

DEEP LEARNING WITH KERAS

Themistoklis Diamantopoulos

Contents

- Part 1:
 - Machine Learning with Python
 - Introduction to Deep Learning
 - Optical Character Recognition
 - Image Recognition
- Part 2
 - Text Classification/Sentiment Analysis
 - Text Generation
 - Neural Doodle/Neural Style Transfer
 - AI Game Learning

INTRODUCTION TO MACHINE LEARNING

What is Machine Learning?

- Subfield of Artificial Intelligence
- Term coined in 1959 by Arthur Samuel

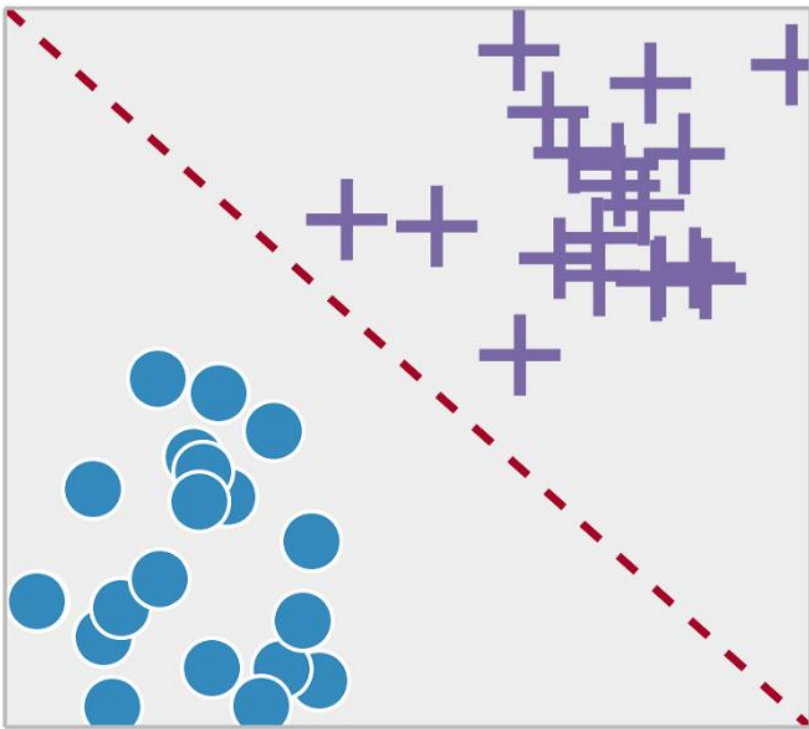
Progressively improve performance on a specific task with data, without being explicitly programmed

Types of Machine Learning tasks

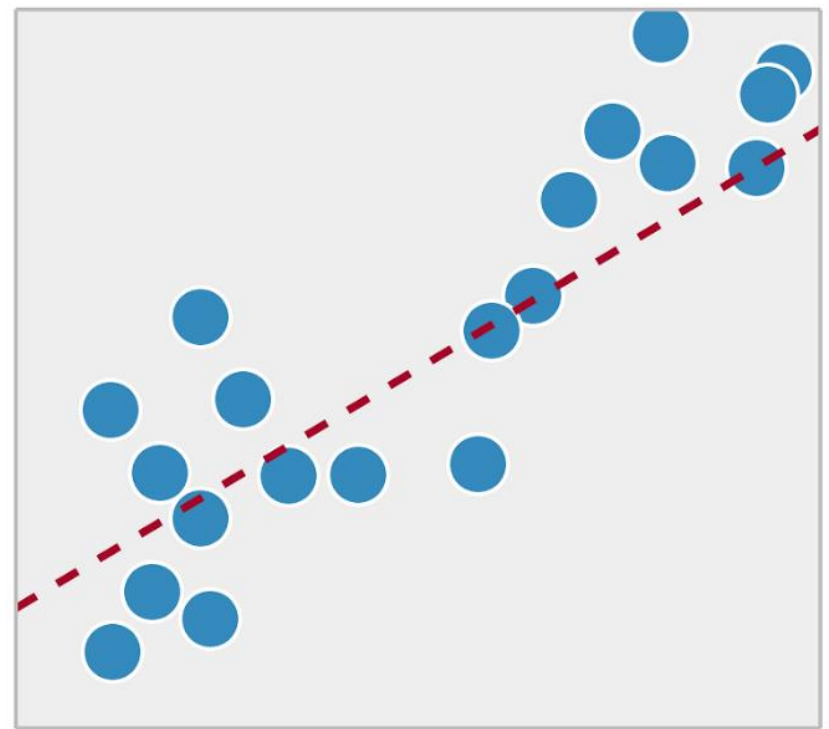
- Supervised Learning
 - Learn output based on input data
- Unsupervised Learning
 - Find structure in given data
- Reinforcement Learning
 - Learn from the environment

Supervised Learning tasks

Classification



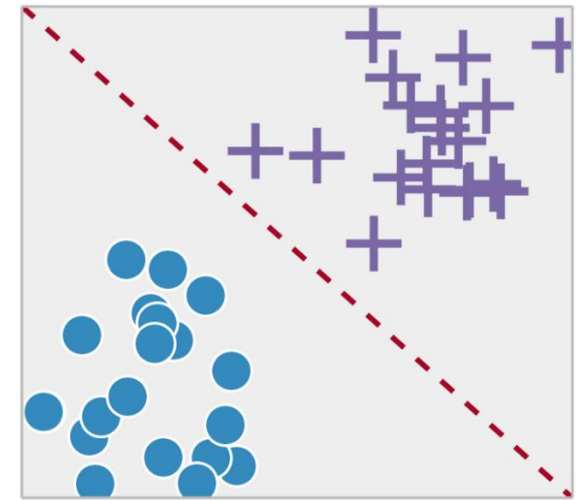
Regression



Classification

- Classify data to 1, 2 or more classes
- Confusion Matrix

| | | Actual Class | |
|-----------------|----------|---------------|---------------|
| | | Positive | Negative |
| Predicted Class | Positive | TP | FP |
| | Negative | FN | TN |
| | | $P = TP + FN$ | $N = FP + TN$ |



- Evaluation Metrics
 - Accuracy = $(TP + TN) / (P + N)$
 - Precision = $TP / (TP + FP)$
 - Recall = TP / P

Bonus: Cross Entropy

$$H(y, \hat{y}) = \sum_i y_i \log \frac{1}{\hat{y}_i} = - \sum_i y_i \log \hat{y}_i$$

Regression

- Build a model that fits the data
- Actual (y_i) and predicted values (\hat{y}_i)

- Mean Absolute Error

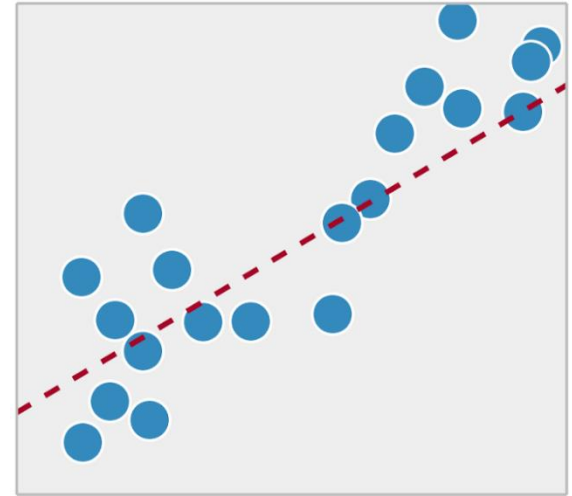
$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|$$

