Data Science

Machine Learning Techniques

Hang Zhang, Ph.D.

December 3rd, 2018

Announcements

- Next lecture will be the last (December 10th)
- Next week, bring a laptop with Python 3.5/3.6 and TensorFlow installed (TensorFlow CPU version OK) for hands-on deep learning labs
- Capstone project report due December 9th, 11:59pm
- Capstone project report requirements
- No extension will be honored due to the hard deadline for the final score required by UW (1 week after the last lecture)

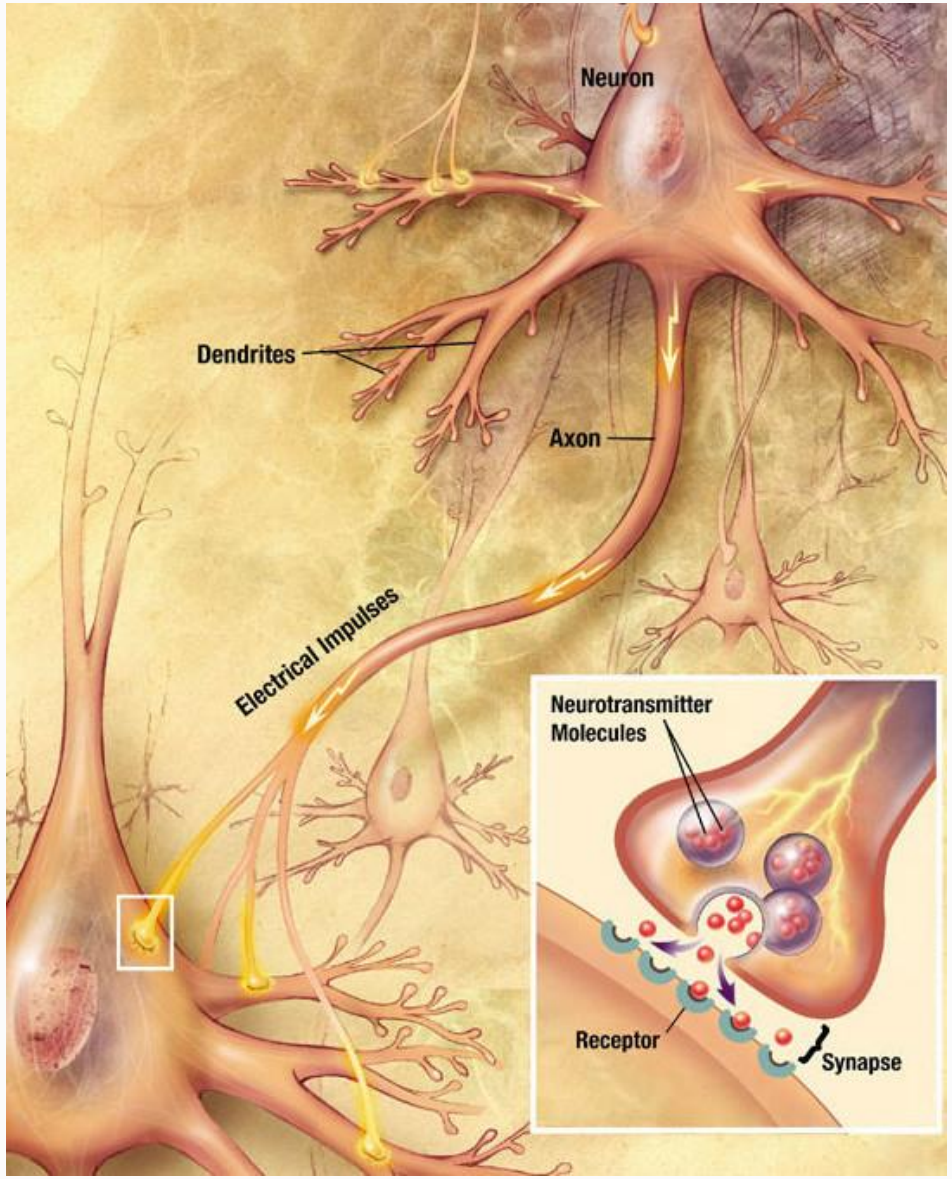
Where Are you (As of November 28th)

Team Name	Public Ranking	Best Score	# of Submissions
Datas R Us	694(626)	5.380(32.265)	7(2)
DS420_PandaPlayers	77 (406)	1.029(1.958)	26(3)
DS420_astroclass_capstone (Merged with datarangers)	410(303)	1.425(1.425)	2(1)
DS420_TYF	**766**	**30.267**	**1**
DS420-GRJT	332(295)	1.169(1.420)	10(4)
DS420_Galileo's_Gala	731(612)	7.250(24.950)	5(3)
DS420_Galaxy	446(502)	1.599(2.492)	9(2)

It is very difficult for me to give you score on your capstone project if you just make 1 submission or you cannot improve from your baseline model although you have multiple submissions.

Artificial Neural Networks

Neural Networks



Source: https://en.wikipedia.org/wiki/Neuron

Source: https://techbuf.com/human-brain-neural-network/

How We Investigate Neural Control of Our Motion, Behavior, Emotion, etc?

Source: https://www.technologyreview.com/s/603492/10-breakthrough-technologies-2017-reversing-paralysis/

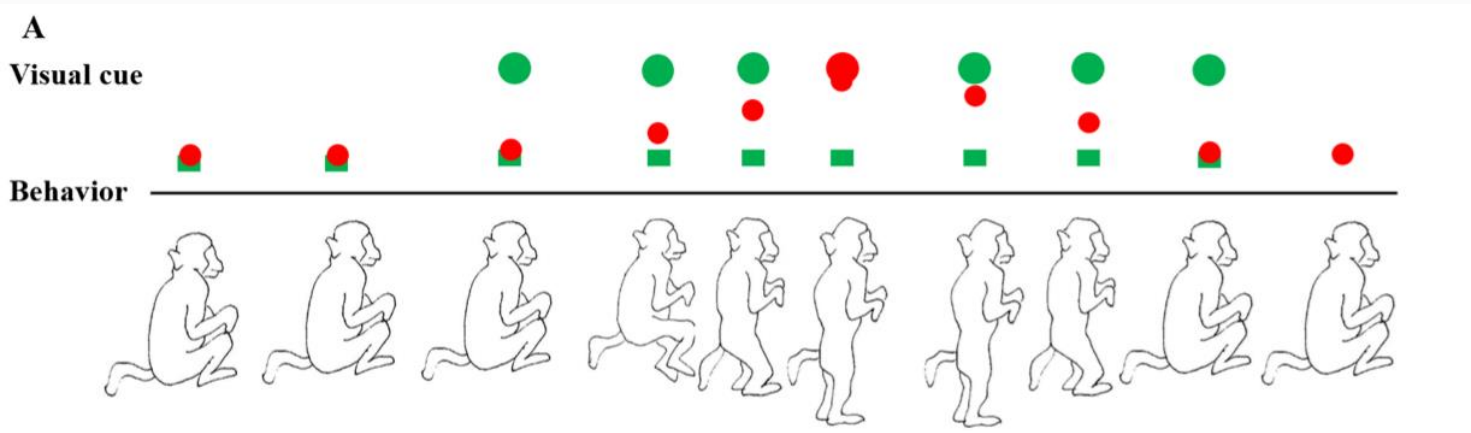
Ma *et al. Behavioral and Brain Functions* (2015) 11:15
DOI 10.1186/s12993-015-0061-0

BBF BEHAVIORAL AND BRAIN FUNCTIONS

RESEARCH Open Access

Neuronal representation of stand and squat in the primary motor cortex of monkeys

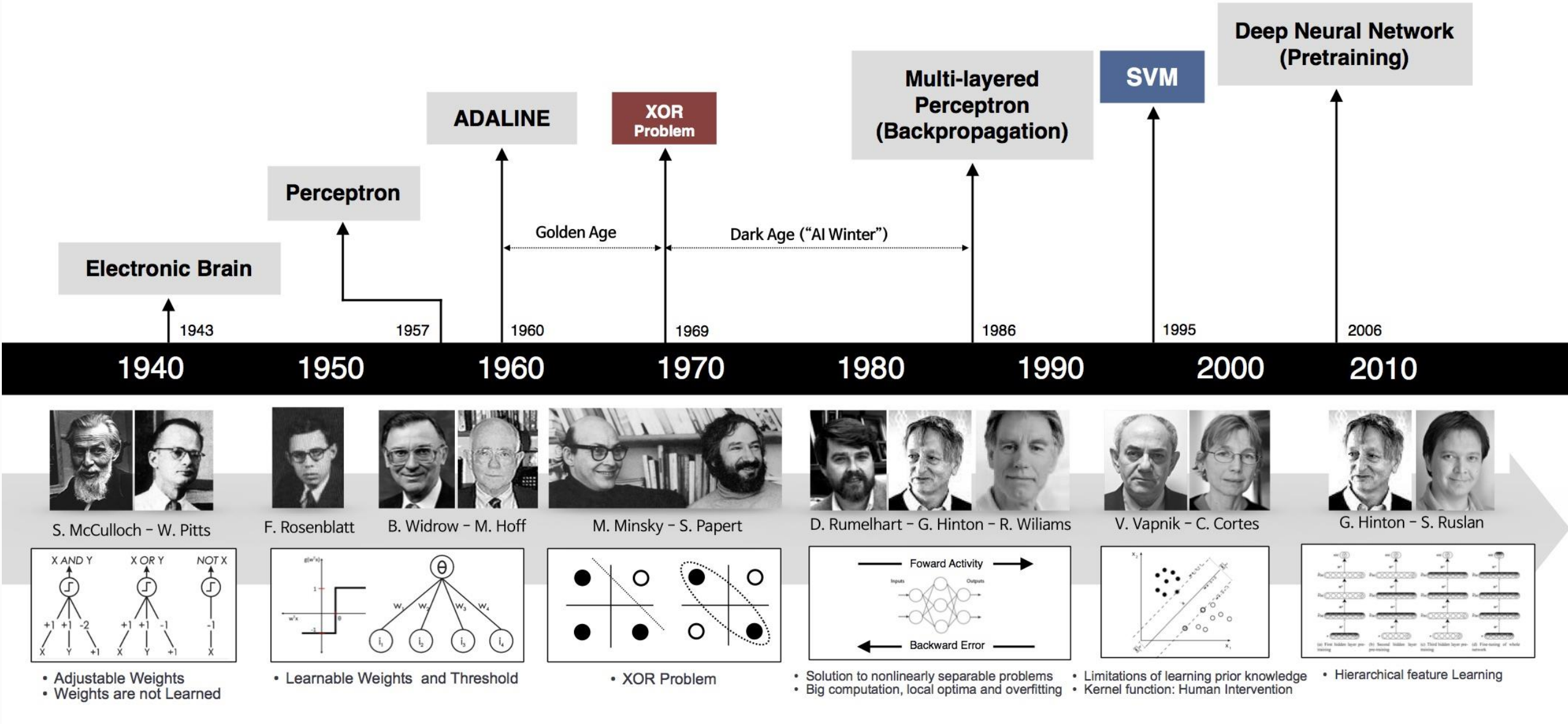
Chaolin Ma[1,2*], Xuan Ma[2,3], Hang Zhang[2], Jiang Xu[3] and Jiping He[2,3*]



Artificial Neural Networks

- We call it artificial because:
  - It is a simplification of the real neural network
  - It is analog to using spike rate as the information conveyer.
  - Every neuron (input, hidden, output) is outputting a single value (analog to the spike rate)

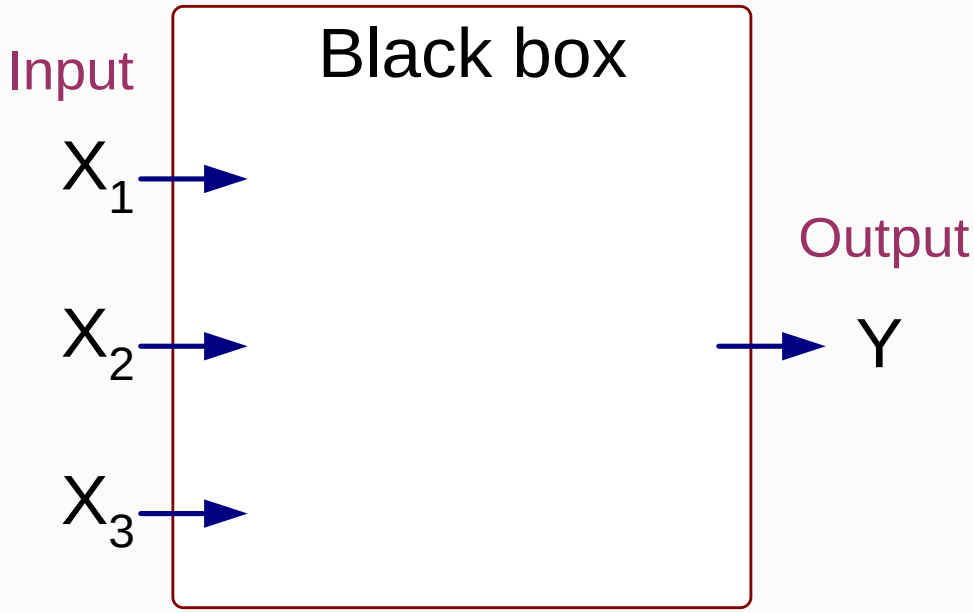
History of Artificial Neural Networks



Source:http://qingkaikong.blogspot.com/2016/11/machine-learning-3-artificial-neural.html

