Réseau de neurones convolutionnel

Introduction au deep learning

Théo Lopes Quintas

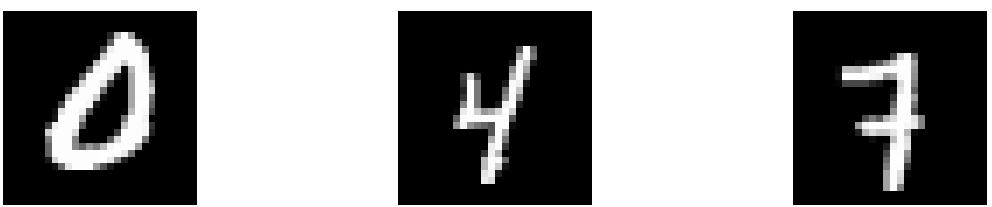
BPCE Payment Services,
Université Paris Dauphine

2023-2026

Introduction

Dataset MNIST

Le dataset MNIST [1] [LeCun et al., 2010] est composé de 70 000 images de chiffres manuscrit : 60 000 pour l'entraînement et 10 000 pour le test.



(a) $y = 0$ **(b)** $y = 4$ **(c)** $y = 7$

Figure – Exemples d'images du dataset MNIST

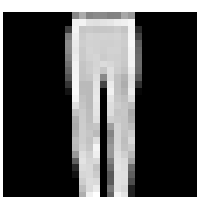
La première utilisation de ce dataset est dans un papier de recherche portant sur les SVM en 1998, développé au laboratoires Bell Labs. Les meilleurs résultats sont obtenus plus tard par les réseaux convolutionnel.

1. *Modified National Institute of Standards and Technology database* car issu du dataset NIST.

Introduction

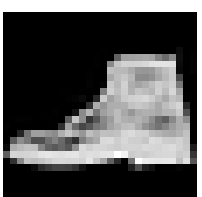
Dataset Fashion MNIST

Le dataset Fashion MNIST[Xiao et al., 2017] est proposé en 2017 pour poser un plus grand challenge que MNIST et servir également de benchmark. Il est composé de 70 000 images de vêtements issus de Zalando. Le format du dataset est identique à celui d'MNIST.



(a) Pantalon : $y = 1$

(b) Veste : $y = 4$

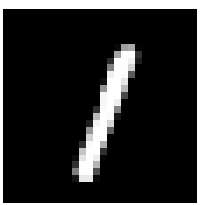


(c) Bottine : $y = 9$

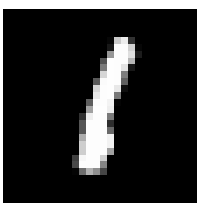
Figure – Exemples d'images du dataset Fashion MNIST

Introduction

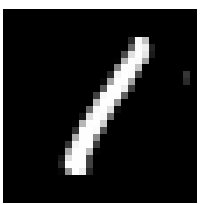
Comment apprendre à partir d'une image ?



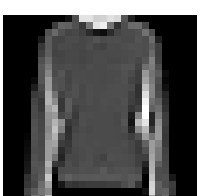
(a) $y = 1$



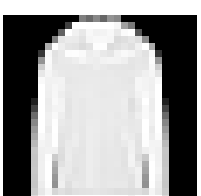
(b) $y = 1$



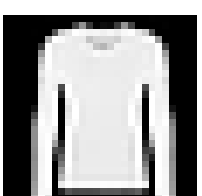
(c) $y = 1$



(d) Pull : $y = 2$



(e) Veste : $y = 4$



(f) Pull : $y = 2$

Figure – Exemples d'image du dataset MNIST et Fashion MNIST

Couche de convolution

Image et filtre

On considère une image de taille $25 \times 25 \times 3$: c'est une image carré de 25 pixels avec couleur RGB. On considère un filtre de taille $5 \times 5 \times 3$: c'est une matrice de nombre qui sont les poids que l'on va apprendre.

Il y a donc $5 \times 5 + 1 = 26$ paramètres car il y a un terme de biais qui est ajouté. Si l'on considère une couche dense, un neurone aurait eu 626 paramètres pour travailler avec cette image en entrée !

