Introduction to Machine Learning & Deep Learning

PEAT

15/03/2017

Outline

I. Machine Learning

II. Deep Learning & ConvNets

Outline

I. Machine Learning

a. Context

b. Capacity / Underfitting / Overfitting

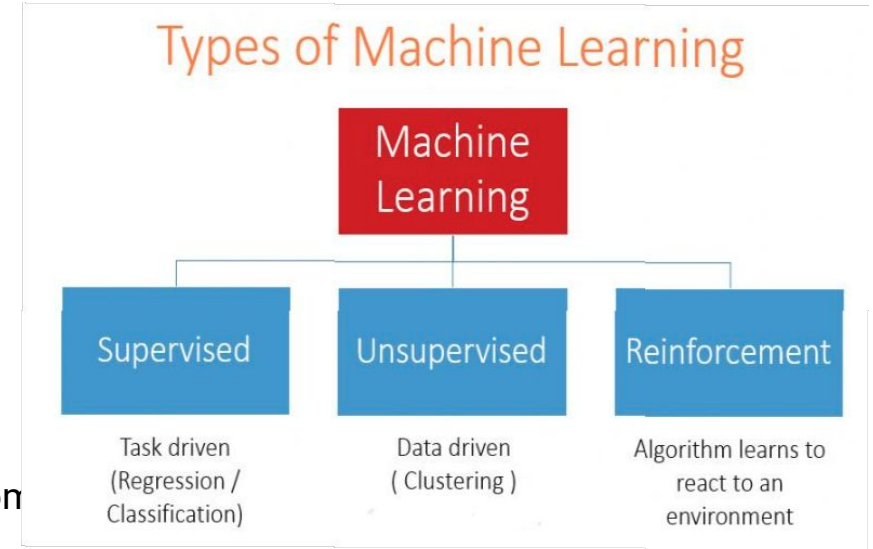
c. Choice of model

d. Hyperparameters / Validation Set

II. Deep Learning & ConvNets

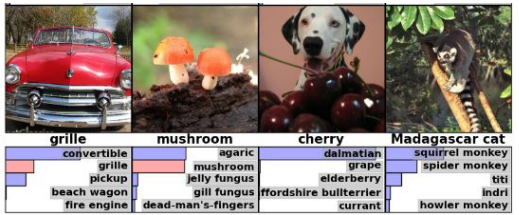
What is Machine Learning ?

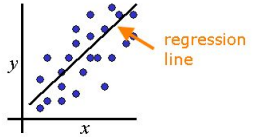
- **Machine Learning** = algorithms able to automatically learn from data or experience
- **Supervised Learning**
  - Data is labeled
- **Unsupervised Learning**
  - Data is not labeled
- **Reinforcement Learning**
  - AI strategy refined by interactions/rewards from an environment

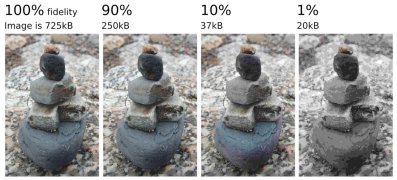


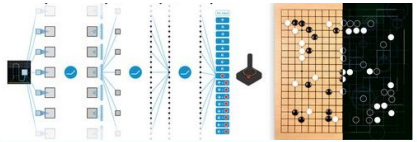
Examples of tasks

- **Classification**
  - From input data **X**, predict probability $\mathbf{p_k}$ for each class (PEAT example: **X**=image, **k**=1 for healthy, **k**=2 for aphid, **k**=3 for anthracnose, ...)
- **Regression**
  - From input data **X,** predict a value **Y** (example: forecasting market prices)
- **Compression**
  - Compress signals with minimum information loss (example: jpeg)
- **Game AI**
  - Play ATARI games or GO

