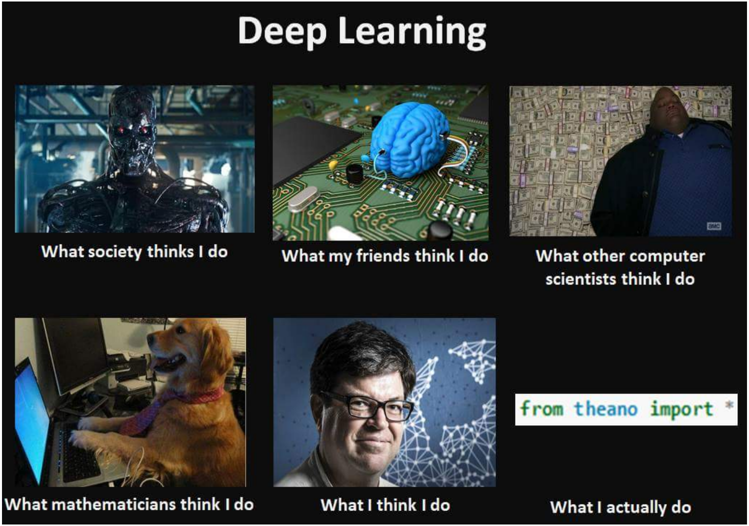
An Introduction to Deep Learning

Simon Carbonnelle

Université Catholique de Louvain, ICTEAM
1st of December, 2016


Deep Learning
What society thinks I do
What my friends think I do
What other computer scientists think I do
What mathematicians think I do
What I think I do
from theano import *
What I actually do

What is Deep Learning and how does it work?

What is Deep Learning currently capable of?

Why does Deep Learning work so well?

What is Deep Learning and how does it work?

What is Deep Learning currently capable of?

Why does Deep Learning work so well?

The Machine Learning approach.

Explicitly program how to solve a task	⟺	Explicitly program how to learn from data how to solve a task
+ Easy use of human knowledge for a specific task		+ Potentially solves tasks that humans can't program
		+ Potentially solves tasks that humans can't solve
		+ General-purpose algorithms

The Machine Learning approach.

Explicitly program
how to solve a task

Explicitly program
how to learn from data
how to solve a task

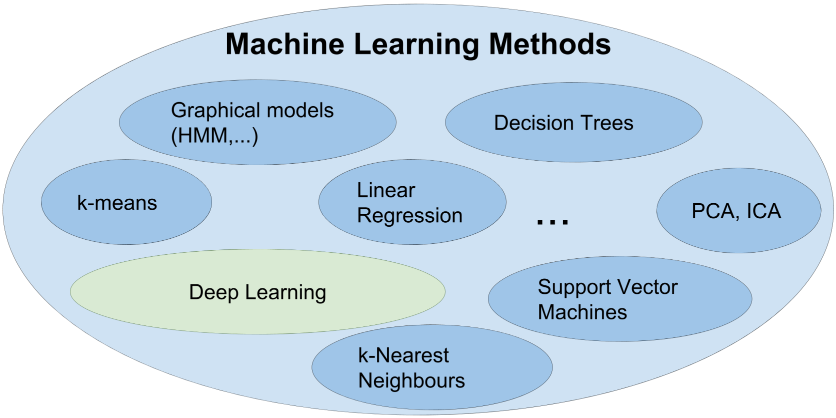
- + Easy use of human knowledge for a specific task

- + Potentially solves tasks that humans can't program
- + Potentially solves tasks that humans can't solve
- + General-purpose algorithms

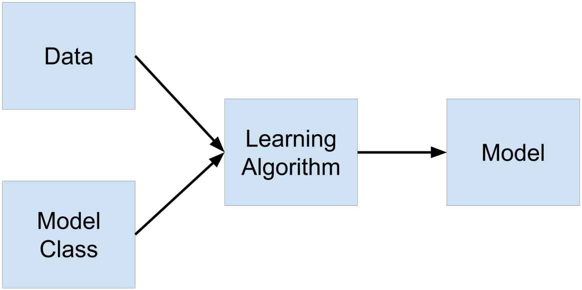
E.g.: Integer addition, Sorting algorithms,...

E.g.: Spam filtering, Playing Go, Object Recognition,...

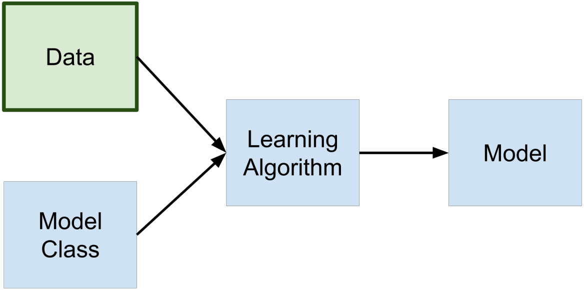
Deep Learning is a sub-field of Machine Learning.



The Machine Learning Flow.

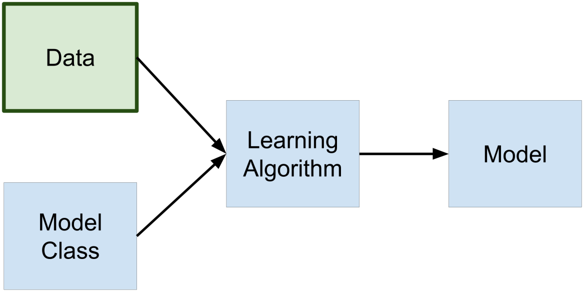


What data is available to learn from?



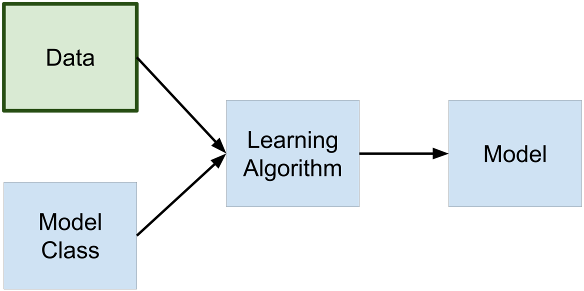
1. Supervised Learning
   $\Rightarrow (X, y)$, inputs and outputs of a task

What data is available to learn from?



