

Strengths, weaknesses, opportunities and threats of deep learning

E. Decencière

Mines Paris
PSL University
Center for Mathematical Morphology



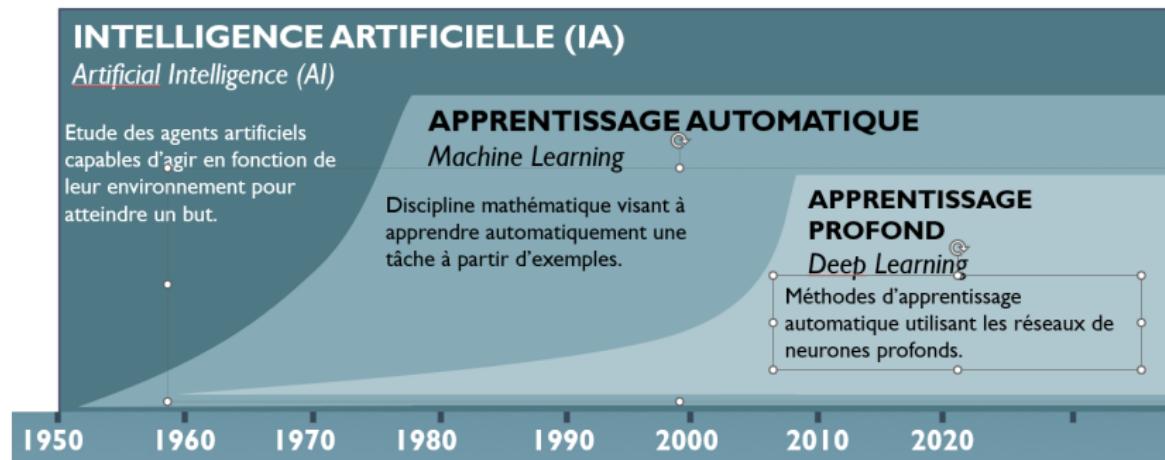
Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

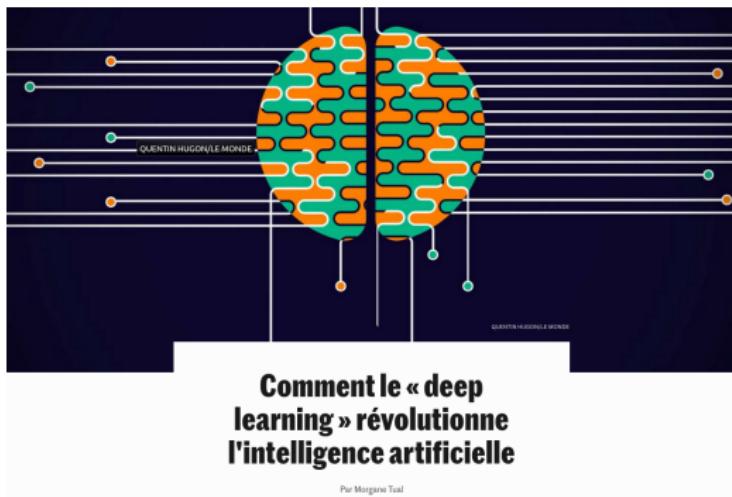
Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

Définitions



The rise of deep learning

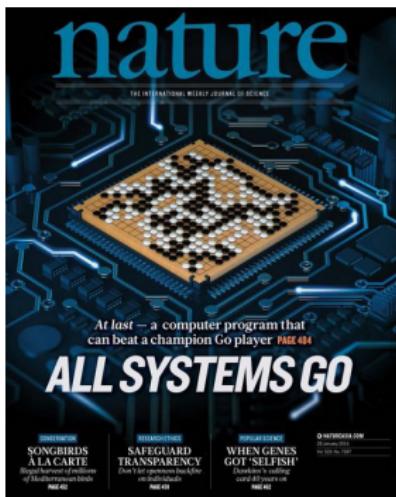


Comment le « deep learning » révolutionne l'intelligence artificielle

Par Morgane Teal

Le Monde, juillet 2015

The rise of deep learning



Nature, 2016

The rise of deep learning

Le prix Turing récompense trois pionniers de l'intelligence artificielle (IA)

L'association américaine ACM a remis son prestigieux prix aux chercheurs français, canadien et britannique : Yann LeCun, Yoshua Bengio et Geoffrey Hinton.

Par David Larousserie · Publié le 27 mars 2019 à 11h01 - Mis à jour le 29 mars 2019 à 12h11

Le Monde, March 2019

Pour Elon Musk, l'intelligence artificielle pourrait menacer la civilisation

L'entrepreneur américain, qui a fondé Tesla, a alerté les politiques américains sur la nécessité de réguler l'intelligence artificielle.

Par **Le Figaro**

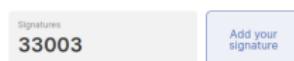
Publié le 18/07/2017 à 06:00, mis à jour le 18/07/2017 à 11:25

Le Figaro, July 2017

AI six months ban letter by the Future of Life Institute

Pause Giant AI Experiments: An Open Letter

We call on all AI labs to immediately pause for at least 6 months the training of AI systems more powerful than GPT-4.



PUBLISHED
March 22, 2023

Prominent signatories:

- Yoshua Bengio, Founder and Scientific Director at Mila, Turing Prize winner and professor at University of Montreal
- Stuart Russell, Berkeley, Professor of Computer Science, director of the Center for Intelligent Systems, and co-author of the standard textbook “Artificial Intelligence: a Modern Approach”
- Bart Selman, Cornell, Professor of Computer Science, past president of AAAI
- Elon Musk, CEO of SpaceX, Tesla, Twitter
- Steve Wozniak, Co-founder, Apple
- Yuval Noah Harari, Author and Professor, Hebrew University of Jerusalem.
- Emad Mostaque, CEO, Stability AI

Generative Pre-trained Transformer

ÉCONOMIE • INTELLIGENCE ARTIFICIELLE

Après ChatGPT, OpenAI dévoile GPT-4, encore plus performant malgré des limites persistantes

L'entreprise californienne a présenté une nouvelle version du logiciel de traitement du langage sur lequel s'appuie son désormais célèbre robot conversationnel ChatGPT.

Par Alexandre Piquard

Le Monde, March 2023

Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

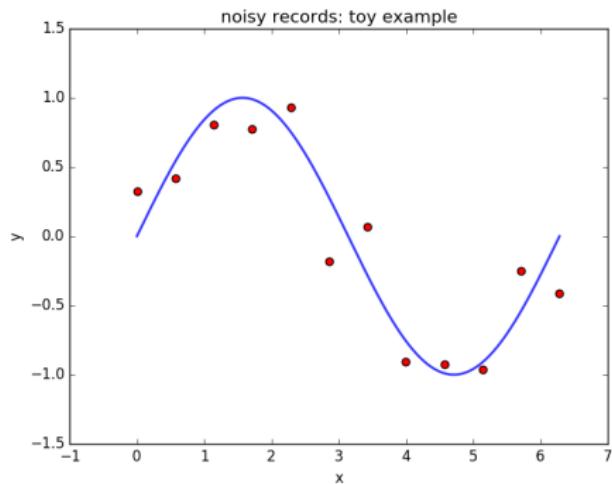
Machine Learning: basic definitions

- Machine Learning aims at predicting some output y from an input (or measurement) x :

$$y = f(x) \tag{1}$$

- In this formulation, Machine Learning aims at finding (learning) f among a set of given possibilities, using available data.
- The data that is used to learn f is called **training set**.
- In this general formulation, there is no particular limitation as to the mathematical nature of x and y .

A simple example: polynomial curve fitting¹



¹Example adapted from [Bishop, 2006]

A simple example: polynomial curve fitting

- We use the following polynomial model:

$$\begin{aligned}f(x) &= a_0 + a_1x + a_2x^2 + \dots + a_mx^m \\&= \boldsymbol{\theta}^T \boldsymbol{\phi}(x)\end{aligned}\tag{2}$$

- Parameter vector: $\boldsymbol{\theta} = (a_0, a_1, \dots, a_m)^T$

A simple example: polynomial curve fitting

- One classical approach is to minimize the least squared error between measured and predicted values:

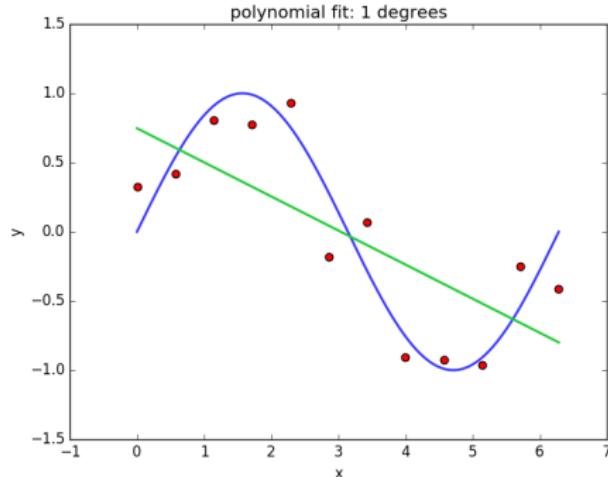
$$\min_{\theta} L(\theta) = \min_{\theta} \sum_{i=1}^N (y_i - f(x_i))^2$$

- This can be achieved by setting the gradient with respect to θ to zero:

$$\nabla_{\theta} L = \left(\frac{\partial L}{\partial a_0}, \frac{\partial L}{\partial a_1}, \dots, \frac{\partial L}{\partial a_m} \right)^T = 0 \quad (3)$$

- Unlike most optimization problems in deep learning, this leads to an analytical solution for θ . This is known as **linear regression**. For more details, we refer to [Hastie et al., 2009].

