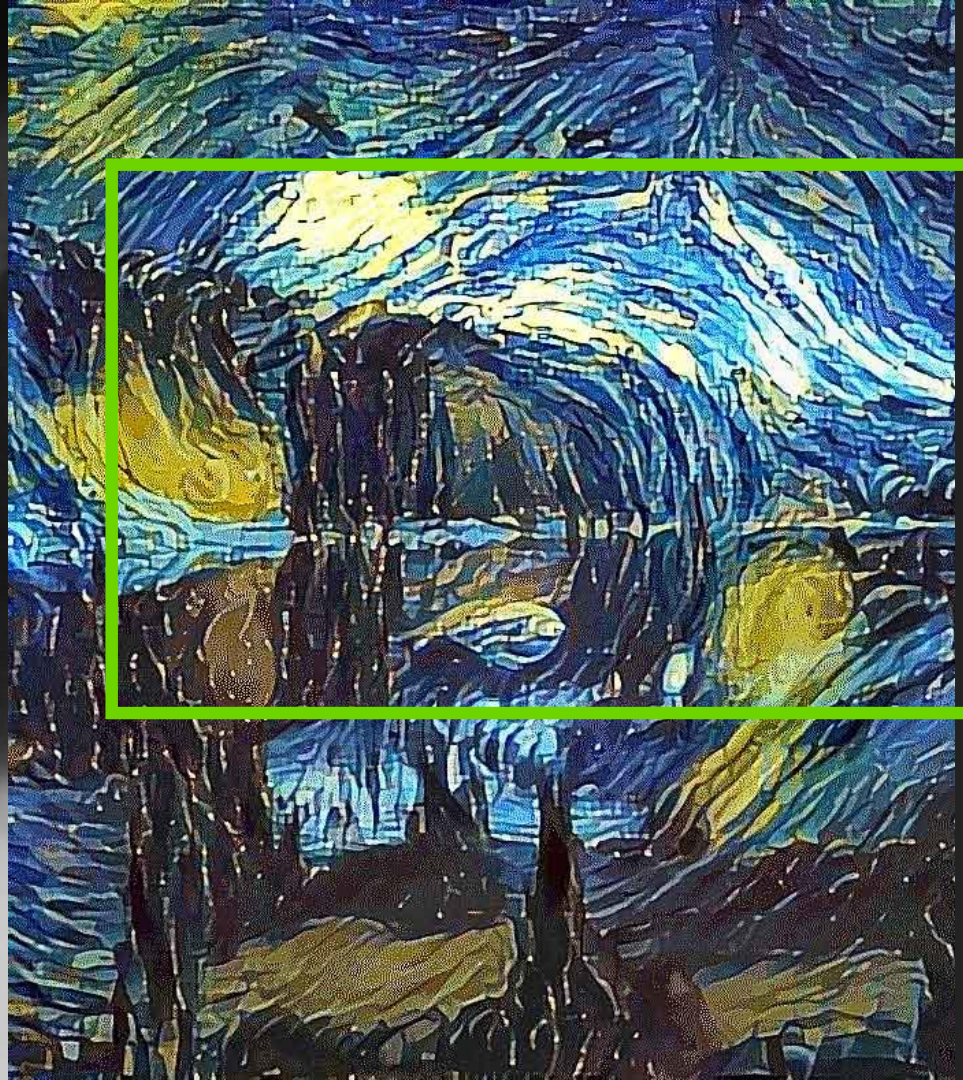
$$\min_{w} \frac{\lambda}{2}||\boldsymbol{w}||_2^2 + \sum_{i \in D} \left(\ln \sum_{\hat{y}} \exp F_{nn}(\boldsymbol{w}, x^{(i)}, \hat{y}) - F_{nn}(\boldsymbol{w}, x^{(i)}, y^{(i)}) \right)$$

Look familiar?

What is the logic here?