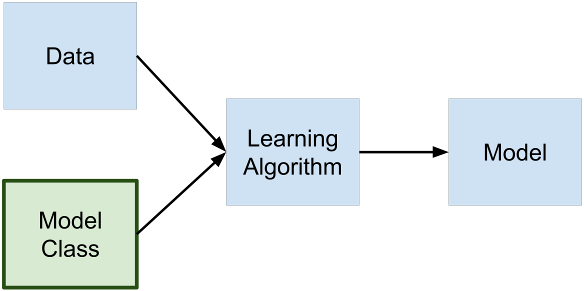
1. Supervised Learning
   $\Rightarrow (X, y)$, inputs and outputs of a task
2. Unsupervised Learning
   $\Rightarrow (X)$, inputs of a task

What data is available to learn from?



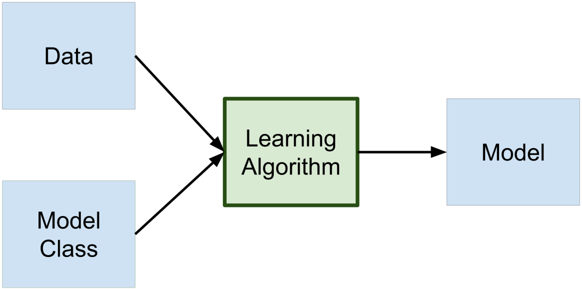
1. Supervised Learning
   $\Rightarrow (X, y)$, inputs and outputs of a task
2. Unsupervised Learning
   $\Rightarrow (X)$, inputs of a task
3. Reinforcement Learning
   $\Rightarrow (X, r)$, inputs and rewards for a task

What type of model will I use?



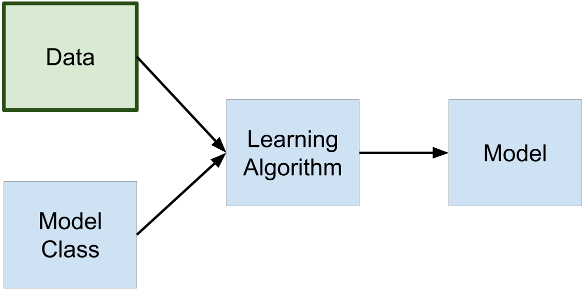
1. Linear
2. Output $\in [0, 1]$
3. Tree-based
4. Probabilistic
5. ...

How do I learn the final model?



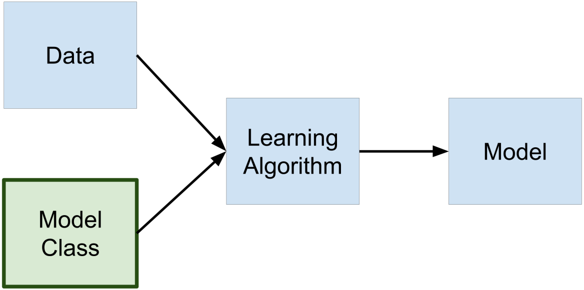
1. Optimization with gradient descent
2. Optimization with closed form solution
3. Iterative algorithms
4. Heuristic search
5. ...

Example: Linear regression



Supervised Learning: x_i, y_i

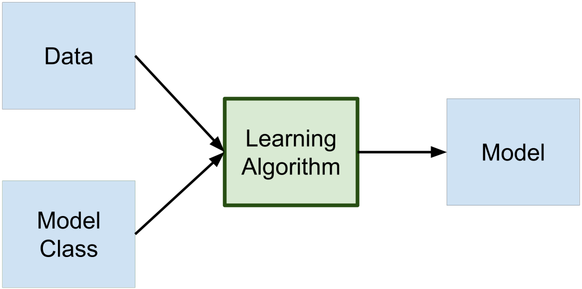
Example: Linear regression



Supervised Learning: x_i, y_i

Linear: $\hat{y}_i = Ax_i + b$

Example: Linear regression

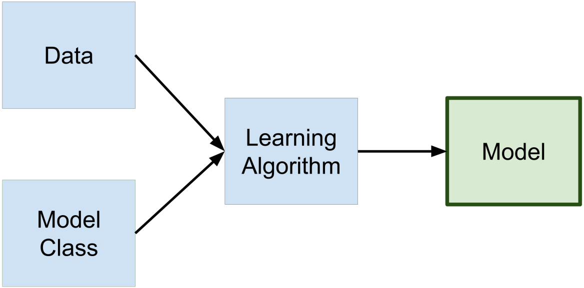


Supervised Learning: x_i, y_i

Linear: $\hat{y}_i = Ax_i + b$

$\min_{A,b} \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

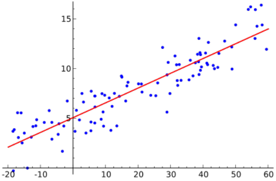
Example: Linear regression



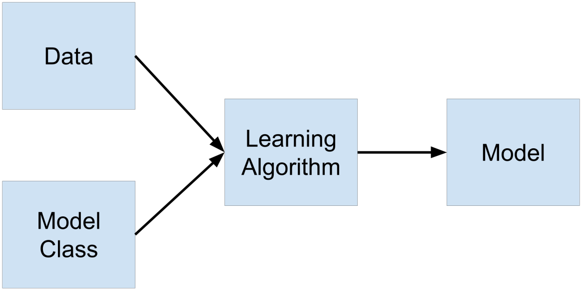
Supervised Learning: x_i, y_i

Linear: $\hat{y}_i = Ax_i + b$

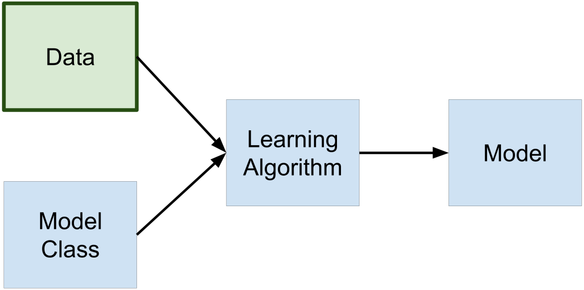
$\min_{A,b} \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$



What about Deep Learning?