Artificial Neural Networks (ANN)

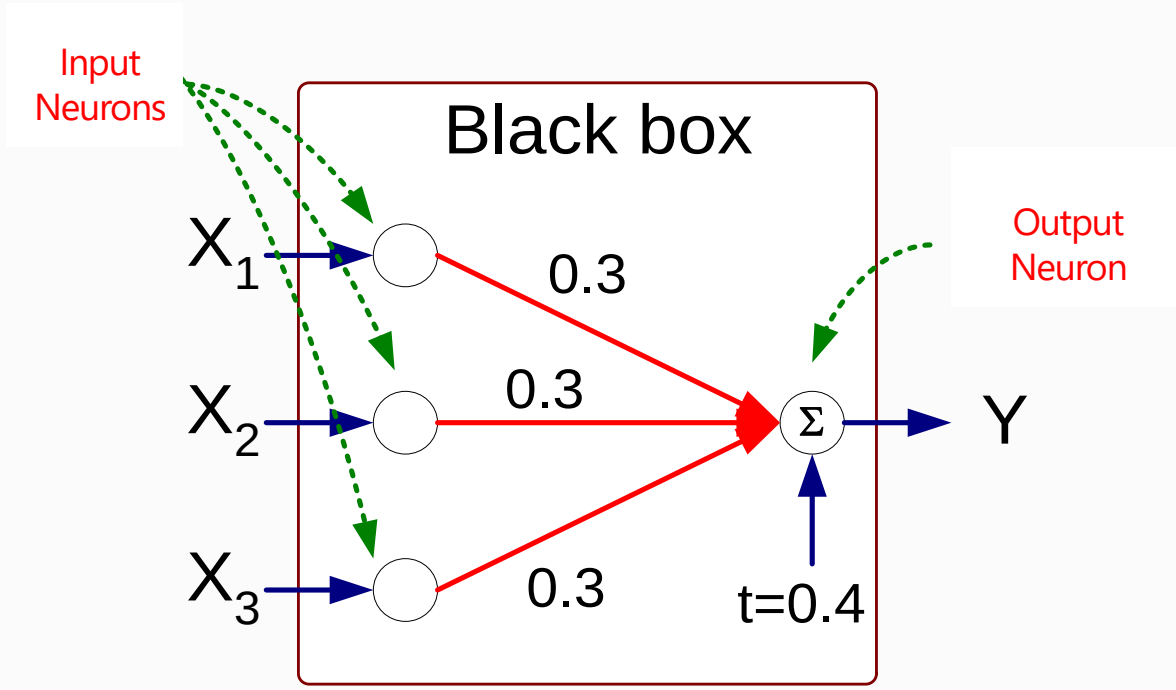
X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



Output Y is 1 if at least two of the three inputs are equal to 1.

Artificial Neural Networks (ANN)

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

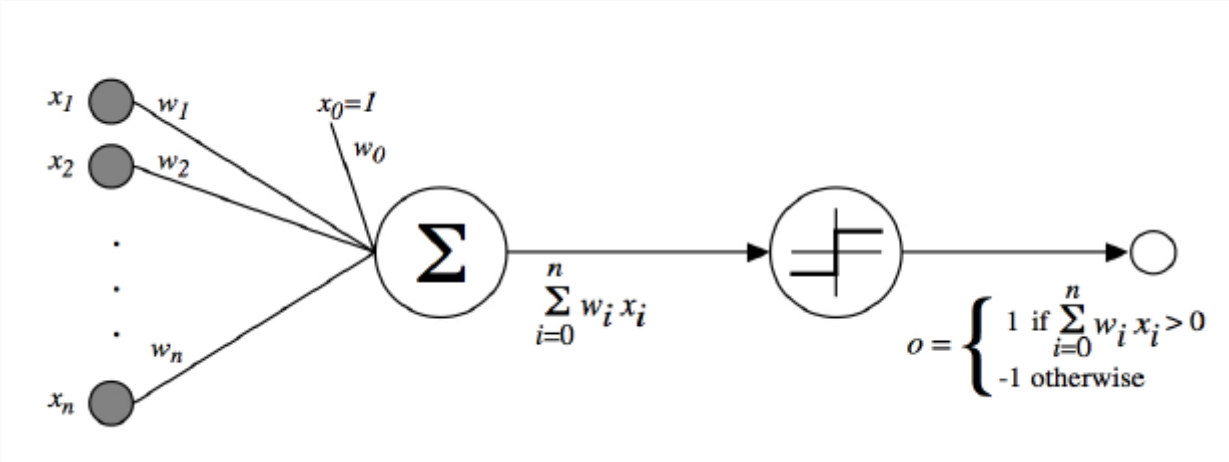


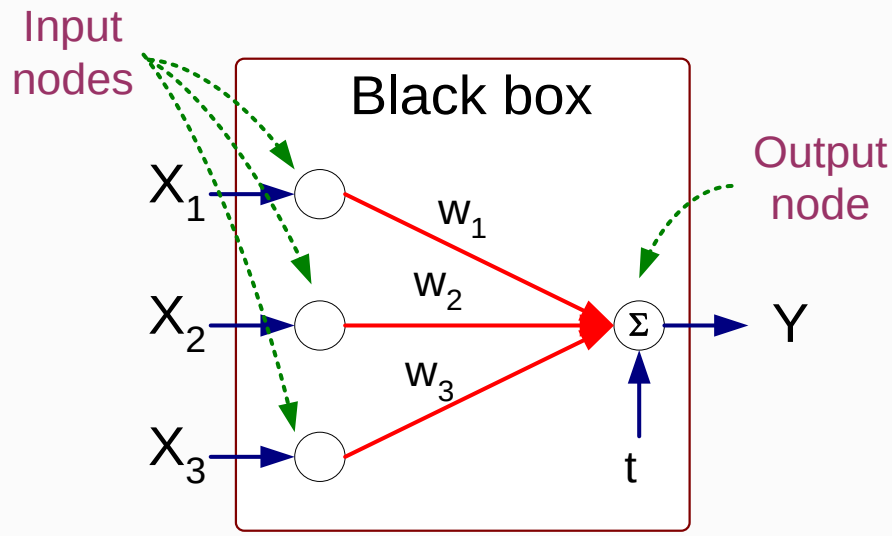
$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where } I(z) = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Artificial Neural Networks (ANN)

- Model is an assembly of inter-connected neurons (nodes) and links with weights;
- Output node sums up each of its input value according to the weights of its links;
- Compare what the output node receives against some threshold t and decide the output value (label)





Perceptron Model

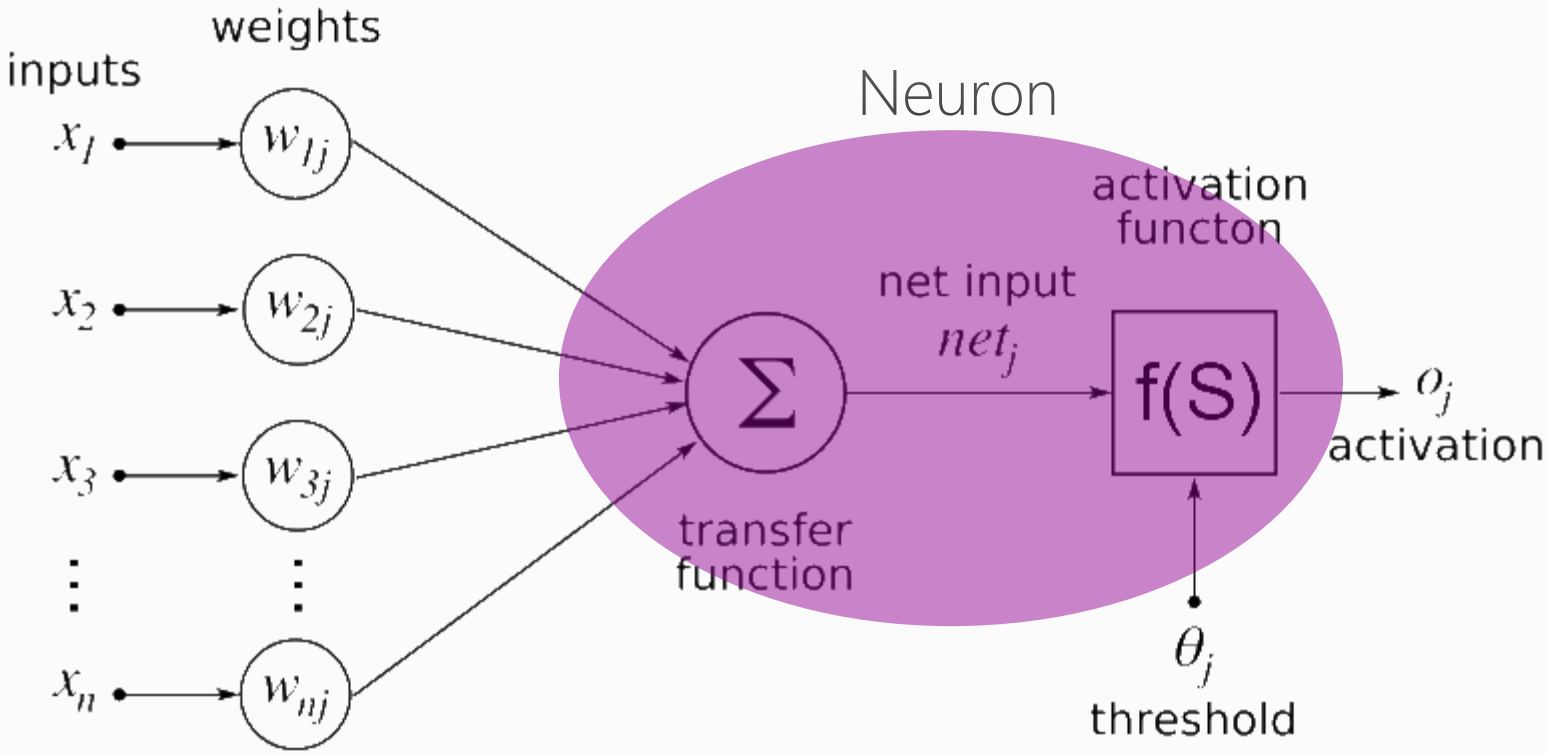
$$Y = I(\sum_i w_i X_i - t) \quad \text{or}$$