http://vault.acm.illinois.edu:5000/models/20180407-165432-abe5

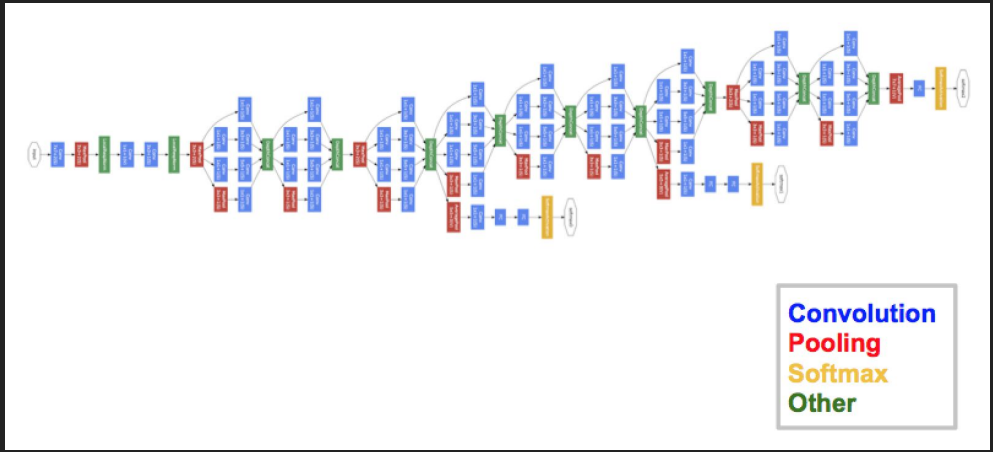




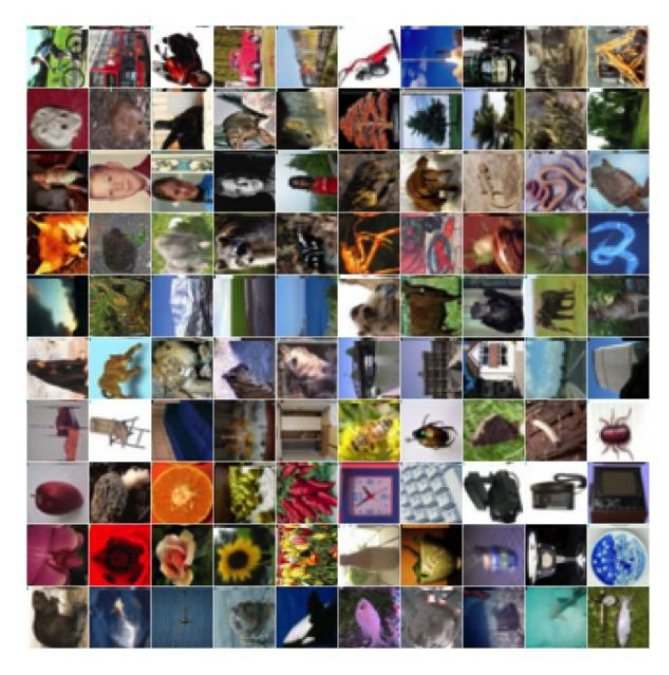
Time to Learn!

A more interesting Example

GoogleNet https://arxiv.org/pdf/1409.4842.pdf



CIFAR-100

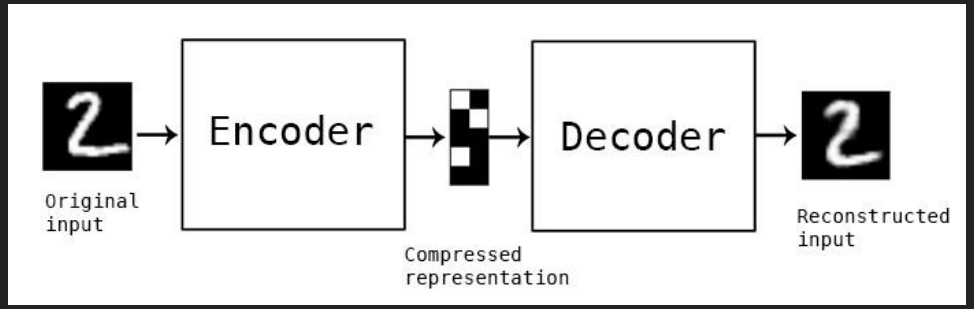


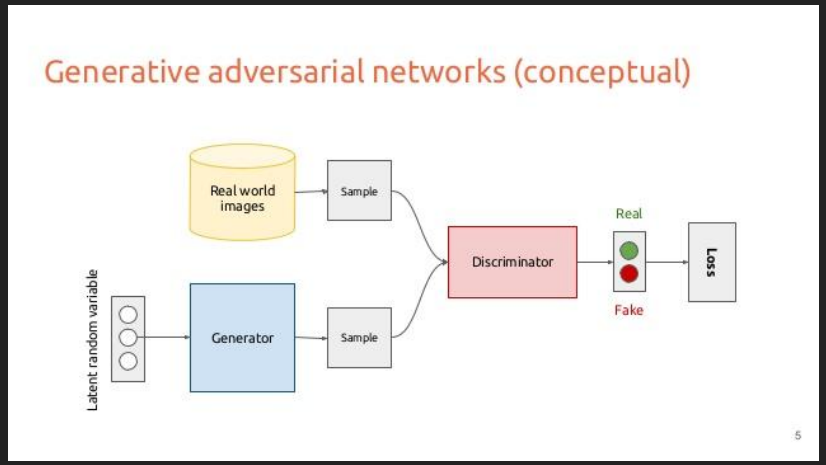
http://vault.acm.illinois.edu:5000/models/20180407-054333-cb20