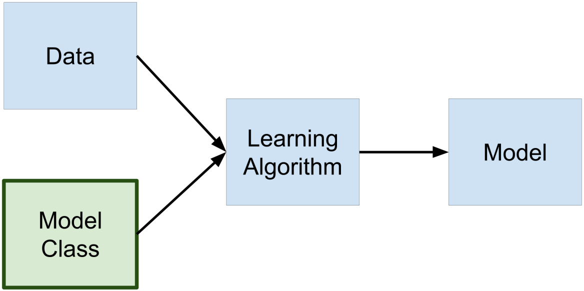


What about Deep Learning?



All three (see next section)

What about Deep Learning?

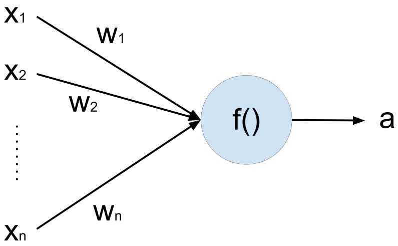


All three (see next section)

Deep Neural Networks
(DNN, CNN, RNN)

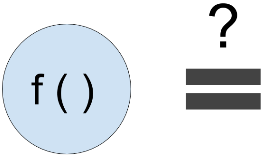
The Artificial Neuron.

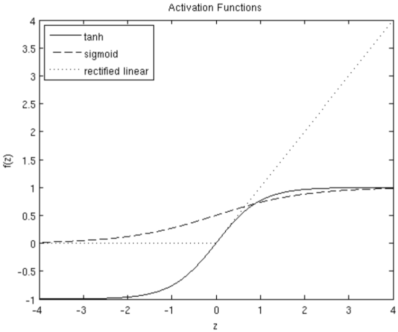
A very simple building block.



$$a = f\left(\sum_{j=1}^{n} w_j x_j\right)$$

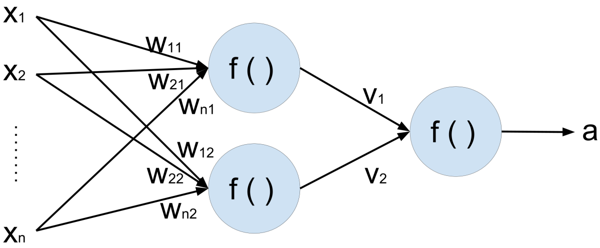
The Activation Function.





Sigmoid	Hyperbolic tangent	Rectified Linear Unit
$f(z) = \frac{1}{1+e^{-z}}$	$f(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$f(z) = max(0, z)$

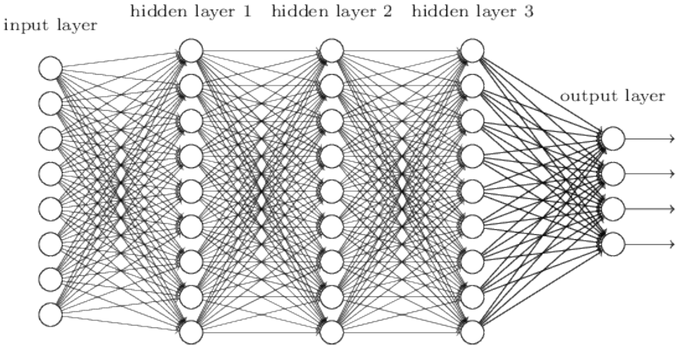
Artificial Neural Networks.



$$a = f\left(\sum_{j=1}^{2} v_j \cdot f\left(\sum_{k=1}^{n} w_{kj} x_j\right)\right)$$

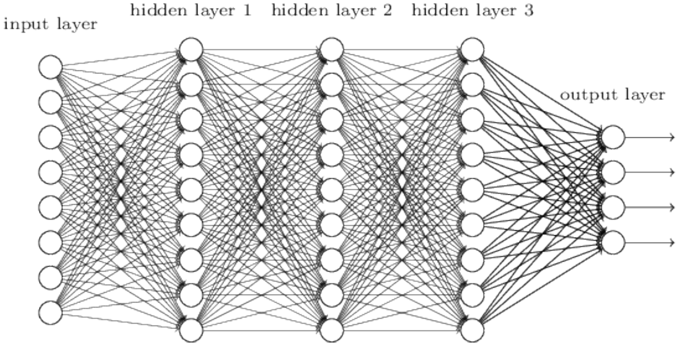
Deep Neural Networks.

Artificial Neural Networks with many layers.



Deep Neural Networks.

Artificial Neural Networks with many layers.

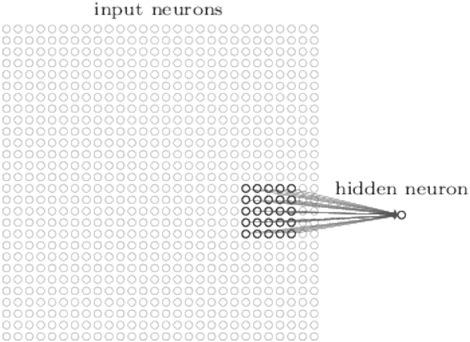


Problem

Number of connections grows exponentially.

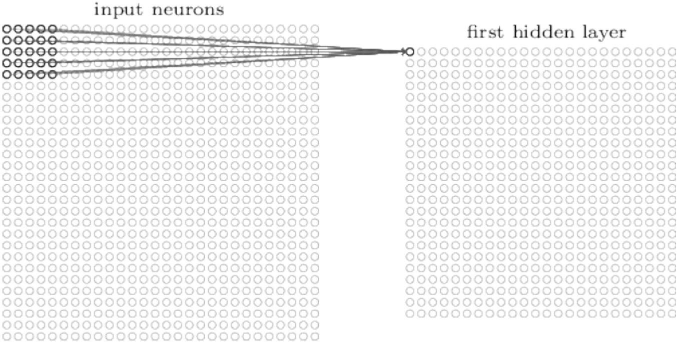
Convolutional Neural Networks

Modelling **local structure** and translation invariance explicitly.