$$Y = sign(\sum_i w_i X_i - t) = sign(\sum_i w_i X_i + w_0)$$

$$= sign(\sum_{i=0}^{n} w_i X_i^{'})$$

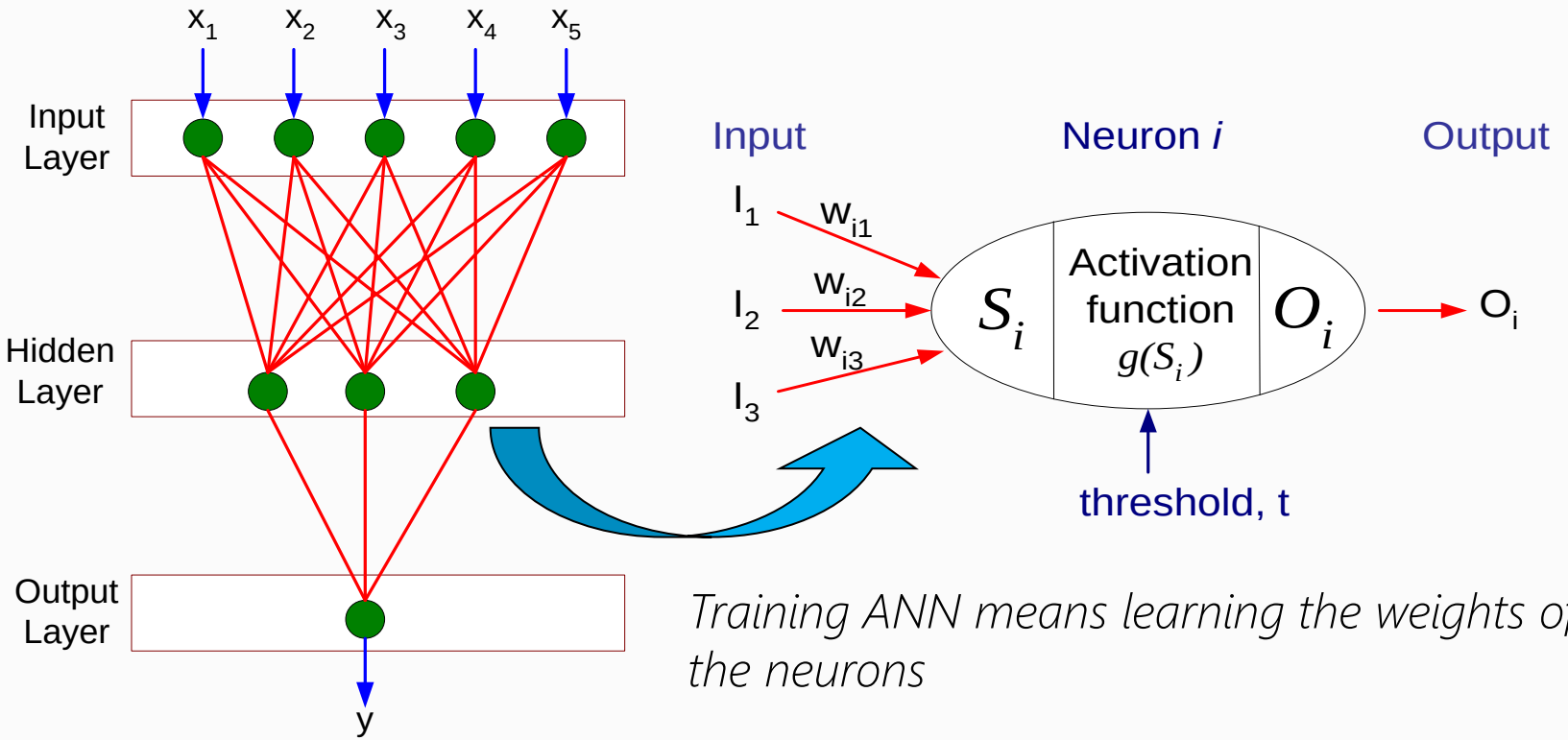
$$where\ X_i^{'} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

Neurons in Artificial Neural Networks



- Linear unit if the activation function is linear
- Nonlinear neuron if otherwise

General Structure of ANN



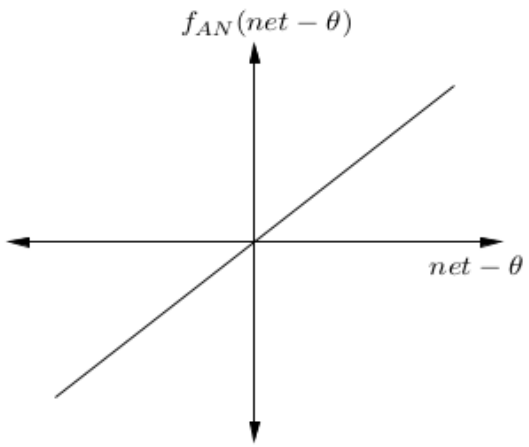
Training ANN means learning the weights of the neurons

- Perceptrons only have two layers: the input layer and the output layer
- Perceptrons only have one output unit;
- Multi-layer ANNs consist of an input layer, hidden layer (s), and output layer
- Multi-layer neural networks can have several output units.
- Input layer neurons only outputs the input variables

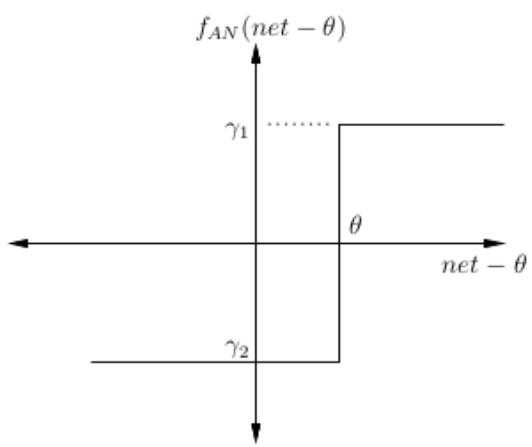
Multi-Layer Artificial Neural Networks

- The units of the hidden layer function as input units to the next layer
- It is said that multi-layer ANN can express any nonlinear relationship between input and output variables
- However, multiple layers of linear units still produce only linear functions

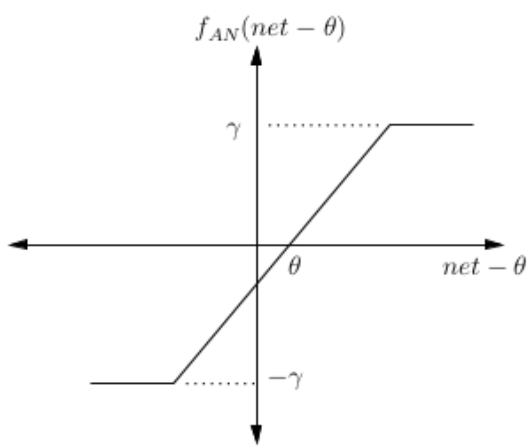
Popular Activation Functions for Classification



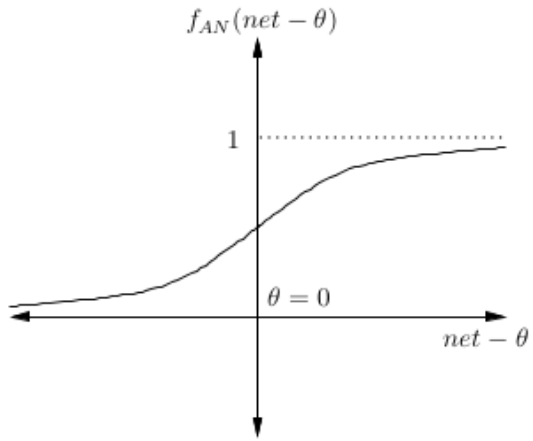
(a) Linear function



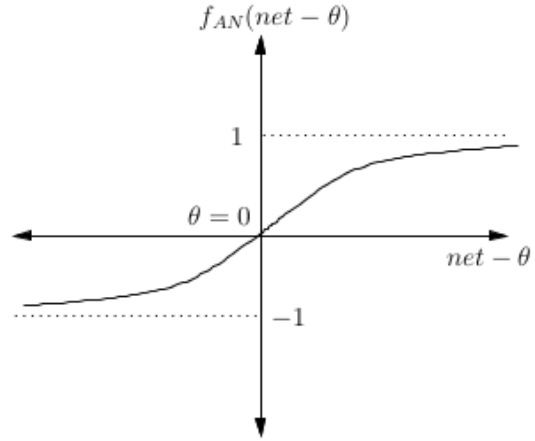
(b) Step function



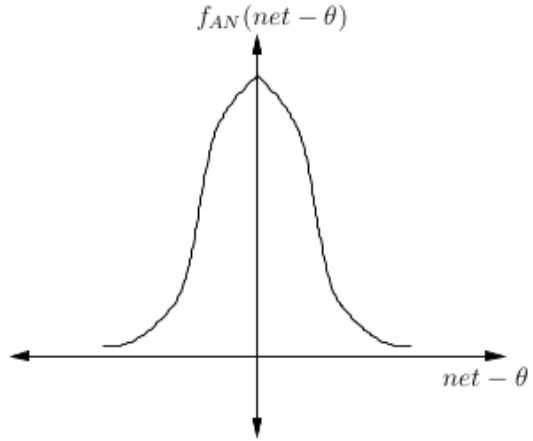
(c) Ramp function



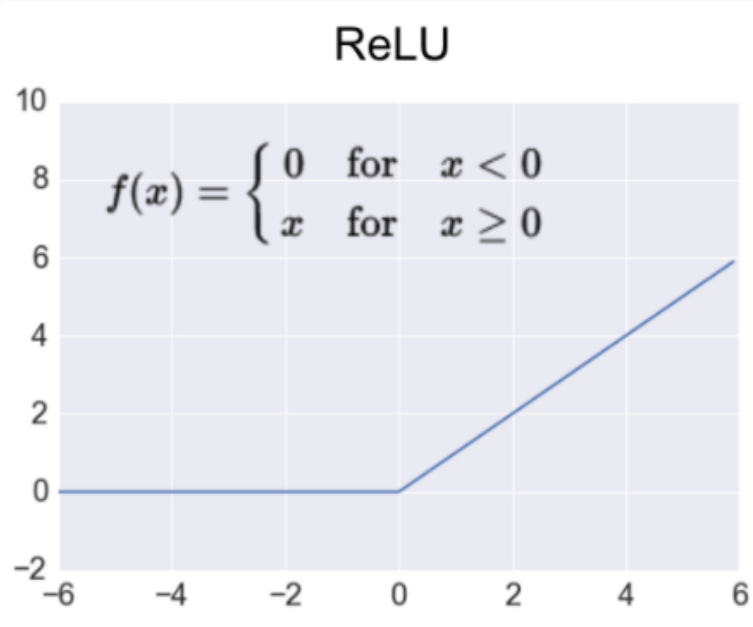
(d) Sigmoid function



(e) Hyperbolic tangent function



(f) Gaussian function

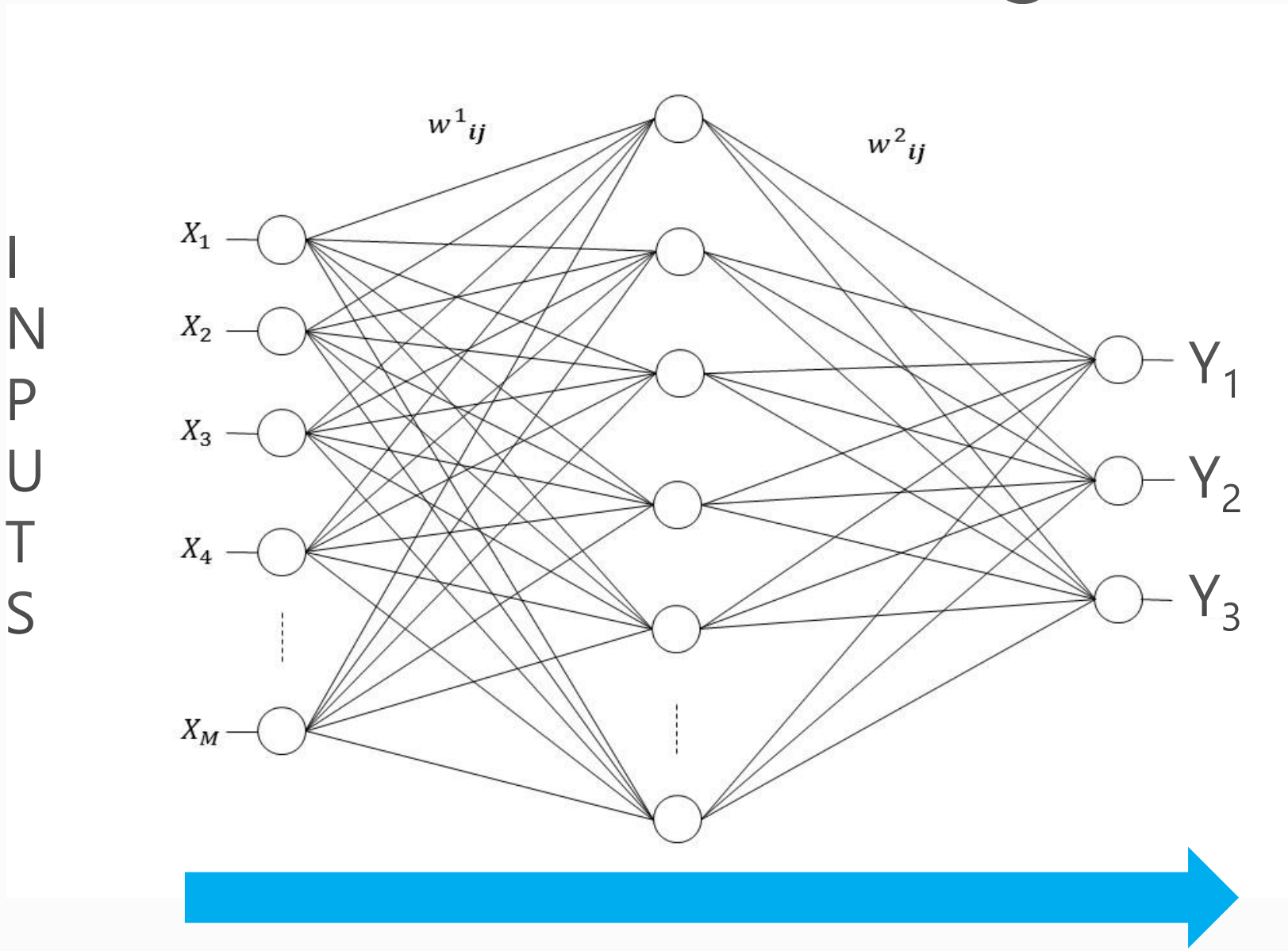


- Rectified Linear Unit
- Most popular activation function in deep neural network

How A Multi-Layer Neural Network Works

- The inputs to the network correspond to the attributes measured for each training tuple
- Inputs are fed simultaneously into the units making up the input layer
- They are then weighted and fed simultaneously to a hidden layer
- The number of hidden layers is arbitrary, although usually only one
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction
- The network is feed-forward: None of the weights cycles back to an input unit or to an output unit of a previous layer
- From a statistical point of view, networks perform nonlinear regression: Given enough hidden units and enough training samples, they can closely approximate any function

Feedforward Phase of Learning



Artificial Neural Network Topology

- How many hidden layers?
- How many neurons in each layer?
  - Input layer: number of neurons = number of independent variables (X)
  - Hidden layer: no golden rule to choose the number of hidden nodes. Trial and fail.
    - You need to decide based on # of training observations, complexity of the problem, etc.
    - If you have 50 input variables, you choose 100 hidden nodes, 1 output variable. Then, you need to estimate 50*100 + 100*1 = 5100 weights from your training observations. If you only have 1000 observations, you are seriously overfitting.
  - Output layer: if it is binary classification, 1 output node. Multiclass classification, 1 output node for each class. Regression, 1 output node.
- What is the activation function of each neuron?