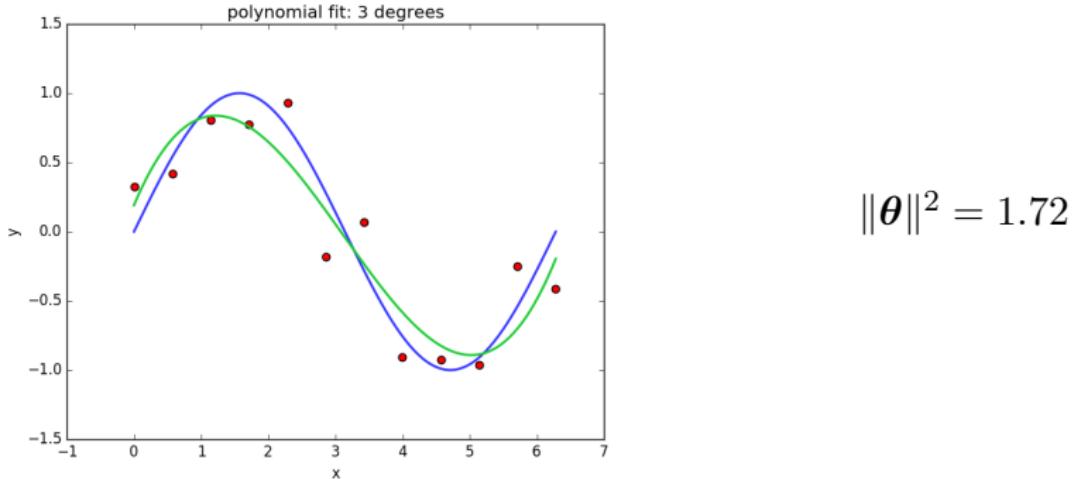
Overfitting and underfitting



$$\|\theta\|^2 = 0.67$$

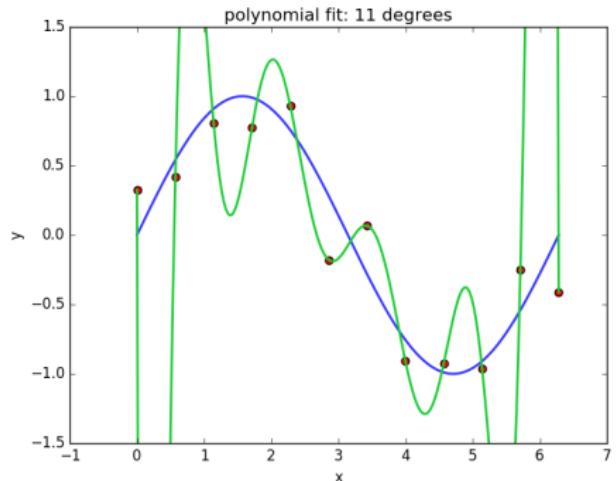
With a polynomial of degree 1, the solution is not capable of modeling the measured data points; we get a poor approximation of the original function. The family of functions we have used was not complex enough to model the true data distribution. We also speak of **underfitting**.

Overfitting and underfitting



For $m = 3$, we obtain a solution that seems to be quite right: it is sufficiently complex to model the true data distribution, but not too complex to model the small variations which are due to noise.

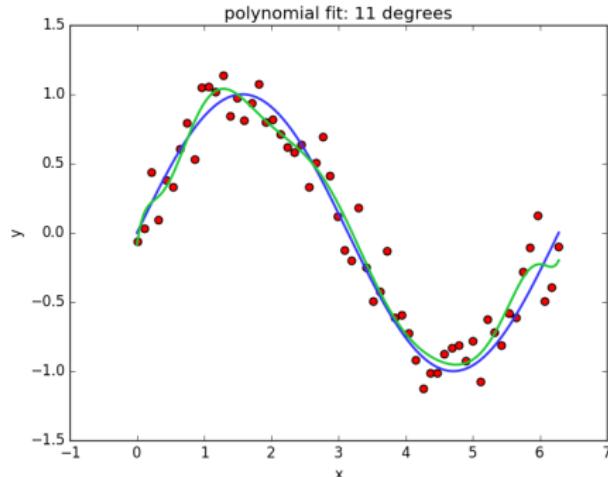
Overfitting and underfitting



$$\|\theta\|^2 \approx 10^7$$

For $m = 11$, we obtain a solution that has zero error (the function passes through every point of the training set). But the coefficients with large absolute values that cancel each other precisely on the training points lead to a highly unstable function. We speak of **overfitting** and **poor generalization**.

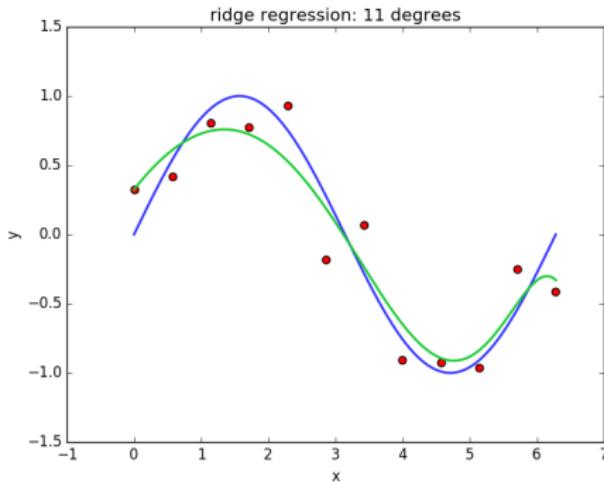
Overfitting and underfitting



$$\|\theta\|^2 = 5647$$

One way of reducing overfitting is to increase the number of samples. Even if the function is complex, it cannot be “too wild”, as it has to find a compromise between many training samples. This however implies the annotation (or measurement) of more samples.

Overfitting and underfitting



$$\|\boldsymbol{\theta}\|^2 = 0.41$$

Another way of preventing overfitting without increasing the number of samples, is to add a penalization term in the optimization procedure. This is also known as **regularization**:

$$L = \sum_{i=1}^N (y_i - \boldsymbol{\theta}^T \phi(x_i))^2 + \lambda \|\boldsymbol{\theta}\|^2 \quad (4)$$

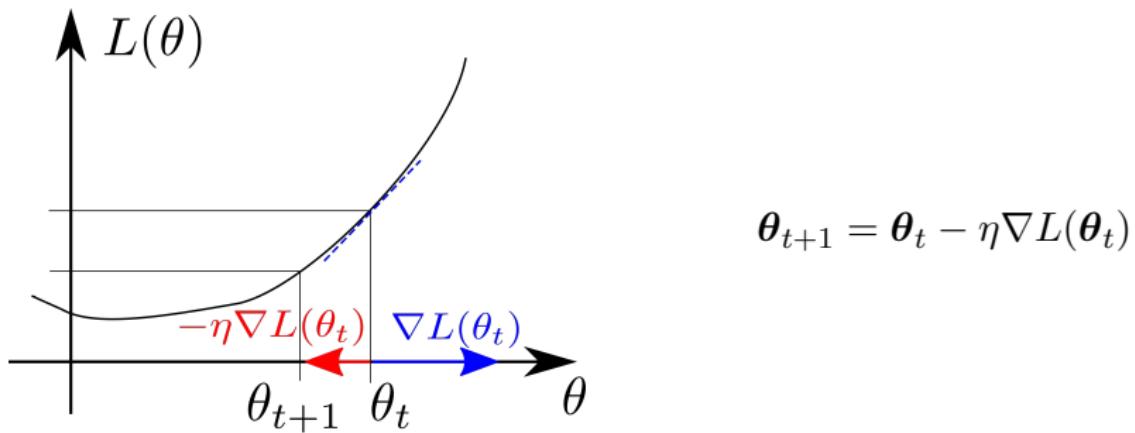
Optimization

- In the general case, **gradient descent** is used to minimize the loss function L :

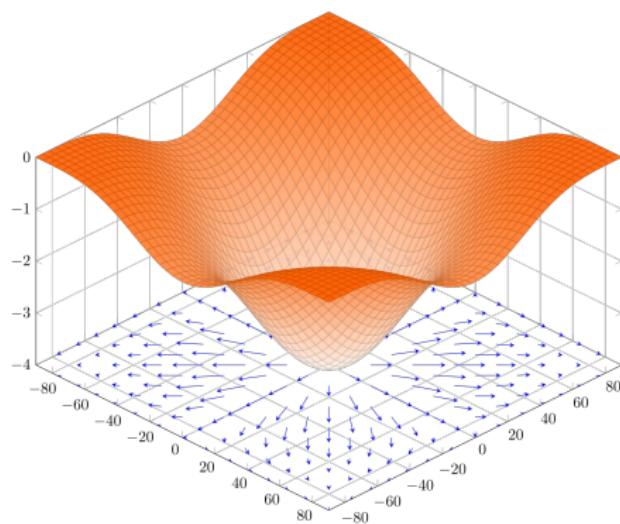
$$L = \sum_{i=1}^N (y_i - f_{\theta}(x_i))^2 \quad (5)$$

- For this to be possible, f_{θ} must be differentiable.

Gradient descent in the scalar case



How to minimize a function?