- Mean Squared Error

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

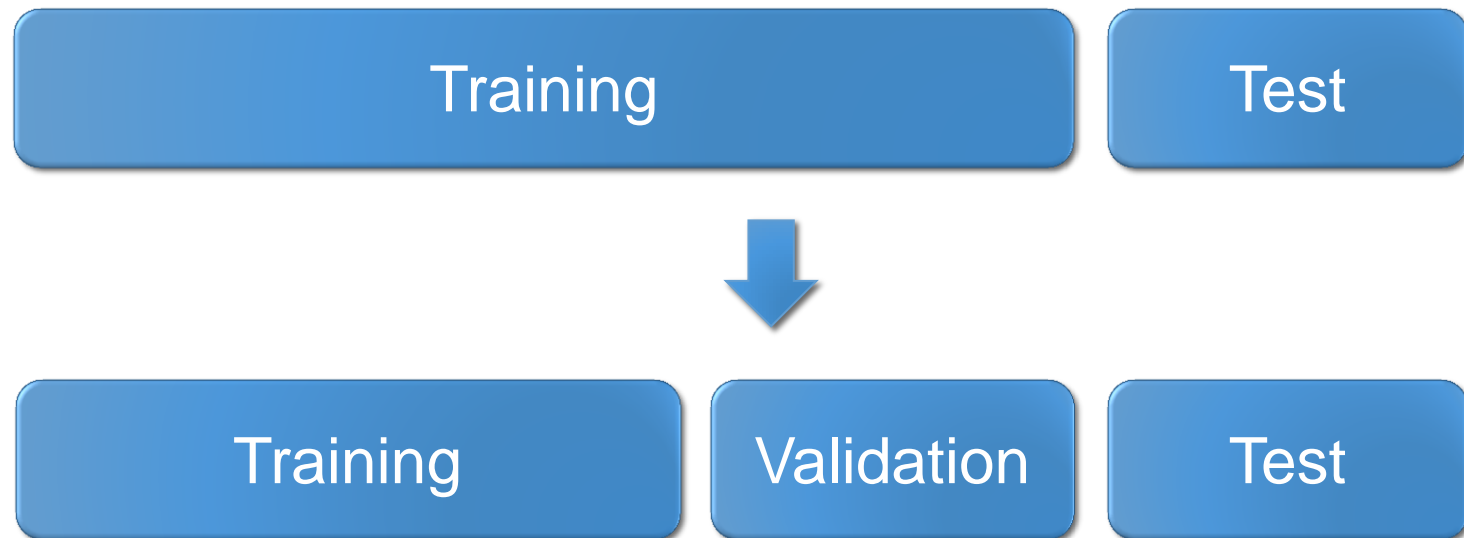
- Coefficient of Determination

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad \text{where} \quad SS_{res} = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad \text{and} \quad SS_{tot} = \sum_{i=1}^n (y_i - \bar{y})^2$$



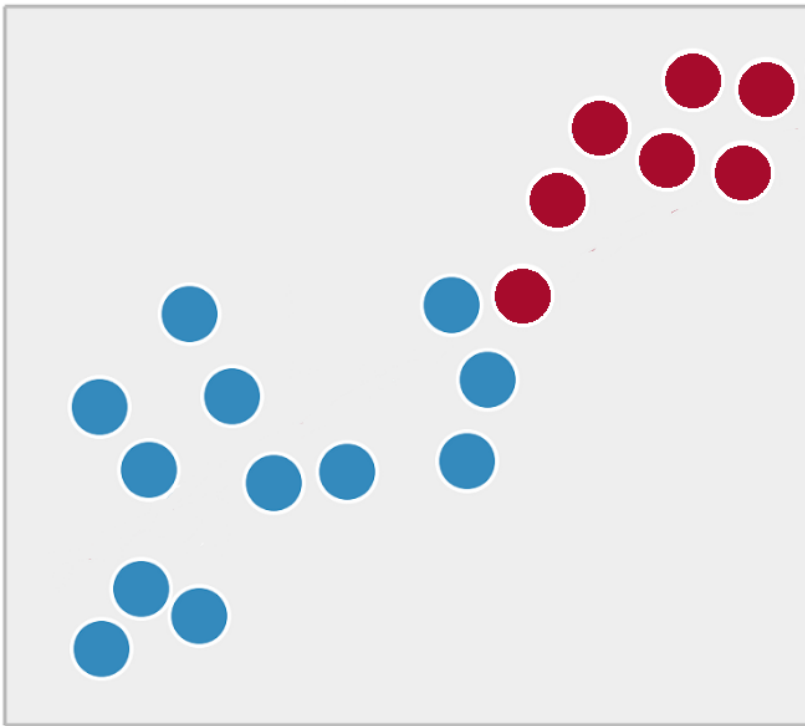
Data Splitting

- Use training data to train the model
 - Some data can be used to validate the model → validation set
 - Use folds of training data for validation → Cross-validation
- Evaluate the model on test data
 - Test set must not overlap with training data