Hyperparameters to Tune in Artificial Neural Networks

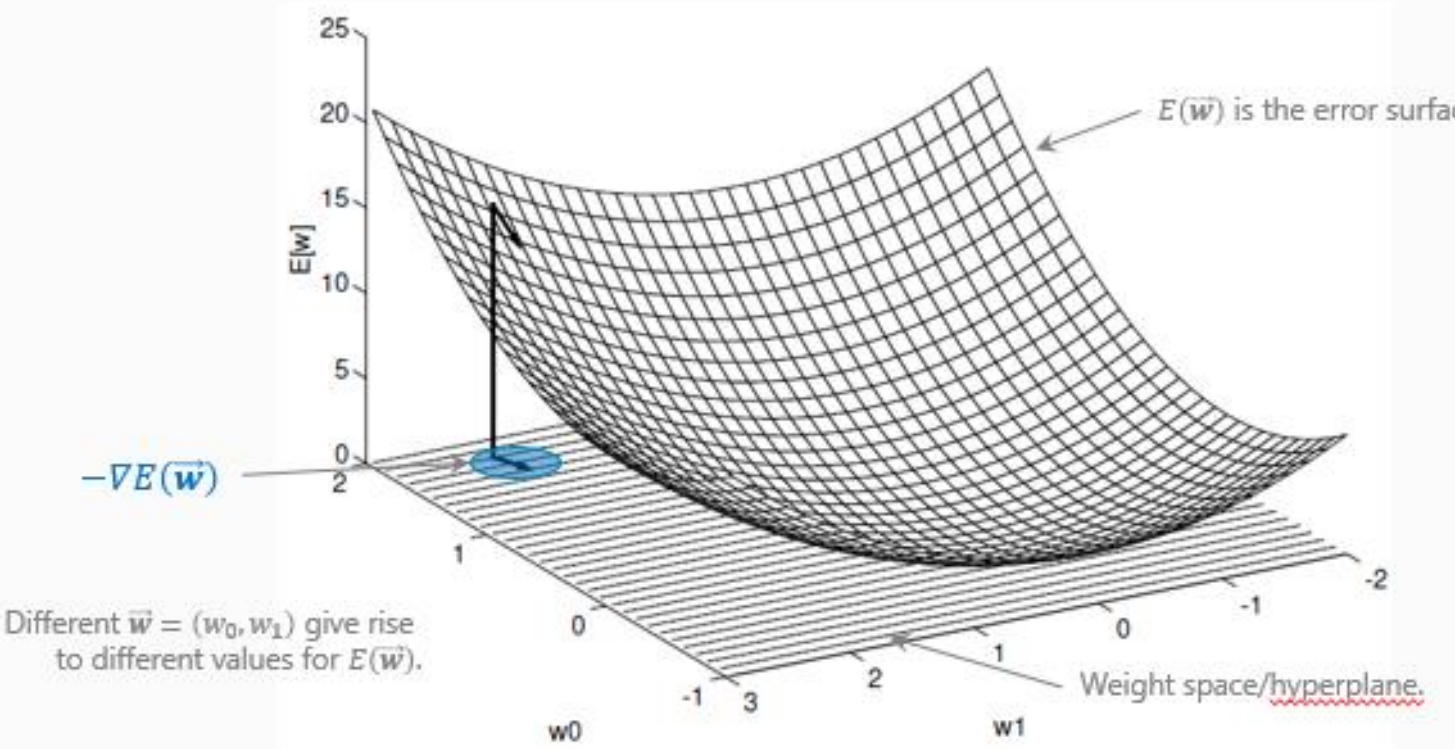
- Topology of the artificial neural networks
- Learning rate of the stochastic gradient descent algorithm
- Learning iterations of the stochastic gradient descent algorithm
- Variation of the initial weight matrixes.
- Once a network has been trained and its accuracy is unacceptable, repeat the training process with a *different set of hyperparameters*

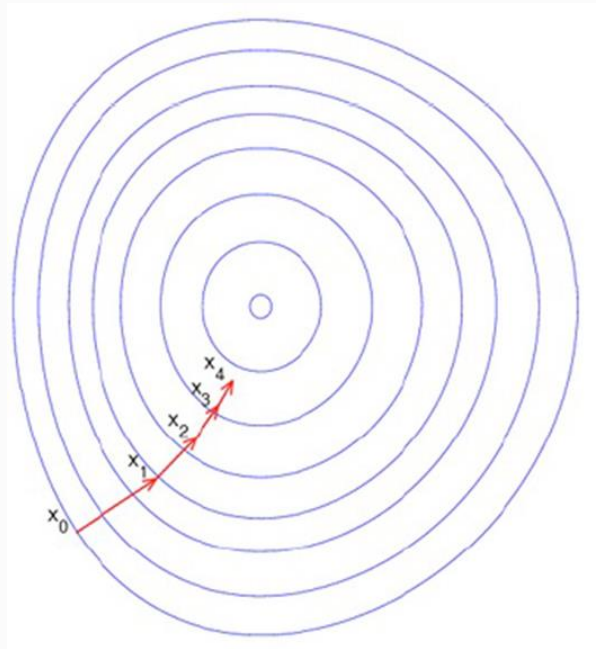
Stochastic Gradient Descent

The gradient of the error: $\nabla E(\vec{w}) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_d}\right]$

(a vector in weight space) specifies the direction of the argument that leads to the steepest increase for the value of the error.

The negative of the gradient gives the direction of the steepest decrease.





Algorithm for Learning ANN

- Initialize the weights (w_0, w_1, …, w_k)
- Adjust the weights in such a way that the output of ANN is consistent with class labels of training examples
  - Objective (Loss) function:

$$E = \sum_i \left[Y_i - f(w_i, X_i) \right]^2$$

  - Find the weights w_i's that minimize the above objective function
    - e.g., backpropagation algorithm

Backpropagation

- Iteratively process a set of training examples & compare the network's prediction with the actual known target value
- In each iteration, the training examples are shuffled
- For each training example, the weights are modified to minimize the mean squared error between the network's prediction and the actual target value
- Modifications are made in the "backwards" direction: from the output layer, through each hidden layer down to the first hidden layer, hence "backpropagation"
- Steps
  - Initialize weights to small random numbers, associated with biases
  - Propagate the inputs forward (by applying activation function)
  - Backpropagate the error (by updating weights and biases)
  - Terminating condition (when error is very small, etc.)

Artificial Neural Networks

Known samples (historical data) are used to "train" the network.

Input data (x_i) are assigned weights (w_i) and combined in the "hidden" layer – like a set of linear regressions. These sets can then combined in additional layers – like regressions of regressions.

The sum of data and weights are transformed ("squashed") to the range of the training data and error is measured.

A supervised training algorithm uses output error to adjust network weights to minimize errors.

Practical Considerations

- All features need to be scaled to mean 0 and standard deviation 1
- Many parameters must be carefully selected to ensure a good performance.
- Although the deficiencies of Backpropagation nets cannot be completely cured, some of them can be eased by some practical means.
- Initial weights (and biases)
  - Random, [-0.05, 0.05], [-0.1, 0.1], [-1, 1]
- Learning Rate: 0.01 – 0.1
- Number of iterations: 20 – 500
- Use validation data (or cross validation) to choose the optimal parameters

Over-Training/Over-Fitting

- Trained net fits very well with the training samples, but not with new input patterns
- Over-training may become serious if
  - Training samples were not obtained properly
  - Training samples have noise

Control over-training for better generalization

- Either divide the samples into two sets
  - - 90% into training set: used to train the network
  - - 10% into test set: used to validate training results

  periodically test the trained net with test samples, stop training when test results start to deteriorating, or
  - Use Cross-validation

- Stop training early
- Add noise to training samples: x:target becomes x+noise:target

Hyperparameters

Gradient descent

- Initial weights
- Learning rate schedule
- Batch size
- Stopping criteria/number of training iterations

Tuning

- Split data set into training, validation (or cross-validation (cv)), and test
- Fit model on training set
- Tune hyperparameters on Cross-Validation set
- Evaluate on test set

Neural Network as a Classifier

Strength

- High tolerance to noisy data
- Ability to classify untrained patterns
- Well-suited for continuous-valued inputs *and outputs*
- Successful on an array of real-world data, e.g., hand-written letters
- Algorithms are inherently parallel
- Techniques have been developed for the extraction of rules from trained neural networks

Weakness

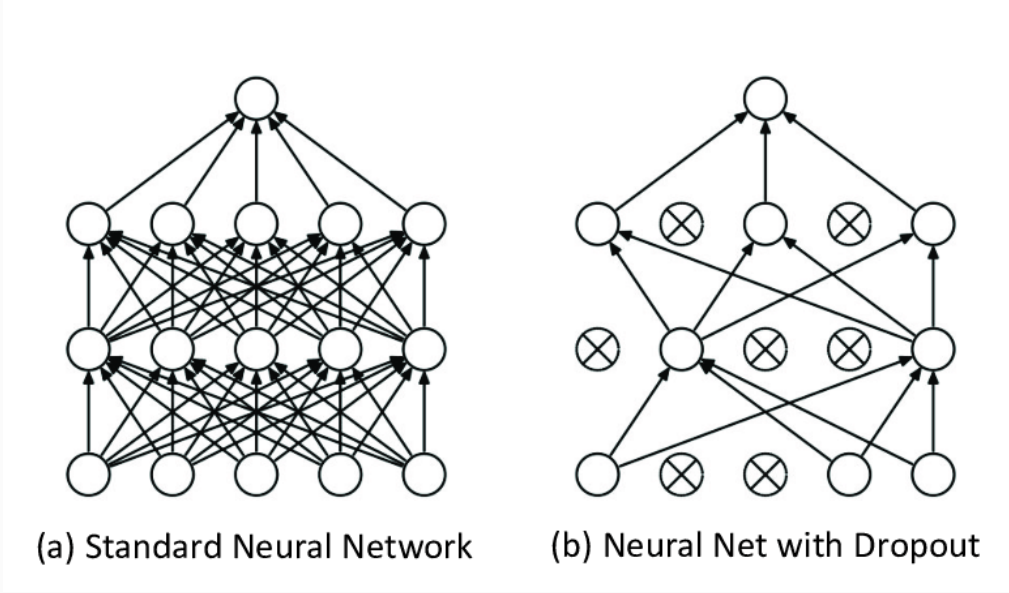
- Long training time
- Require a number of parameters typically best determined empirically, e.g., the network topology or "structure."
- Poor interpretability: Difficult to interpret the symbolic meaning behind the learned weights and of "hidden units" in the network

Overfitting in Artificial Neural Networks

- Assuming the training set is fixed
- Larger number of hidden nodes, larger number of hidden layers (deeper neural network)
  - Pro: give you better opportunity to model the complex relationship between X and Y
  - Con: you have higher risk of overfitting
- Larger number of iterations
  - Pro: reduce the training error, possibly generalize better
  - Cons: reduce the training error too much, possibly overfitting

Ways of Preventing Overfitting

- Select the optimal hyperparameters based on accuracy on validation
- Regularization:
  - L2 regularization $E = \sum_i [Y_i - f(\mathbf{w}, X_i)]^2 + \lambda \|\mathbf{w}\|_2$
- Dropout



Neural network with dropout won the Merck competition in 2012

Neural Networks: In-class Lab

SVM vs. Neural Network

SVM

- Deterministic algorithm
- Nice generalization properties
- Hard to learn – learned in batch mode using quadratic programming techniques
- Using kernels one can learn very complex functions (hypothesis space)

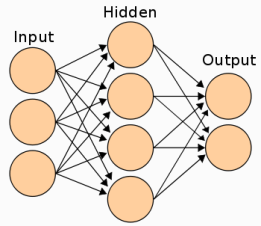
Neural Network

- Nondeterministic algorithm
- Generalizes well but doesn't have strong mathematical foundation
- Can easily be learned in incremental fashion
- To learn complex functions—use multilayer perceptron (nontrivial)

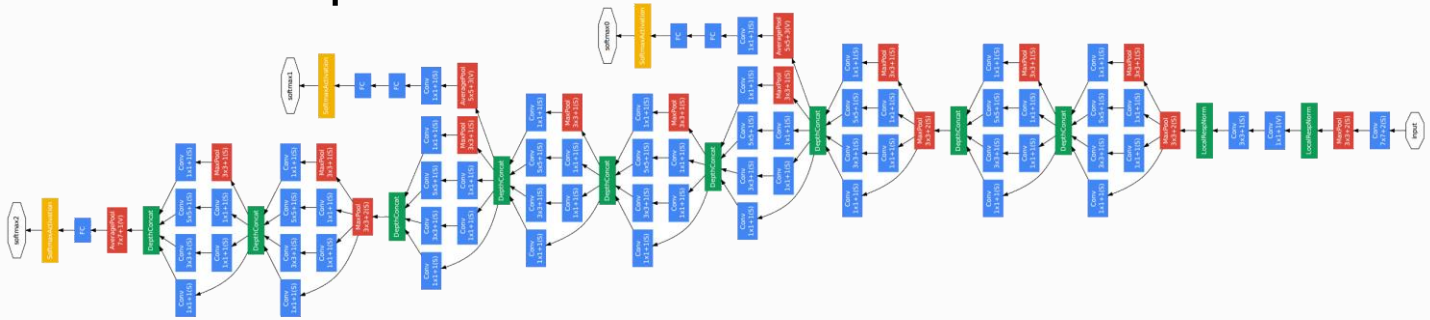
What is this DNN thing of which you speak?