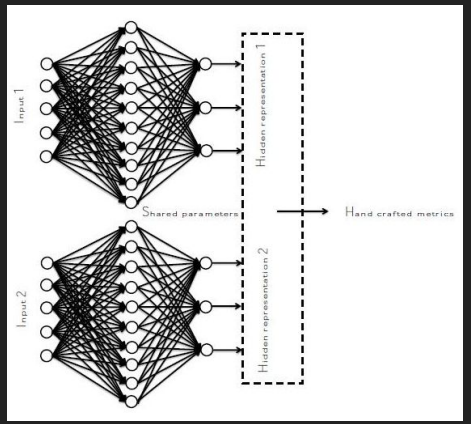
Where Can you go from here?

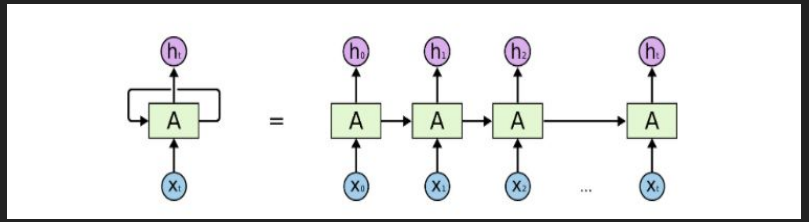
What do the building blocks of CNNs enable?