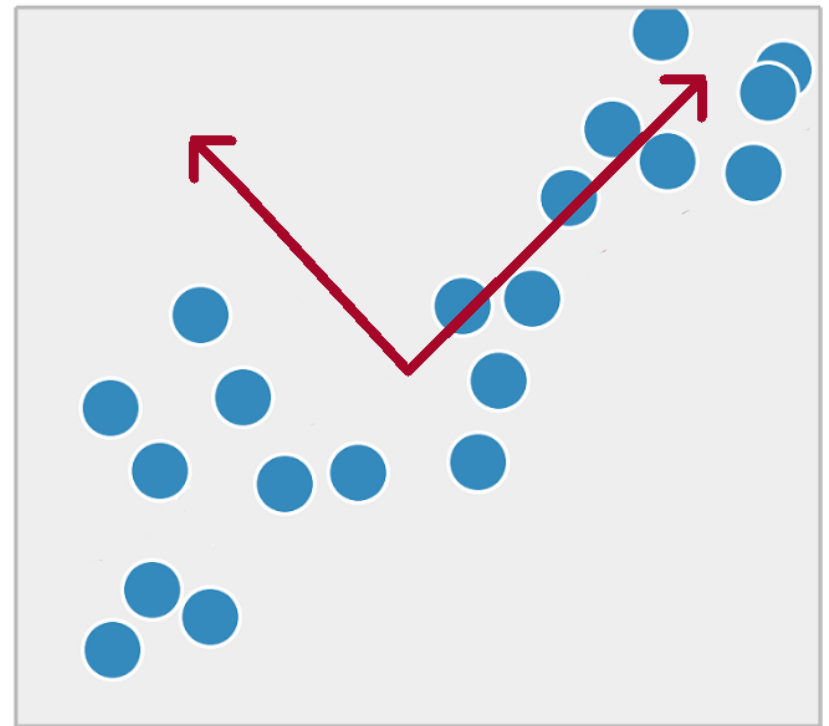


Unsupervised Learning tasks

Clustering



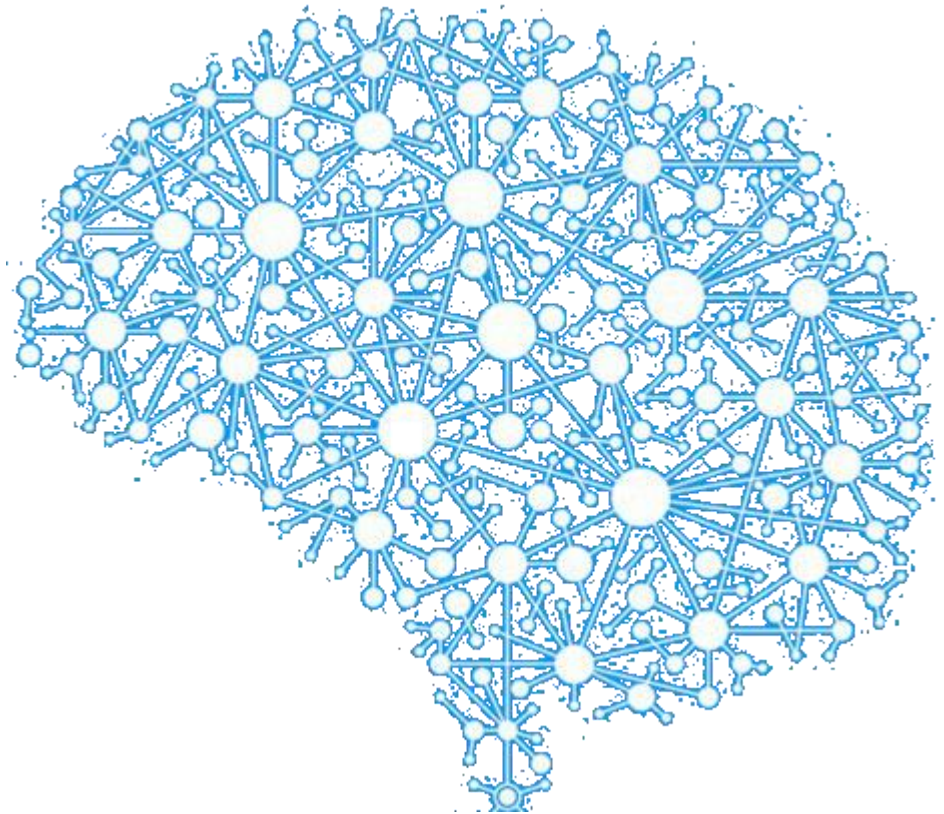
Dimensionality Reduction



INTRODUCTION TO NEURAL NETWORKS

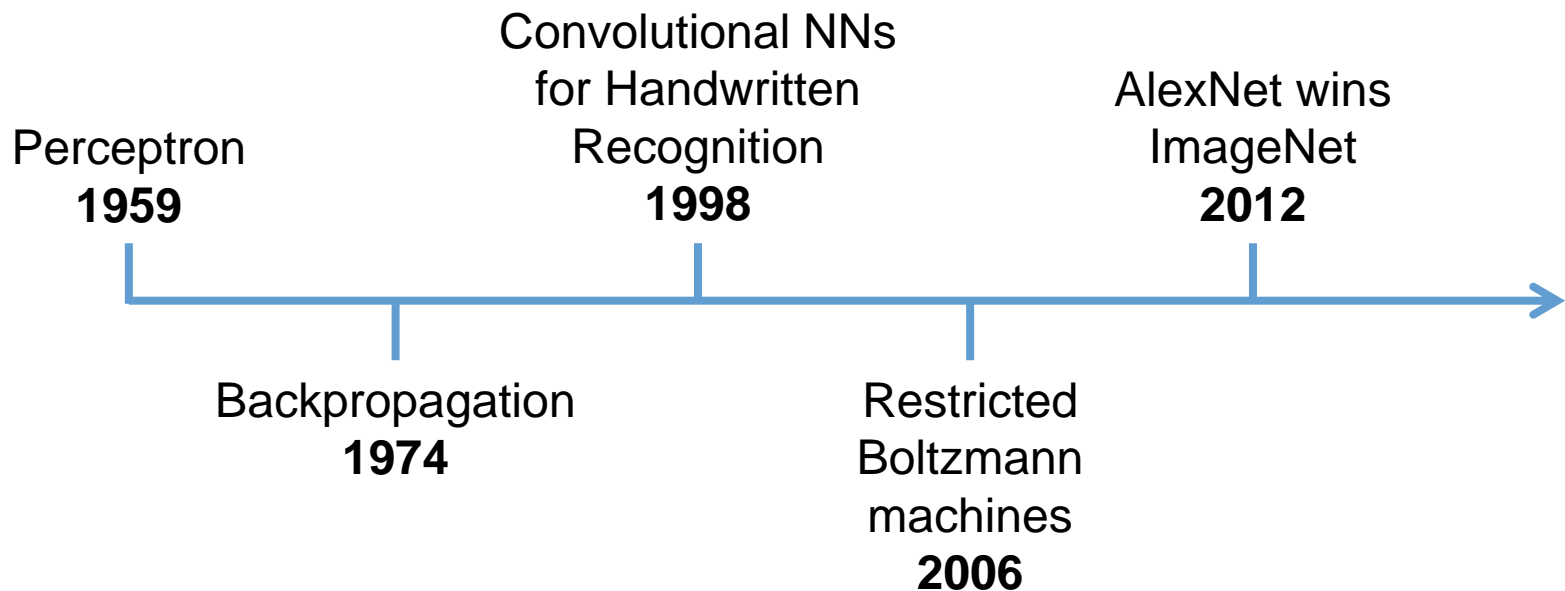
Why Neural Networks?

- Cognitive features
- Inspired by the brain
 - 10^{11} neurons
 - 0.001 sec switching time
 - $>10^4$ connections per neuron
 - 0.1 sec for scene recognition