Definition: gradient

Let L be a derivable function from \mathbb{R}^n into \mathbb{R} . Its gradient ∇L is:

$$\nabla L(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial L}{\partial \theta_1}(\boldsymbol{\theta}) \\ \vdots \\ \frac{\partial L}{\partial \theta_n}(\boldsymbol{\theta}) \end{pmatrix}$$

Credits: By MartinThoma, CC0,
<https://commons.wikimedia.org/>

Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

Computational graph

Definition

A computational graph is a direct acyclic graph such that:

- A node is a mathematical operator
- To each edge is associated a value (scalar, vector, matrix, tensor, ...)
- Each node can compute the values of its output edges from the values of its input edges

Computing a *forward pass* through the graph means choosing its input values, and then progressively computing the values of all edges.

Computational graph example

Computational graph of:

$$\sigma(w_1x + w_2y + b)$$

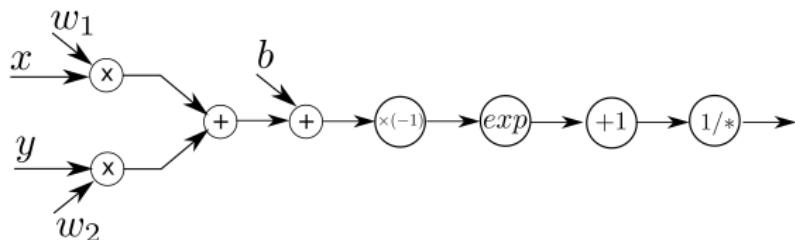
where σ is the sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$

Computational graph example

Computational graph of:

$$\sigma(w_1x + w_2y + b)$$

where σ is the sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$

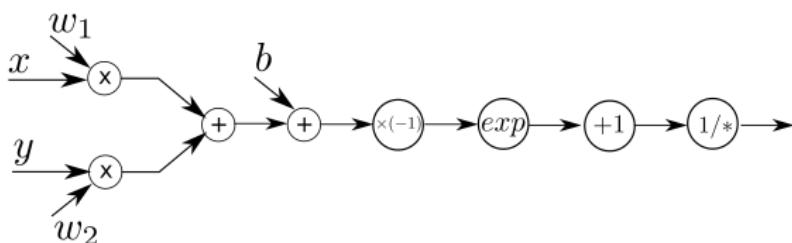


Computational graph example

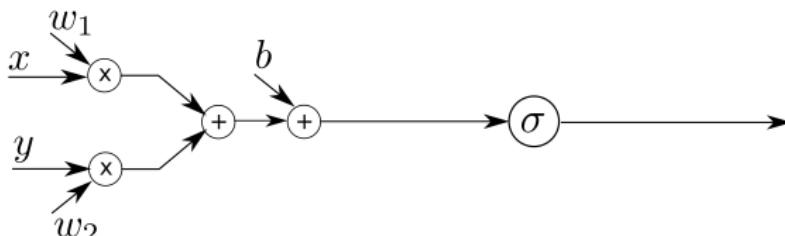
Computational graph of:

$$\sigma(w_1x + w_2y + b)$$

where σ is the sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$



The graph can be represented at different levels of detail:



Gradient descent applied to computational graphs

In the case of computational graphs, the loss L depends on each parameter θ_i via the composition of several simple functions. In order to compute the gradient $\nabla_{\theta} L$ we will make extensive use of the chain rule theorem.

Chain rule theorem

Let f_1 and f_2 be two derivable real functions ($\mathbb{R} \rightarrow \mathbb{R}$). Then for all x in \mathbb{R} : :

$$(f_2 \circ f_1)'(x) = f'_2(f_1(x)) \cdot f'_1(x)$$

Leibniz notation

Let us introduce variables x , y and z :

$$x \xrightarrow{f_1} y \xrightarrow{f_2} z$$

Then:

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$

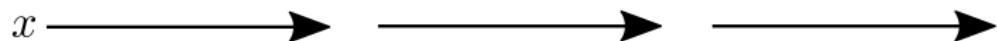
The backpropagation algorithm

- The backpropagation algorithm is used in a computational graph to efficiently compute the partial derivatives of the loss with respect to each parameter of the network.
- One can trace the origins of the method to the sixties
- It was first applied to NN in the eighties
[Werbos, 1982, LeCun, 1985]

The backpropagation algorithm: intuition

- Given a computational graph, the main idea is to compute the local partial derivatives during a forward pass
- Then, during a backward pass, the partial derivatives of the loss with respect to each parameter is computed

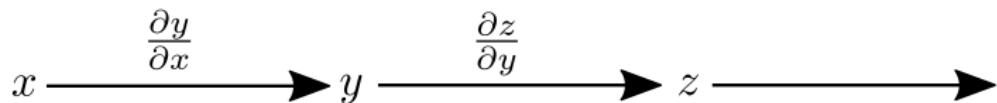
Simple backpropagation example



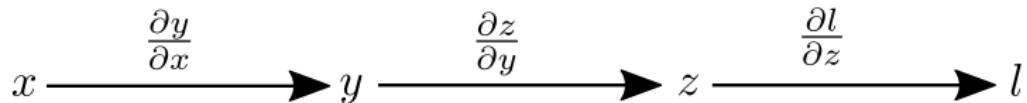
Simple backpropagation example



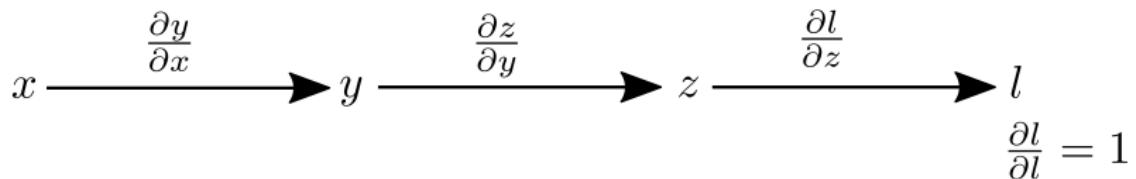
Simple backpropagation example



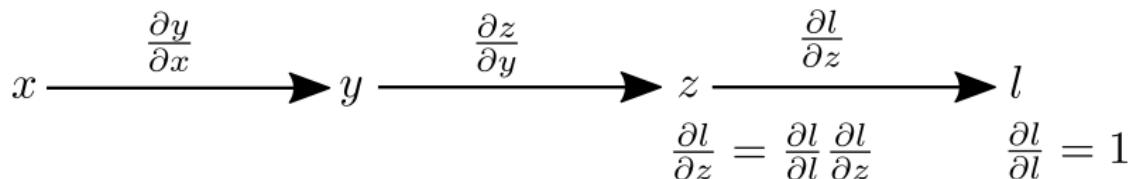
Simple backpropagation example



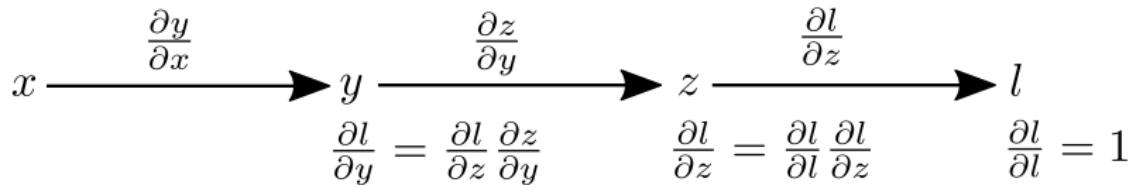
Simple backpropagation example



Simple backpropagation example



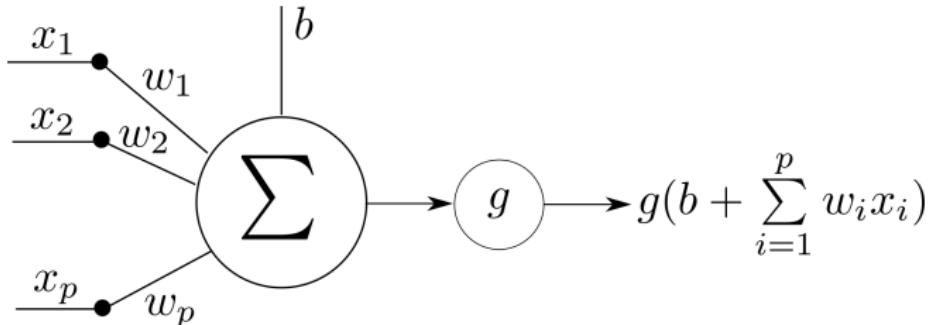
Simple backpropagation example



Simple backpropagation example

$$x \xrightarrow{\frac{\partial y}{\partial x}} y \xrightarrow{\frac{\partial z}{\partial y}} z \xrightarrow{\frac{\partial l}{\partial z}} l$$
$$\frac{\partial l}{\partial x} = \frac{\partial l}{\partial y} \frac{\partial y}{\partial x}$$
$$\frac{\partial l}{\partial y} = \frac{\partial l}{\partial z} \frac{\partial z}{\partial y}$$
$$\frac{\partial l}{\partial z} = \frac{\partial l}{\partial l} \frac{\partial l}{\partial z}$$
$$\frac{\partial l}{\partial l} = 1$$