Autoencoders, RNNs, Siamese Networks and GANs



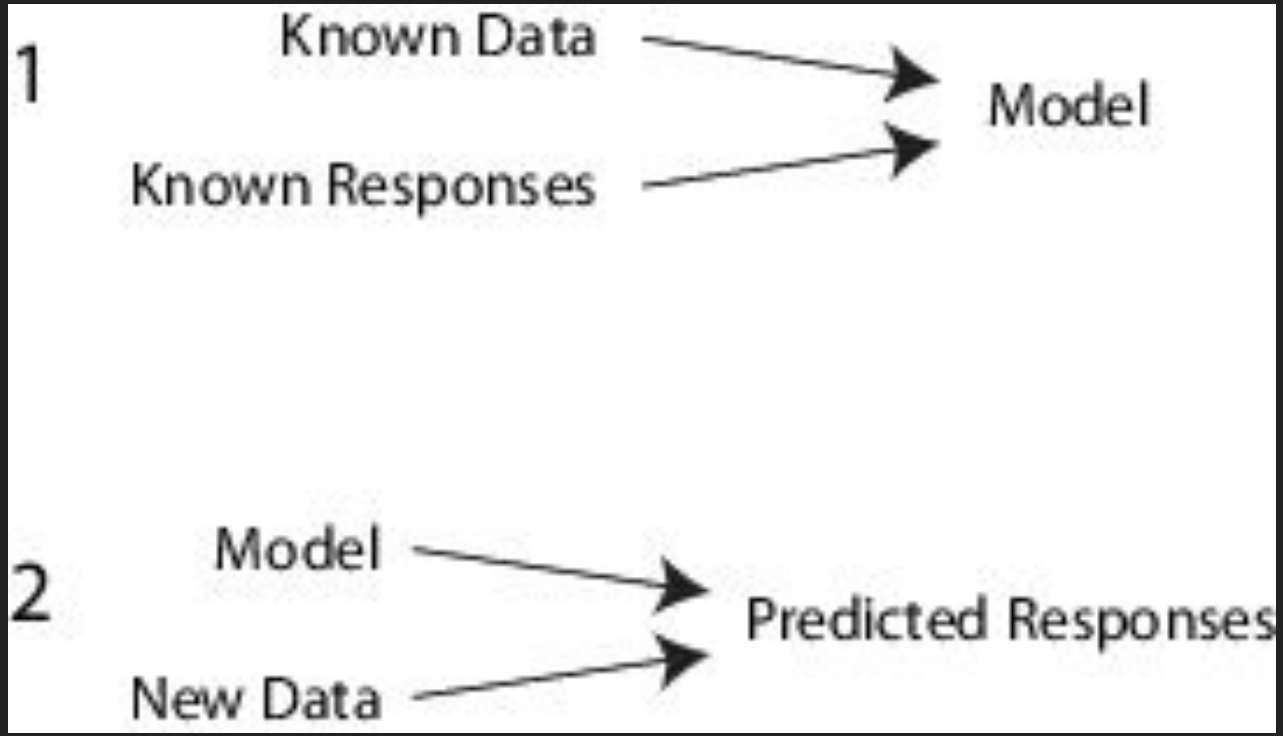




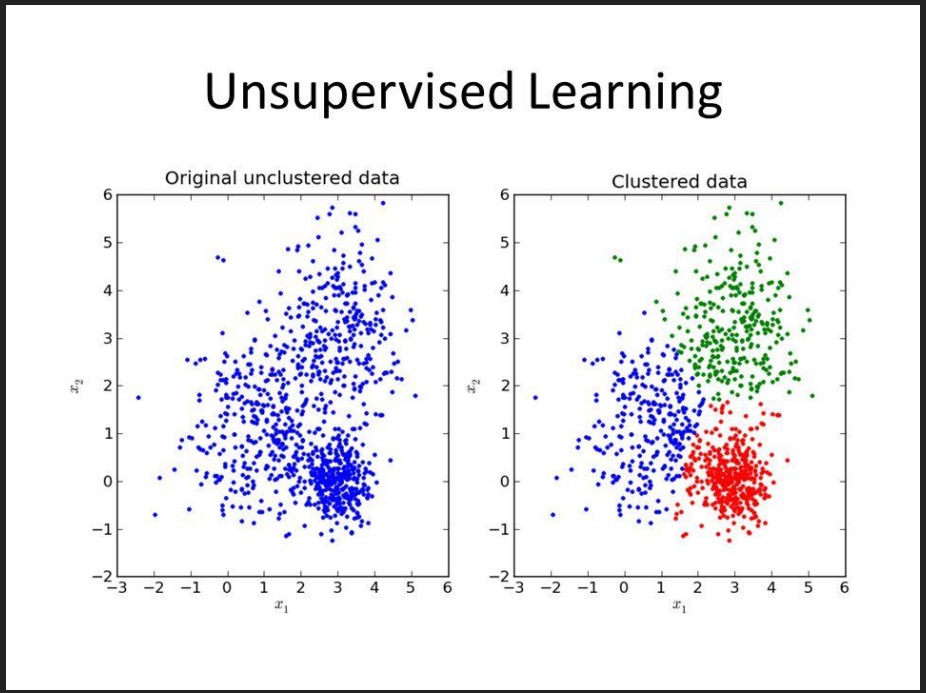


Types of Learning

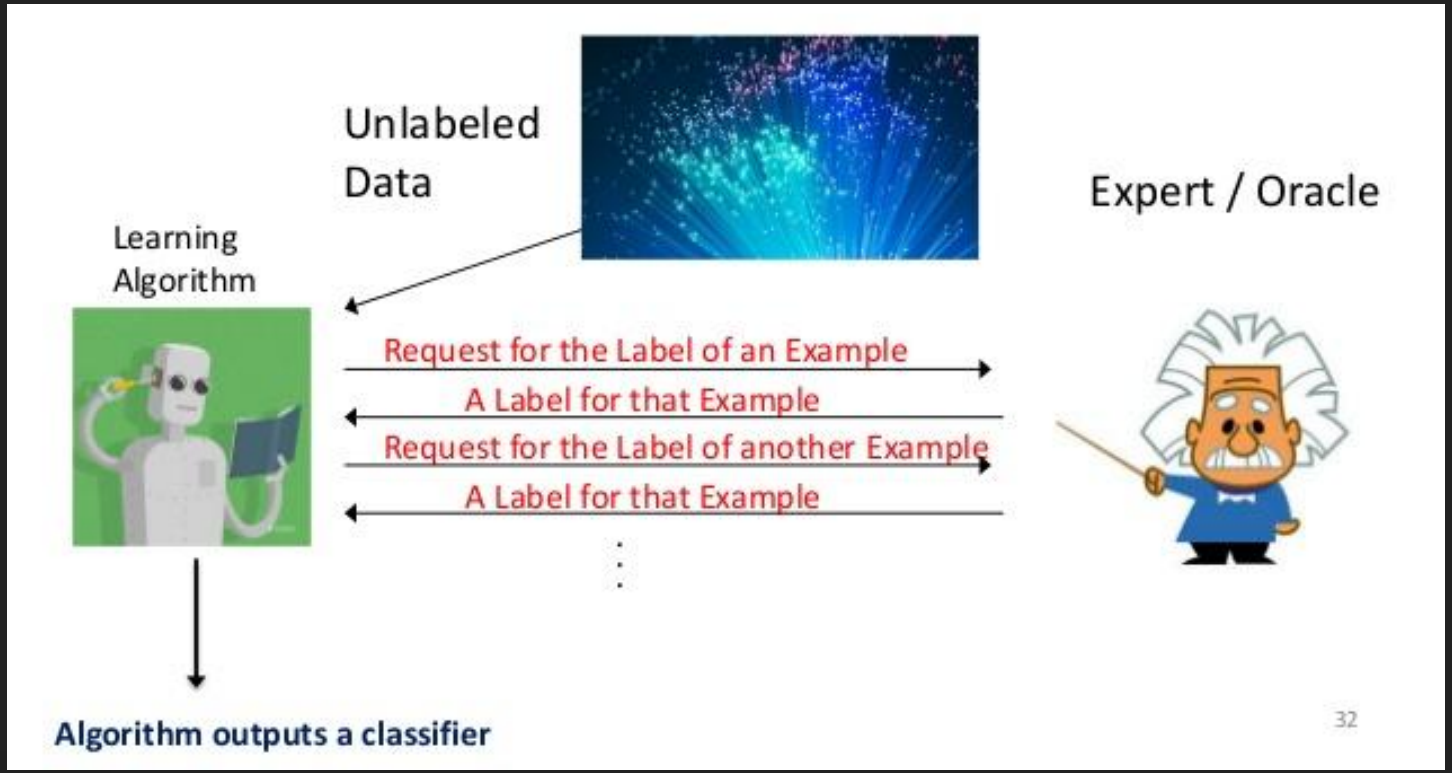
Supervised Learning



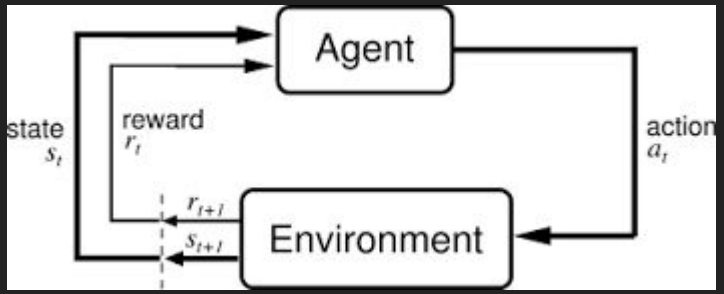
Unsupervised Learning

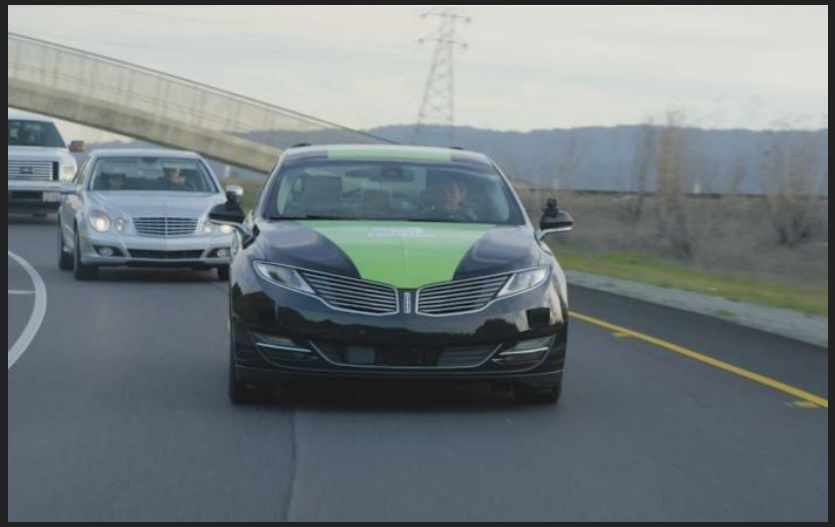


Semi Supervised Learning and Active Learning



Reinforcement and Imitation Learning





What does AI look like in industry?

- Self Driving Cars
- Robotics
- Recommendation Engines
- Language Translation and Understanding
- Speech Synthesis

What does AI look like in Academia?

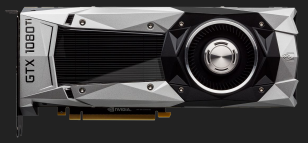
- Teaching Art to Computers
- Hyper Realistic CGI
- Bots
- New Archs/Components for NNs
- One Shot Learning
- Visual Question Answering

What is required to create AIs yourself ?

High End Workstations

Cloud (recommended)

CUDA
(standard for DL)



Training

DL Frameworks

Inference

UIs

NVIDIA DIGITS