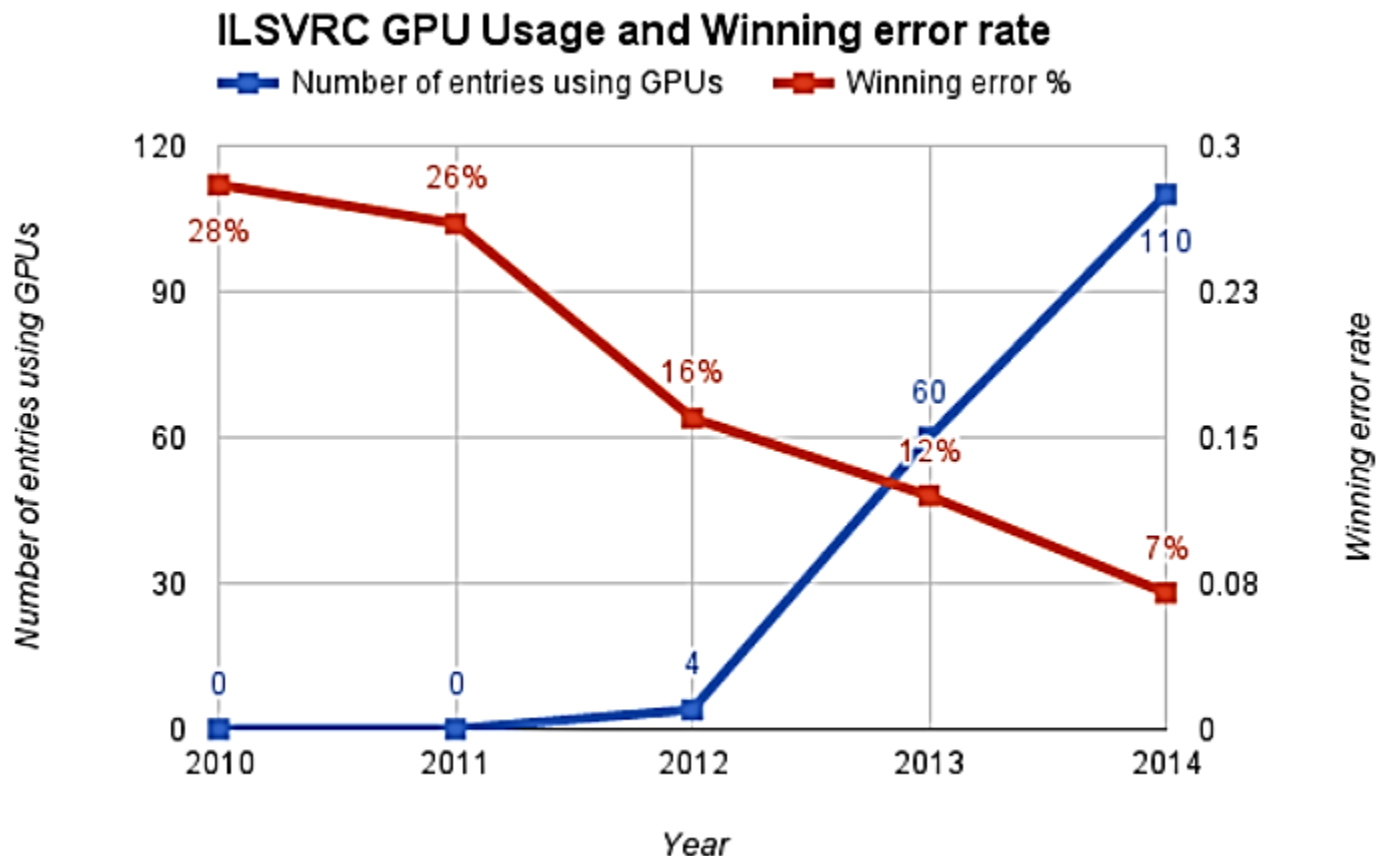
A brief history course

- From the perceptron to deep learning
- AI Winter 1969 – 1986



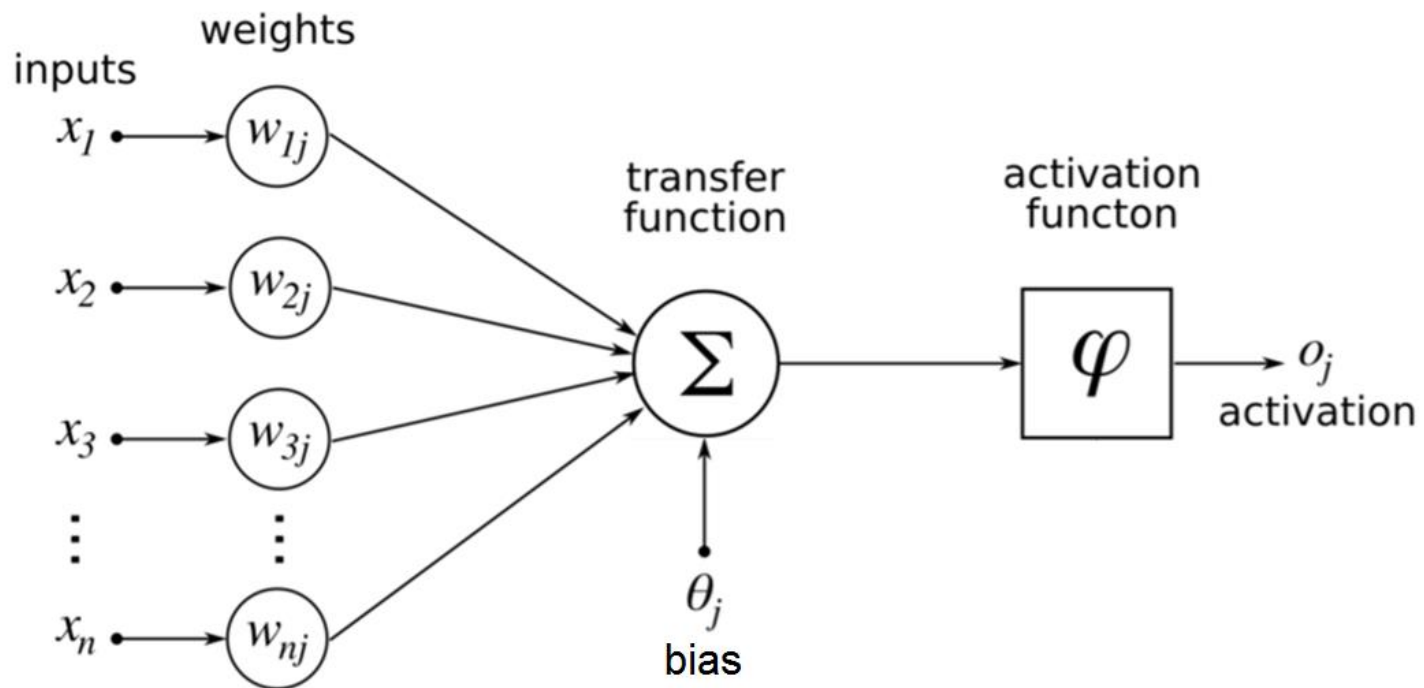
The first breakthrough

- ImageNet image recognition challenge



The perceptron - where it all started

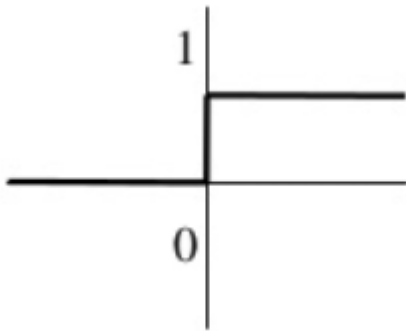
- Invented in 1959 by Frank Rosenblatt
- Practically also the first Neural Network



Activation functions

- Different types of functions

Step function



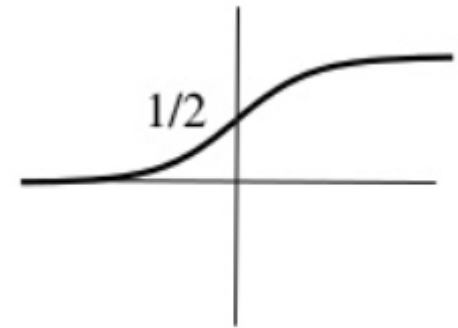
$$step_t(x) = \begin{cases} 1 & x > t \\ 0 & \text{otherwise} \end{cases}$$

Sign function



$$sign(x) = \begin{cases} +1 & x \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

Sigmoid function

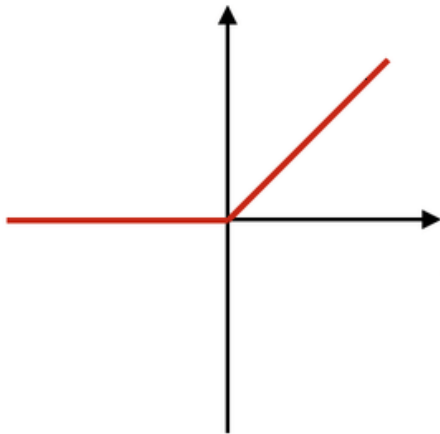


$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

Activation functions (continued)

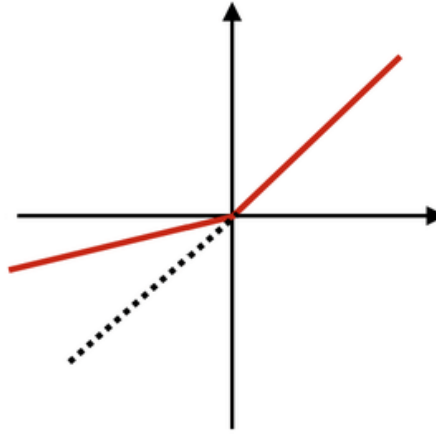
- Different types of functions

ReLU
(Rectified Linear Unit)



$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

PReLU
(Parametric ReLU)

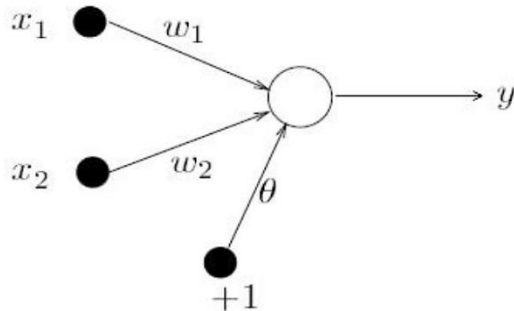


$$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

Softmax
(similar to Sigmoid -
generalized logistic function
used for multiclass
classification problems)

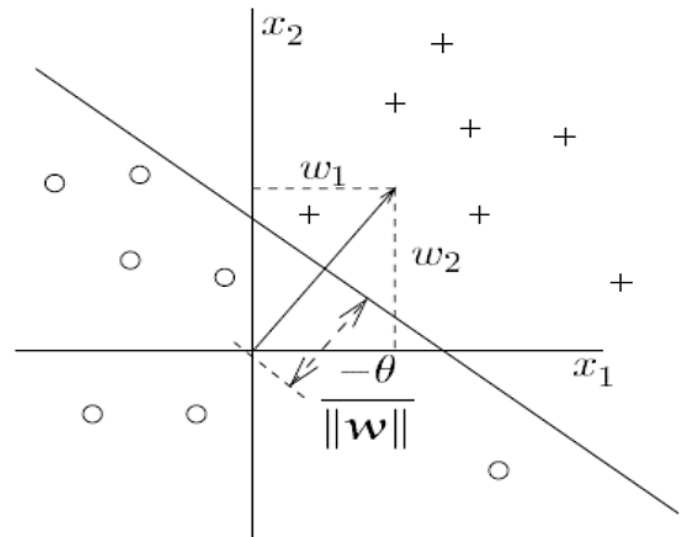
Example use of perceptron

- Bias \rightarrow Offset from the origin
- Weights \rightarrow Slope of the line



$$w_1x_1 + w_2x_2 + \theta = 0$$

$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{\theta}{w_2}$$



$$y = \text{sgn} \left(\sum_{i=1}^2 w_i x_i + \theta \right)$$

$$\text{sgn}(s) = \begin{cases} 1 & \text{if } s > 0 \\ -1 & \text{otherwise.} \end{cases}$$

REPEAT {

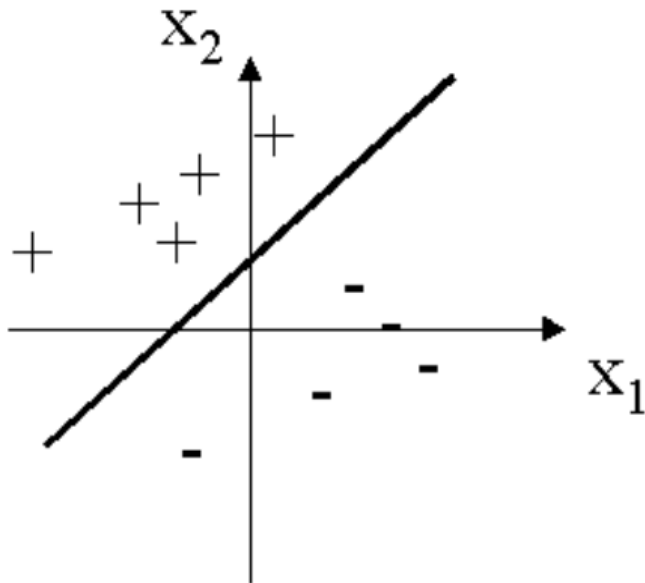
1. Select random sample from training set
2. If classification is correct, do nothing
3. If classification is incorrect, modify w :