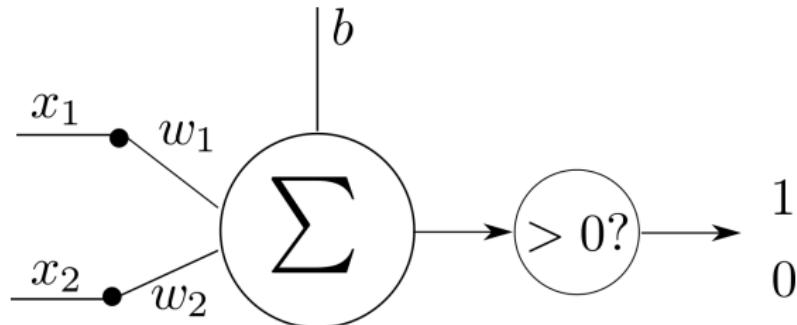
Artificial neuron



Geometric interpretation

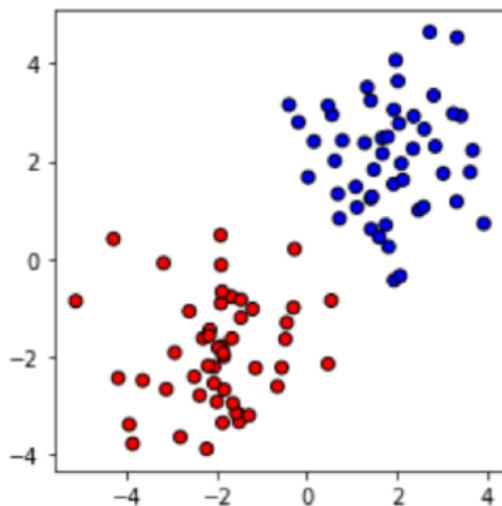
In \mathbb{R}^p , $b + \sum_{i=1}^p w_i x_i = 0$ corresponds to a hyperplane H . For a given point $\mathbf{x} = \{x_1, \dots, x_p\}$, decisions are made according to the side of the hyperplane it belongs to.

Example in 2D

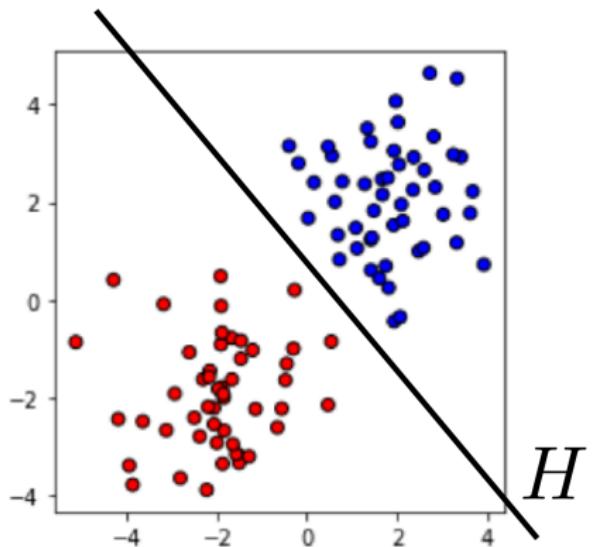


- $p = 2$: 2-dimensional inputs (can be represented on a screen!)
- Activation: binary
- Classification problem

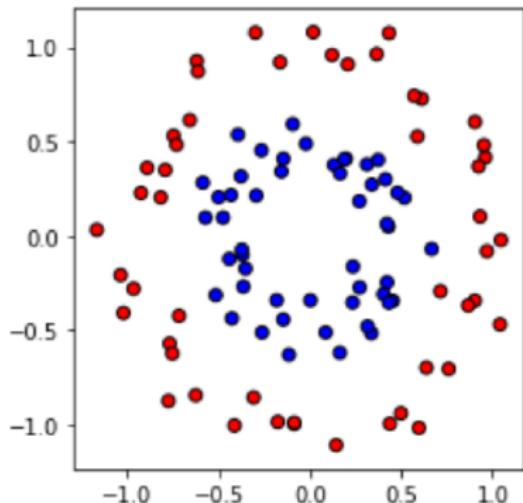
Gaussian clouds



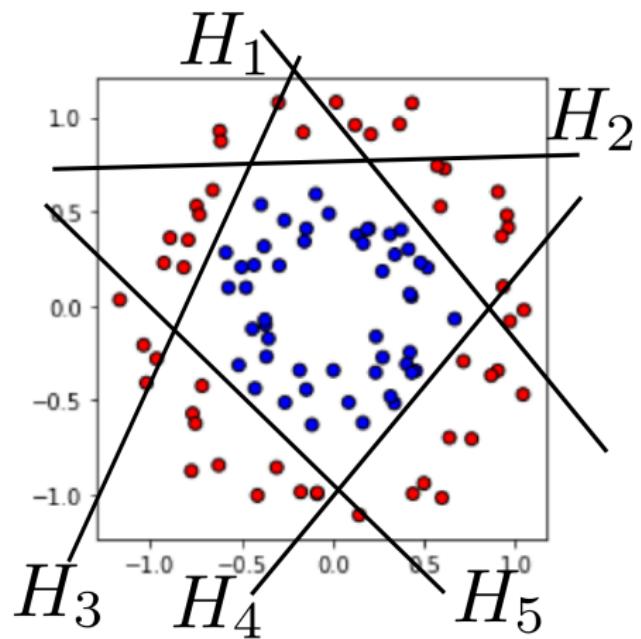
Gaussian clouds



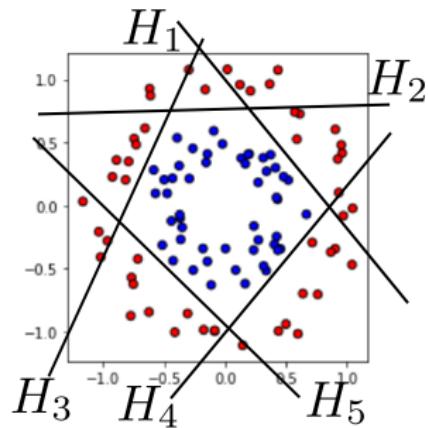
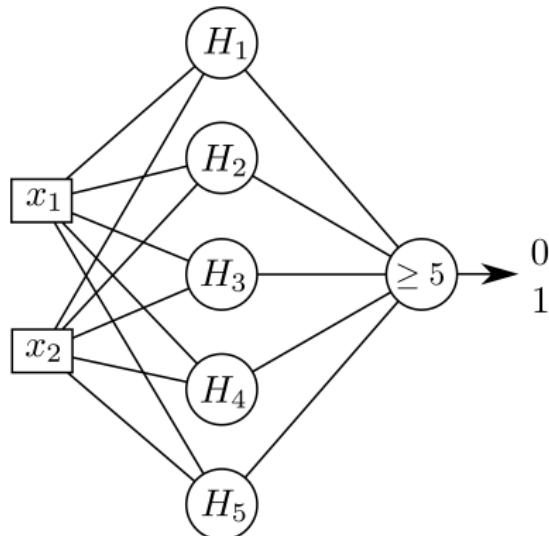
Circles



Circles



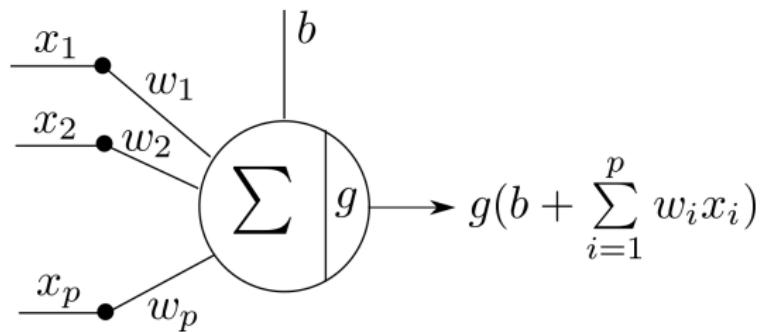
Solution with a simple neural network



Intuition

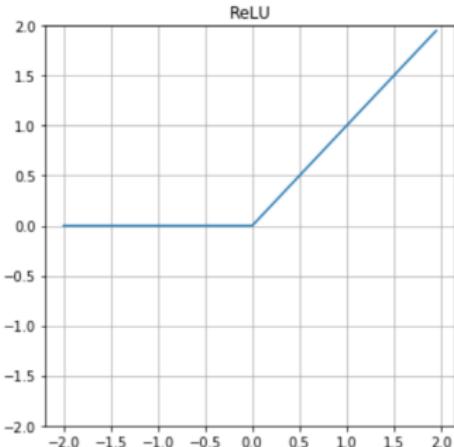
Combining several neurons one can build complex classifiers.