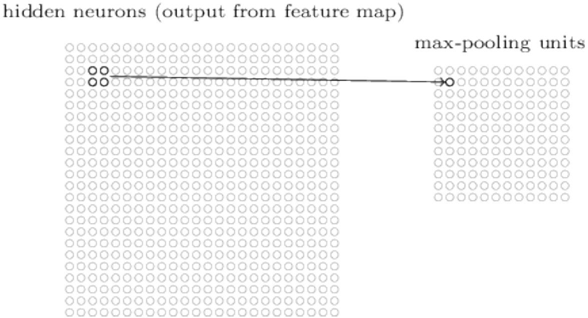
Convolutional Neural Networks

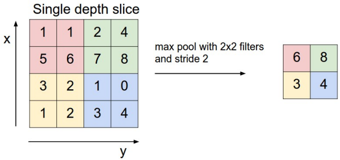
Modelling local structure and **translation invariance** explicitly.



Convolutional Neural Networks

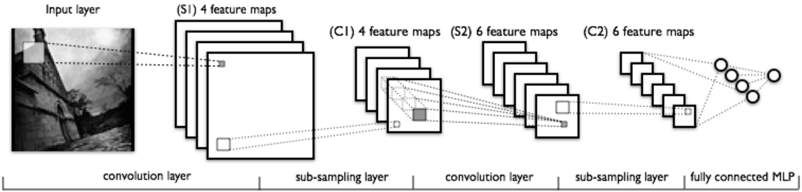
Reducing the **spatial dimensions**.





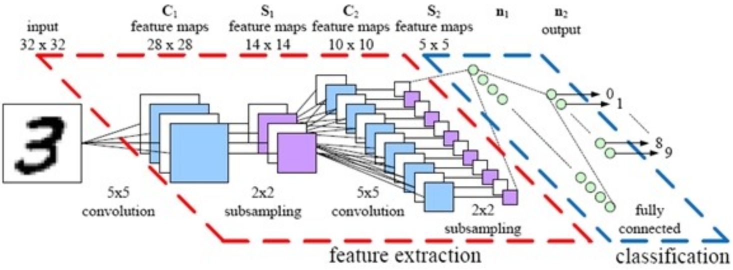
Convolutional Neural Networks

Assembling mutliptle **feature maps** and **layers**.

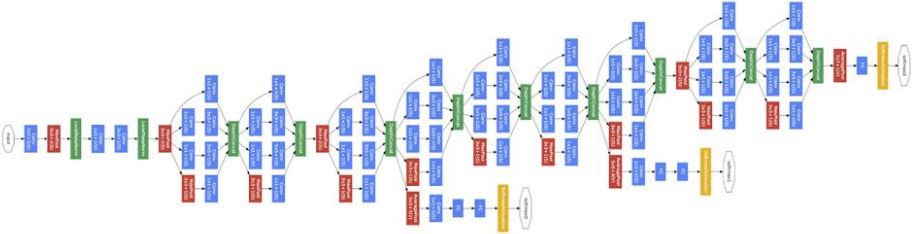


Convolutional Neural Networks

Designing a network for digit recognition.



Convolutional Neural Networks

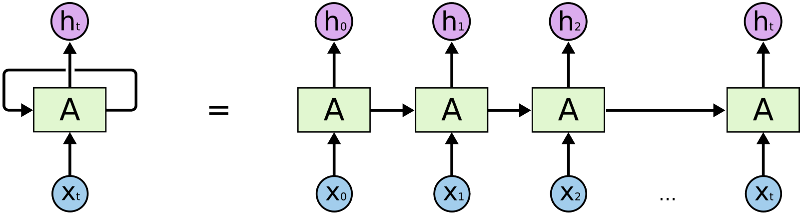


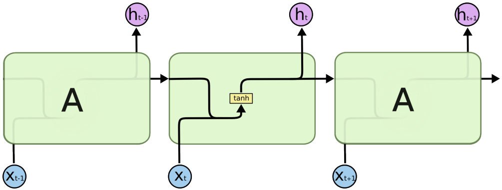
Szegedy, C. et al., Going deeper with convolutions. CVPR 2015

VGG-Net (Visual Geometry Group, Oxford) has >**130M parameters**.

Recurrent Neural Networks

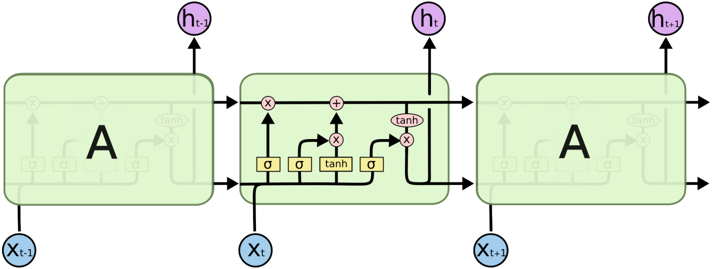
Adding an internal memory state to Neural Networks.





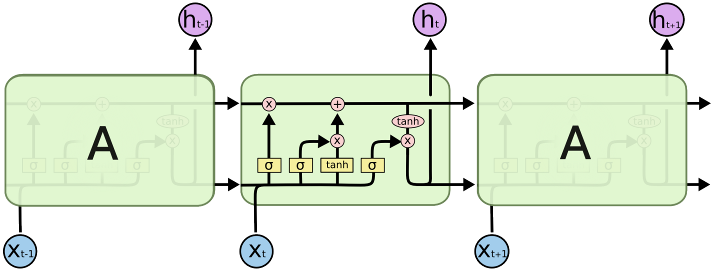
Recurrent Neural Network variants

The Long Short-Term Memory Network is the most popular:

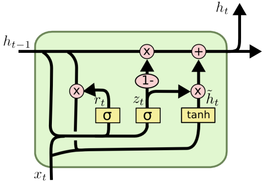


Recurrent Neural Network variants

The Long Short-Term Memory Network is the most popular:



The Gated Recurrent Unit (GRU) has very simple equations:



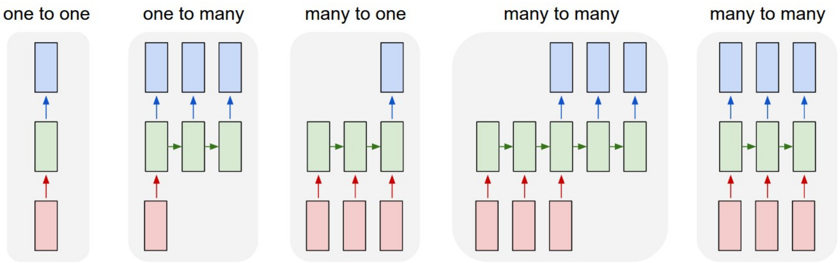
$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

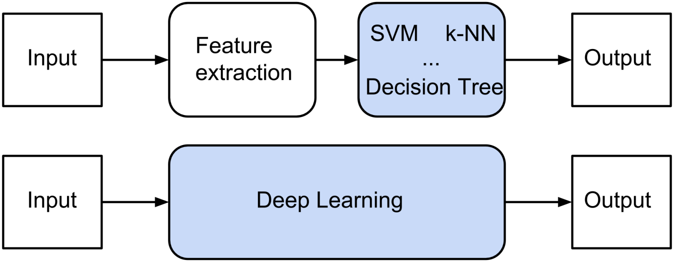
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

RNNs for sequence to sequence tasks

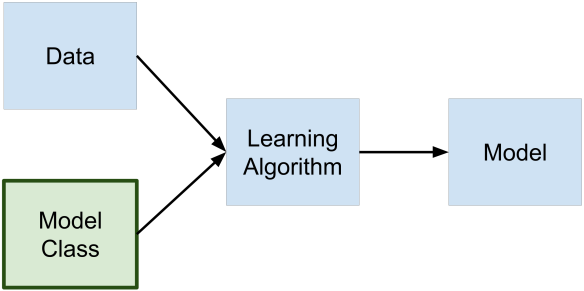


CNNs and RNNs avoid Feature Extraction.



The networks are designed to take advantage of the input structure

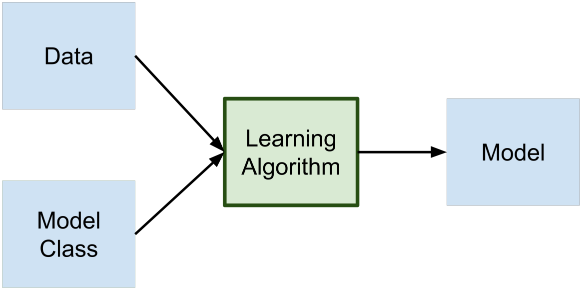
What about Deep Learning?



All three (see next section)

Deep Neural Networks
(DNN, CNN, RNN)

What about Deep Learning?



All three (see next section)

Deep Neural Networks
(DNN, CNN, RNN)

$\min_w C(w, X, y)$
Mini-batch gradient descent

Learning algorithm

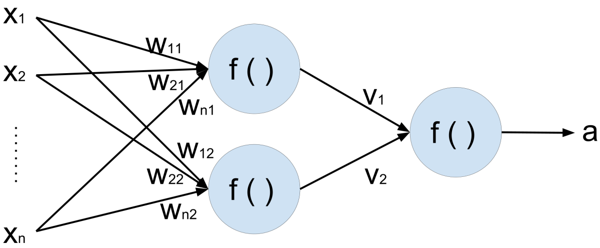
$$\min_w C(w, X, y) = \min_w \frac{1}{N} \sum_{X_i \in X, y_i \in y} C_s(w, X_i, y_i)$$

Mini-batch gradient descent
At each iteration, select B samples $(X^{(b)}, y^{(b)})$
Apply gradient descent on:

$$\min_w \frac{1}{B} \sum_{X_i \in X^{(b)}, y_i \in y^{(b)}} C_s(w, X_i, y_i)$$

$$\Rightarrow w = w + \frac{\lambda}{B} \sum_{X_i \in X^{(b)}, y_i \in y^{(b)}} \nabla_w C_s(w, X_i, y_i)$$

Computing the derivatives

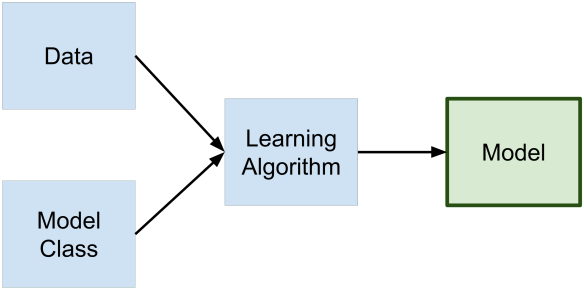


$$a = f\left(\sum_{j=1}^{2} v_j \cdot f\left(\sum_{k=1}^{n} w_{kj} x_j\right)\right)$$

Chain rule all the way! $\Rightarrow$ The back-propagation algorithm

(Rumelhart et al., 1986)

What about Deep Learning?



All three (see next section)

Deep Neural Networks
(DNN, CNN, RNN)

$\min_w C(w, X, y)$
Mini-batch gradient descent

Pratical implementation of Deep Learning

Libraries do it for you:

- Automatic differentiation
- GPU optimized
- High-level libraries

```
from keras.models import Sequential

model = Sequential()
```

```
from keras.layers import Dense, Activation

model.add(Dense(output_dim=64, input_dim=100))
model.add(Activation("relu"))
model.add(Dense(output_dim=10))
model.add(Activation("softmax"))
```

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

```
model.fit(X_train, Y_train, nb_epoch=5, batch_size=32)
```

First part conclusion

- Machine Learning: Coding how to learn from data
- Deep Learning
  - Data: Supervised, Unsupervised, Reinforcement
  - Model: Artificial Neural Networks
  - Algorithm: mini-batch gradient descent

First part conclusion

- Machine Learning: Coding how to learn from data
- Deep Learning
  - Data: Supervised, Unsupervised, Reinforcement
  - Model: Artificial Neural Networks
  - Algorithm: mini-batch gradient descent
- **Deep Learning's big advantage:**
  Network design to take advantage of input structure!!!