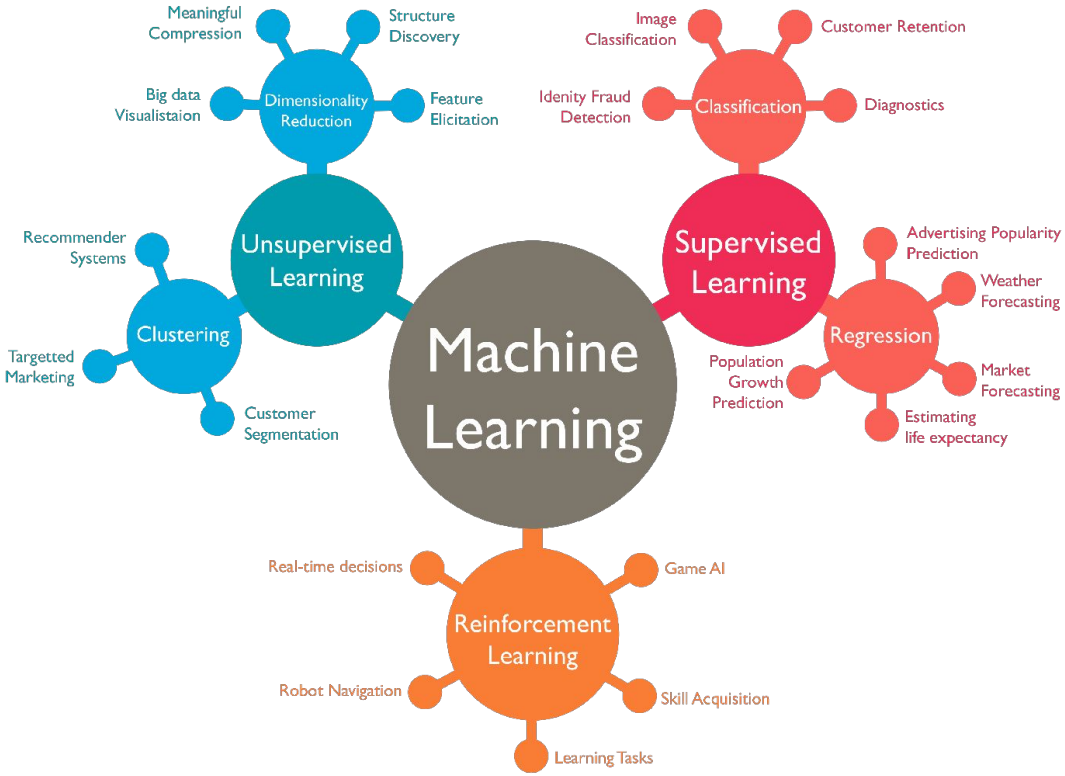








Examples of tasks



Performance Measures

- Supervised Learning: Input **X**, Target **Y**, Prediction $\hat{\mathbf{Y}}(\mathbf{X})$
- Loss function defines the cost of mistakes: $\mathbf{L}(\mathbf{Y},\hat{\mathbf{Y}})$

Classification	**Regression**
• Target **Y**: $\mathbf{Y} = (Y_1,...,Y_k,...,Y_n) = (0,...,1,...,0)$ • Prediction $\hat{\mathbf{Y}}(\mathbf{X})$ = proba of each class: $\hat{\mathbf{Y}} = (p_1,...,p_k,...,p_n)$ with $\sum_i p_i = 1$ • Top-1 accuracy: 1 if $\mathbf{p}_k$ is largest $\mathbf{p}_i$, 0 otherwise • Top-5 accuracy: 1 if $\mathbf{p}_k$ in 5 largest $\mathbf{p}_i$, 0 otherwise	Target **Y** Prediction $\hat{\mathbf{Y}}(\mathbf{X})$ Absolute Error Loss: $\mathbf{L} = \lvert\mathbf{Y} - \hat{\mathbf{Y}}\rvert$ Square Error Loss: $\mathbf{L} = (\mathbf{Y} - \hat{\mathbf{Y}})^2$

Model Training

- We are given:
  - Task **T**
  - Loss **L** for mistakes
  - Training set: $(X^1,Y^1), \dots, (X^N,Y^N)$

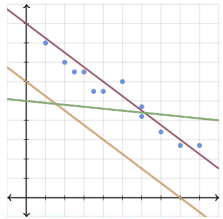
- We choose class of models (e.g. ConvNets, Decision Trees, ...):

$$\hat{Y}(X) = f_\theta(X)$$

- For a given choice of parameter θ, total loss function on training set:

$$L_{tot} = \sum_j L(Y^j,\hat{Y}^j) = \sum_j L(Y^j,f_\theta(X^j))$$

- Model training = Optimizing θ in order to minimize total loss



Outline

I. Machine Learning

a. Context

b. Capacity / Underfitting / Overfitting

c. Choice of model

d. Hyperparameters / Validation Set

II. Deep Learning & ConvNets

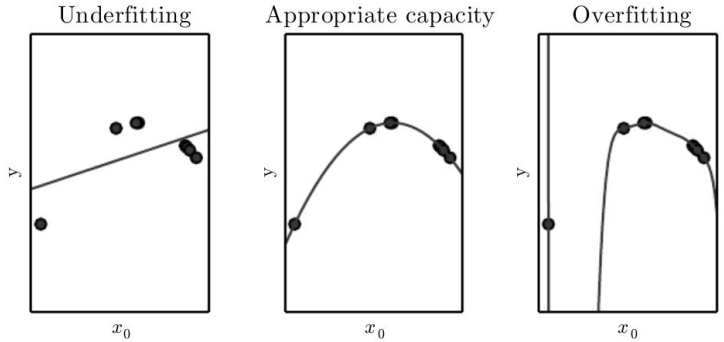
Generalization / Test Set

- **Main goal of Machine Learning:** predict new input correctly
- So far, we have simply described a **minimization of the loss** on **training data**
- However **Machine Learning ≠ pure Optimization**
  - What we care about is the loss on new input (= generalization loss)
  - Generalization loss is estimated on a separate **test set** (not seen during training)
- A Machine Learning model performs well under **2 conditions**:
  - The training loss is small
  - The gap between training loss and generalization loss is small

Underfitting / Overfitting

- Underfitting:
  - The training loss is too large
  - This happens when the model space is too constrained