Compact representation



Activation: rectified linear unit (ReLU)

$$g(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

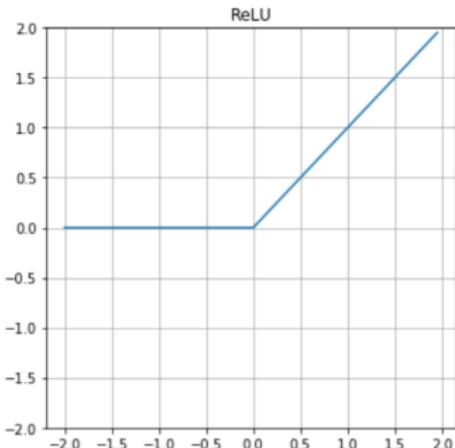


Remarks

- + Usable gradient when activated
- + Fast to compute
- + High abstraction

Activation: rectified linear unit (ReLU)

$$g(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$



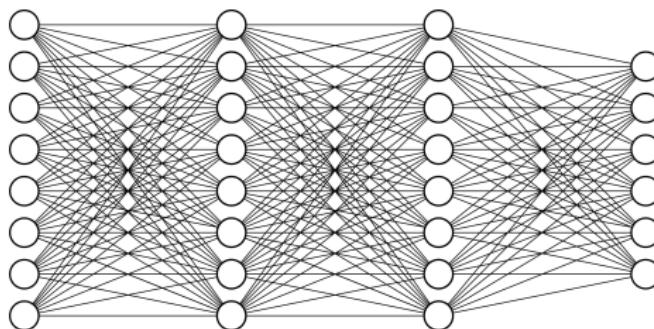
Remarks

- + Usable gradient when activated
- + Fast to compute
- + High abstraction

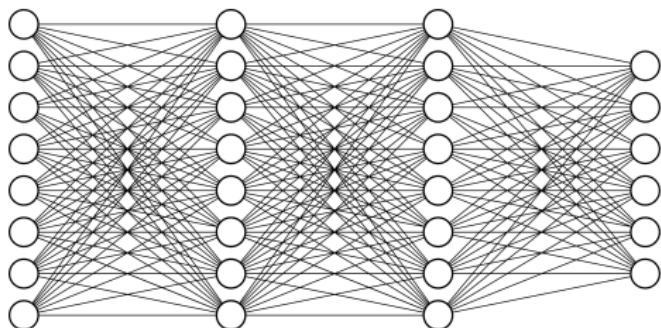
ReLU is the most commonly used activation function.

Multi-layer perceptron (MLP)

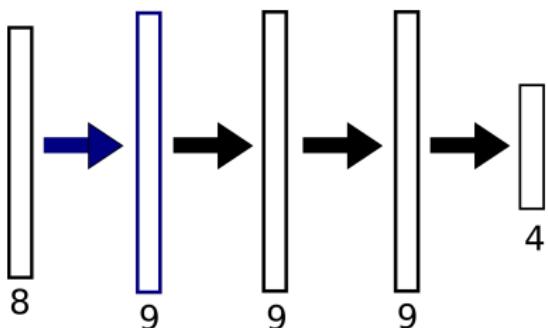
- A layer is said to be fully-connected (FC) if each of its neurons is connected to all the neurons of the previous layer
- If a FC layer contains r neurons, and the previous layer q , then its weights are a 2D dimensional array (a matrix) of size $q \times r$



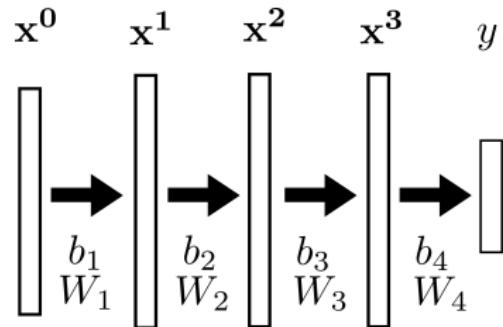
Graphical representation of NNs



- Data is organized into arrays, linked with operators
- A layer corresponds to an operator between arrays.



The equations of fully-connected layers



$$\mathbf{x}^i = g_i(\mathbf{W}_i \mathbf{x}^{i-1} + \mathbf{b}_i), \quad i = 1, 2, 3$$

$$y = g_4(\mathbf{W}_4 \mathbf{x}^4 + \mathbf{b}_4)$$

Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

A picture is worth a thousand words

Definition

- Classically, an image is a matrix of values belonging to $[0, \dots, 255]$ (grey level images) or to $[0, \dots, 255]^3$ (color images).
- More generally, an image is a q -dimensional array of values belonging to R^d .



Grey level values around the left eye of the faun

The role of annotated image databases

Image databases including *annotations* (typically some kind of high level information) are essential to the development of *supervised* machine learning methods for image analysis.

Annotations

- Image class
- Measure(s) obtained from the image
- Position of objects within the image
- Segmentation

MNIST database [Lecun et al., 1998]

- The Modified National Institute of Standards and Technology (MNIST) database contains 60 000 training images of hand-written digits, and 10,000 test images.
- Image size: 28×28
- It has been used since 1998
- Human performance on a similar database (NIST) is reported to be around 1.5% error [Simard et al., 1993]
- Best methods, based on convolutional neural networks, give around 0.21% test error.

MNIST database

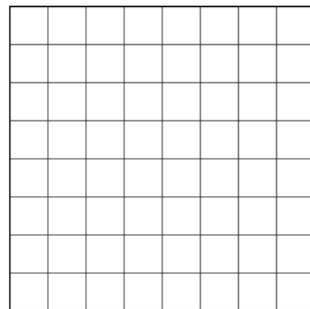


Credits: Images from MNIST assembled
by Josef Stepan (licensed under CC
BY-SA 4.0)

Layers representation

For illustration purposes, in the following slides images and filters will be displayed as rows of neurons – these can be seen as 1D arrays or as sections of 2D arrays.

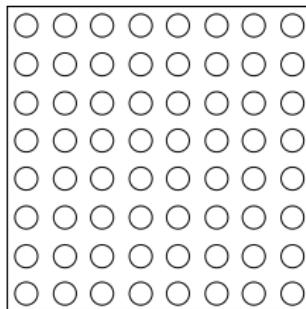
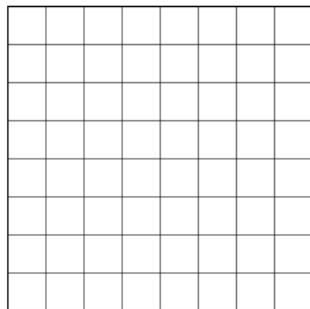
We represent some connections between neurons. Each such connection is associated to a weight. The bias are not represented, to avoid clutter, but must not be forgotten.



Layers representation

For illustration purposes, in the following slides images and filters will be displayed as rows of neurons – these can be seen as 1D arrays or as sections of 2D arrays.

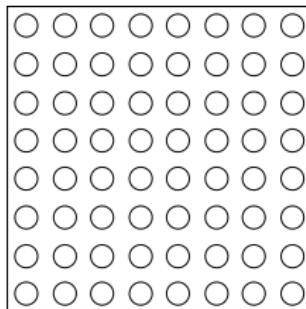
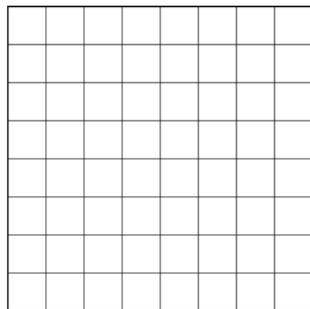
We represent some connections between neurons. Each such connection is associated to a weight. The bias are not represented, to avoid clutter, but must not be forgotten.



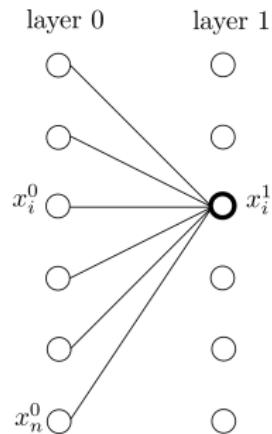
Layers representation

For illustration purposes, in the following slides images and filters will be displayed as rows of neurons – these can be seen as 1D arrays or as sections of 2D arrays.

We represent some connections between neurons. Each such connection is associated to a weight. The bias are not represented, to avoid clutter, but must not be forgotten.

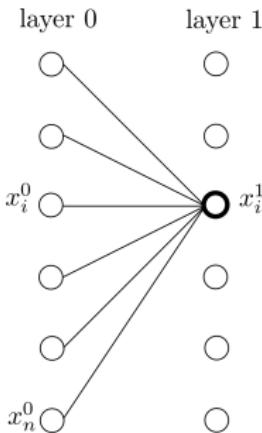


Towards convolutional layers

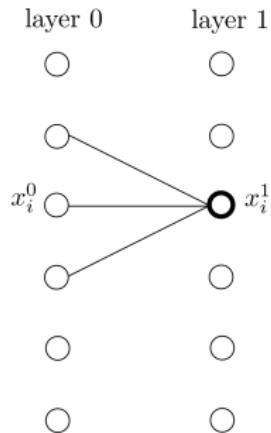


Fully connected layer:
 $n(n + 1)$ weights

Towards convolutional layers

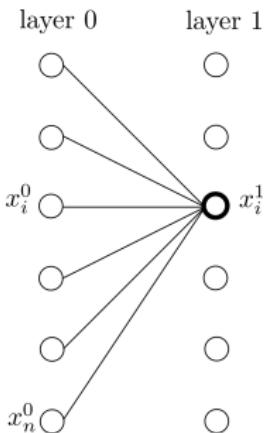


Fully connected layer:
 $n(n + 1)$ weights

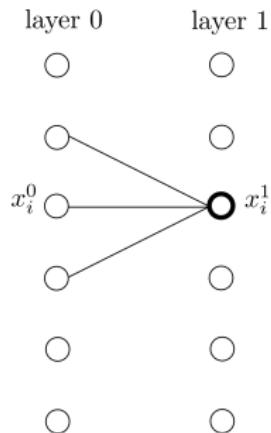


Locally conn. layer:
 $n(s + 1)$ weights

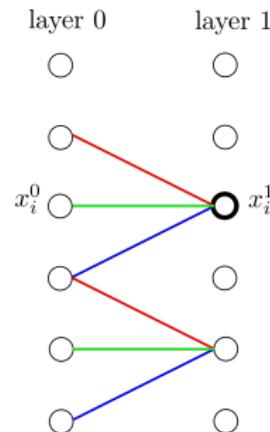
Towards convolutional layers



Fully connected layer:
 $n(n + 1)$ weights



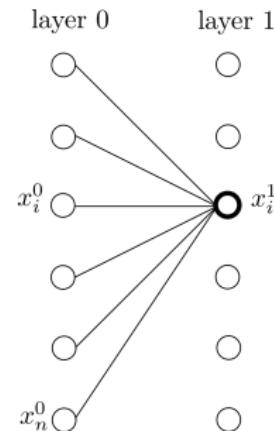
Locally conn. layer:
 $n(s + 1)$ weights



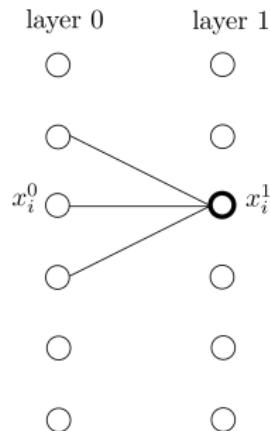
Weight replication: $s + 1$ weights.
Convolutional layer.