What is Deep Learning and how does it work?

What is Deep Learning currently capable of?

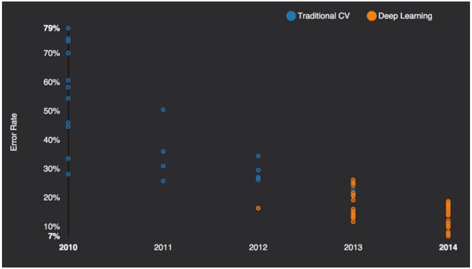
Why does Deep Learning work so well?

Supervised Learning

Image Classification

Alaskan Malamute

Siberian Husky



Supervised Learning

Speech Recognition

Achieving Human Parity In Conversational Speech Recognition
(Xiong et al.; 2016)

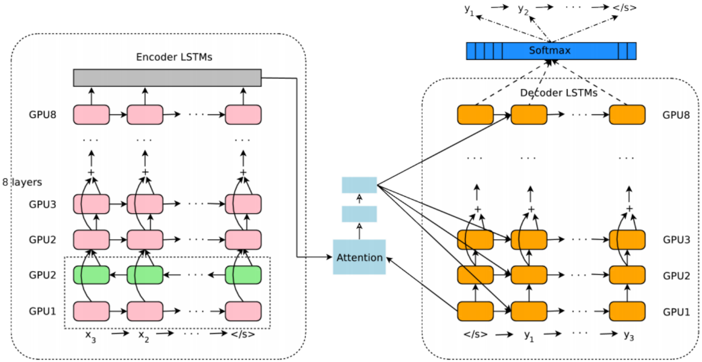
Model	N-gram LM		RNN-LM		LSTM-LM	
	CH	SWB	CH	SWB	CH	SWB
300h ResNet	19.2	10.0	17.7	8.2	17.0	7.7
ResNet GMM alignment	15.3	8.8	13.7	7.3	12.8	6.9
ResNet	14.8	8.6	13.2	6.9	12.5	6.6
VGG + ResNet	14.5	8.4	13.0	6.9	12.2	6.4
VGG	15.7	9.1	14.1	7.6	13.2	7.1
LACE	14.8	8.3	13.5	7.1	12.7	6.7
BLSTM	16.6	8.9	15.1	7.4	14.4	7.0
BLSTM 27k senones	16.2	8.7	14.6	7.5	13.6	7.0
BLSTM 27k, spatial smoothing	14.9	8.3	13.7	7.0	13.0	6.7
Final ASR System	13.3	7.4	12.0	6.2	**11.1**	**5.9**
Human Performance	-	-	-	-	11.3	5.9

Word error rates (%). Trained on 2000 hours of data.

Supervised Learning

Automatic translation

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation (Wu et al.; 2016)

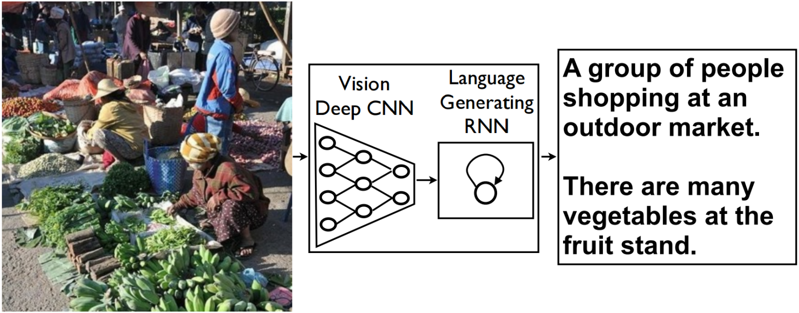


Supervised Learning

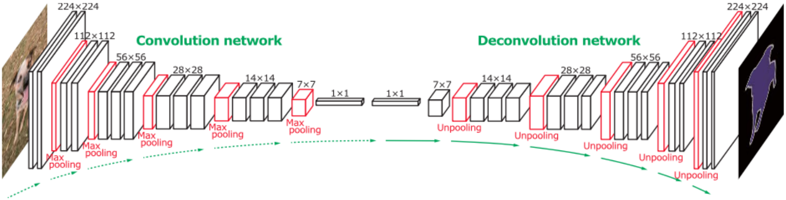
Image Captioning

Show and Tell: A Neural Image Caption Generator

(Vinyals et al.; 2016)



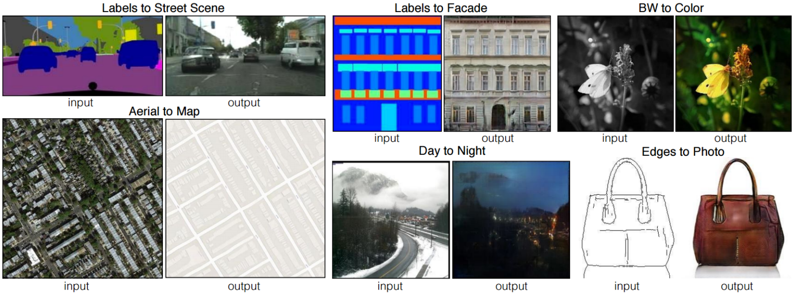
Learning Deconvolution Network for Semantic Segmentation
(Noh et al.; 2015)



Supervised Learning

Image Translation

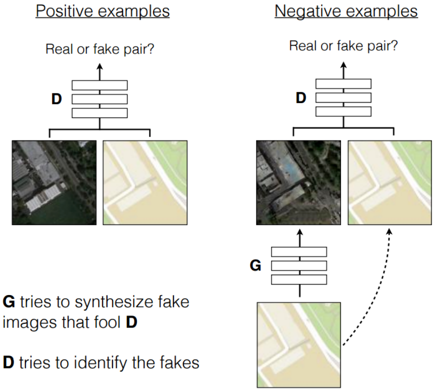
Image-to-Image Translation with Conditional Adversarial Networks (Isola et al.; 2016). Code: `https://phillipi.github.io/pix2pix/`