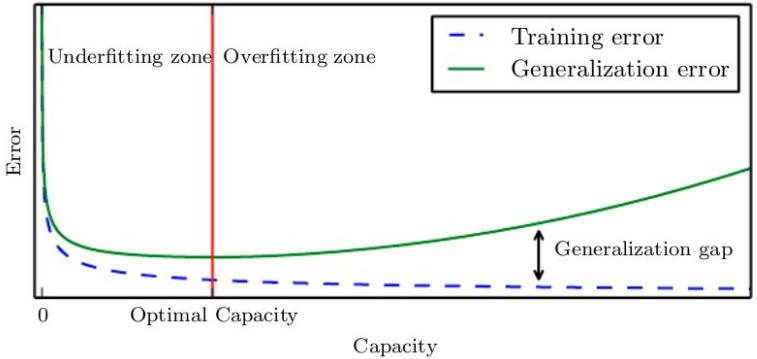
- Overfitting:
  - The gap between training loss and generalization loss is too large
  - This happens when the model space is not constrained enough



Model Capacity / Regularization

- Optimal point comes from a **balance of Underfitting** and **Overfitting**:
  - A model space sufficiently large to have small training loss
  - A model space not too large to avoid large gap between training loss and generalization loss

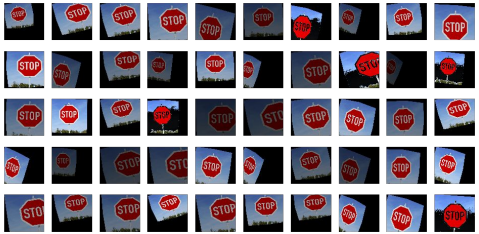
- We control these 2 effects by altering the **model capacity:**
  - It can be controlled by varying the number of parameters in the model
  - It can also be controlled with **regularization**



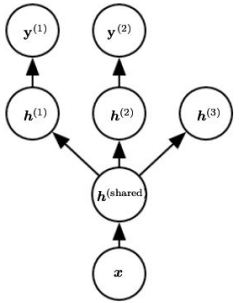
Regularization

- Can be seen as introducing a **prior knowledge**

- Dataset Augmentation
  - Create **fake data** that increase the size of the training set
  - For image classification, we can introduce random translations, rotations, change of luminosity, ...
  - Introduced prior knowledge: image class is invariant under these transformations



- Multi-task learning
  - **Share parts** of the model between different tasks
  - For image classification, we can share first layers between several image classification tasks
  - Introduced prior knowledge: some features can be shared between different tasks (edges, corners, textures, …)



Outline

I. Machine Learning

a. Context

b. Capacity / Underfitting / Overfitting

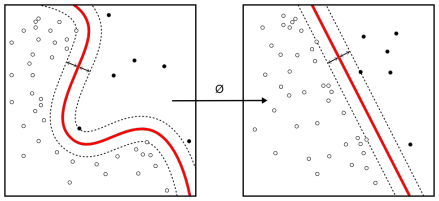
c. Choice of model

d. Hyperparameters / Validation Set

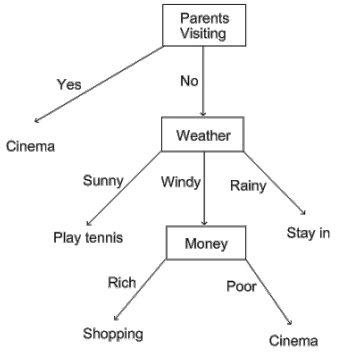
II. Deep Learning & ConvNets

Usual models

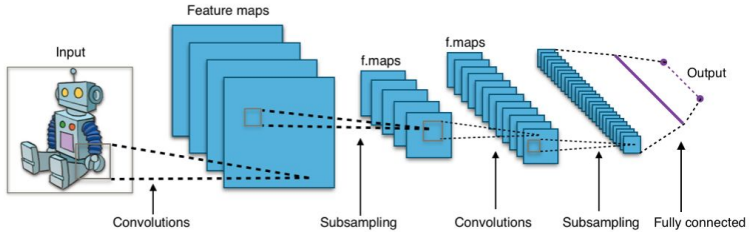
- Support Vector Machines (classification) / Support Vector Regression (regression)



- Decision trees / Random Forests



- Deep Learning
  - Fully-Connected Networks
  - Convolutional Neural Networks (CNN - ConvNets)
  - Recurrent Neural Networks (RNN)

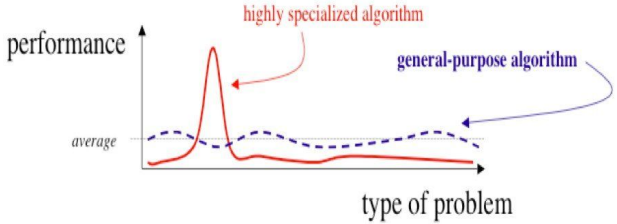


No Free Lunch Theorem

- Averaged over all possible random data distribution for **X,Y**
  - every Machine Learning model has the same error
  - no Machine Learning model is universally better

- So why are Deep Learning models working so well ?
  - Real-world distributions are fortunately not completely random

- Different Machine Learning models encode different prior knowledge about the function **$\hat{Y} = f(X)$**
  - ConvNets encode priors close to translation-invariance, and that task can be accomplished by processing through a hierarchy of features
  - Recurrent Neural Networks encode prior of stationarity
  - ...