Towards convolutional layers: some figures

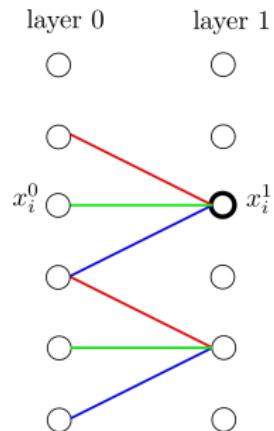
- 3×3 convolutions: $s = 9$
- Toy image: $n = 28 \times 28 = 784$
- Typical image: $n = 1000 \times 1000 = 10^6$



Fully connected layer:
 $n(n + 1)$ weights
 $\approx 6.10^5$
 $\approx 10^{12}$



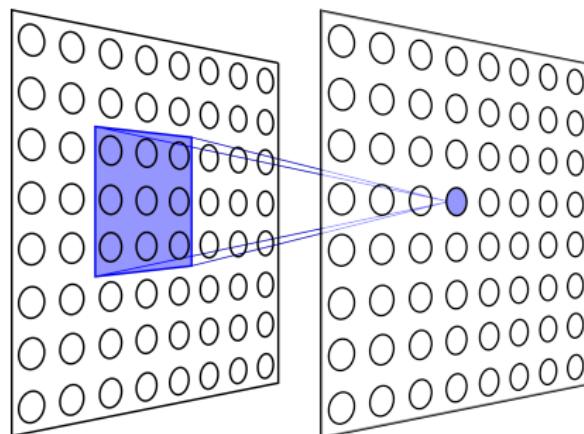
Locally conn. layer:
 $n(s + 1)$ weights
7840
 10^7



Weight replication: $s + 1$ weights.
10
 10^6

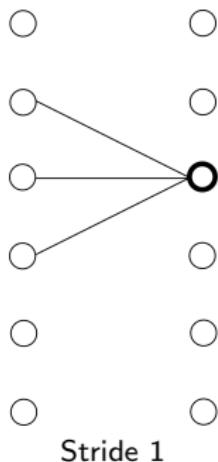
Convolutional layer illustration in 2D

- Illustration of a convolution of size 3×3



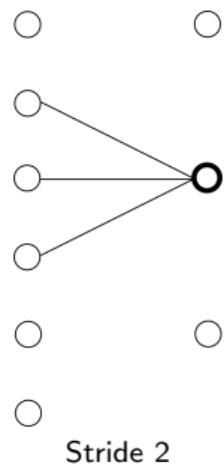
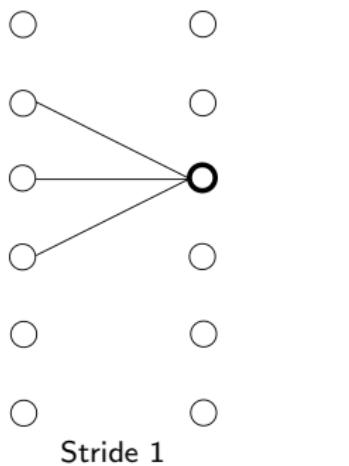
Stride

A convolutional layer can at the same time downsample the image by applying a sampling step, or *stride*.



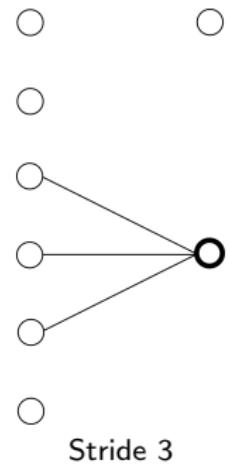
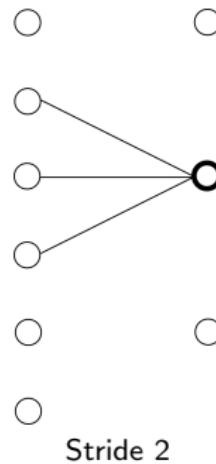
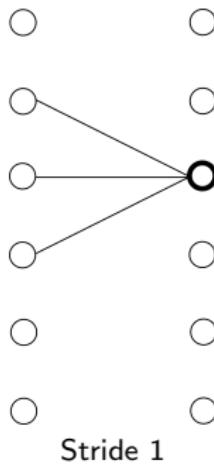
Stride

A convolutional layer can at the same time downsample the image by applying a sampling step, or *stride*.

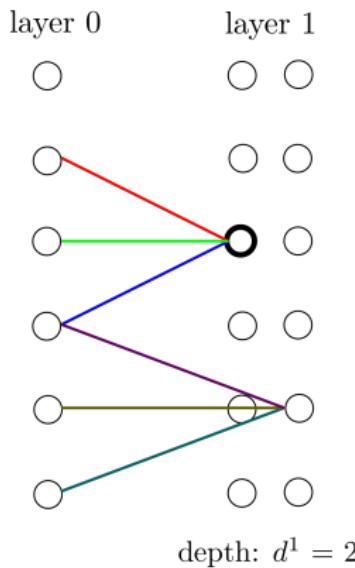


Stride

A convolutional layer can at the same time downsample the image by applying a sampling step, or *stride*.



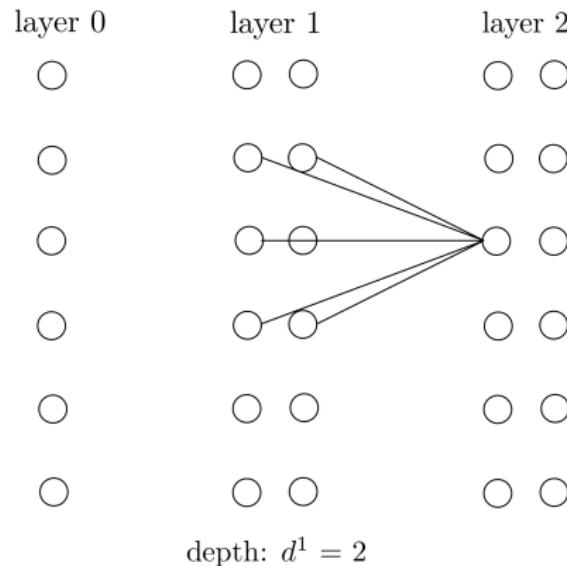
Several filters in the same convolutional layer



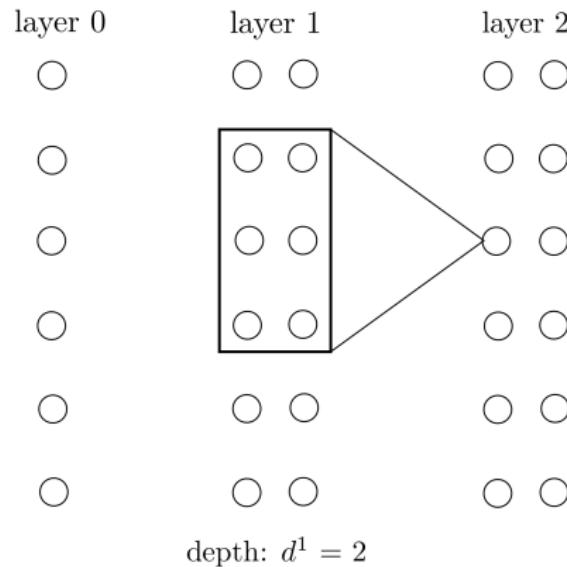
Note on vocabulary

The depth of a layer is often called the **number of filters**.