$$w_i = w_i + \eta d(n) x_i(n)$$

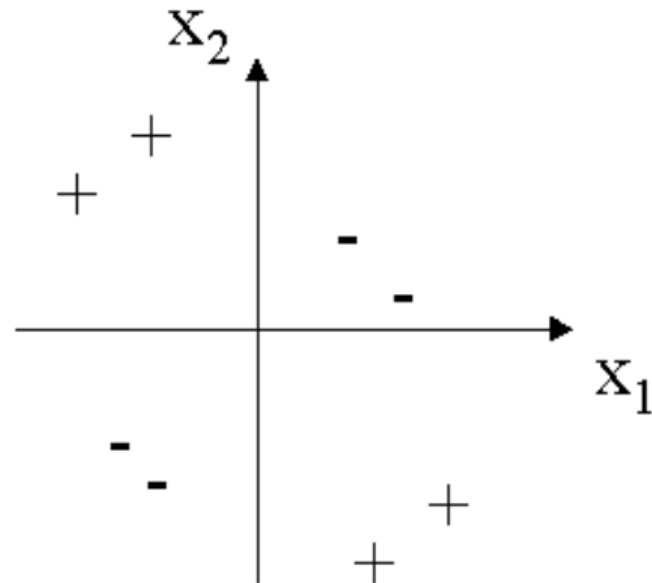
Limitations of perceptron

- Can be used only for Linearly Separable Data

Linearly Separable



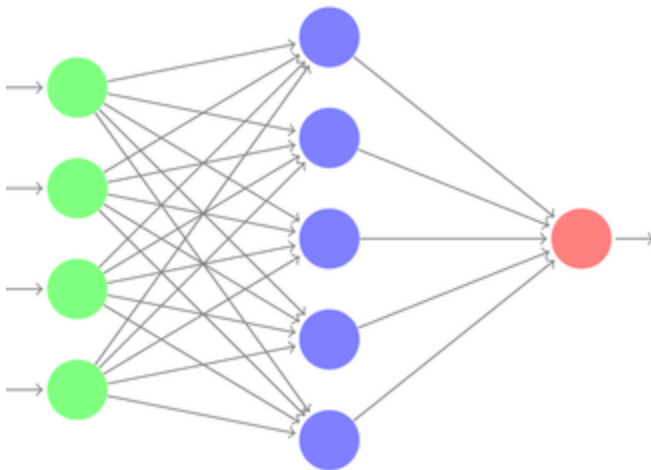
Not Linearly Separable



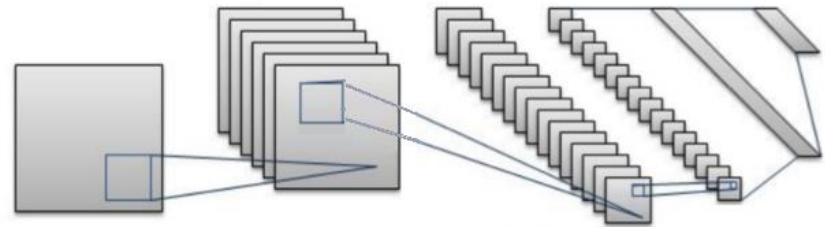
Neural Network Topologies

- Used for different types of problems

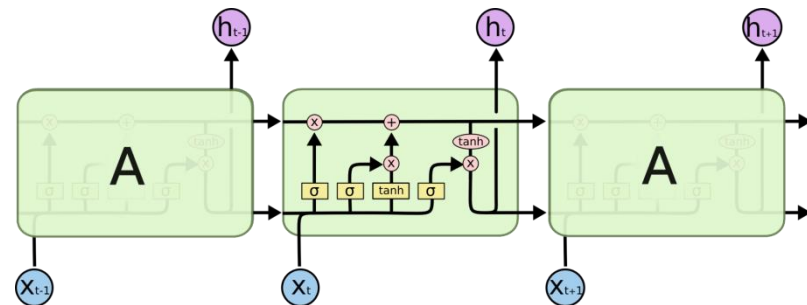
Multi-Layer Perceptron



Convolutional Neural Network



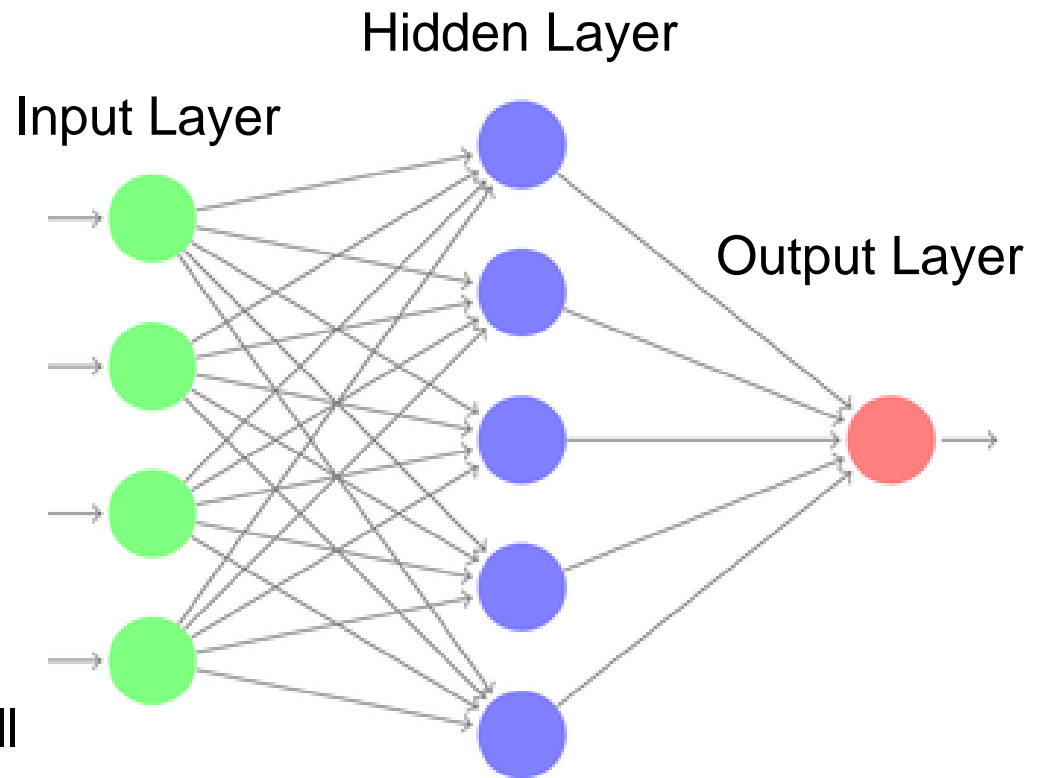
Recurrent Neural Network



MULTI-LAYER PERCEPTRON

Training a Multi-Layer Perceptron

- Gradient Descent
 - Start with some initial parameters θ
 - Update them using:
$$\theta \leftarrow \theta - \eta \cdot \nabla_{\theta} E(x, \theta, y)$$
where:
 - η : learning rate
 - $E(x, \theta, y)$: error
 - Continue until error is small