Outline

I. Machine Learning

a. Context

b. Capacity / Underfitting / Overfitting

c. Choice of model

d. Hyperparameters / Validation Set

II. Deep Learning & ConvNets

Hyperparameters / Validation set

- **Standard model parameters**
  - They are fit during model training to minimize loss on training data

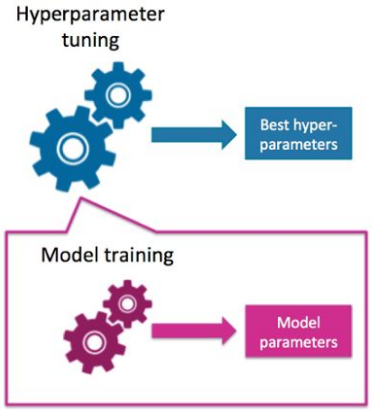
- **Hyperparameters:**
  - Define higher level concepts about the model
  - Cannot be learned during model training
  - They need to be predefined before model training
  - They are optimized to minimize **validation loss** (=loss on validation set after model training)

- Some examples of **hyperparameters**:
  - Number and type of layers in a Convnet
  - Regularization parameters
  - Learning rate
  - …

Usual Workflow

- For usual Machine Learning tasks, we will thus divide our data into 3 sets:
  - **Training set** (typically 70% of data)
  - **Validation set** (typically 15% of data)
  - **Test set** (typically 15% of data)

- Then:
  - For different choices of hyperparameters, we perform model training on the training set, and then compute the validation loss
  - We **fix hyperparameters** at the value **minimizing validation loss**
  - We then perform **model training** on the **union of** the **training set** and **validation set** (to have more training data)
  - We evaluate the **final precision** of our model by computing the **test loss** on the test set



Outline

I. Machine Learning

II. Deep Learning & ConvNets

a. History of Deep Learning

b. Deep Learning

c. Convets

History of Deep Learning

- **Connectionism** wave in the 1980s - 1990s
  - Central idea: a large number of simple computational units can achieve intelligent behavior when networked together
  - **Back-propagation** algorithm
  - Lasted until mid 1990s: at this point Neural Networks did not fulfill expectations, and other competing approaches made advances
  - Main problems: **too small datasets**, algorithms **too computationally expensive**

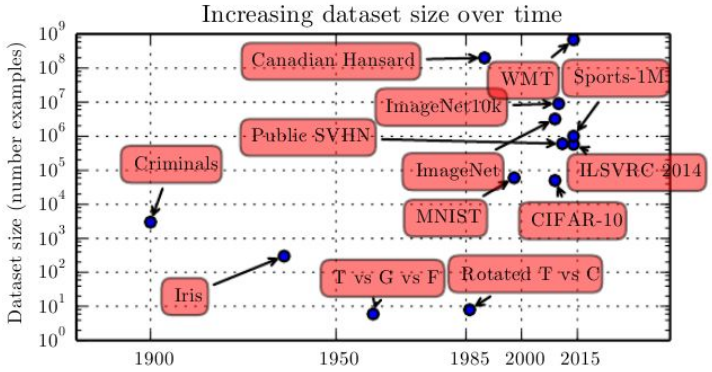
- **Deep Learning** revival from 2006:
  - “Deep” to emphasize the importance of depth
  - **Dataset sizes increased** dramatically (digitization of society, big data)
  - **Model sizes** also increased, thanks to software and hardware advances (in particular general purpose GPU)
  - Some theoretical advances (ReLU, Batch Normalization, Dropout, Inception, ResNet, LSTM, ...)
  - Huge success and impact in many fields (object recognition, speech recognition, robotics, NLP, Recommendation systems…)

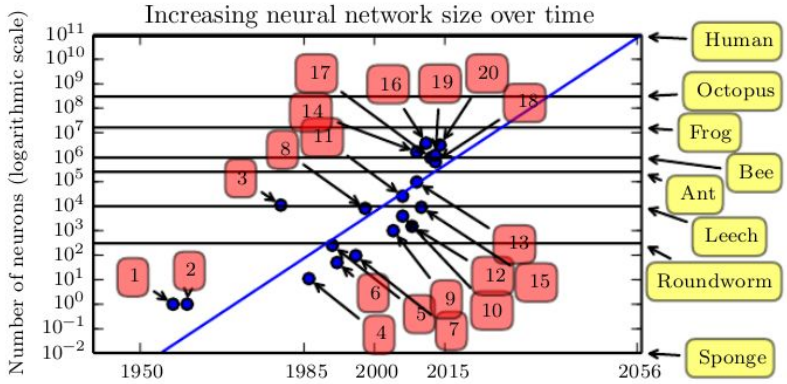
Dataset and Model sizes

- A rough rule of thumb for supervised deep learning algorithm:
  - Acceptable performance with around 5,000 labelec examples per category
  - Match or exceed human performance with a dataset > 10 million labeled examples