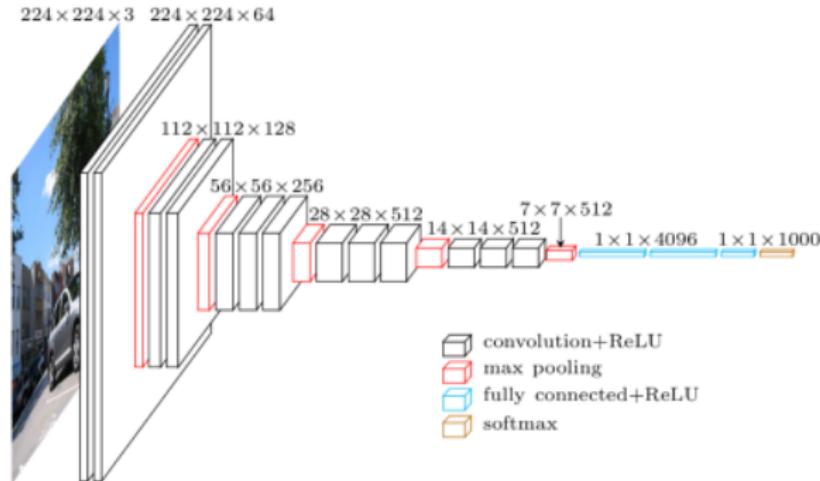
Several filters in the same convolutional layer



Several filters in the same convolutional layer

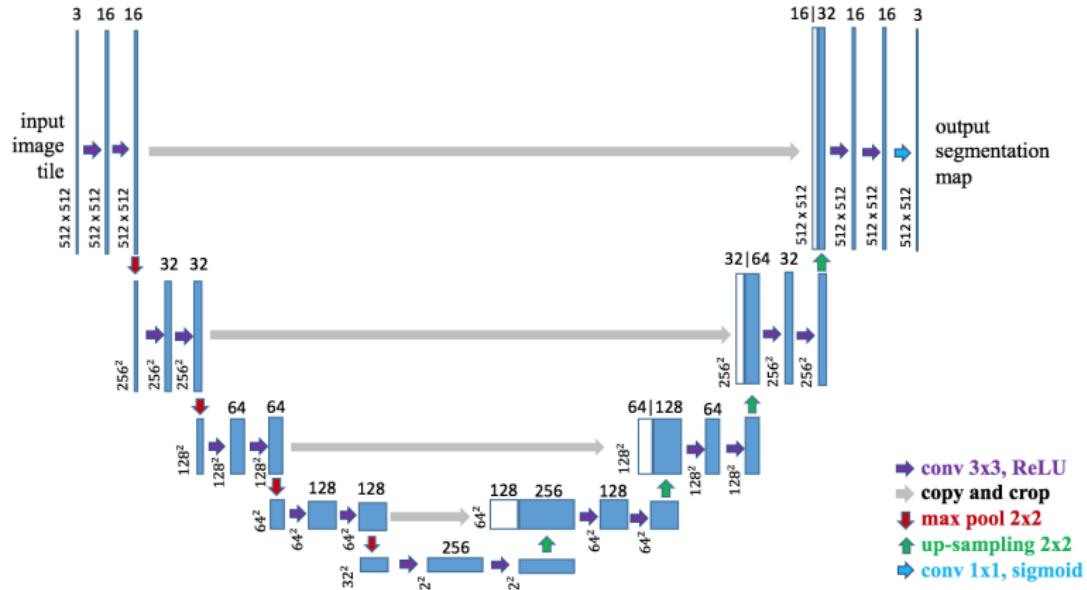


Architecture example: VGG



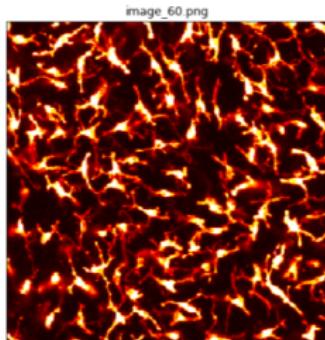
Credits: VGG16 (From
<https://www.cs.toronto.edu/~frossard/post/>)

U-Net architecture [Ronneberger et al., 2015]

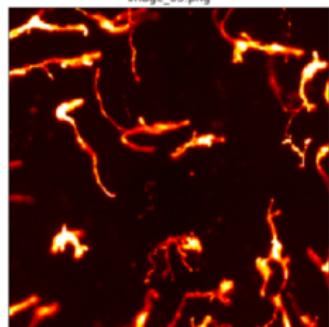


Example: counting cells

image

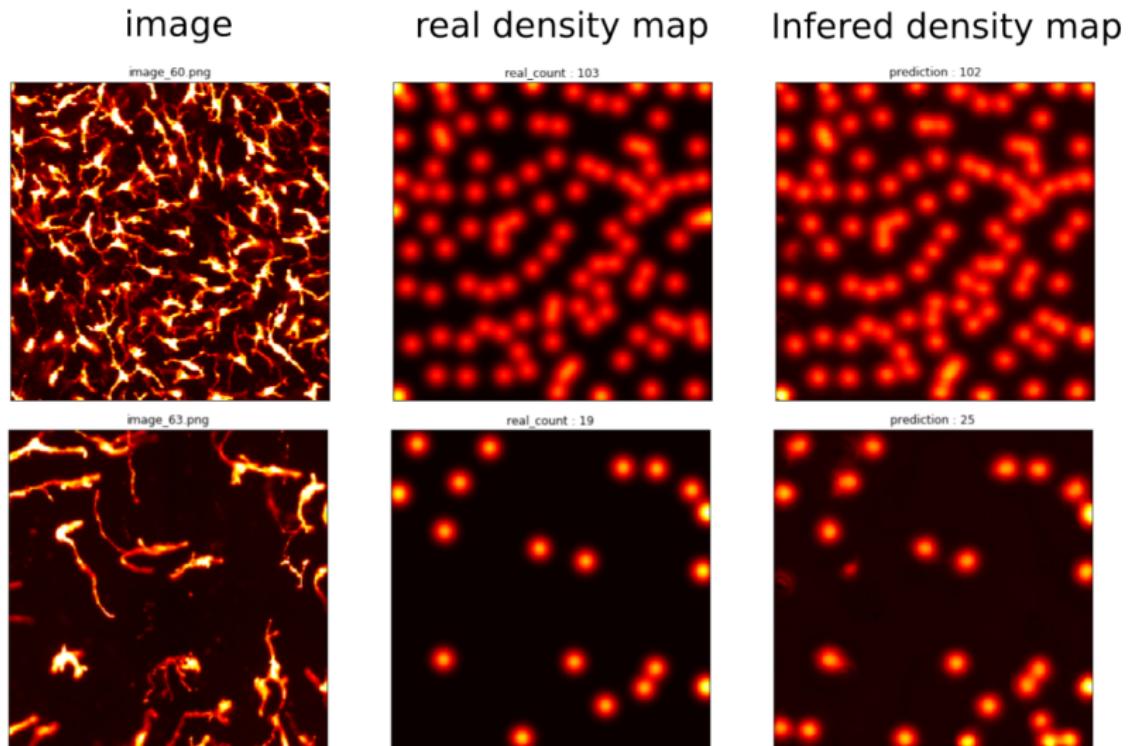


image_63.png



Credits: Tristan Lazard, master thesis. In collaboration with L'Oréal.

Counting cells



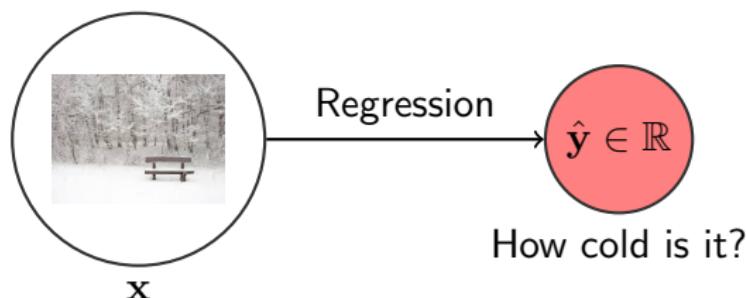
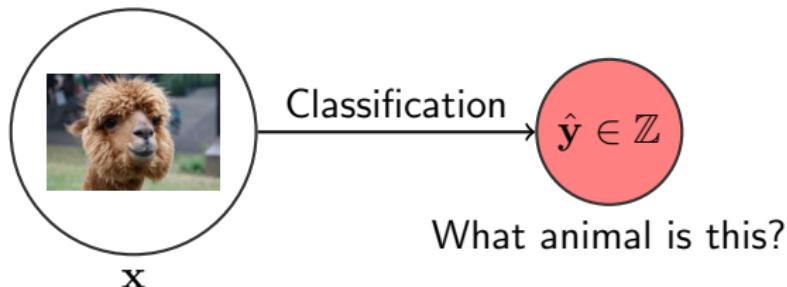
Credits: Tristan Lazard, master thesis. In collaboration with L'Oréal.

Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

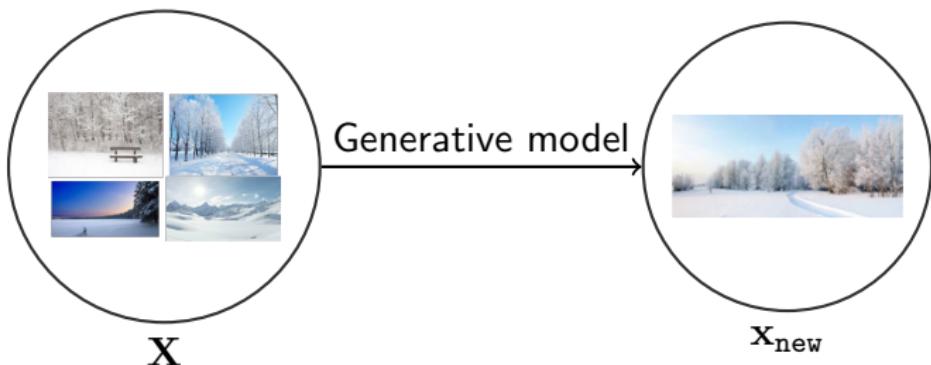
Supervised Learning

Given a labeled dataset (\mathbf{X}, \mathbf{Y}) , we would like to learn a mapping from data space to label space.