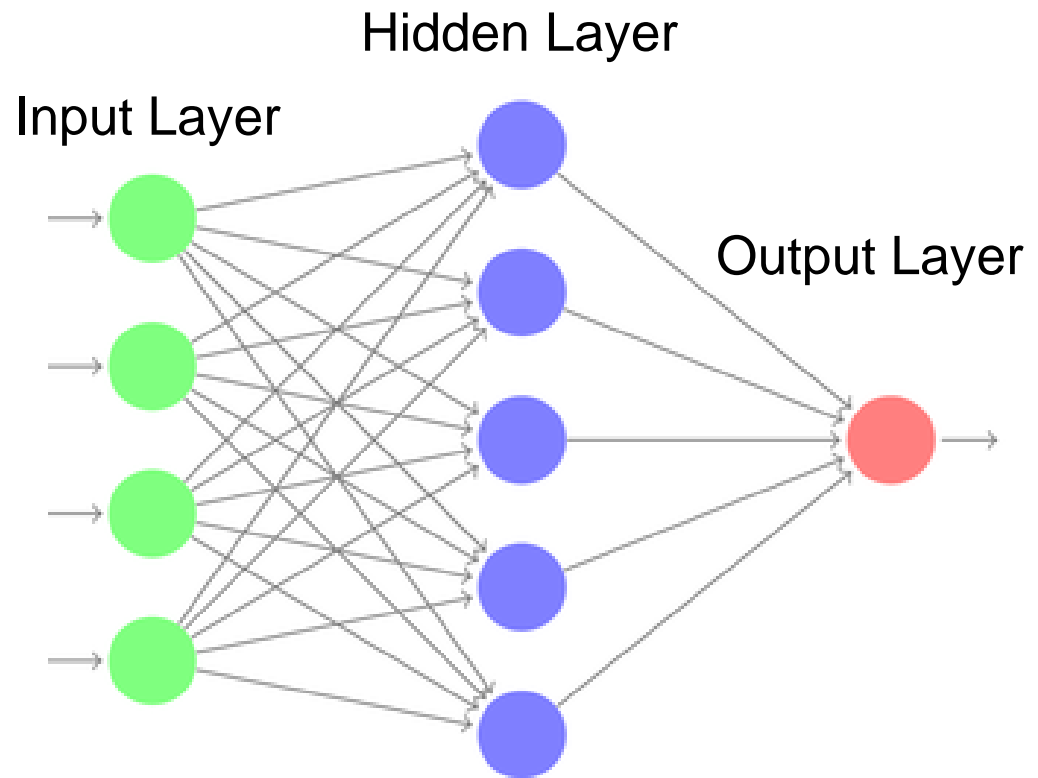
Training a Multi-Layer Perceptron

- Gradient Descent

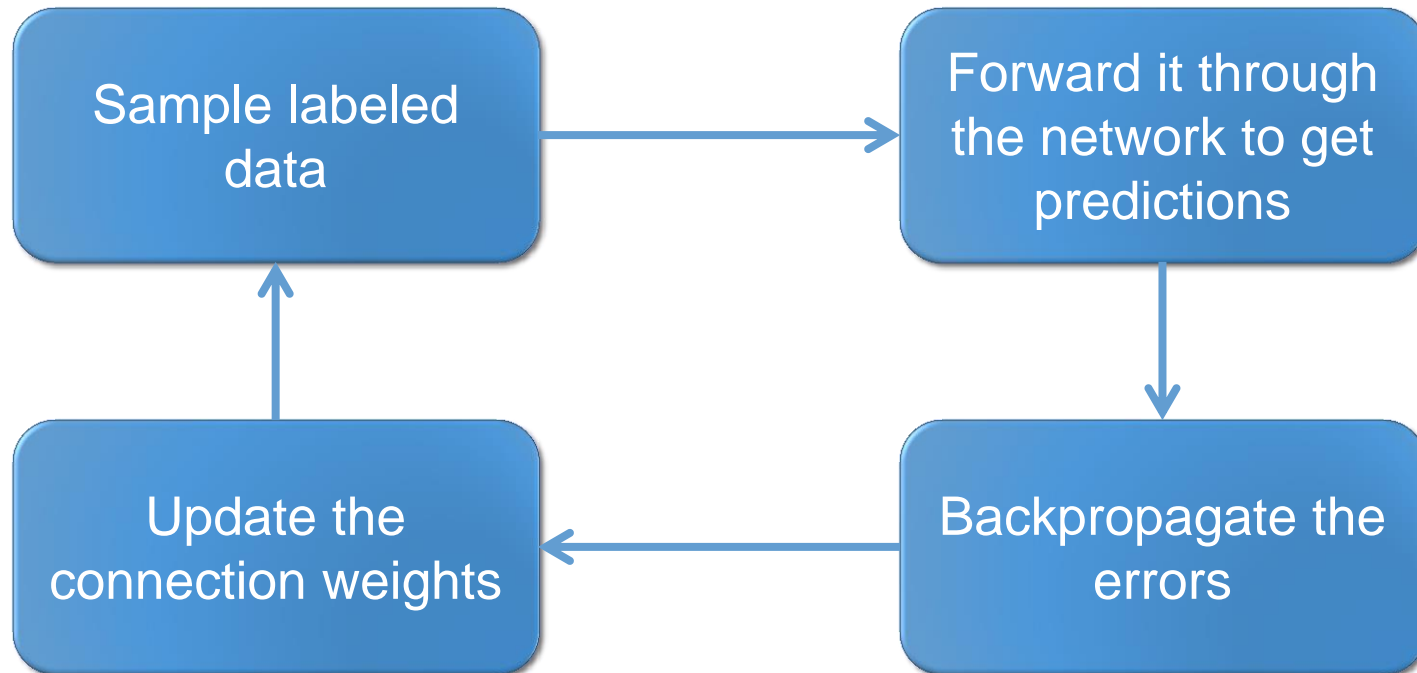
$$\theta \leftarrow \theta - \eta \cdot \nabla_{\theta} E(x, \theta, y)$$

- Backpropagation

- Easy way to compute $\nabla_{\theta} E(x, \theta, y)$



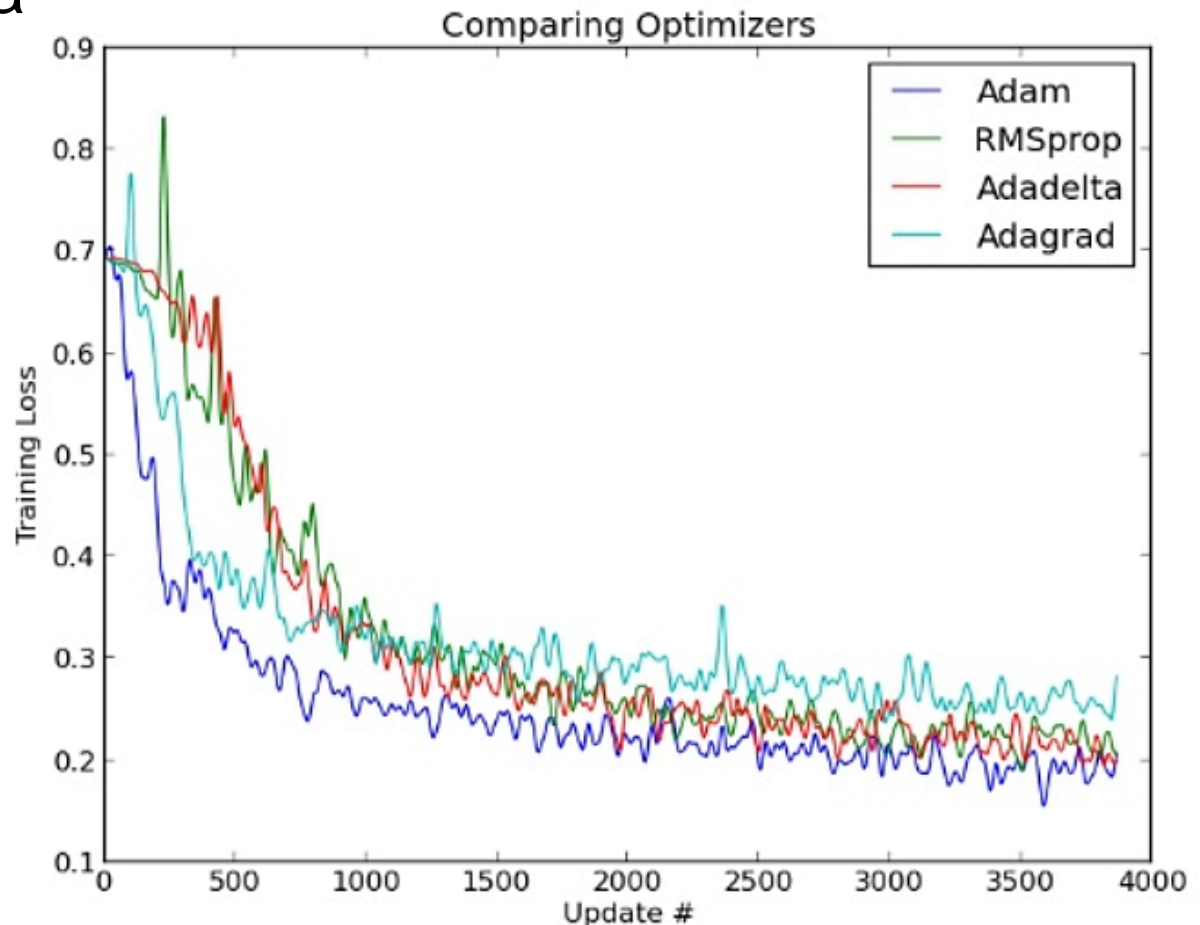
Training a Multi-Layer Perceptron



Generate an error signal that measures the difference between the predictions of the network and the desired values and then use this error signal to change the weights so that predictions get more accurate

Gradient Descent Optimizers

- RMSprop, Adadelta
 - Better later convergence
- Adagrad
 - Early optimization speed
- Adam
 - Combines the advantages of the above



INTRODUCTION TO DEEP LEARNING

What is Deep Learning?

- Subfield of Machine Learning
- Practical definition:

Imitates the workings of the human brain in processing data and creating patterns for use in decision making

What are Deep Neural Networks?

- Simple answer:

Neural Networks with many layers

- Practical answer:

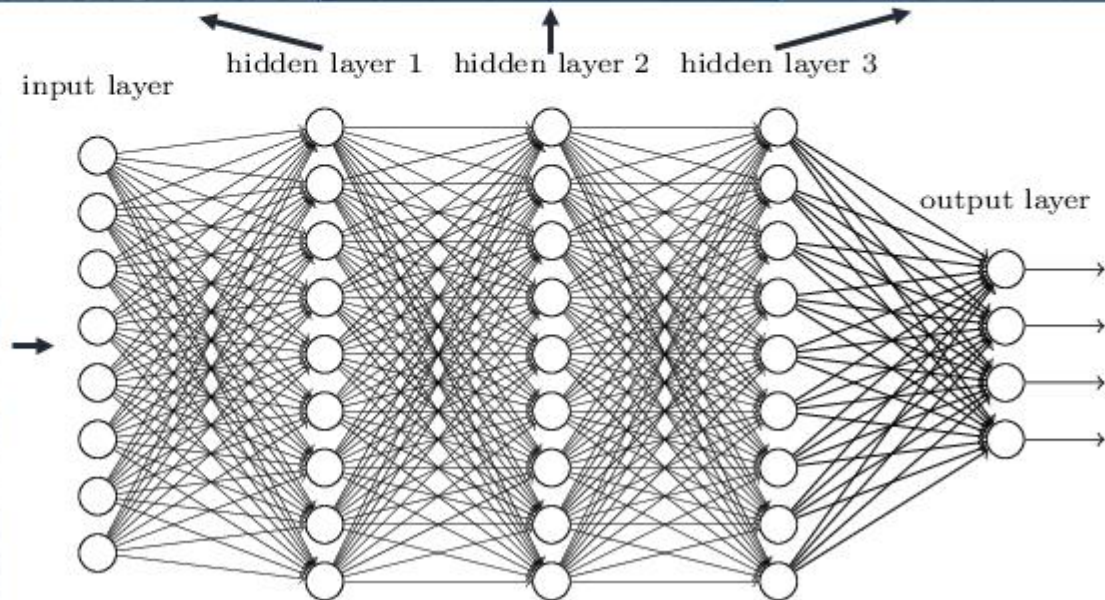
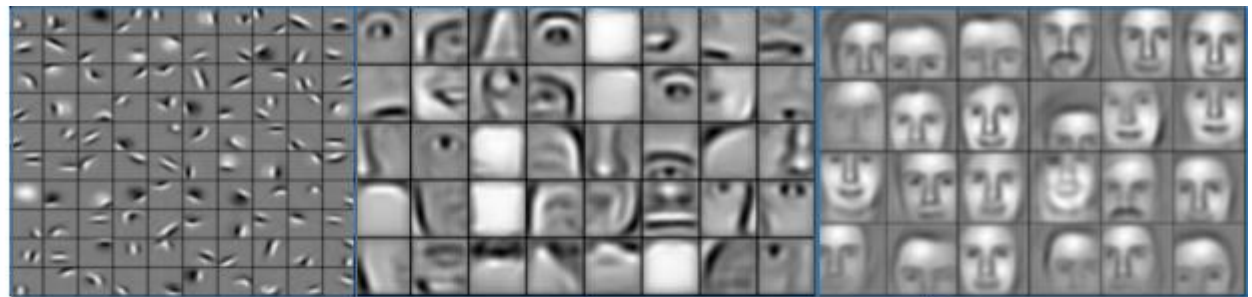
Neural Networks with more than one hidden layer

- Elaborate answer:

Neural Networks that train on a distinct set of features in each layer → Feature Hierarchy