- Dataset sizes increased dramatically (from a few hundreds-thousands to now several million)

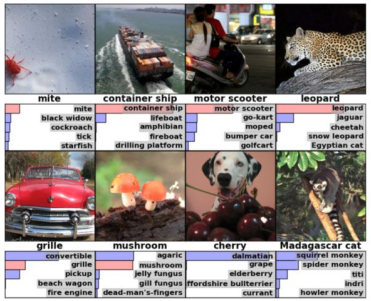
- Model sizes also increased exponentially (interpolate to human brain complexity in 40 years)

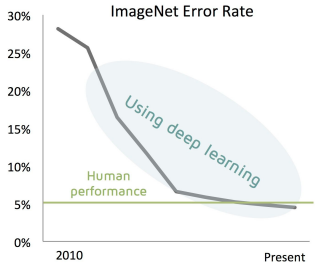




Deep Learning successes

- Super-human performance in **Object Recognition**
- Super-human performance in **Speech Recognition**
- State of the art in many **Natural Language Processing** (NLP) tasks
- Super-human game AI (Atari games, GO, Poker, …)
- **Self-driving cars**





Outline

I. Machine Learning

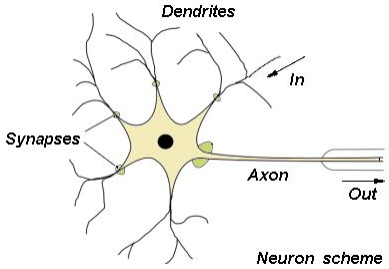
II. Deep Learning & ConvNets

a. History of Deep Learning

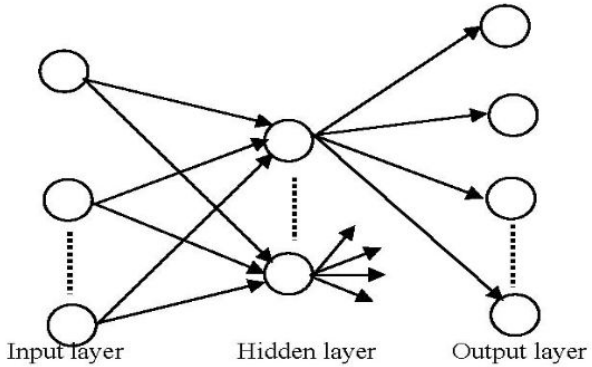
b. Deep Learning

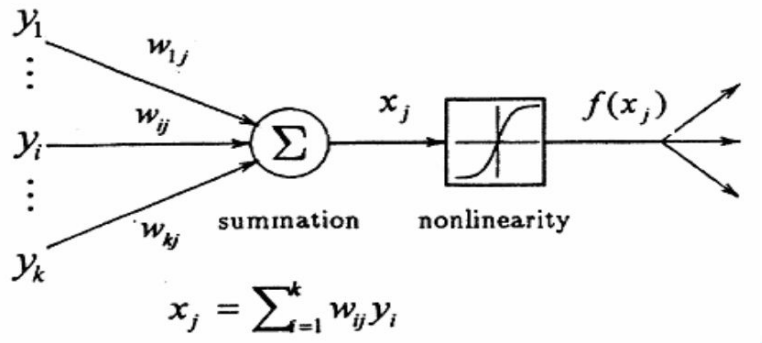
c. ConvNets

Basic Blocks



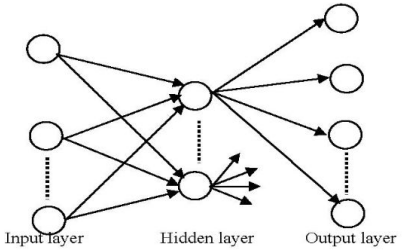
- Neural Networks inspired by biological brain
- Networks are organized in **successive layers**, and each layer consists of many **neurons** (also called units), that are the basic blocks of the network
- Each neuron receives input from neurons in previous layer, makes a weighted sum and applies a nonlinear **activation function**
- Early networks applied a sigmoid activation function, but modern networks mostly use ReLU



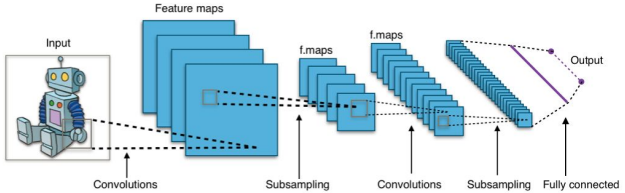


Network Types

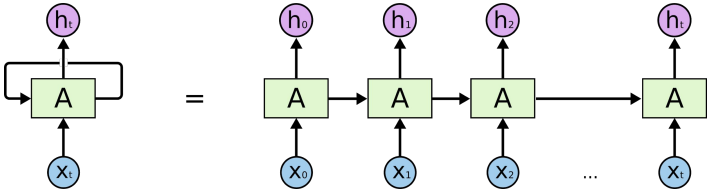
- Fully-Connected Neural Networks
  - Standard Network Architecture
  - Each layer is fully connected to the next layer