It's a brand new (actually, started in the 40's) type of Machine Learning using Neural Networks (well, mostly) to solve all of humanity's problems (umm, no)

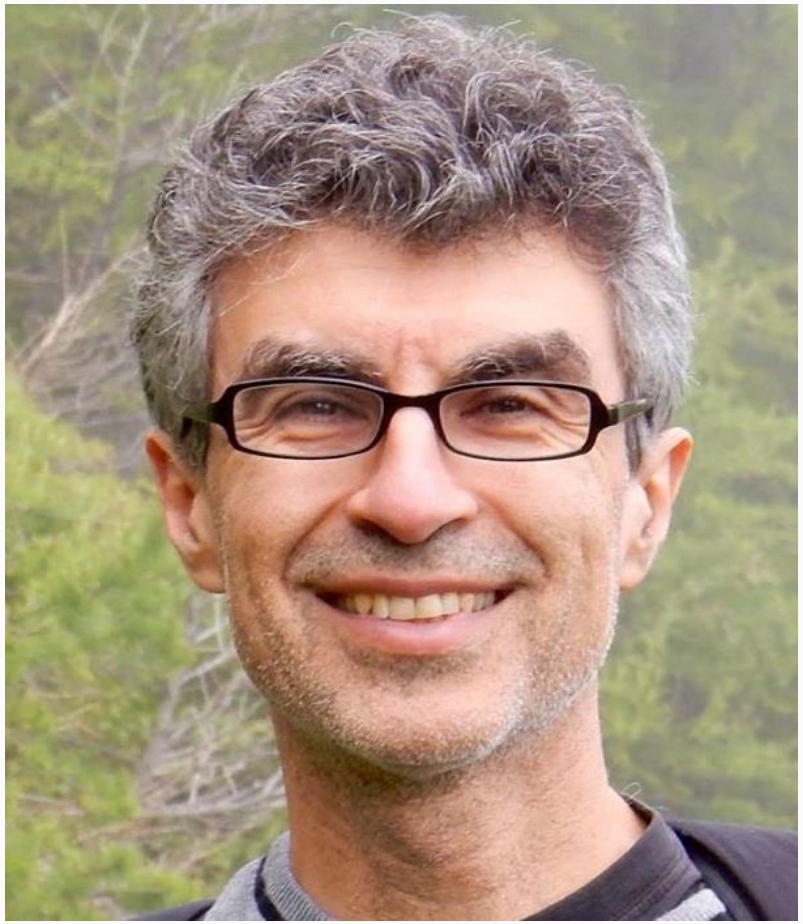
- It's Neural Networks (NNs) *gone deeper*
- "Old" NNs are all



- But "Deep" NNs are all



WHAT IS DEEP LEARNING?



"Learning **multiple levels of representation** to help a learner accomplish a task of interest, with higher levels capturing more abstract concepts through a **deeper composition** of computations"
Yoshua Bengio

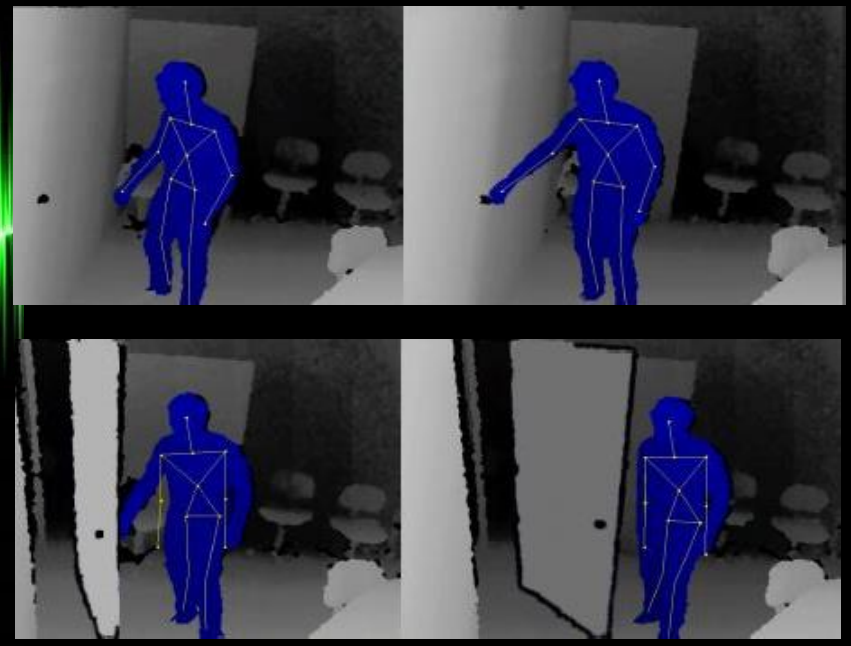
source: http://www.iro.umontreal.ca/~bengioy/yoshua_en/

WHY DEEP LEARNING?

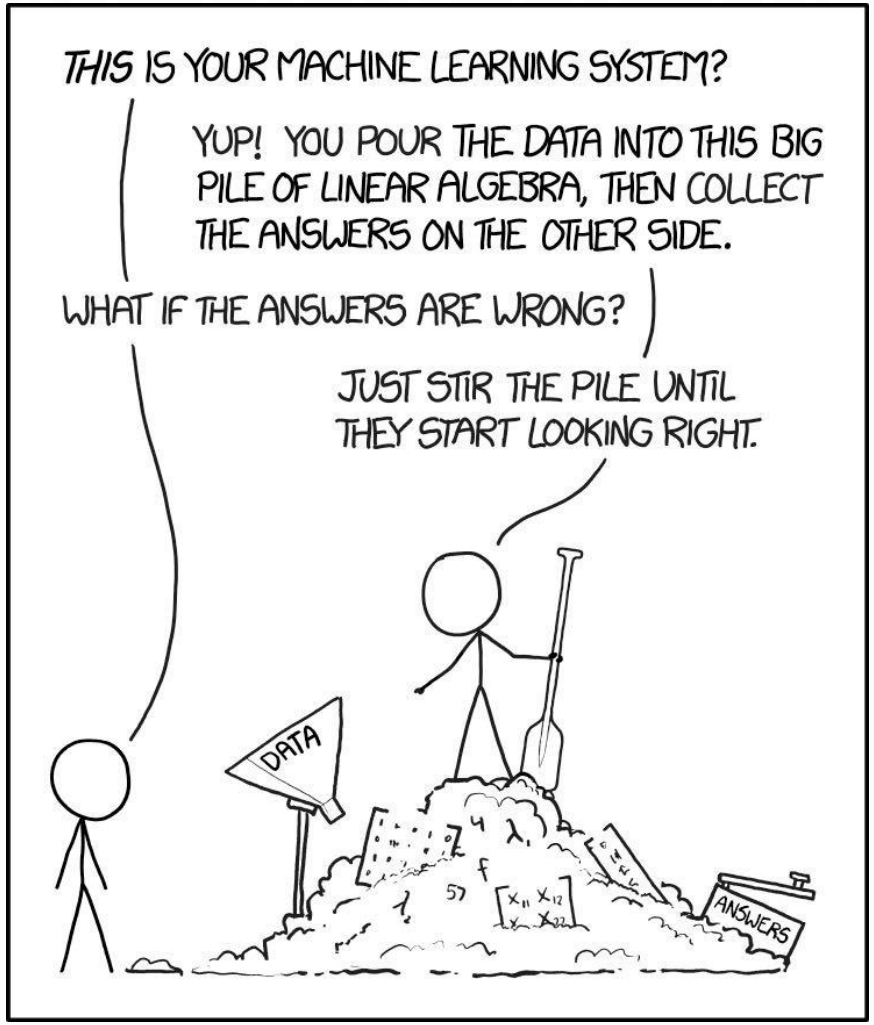
Natural Language Processing

Speech recognition

Computer Vision



WHY DEEP LEARNING?

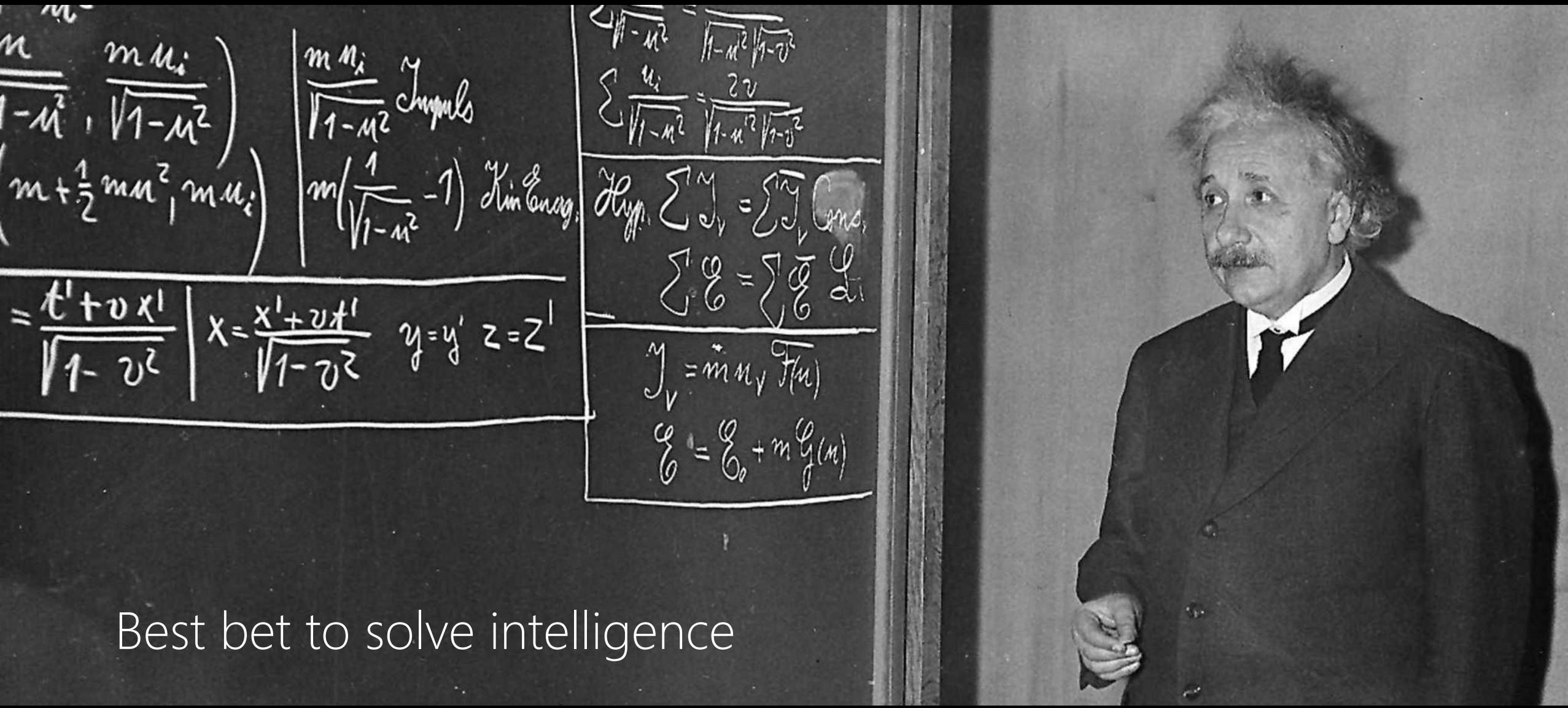


source: https://xkcd.com/1838/

"The more data,
The better it works*"