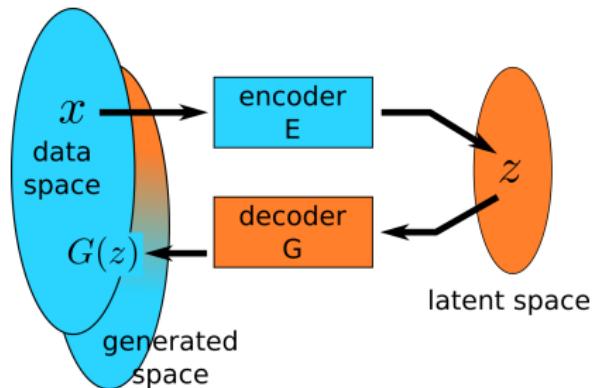


Unsupervised learning: Generative Models

Given an unlabeled dataset (\mathbf{X}), we would like to learn: How to generate a new observation from the same distribution (unknown) of dataset?

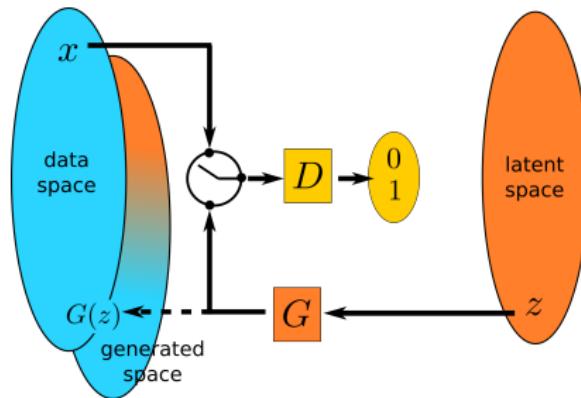


Autoencoders



- Encoder: E ; decoder: G ; autoencoder: $G \circ E$
- In most applications, the latent space is “smaller” than the data space.
- Objective: \hat{x} , i.e. $G \circ E(x)$, “close” to x
- When dealing with images, modern autoencoders use convolutional neural networks

Generative adversarial networks [Goodfellow et al., 2014]



- The **discriminator** D is optimized so that it correctly classifies images as real (1) or fake (0)
- The decoder or **generator** G is optimized so that the produced images are classified as real by the discriminator

Value function

$$V(G, D) = \mathbb{E}_{p_{\mathbf{x}}}(\log(D(\mathbf{x}))) + \mathbb{E}_{p_z(z)}(\log(1 - D(G(z))))$$

Which face is real?

Contents

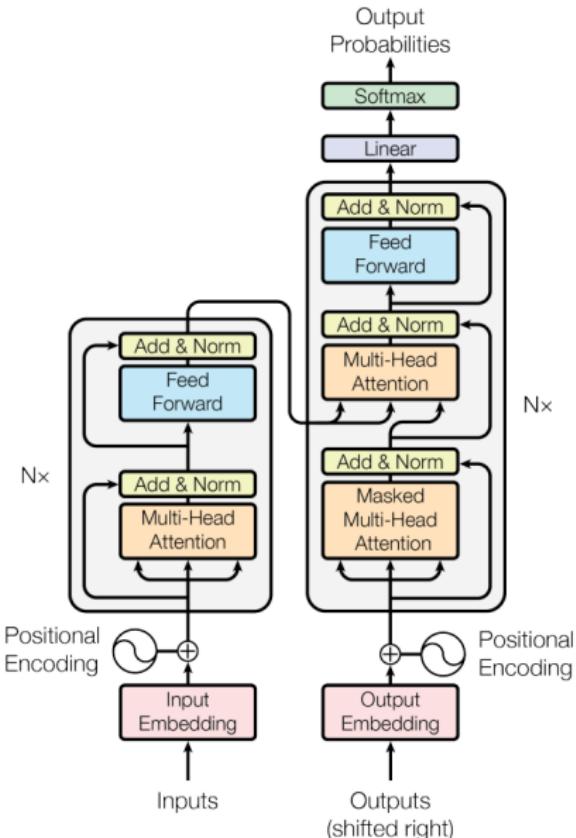
- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

The rise of transformers

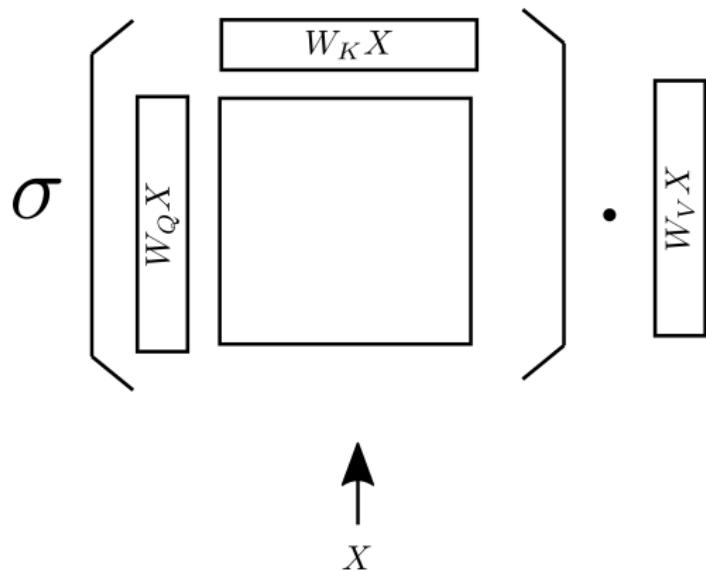
The paper that started it all

Vaswani et al., Attention is all you need, Neurips 2017.

This architecture was developed for text processing.



Self-attention



Example: LLaMA-I [Touvron et al., 2023]

write a conversation between the sun and pluto

Sun: Hey Pluto, what's up?

Pluto: Not much. Just hanging out here at the edge of the solar system.

Sun: Yeah, I've been meaning to talk to you about that. You're not a planet anymore.

Pluto: What? Come on, man. That's not fair.

Sun: I'm sorry, but it's true. You just don't meet the criteria anymore.

Pluto: This is bulls**t! I've been a planet for over 70 years!

Sun: Things change, Pluto. You've just got to accept it.

Pluto: F**k you, Sun! F**k you and your stupid solar system!

Contents

- 1 Introduction
- 2 Machine learning
- 3 Artificial neural networks
- 4 Learning image transformations
- 5 Autoencoders and generative adversarial networks
- 6 Transformers
- 7 Conclusion

Images AND text



Strengths

- Modelling power

Strengths

- Modelling power
- Increased productivity

Strengths

- Modelling power
- Increased productivity
- Relative ease of use (thanks to open source)

Weaknesses

- Supervised models need huge amounts of data

Weaknesses

- Supervised models need huge amounts of data
- Learning needs important resources

Weaknesses

- Supervised models need huge amounts of data
- Learning needs important resources
- Interpretation is often difficult

Weaknesses

- Supervised models need huge amounts of data
- Learning needs important resources
- Interpretation is often difficult
- Their effectiveness is not well understood

Weaknesses

- Supervised models need huge amounts of data
- Learning needs important resources
- Interpretation is often difficult
- Their effectiveness is not well understood
- Not all transformations can be learnt with artificial neural networks

Opportunities

- Improved performance of existing applications

Opportunities

- Improved performance of existing applications
- New applications

Threats

- Biased models

Threats

- Biased models
- Unemployment increase in some sectors

Threats

- Biased models
- Unemployment increase in some sectors
- Unaware users

Threats

- Biased models
- Unemployment increase in some sectors
- Unaware users
- Malevolent use: Fake images and sound; new weapons; etc.

Threats

- Biased models
- Unemployment increase in some sectors
- Unaware users
- Malevolent use: Fake images and sound; new weapons; etc.
- Leakage of private data

Conclusion par ChatGPT

ET

Ecris-moi un texte d'environ 100 mots pour conclure une présentation sur les forces, faiblesses, opportunités et menaces de l'apprentissage profond



En conclusion, l'apprentissage profond présente un potentiel extraordinaire dans de nombreux domaines, mais il n'est pas sans ses défis. Ses forces résident dans sa capacité à traiter de vastes quantités de données et à extraire des modèles complexes, ouvrant ainsi de nouvelles perspectives en matière de reconnaissance d'images, de traduction automatique et de diagnostics médicaux, entre autres. Cependant, ses faiblesses incluent sa dépendance à des ensembles de données de qualité et à des ressources informatiques considérables, ce qui peut entraîner des préjugés et des problèmes d'éthique. Les opportunités sont énormes, avec la possibilité de révolutionner de nombreux secteurs, mais il est crucial de rester conscient des menaces potentielles, notamment la confidentialité des données et les implications socio-économiques. Pour maximiser les avantages de l'apprentissage profond, une approche équilibrée, éthique et responsable est essentielle.

References |

- [Bahdanau et al., 2015] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*. arXiv:1409.0473 [cs, stat].
- [Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.
- [Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc.
- [Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.
- [LeCun, 1985] LeCun, Y. (1985). Une procedure d'apprentissage pour reseau a seuil asymmetrique (A learning scheme for asymmetric threshold networks). In *proceedings of Cognitiva 85*.
- [Lecun et al., 1998] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

References II

- [Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, number 9351 in Lecture Notes in Computer Science, pages 234–241. Springer International Publishing.
- [Simard et al., 1993] Simard, P., LeCun, Y., and Denker, J. S. (1993). Efficient pattern recognition using a new transformation distance. In *Advances in neural information processing systems*, pages 50–58.
- [Touvron et al., 2023] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971 [cs].
- [Werbos, 1982] Werbos, P. J. (1982). Applications of advances in nonlinear sensitivity analysis. In Drenick, R. F. and Kozin, F., editors, *System Modeling and Optimization*, Lecture Notes in Control and Information Sciences, pages 762–770, Berlin, Heidelberg. Springer.