- Convolutional Neural Networks
  - Used for signal and image processing

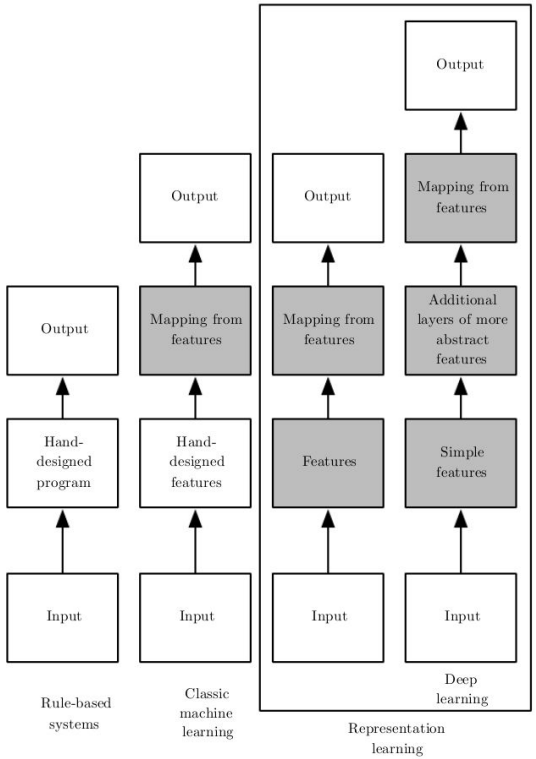


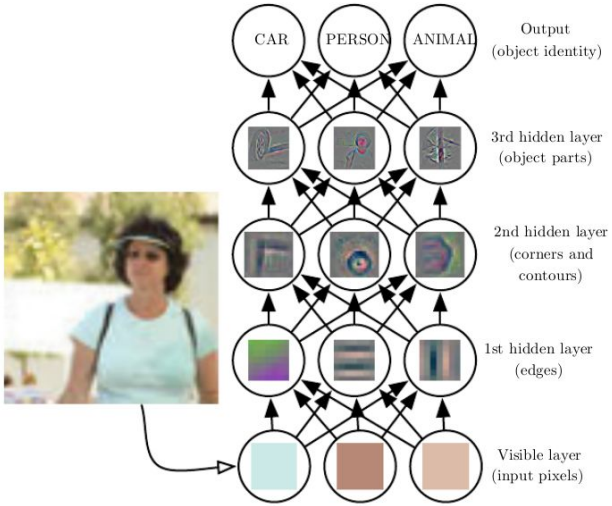
- Recurrent Neural Networks:
  - Used for sequence modeling (e.g. NLP)



Representation Learning - Depth

- Classic Machine Learning algorithms require **hand-crafted features** as inputs
- Artificial Neural Networks **learn features** by themselves
- Deep Neural Networks **learn several layers of features** by themselves





Demo

http://playground.tensorflow.org

Outline

I. Machine Learning

II. Deep Learning & ConvNets

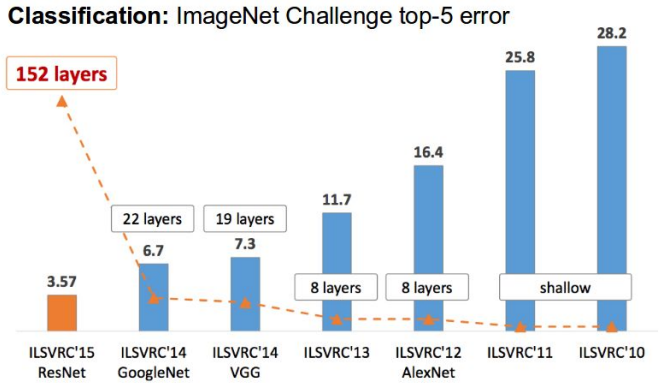
a. History of Deep Learning

b. Deep Learning

c. ConvNets

Success with ConvNets

- ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) competition = annual world cup for Image Classification
- For the first time in 2012, a CNN ranked first and decreased best error rate from 25% to 17%
- From this time, only CNN won the competition
- The error rate continued to drop very quickly, and now the best results are super-human

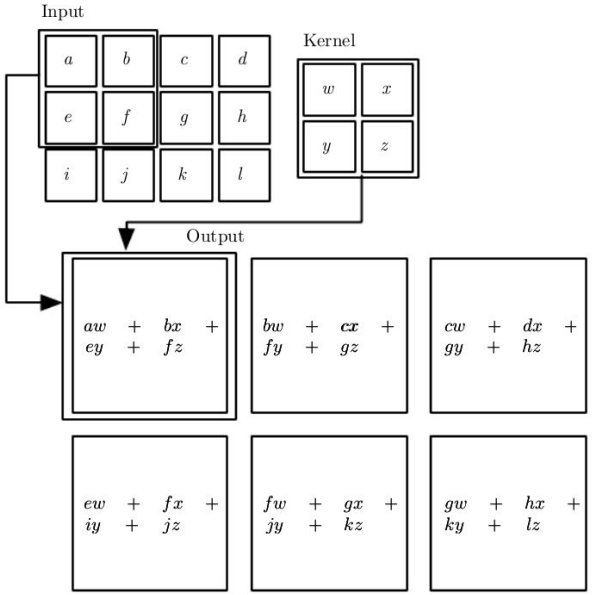


Convolution layers

- The main building blocks of Convolutional Neural Networks are **Convolutional layers**