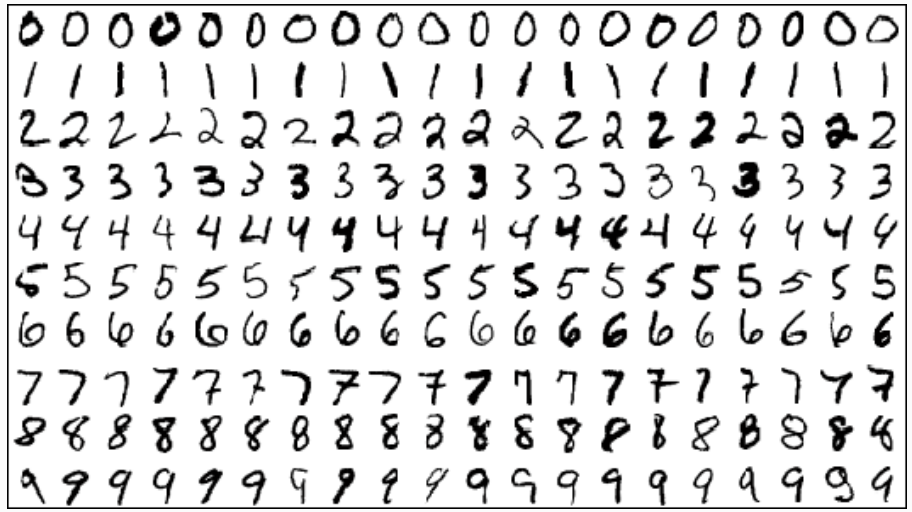
* always read the small print

WHY DEEP LEARNING? BEST BET



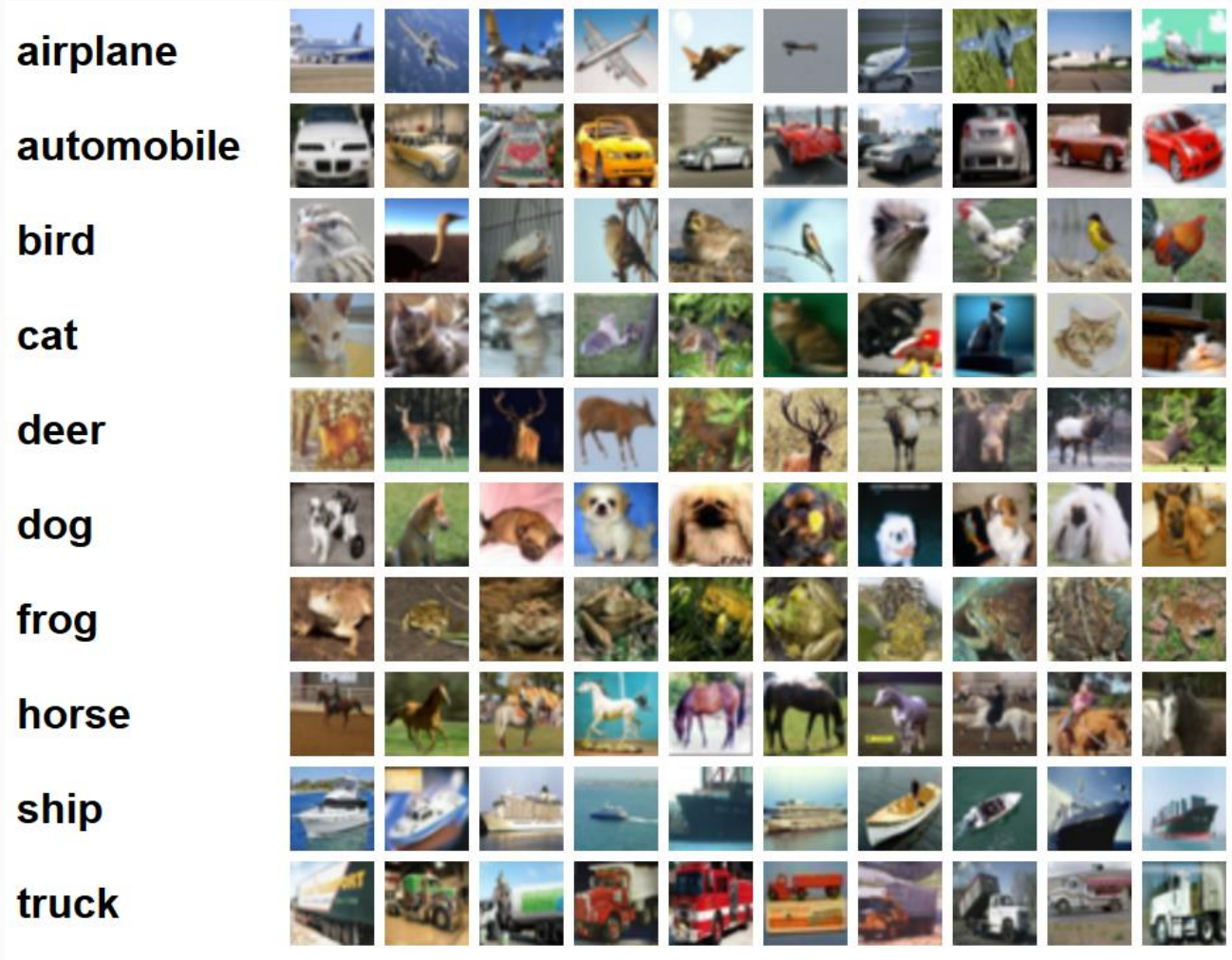
EXAMPLE: DIGIT RECOGNITION

MNIST with CNNs
(source here)



EXAMPLE: Image Classification

CIFAR with ResNet CNN (source here)

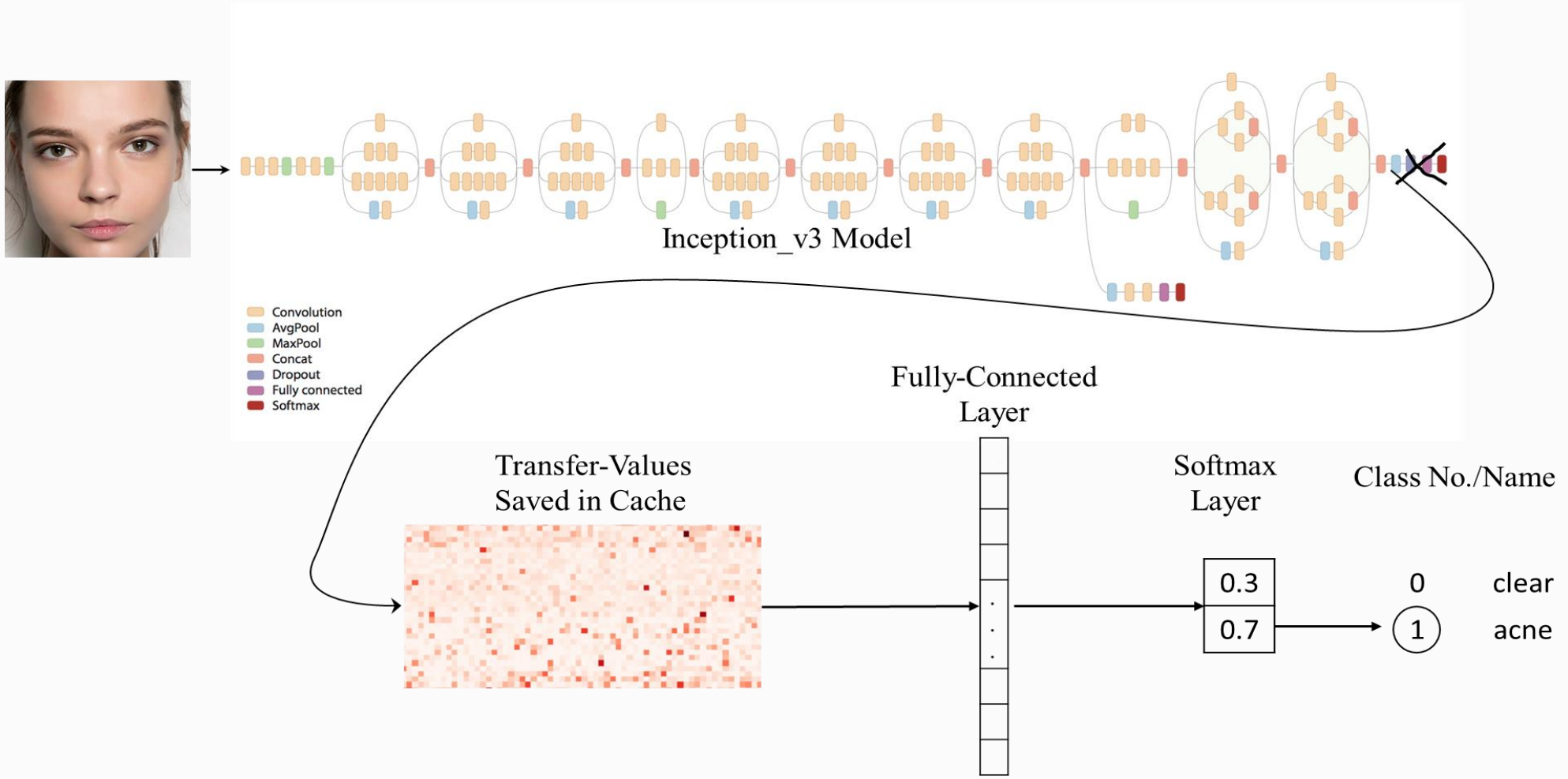


source: https://www.cs.toronto.edu/~kriz/cifar.html

EXAMPLE: Object detection

Object detection with FasterRCNN
(source here)

EXAMPLE: TRANSFER LEARNING



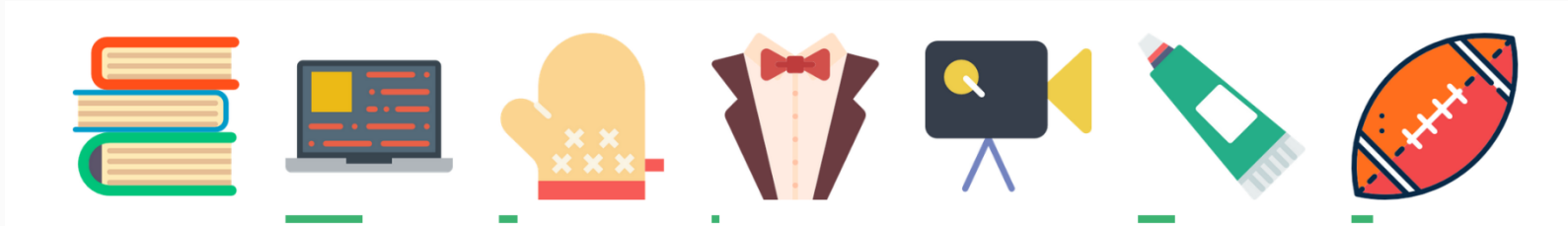
Example: Video Classification

Video classification with 3D CNNs (source here)

Example: Text classification

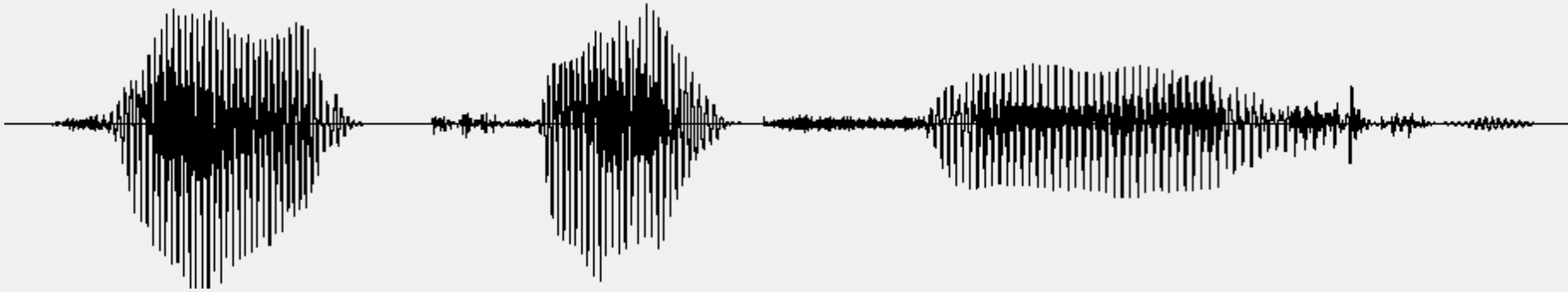
Text classification with CNNs (source here)