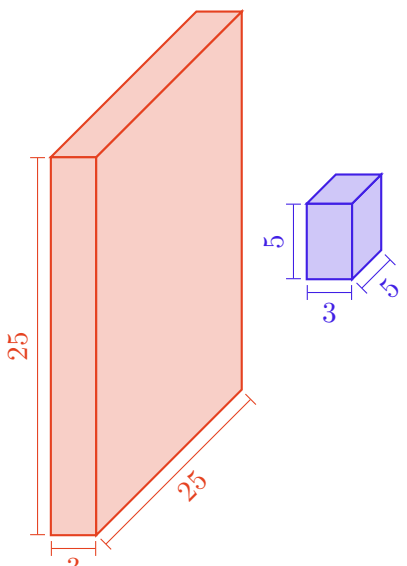
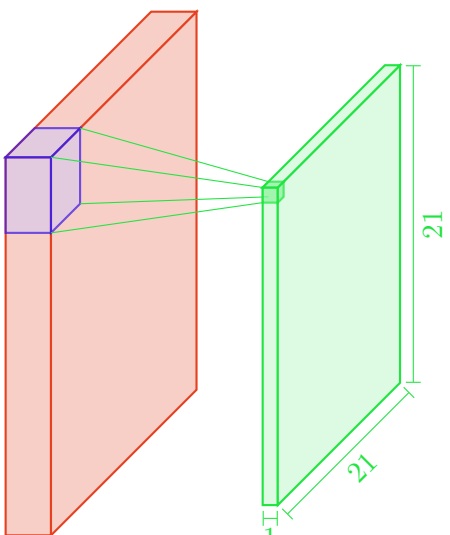


Figure – Exemple d'une image et d'un filtre

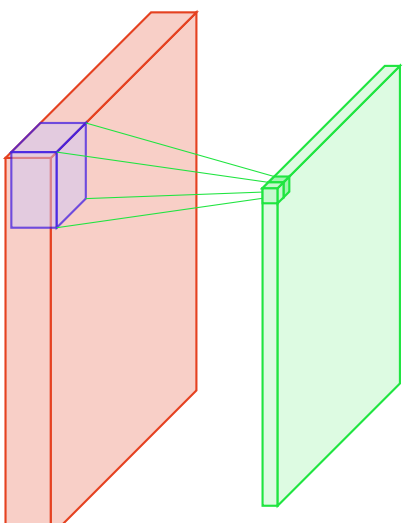
Couche de convolution

Image et filtre

Le produit scalaire entre une sous-image, de la même taille que le filtre, et le filtre renvoie un nombre. En faisait varier la fenêtre qui génère la sous-image, on construit une nouvelle matrice :



(a) Première valeur



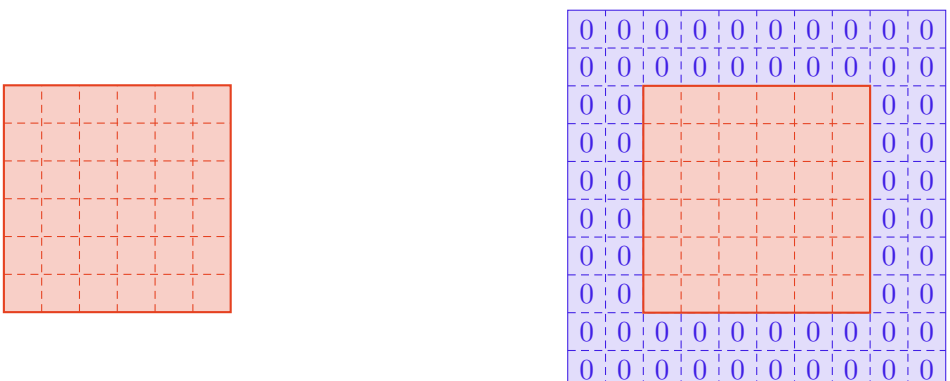
(b) Deuxième valeur

Figure – Calcul des valeurs dans la nouvelle matrice

Couche de convolution

Zero-padding

Les arrêtes ne seront pas forcément parfaitement prise en compte avec le processus actuel. On introduit alors le zero-padding :



(a) Image de départ

(b) Image avec padding

Figure – Illustration du padding avec un zero-padding de 2

Réaliser une couche de convolution avec un filtre de taille 3×3 sur une image avec un zero-padding de 2 ne changera pas la taille de l'image en sortie.