- A convolution is the result of sliding a **filter** on a grid
  - The filter is **local** (its area is called the receptive field)
  - The filter is **constant** throughout the grid

- Introduced prior knowledge:
  - Due to translation-invariance, features describing objects should be independent of the position in the image
  - Features describing objects should only be local

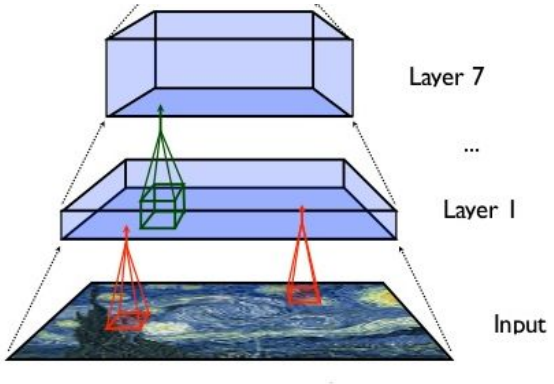


Convolutional layers

- Each convolutional layer has **different channels**, where each channel correspond to a single filter

- Layers are 3D
  - In the input: 3D = 2D (position in the image) + 1D (color channels)
  - In further layers: 3D = 2D (position in the image) + 1D (different filters)

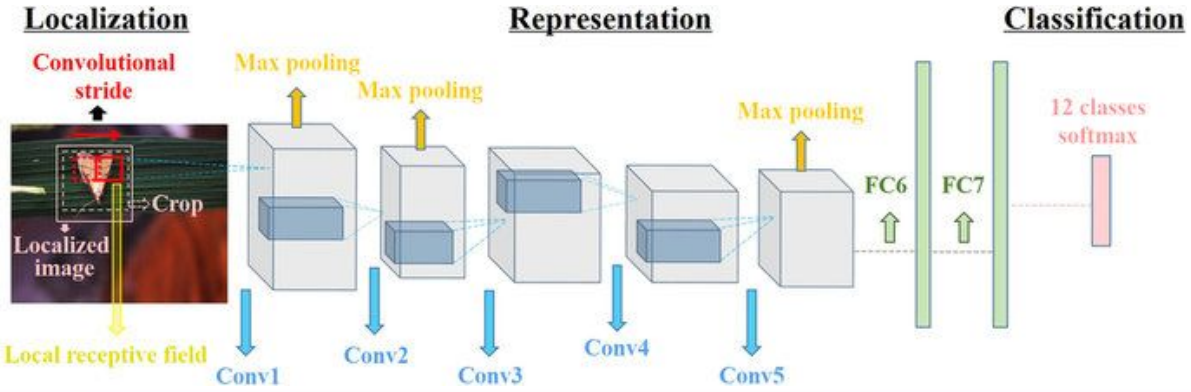
- Filters are 3D
  - local in the 2 image dimensions
  - global in the 3rd dimension



ConvNets in 2012

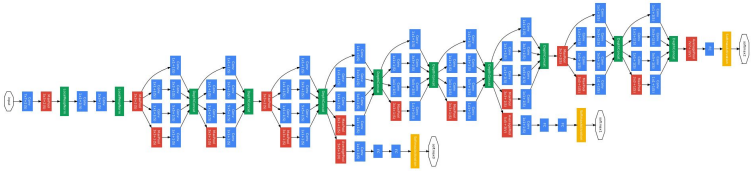
- First CNN that won ImageNet challenge:
  - A few successive convolutional layers
  - Pooling layers that progressively reduce spatial dimensions
  - Followed by a few fully connected layers

- Fully connected layers:
  - only 10% of computation time
  - 90% of the parameters



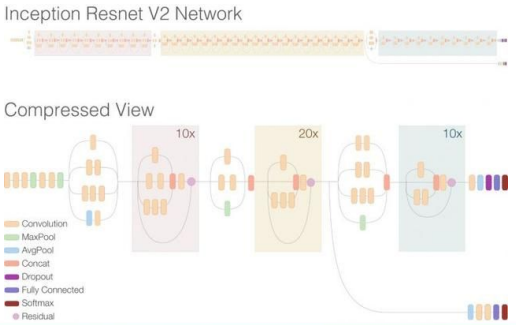
ConvNets now

- Now revolution of depth
- Google Inception v1 (ImageNet 2014 winner): 22 layers