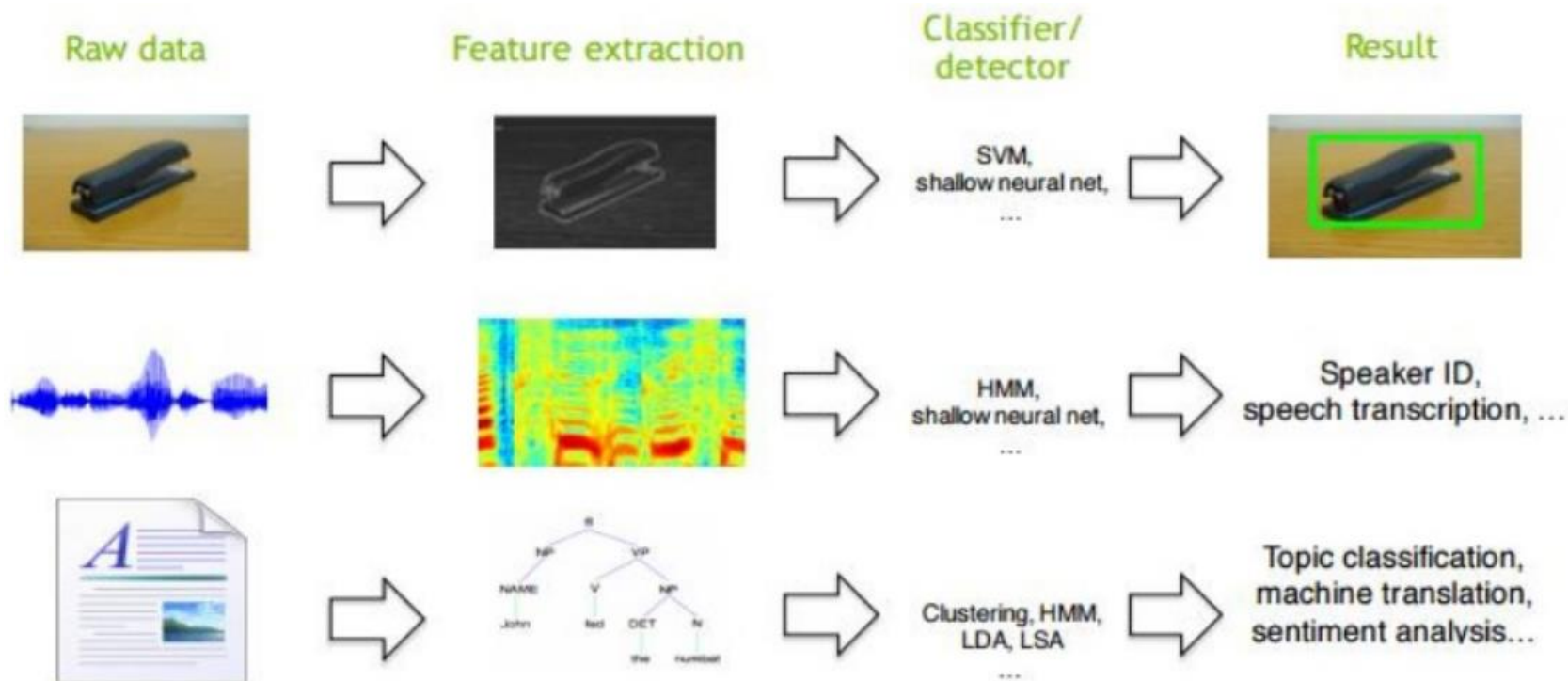
Feature Hierarchy

Deep neural networks learn hierarchical feature representations



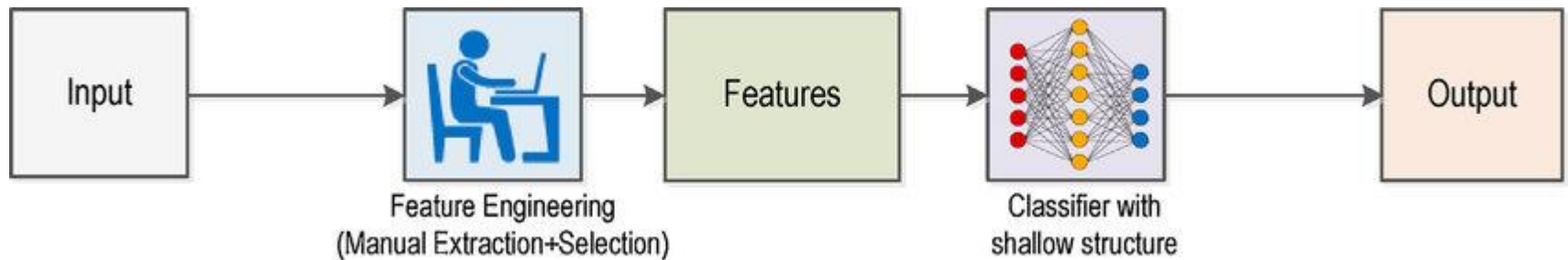
Traditional Machine Perception

- Hand-crafted Feature Extraction

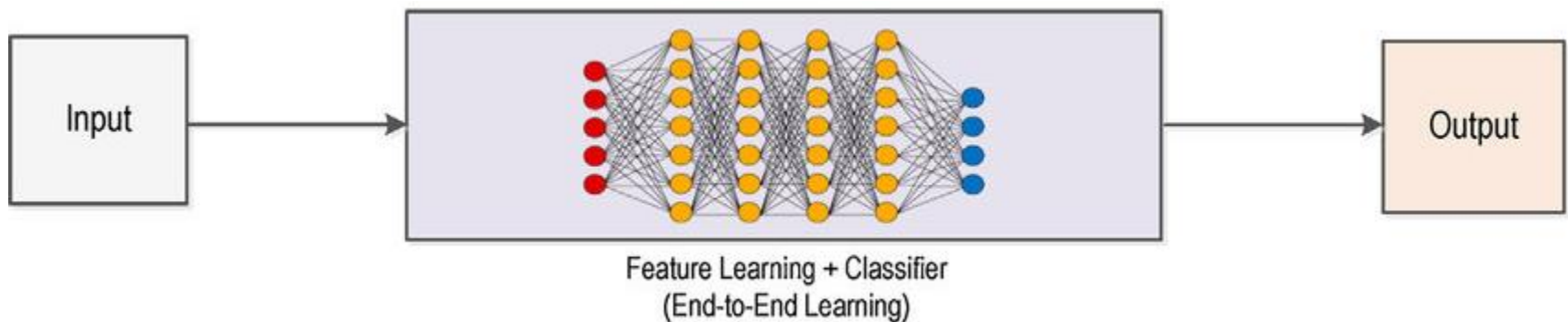


Traditional vs Deep Learning

Traditional Learning

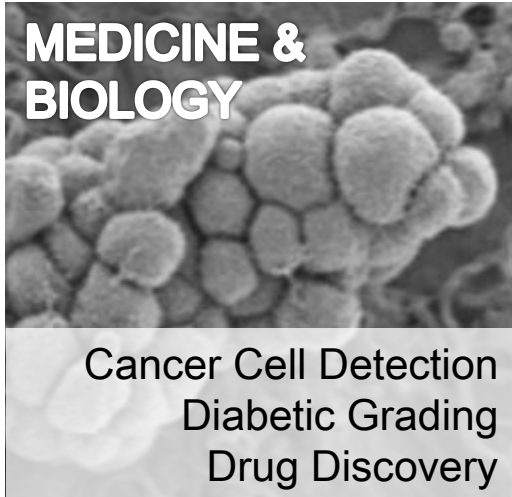


Deep Learning



Deep Learning Applications

MEDICINE & BIOLOGY



Cancer Cell Detection
Diabetic Grading
Drug Discovery

INTERNET & CLOUD



Image Classification
Speech Recognition
Language Translation

MEDIA & ENTERTAINMENT



Video Captioning
Video Search
Real Time Translation

SECURITY & DEFENSE



Face Detection
Video Surveillance
Satellite Imagery

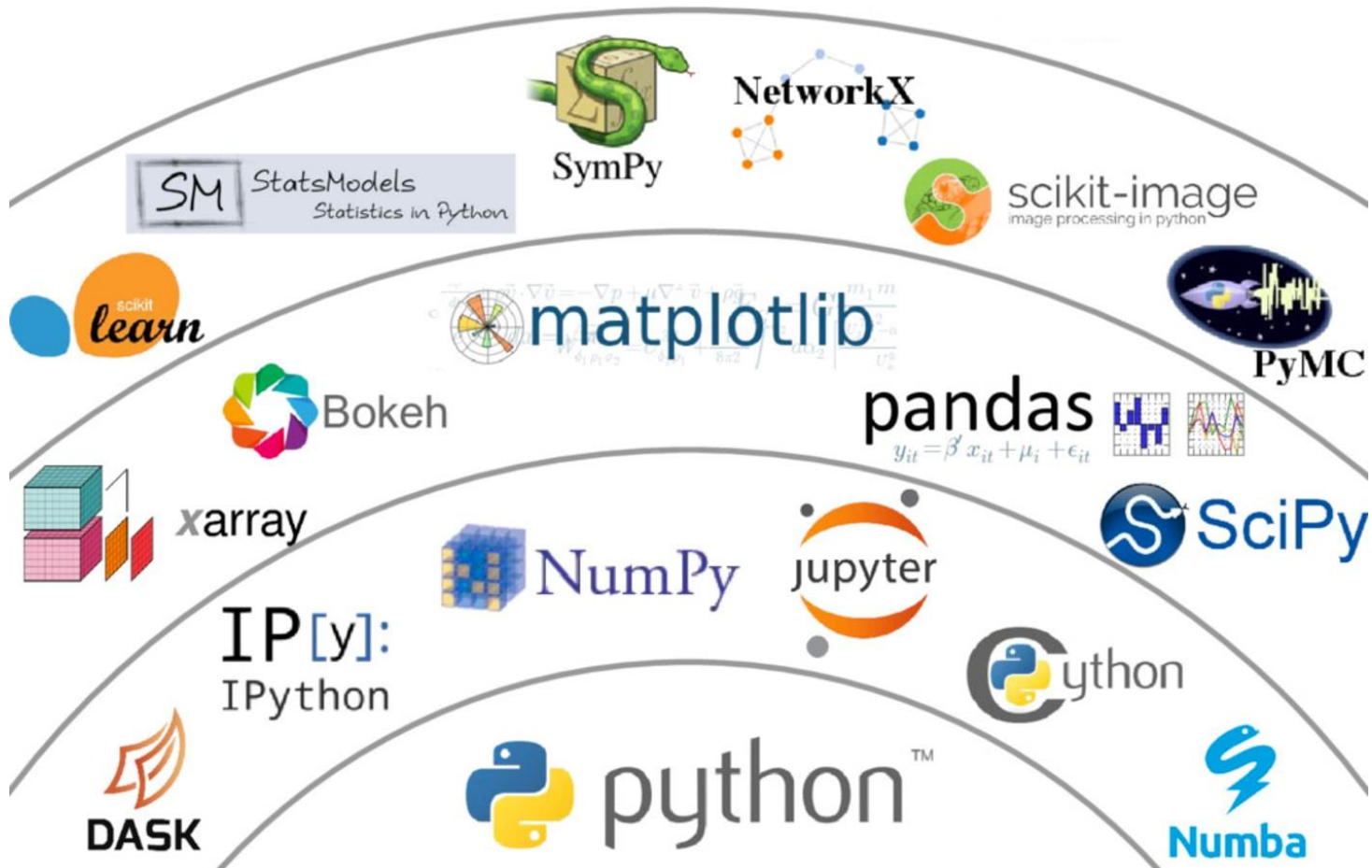
AUTONOMOUS MACHINES



Pedestrian Detection
Lane Tracking
Recognize Traffic Sign

DEEP LEARNING WITH PYTHON

Set of Powerful Libraries



Machine Learning Libraries

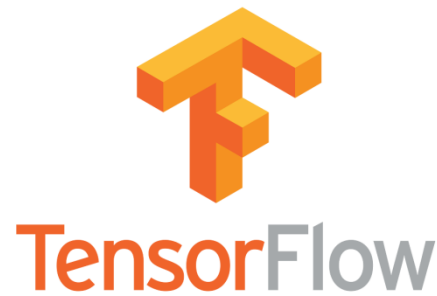
- numpy
 - Arrays: universal point of reference in the python ML world
- pandas
 - Data manipulation made easy
- scipy
 - Basis of scientific computing
- scikit-learn
 - (Almost) all machine learning algorithms you will ever need
- matplotlib
 - Plot all of the above

... and all of these are seamlessly connected!

Deep Learning with Python

- Multiple options
- All equivalent but all different
- Hard to port solutions

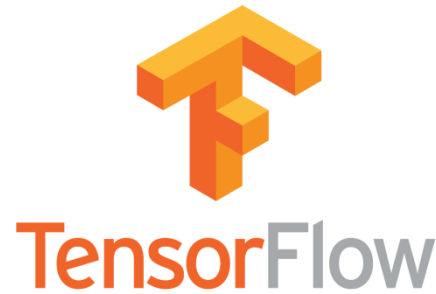
theano



Deep Learning with Keras

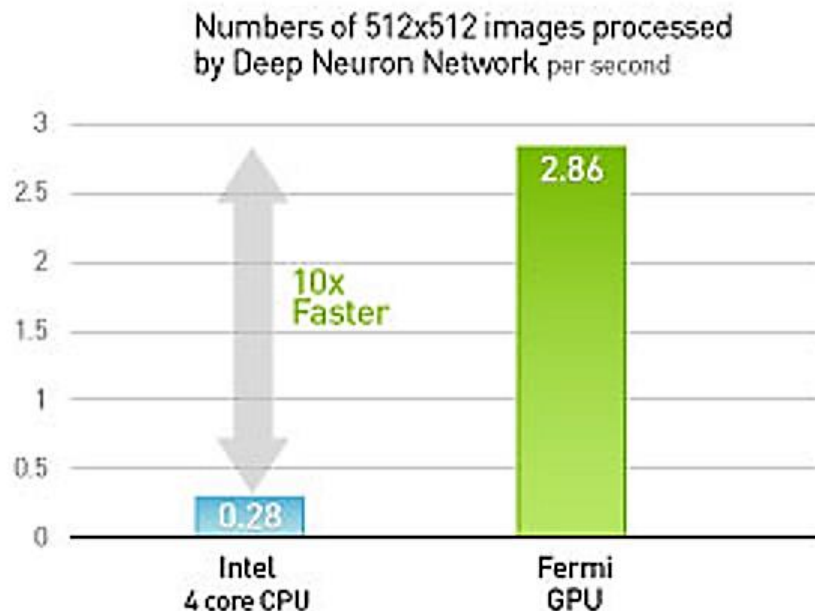
- One framework to rule them all
- Easier to code and read
- Can harness CPU and GPU

theano



Keras Requirements

- An up-to-date python distribution
- The python numpy-scipy-scikit stack
- A fast CPU or GPU



If possible,
use a GPU!

... although your CPU will
do for simple applications!

Neural Networks with Keras

```
model = Sequential()  
model.add(Dense(units=64, activation='relu', input_dim=100))  
model.add(Dense(units=10, activation='softmax'))
```

} Create Model

```
model.compile(loss='categorical_crossentropy',  
              optimizer='sgd', metrics=['accuracy'])
```

} Configure Learning

```
model.fit(x_train, y_train, epochs=5, batch_size=32)
```

} Train the model

```
metrics = model.evaluate(x_test, y_test, batch_size=128)
```

} Get metrics

```
classes = model.predict(x_test, batch_size=128)
```

} Make predictions

Time for hands-on!