Supervised Learning

Image Translation

Image-to-Image Translation with Conditional Adversarial Networks (Isola et al.; 2016). Code: `https://phillipi.github.io/pix2pix/`

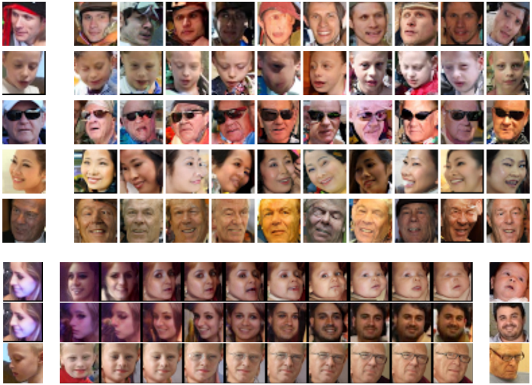


Unsupervised Learning

Conditional Image Generation

Conditional Image Generation with PixelCNN Decoders

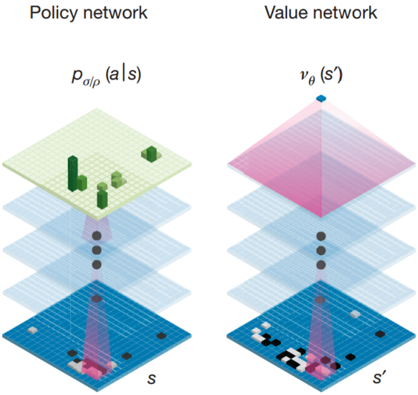
(van den Oord et al.; 2016)



Reinforcement Learning

The Game of Go

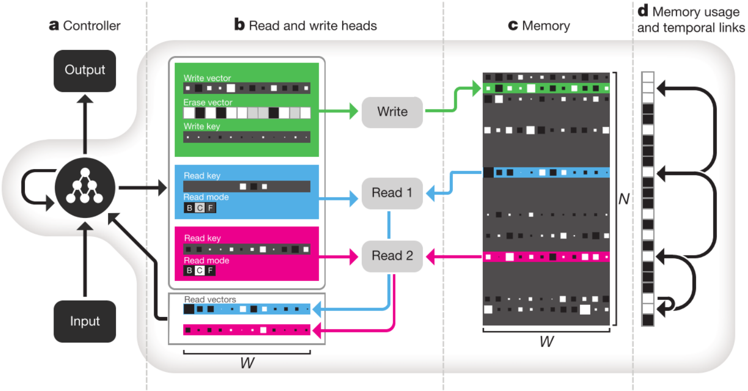
Mastering the game of Go with deep neural networks and tree search (Silver et al.; 2016)



Memory-augmented neural networks

Learning to reason over complex data structures.

Hybrid computing using a neural network
with dynamic external memory (Graves et al.; 2016)

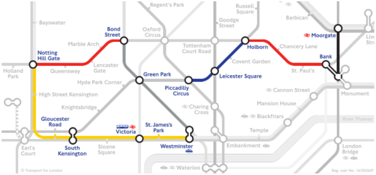


Memory-augmented neural networks

Learning to reason over complex data structures.

Hybrid computing using a neural network
with dynamic external memory (Graves et al.; 2016)

b London Underground



Traversal

Shortest-path

Underground input:
(OxfordCircus, TottenhamCtRd, Central)
(TottenhamCtRd, OxfordCircus, Central)
(BakerSt, Marylebone, Circle)
(BakerSt, Marylebone, Bakerloo)
(BakerSt, OxfordCircus, Bakerloo)
⋮
(LeicesterSq, CharingCross, Northern)
(TottenhamCtRd, LeicesterSq, Northern)
(OxfordCircus, PiccadillyCircus, Bakerloo)
(OxfordCircus, NottingHillGate, Central)
(OxfordCircus, Euston, Victoria)

84 edges in total

Traversal question:
(BondSt, _, Central),
(_, _, Circle), (_, _, Circle),
(_, _, Circle), (_, _, Circle),
(_, _, Jubilee), (_, _, Jubilee),

Answer:
(BondSt, NottingHillGate, Central)
(NottingHillGate, GloucesterRd, Circle)
⋮
(Westminster, GreenPark, Jubilee)
(GreenPark, BondSt, Jubilee)

Shortest-path question:
(Moorgate, PiccadillyCircus, _)

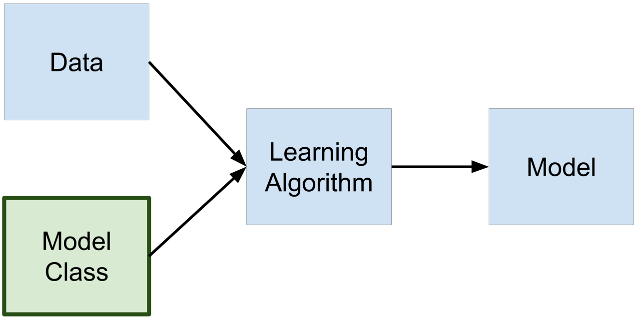
Answer:
(Moorgate, Bank, Northern)
(Bank, Holborn, Central)
(Holborn, LeicesterSq, Piccadilly)
(LeicesterSq, PiccadillyCircus, Piccadilly)

What is Deep Learning and how does it work?

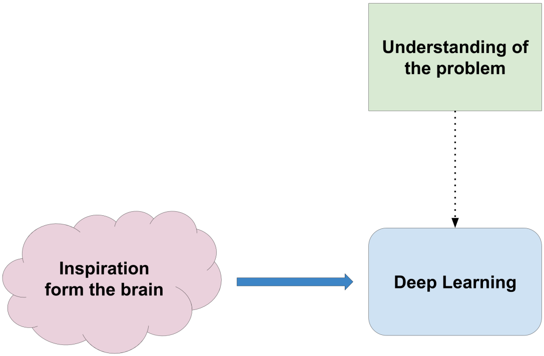
What is Deep Learning currently capable of?