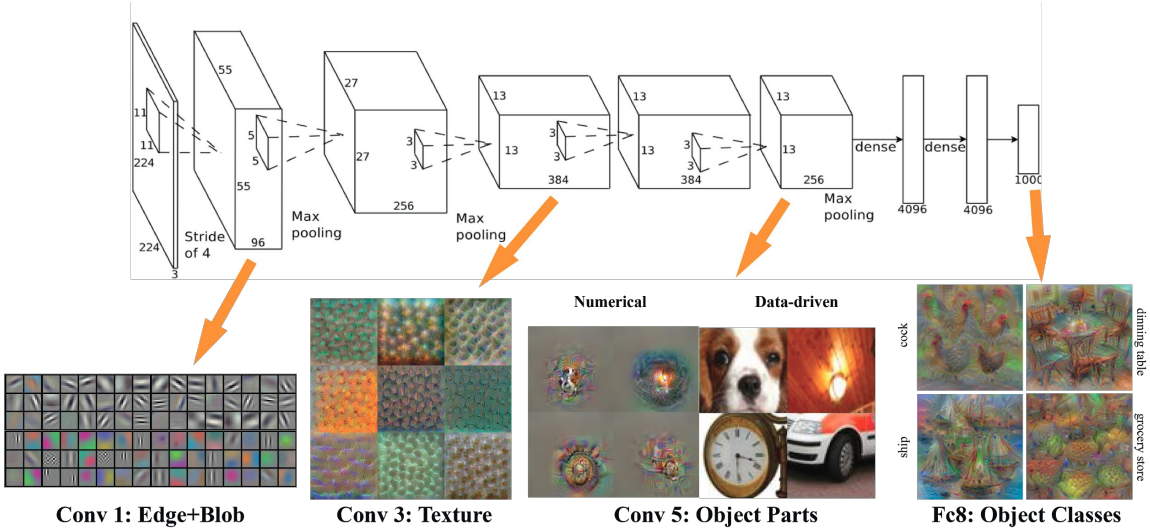


- ResNet (ImageNet 2015 winner): 152 layers

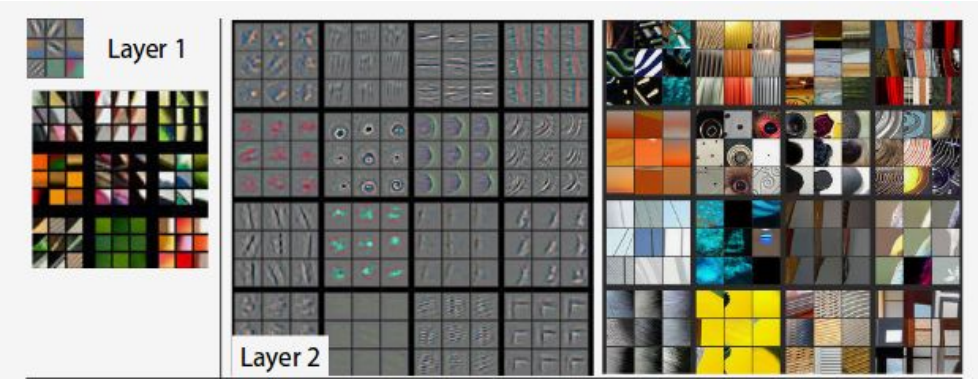
- Inception Resnet v2: 467 layers

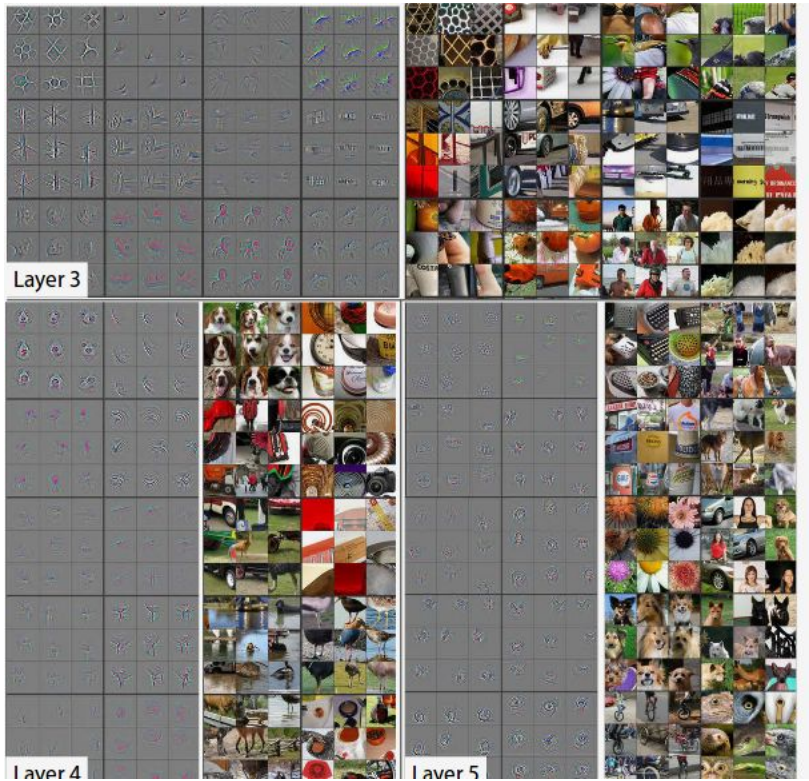


Visualizing ConvNets



Visualizing ConvNets





Thank you !