Input text: *"It was a breeze to configure and worked straight away"*



EXAMPLE: Speech Recognition

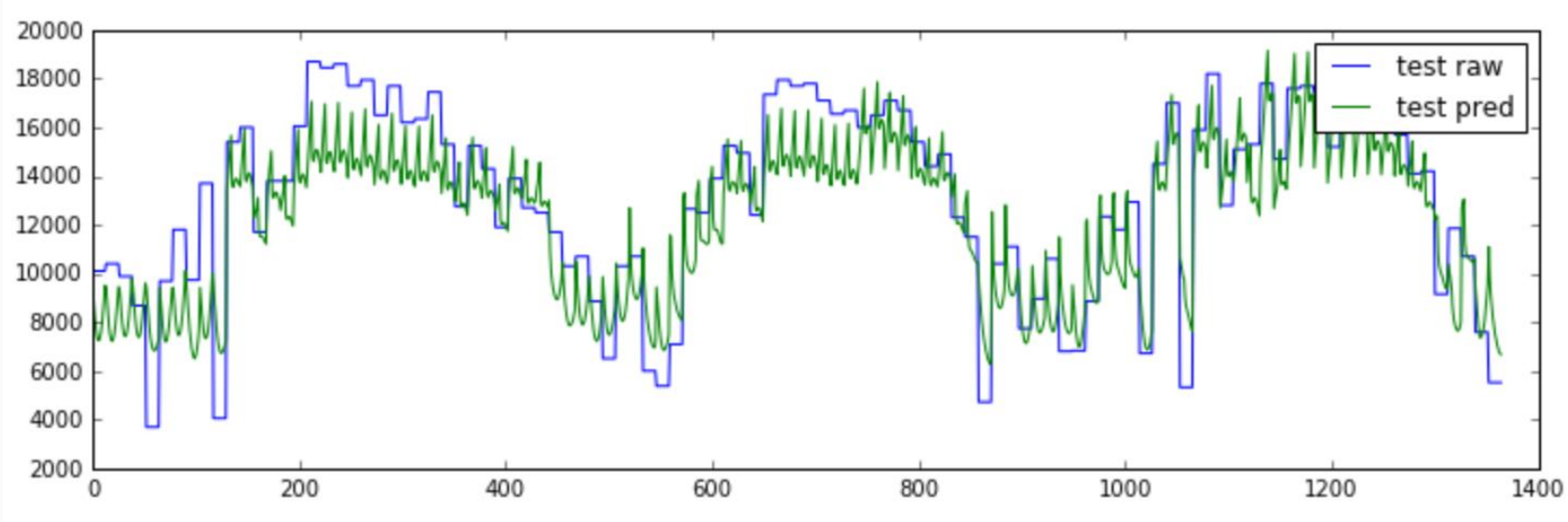
Speech recognition with RNN (source here)



EXAMPLE: TIME SERIES

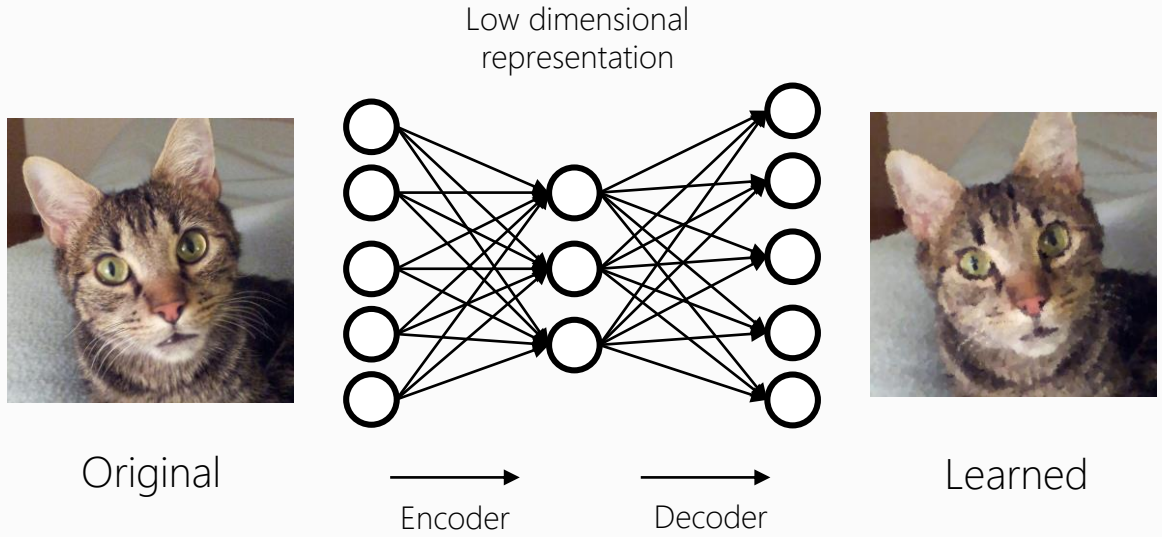
FORECASTING

Time series forecasting with RNN (source here)



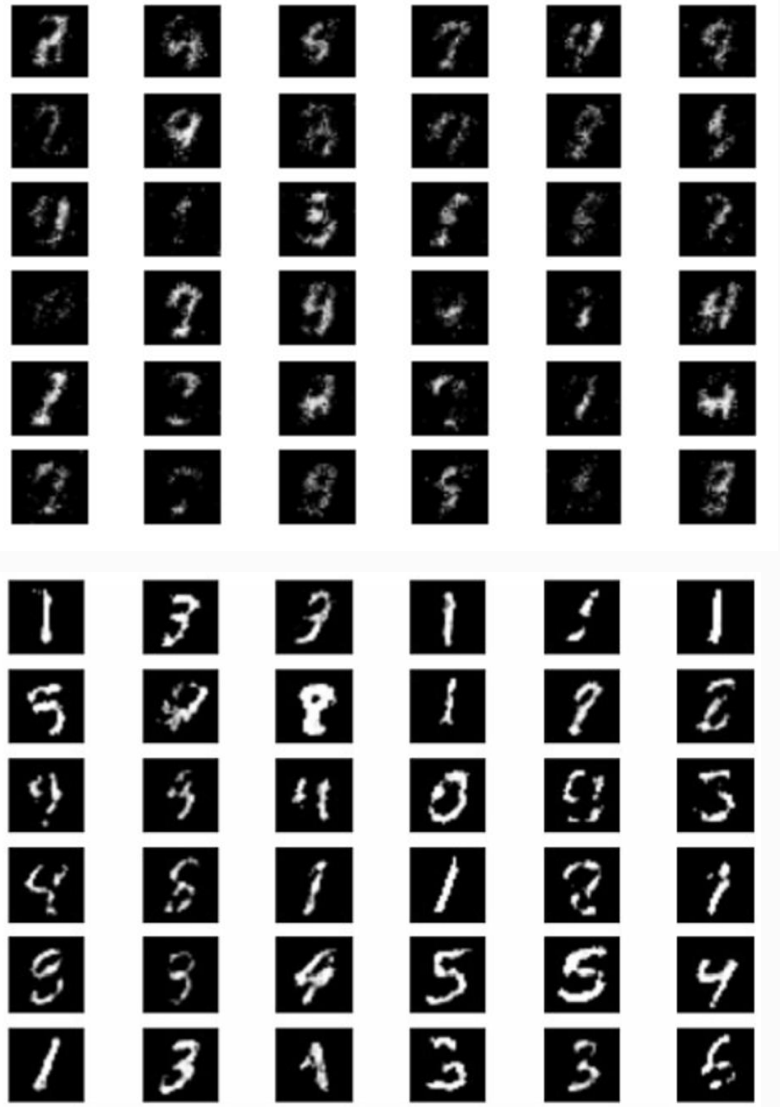
EXAMPLE: Autoencoders

Autoencoders (source here)



Example: GANs

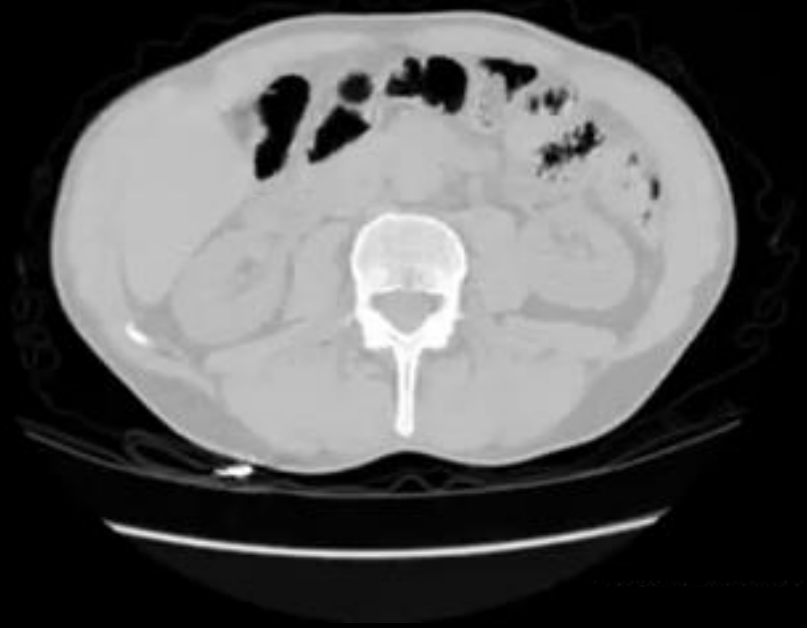
Generative Adversarial Networks (GANs) tutorial (source here)



Sector: Healthcare

Diagnosis augmentation for lung cancer detection
3D CNN for nodule segmentation +
CNN for nodule classification (info1, info2)

Patient health monitoring with sensors (IoT):
Autoencoder on normal class +
reconstruction error abnormal class (info)