Why does Deep Learning work so well?

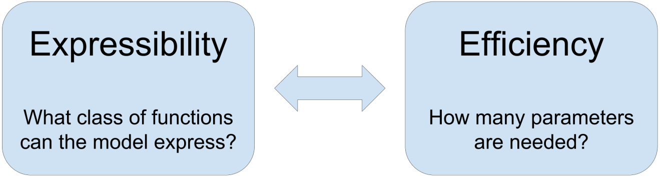
1. What makes the Deep Learning Model good?



Where does the Deep Learning model come from?



A compromise needs to be found.



The best compromise is found when you use the right **priors**.

The Deep Learning priors.

- Smoothness
- Compositionality
  - Distributed Representations
  - Multiple levels of Representations

The Deep Learning priors.

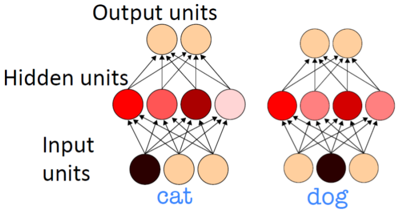
- Smoothness
- Compositionality
  - Distributed Representations
  - Multiple levels of Representations

Used by all the machine learning methods.
Necessary for generalization.

The Deep Learning priors.

- Smoothness
- Compositionality
  - Distributed Representations
  - Multiple levels of Representations

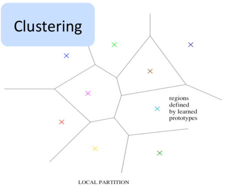
"Every concept is represented by many latent factors, latent factors are used for many concepts."

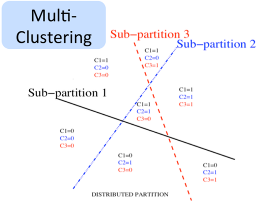


The Deep Learning priors.

- Smoothness
- Compositionality
  - Distributed Representations
  - Multiple levels of Representations

"Every concept is represented by many latent factors, latent factors are used for many concepts."

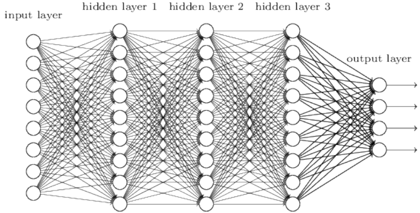




The Deep Learning priors.

- Smoothness
- Compositionality
  - Distributed Representations
  - Multiple levels of Representations

Concepts are hierarchically structured.

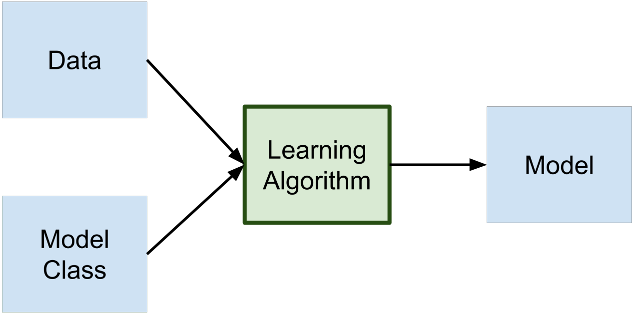


No-flattening Theorems: $x_1 \cdot x_2 \cdot \ldots \cdot x_n$

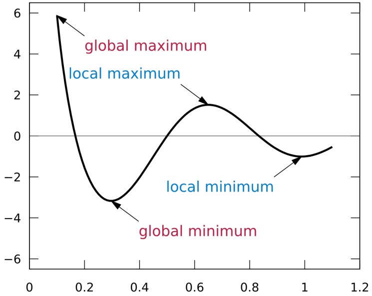
1 hidden layer: 2^n neurons needed

$\log_2 n$ hidden layers: $4n$ neurons needed

2. Why does the learning algorithm work?

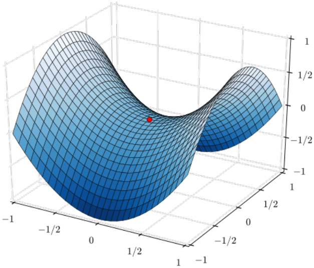


Minimizing a non-convex function with gradient descent...



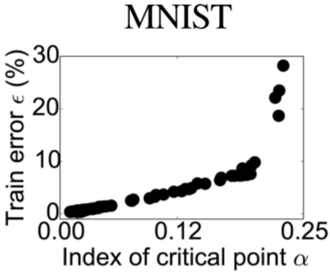
High-dimensional spaces

1. Saddle points are much more prevalent.
   Proved for Gaussian Processes:
   Ratio increases exponentially with N (Rasmussen, Williams; 2005)



High-dimensional spaces

1. Saddle points are much more prevalent.
   Proved for Gaussian Processes:
   Ratio increases exponentially with N (Rasmussen, Williams; 2005)
2. Most local minima are close to the global minima (in value)
   Proved for Gaussian Processes (Bray, Dean; 2007)
   Observed **experimentally** for neural nets (Dauphin et al.; 2014):



Last part conclusion

- Deep Learning uses a un-explored prior: Compositionality.
- Non-convex optimization in high-dimensional spaces
  is not like we would expect.

Last part conclusion

- Deep Learning uses a un-explored prior: Compositionality.
- Non-convex optimization in high-dimensional spaces is not like we would expect.

Thanks!

References

Linear regression image:
https://en.wikipedia.org/wiki/Linear_regression
Activation functions image:
http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/
Max-pooling image:
http://cs231n.github.io/convolutional-networks/
Convolution image: http://neuralnetworksanddeeplearning.com/
LeNet image: http://deeplearning.net/tutorial/lenet.html
RNN images:
http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Keras code: https://keras.io/
Non-convex function image:
https://en.wikipedia.org/wiki/Maxima_and_minima

Part 3 content:
Yoshua Bengio's talk: Deep Learning: Theoretical Motivations
Why does deep and cheap learning work so well? (Lin, Tegmark;2016)