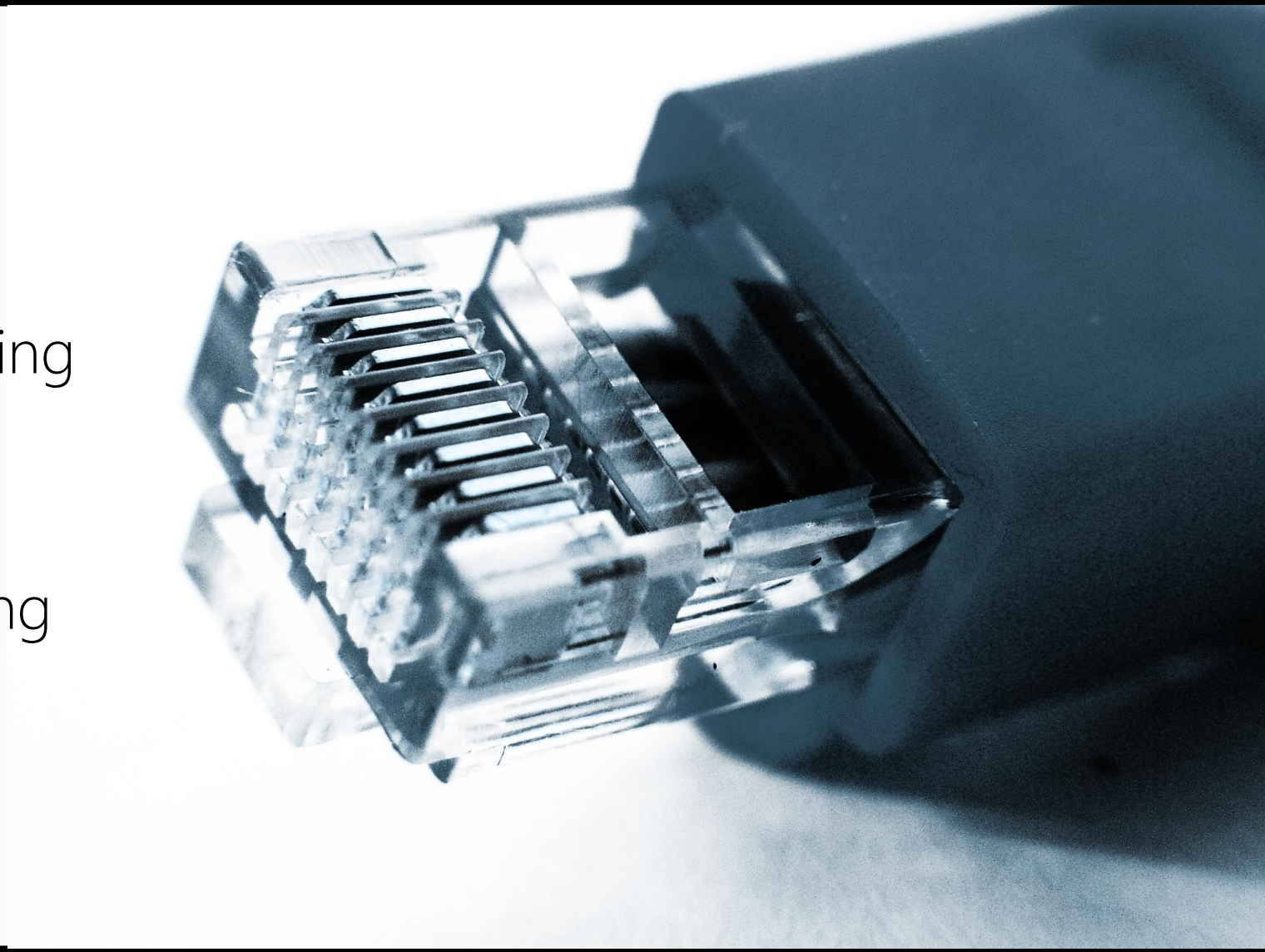


CT scan: horizontal section of lung

Sector: Communications and media

Content recommendation:
CNNs featurization + clustering

Personalized marketing:
CNN featurization + clustering

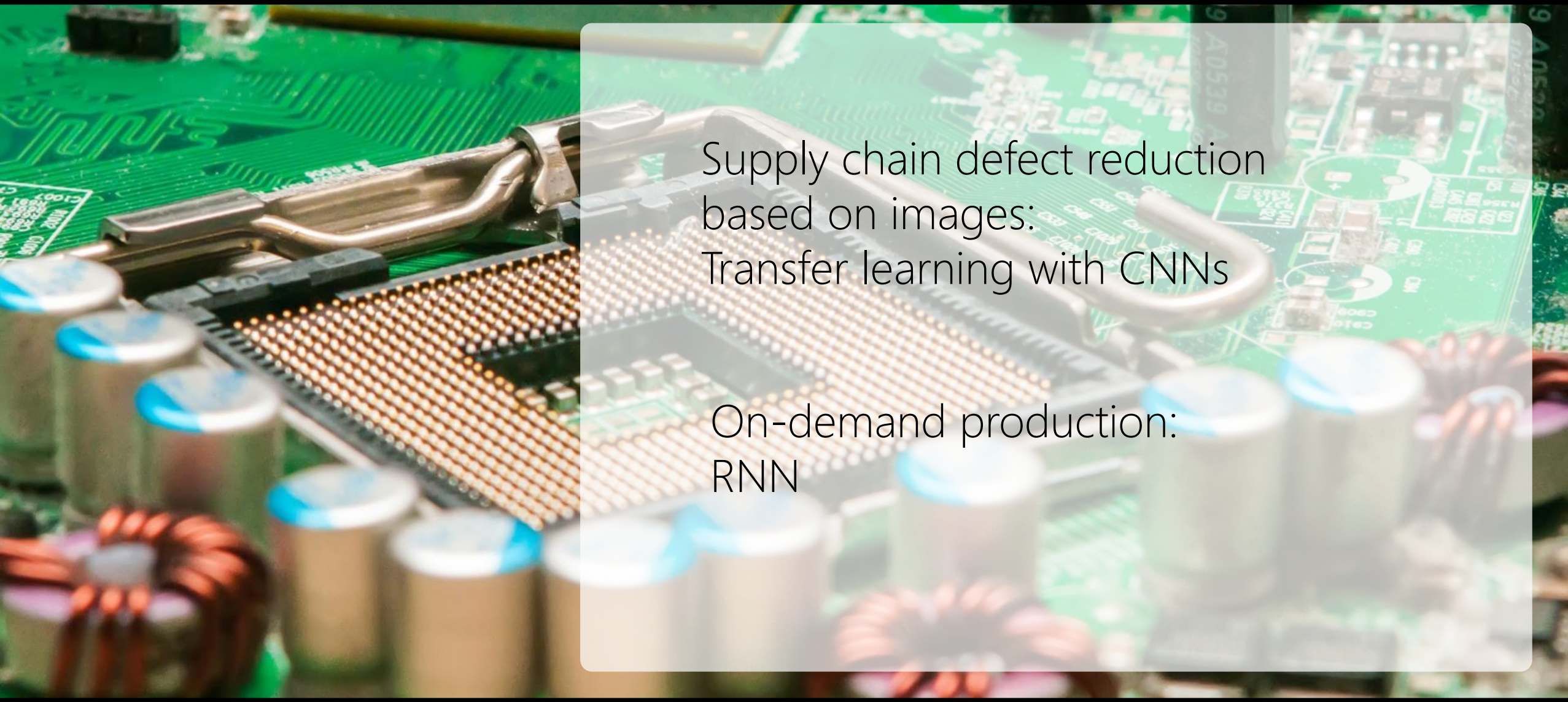


Sector: Retail

Personalized fashion design and production:
CNNs for image classification +
GANs for image generation

Customer demand forecasting:
RNN

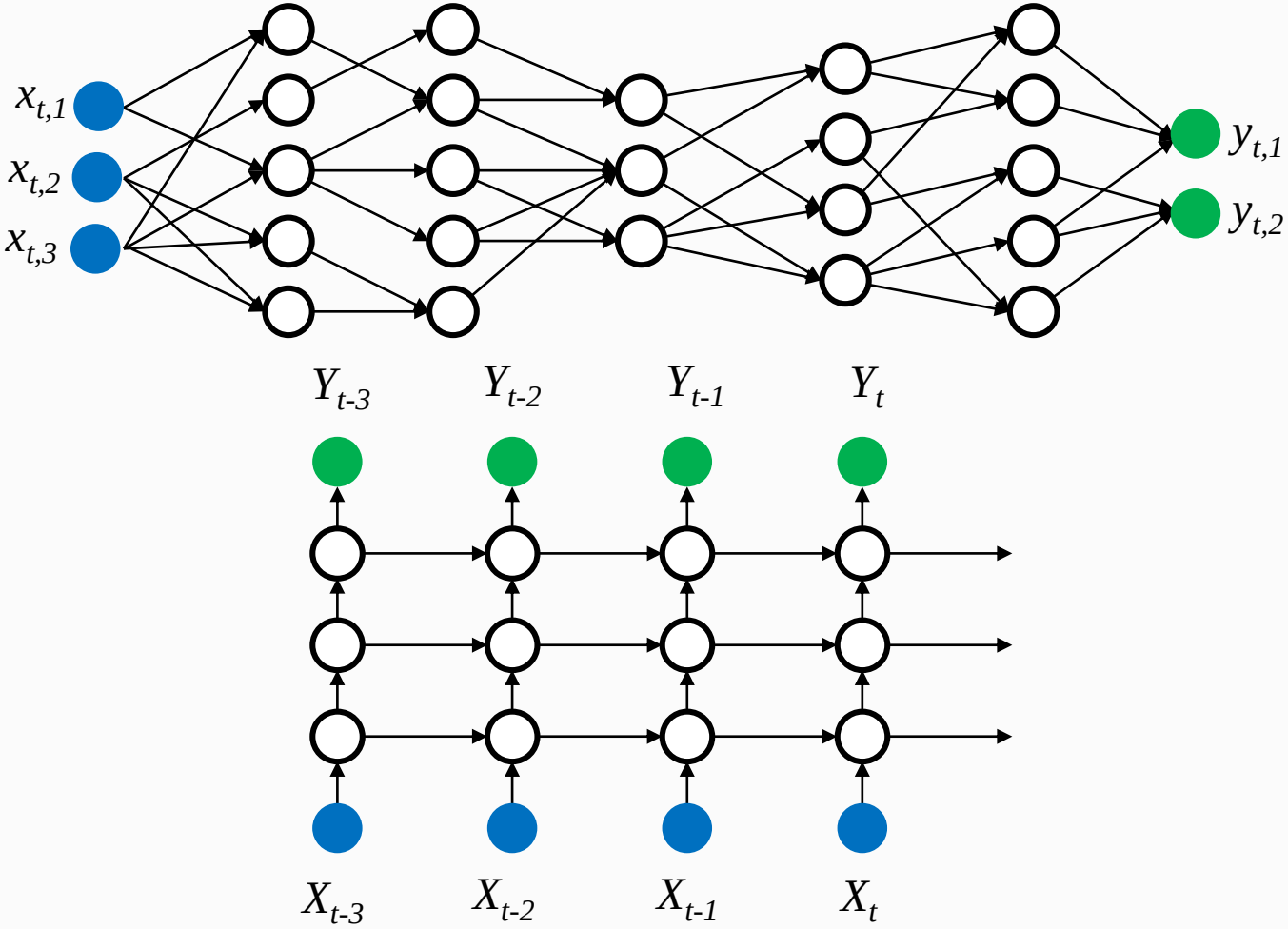
Sector: Manufacturing



Supply chain defect reduction based on images:
Transfer learning with CNNs

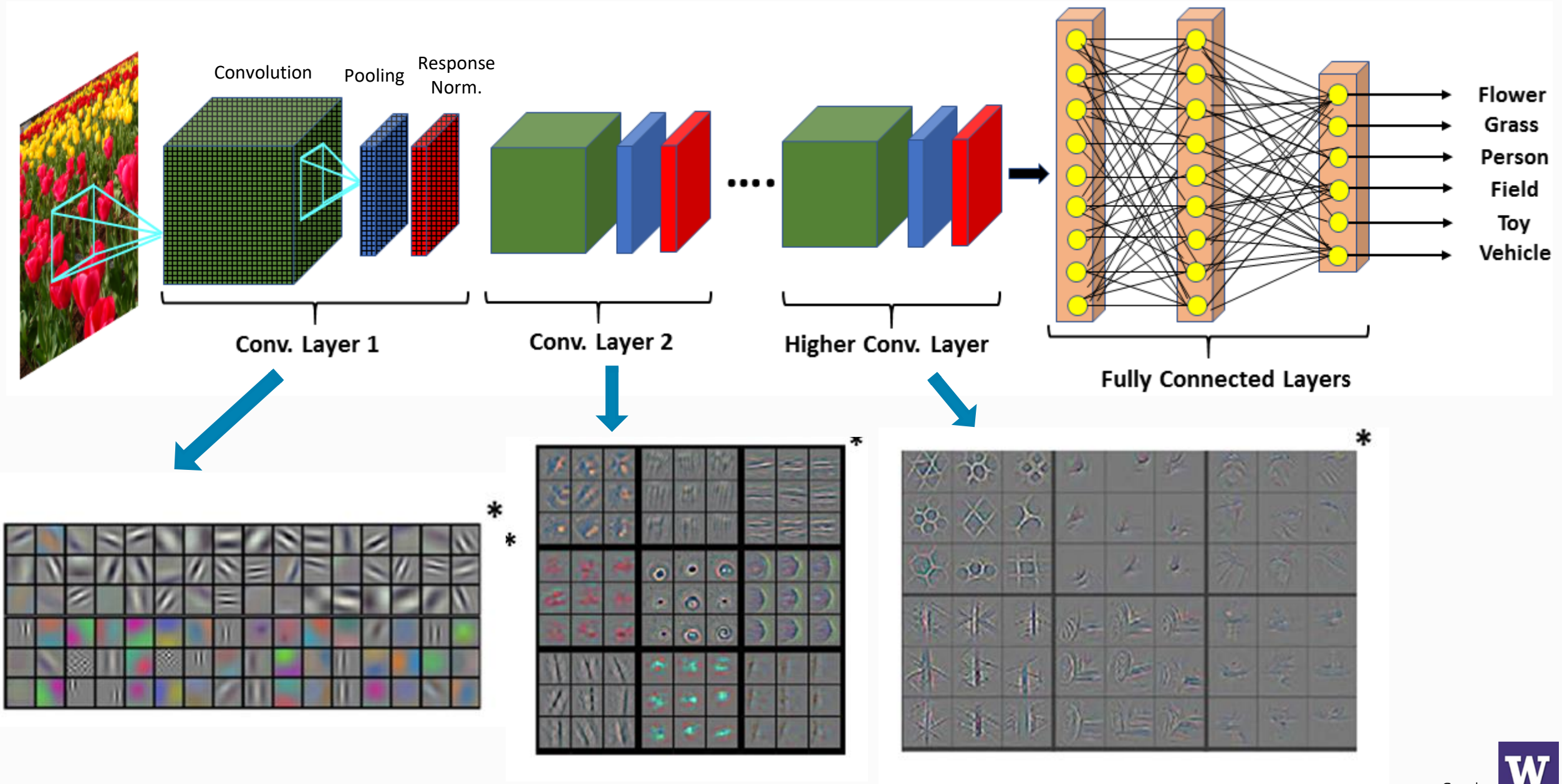
On-demand production:
RNN

Two general kinds of networks



Convolutional Neural Networks (CNN)

Recurrent Neural Networks (RNN)



** Feature visualization images from "Visualizing and Understanding Convolutional Neural Networks", Zeiler and Fergus,*

Polyglot Deep learning

TensorFlow

Deep Learning (Intro) In-class Lab

Data Science

Deriving Knowledge from Data at Scale

That's all for tonight….