Couche de convolution

Dimension de la matrice créée

La fenêtre se déplace d'une case dans l'exemple précédent : on dit que l'on est de *stride* 1. Si on se déplace de n cases, alors on est de *stride* n. Dans l'exemple, nous sommes passés d'une image de taille $(25, 25)$ à une matrice de taille $(21, 21)$. On peut calculer ses dimensions comme :

$$\text{width} = \left\lfloor \frac{\text{width} + 2 \times \text{pad} - \text{filter size}}{\text{stride}} \right\rfloor + 1$$

$$\text{height} = \left\lfloor \frac{\text{height} + 2 \times \text{pad} - \text{filter size}}{\text{stride}} \right\rfloor + 1$$

Chaque matrice ainsi créé passe ensuite dans une fonction d'activation classique comme ReLU par exemple.

Couche de convolution

Exemple

On souhaite réaliser une étape d'une couche de convolution avec un filtre 2×2 et de stride 1 :

-1	2
0	1

2	-1	-1	3
-2	1	2	-1
0	0	2	3
2	-1	-3	2

(a) Matrice de départ

(b) Convolution

(c) Après ReLU

Figure – Résultat d'une couche de convolution avec un filtre 2×2 avec stride de 1, sans biais

Couche de convolution

Exemple - Correction

On souhaite réaliser une étape d'une couche de convolution avec un filtre 2×2 et de stride 1 :

-1	2
0	1

2	-1	-1	3
-2	1	2	-1
0	0	2	3
2	-1	-3	2

(a) Matrice de départ

-3	1	6
4	5	-1
-1	1	6

(b) Convolution

0	1	6
4	5	0
0	1	6

(c) Après ReLU

Figure – Résultat d'une couche de convolution avec un filtre 2×2 avec stride de 1, sans biais

Couche de convolution

Extraction de features

En utilisant plusieurs filtres, on obtient plusieurs nouvelles matrices. Assemblées, elles forment une nouvelle *image* que l'on transmet à la prochaine couche de convolution !

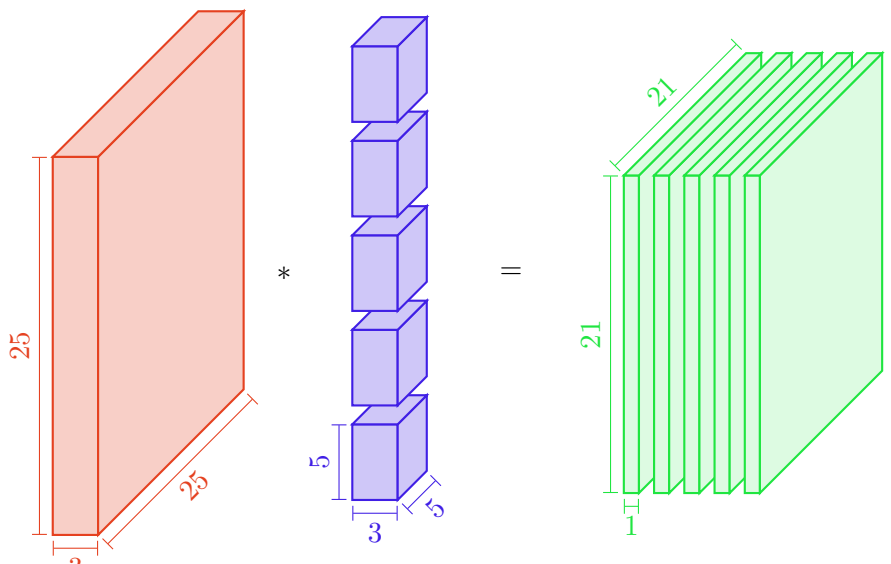


Figure – Formation d'une nouvelle *image* avec une couche de convolution

Couche de convolution

Extraction de features

Une architecture classique[2] des réseaux de neurones convolutionnels, en reprenant l'exemple précédent, ressemble à :

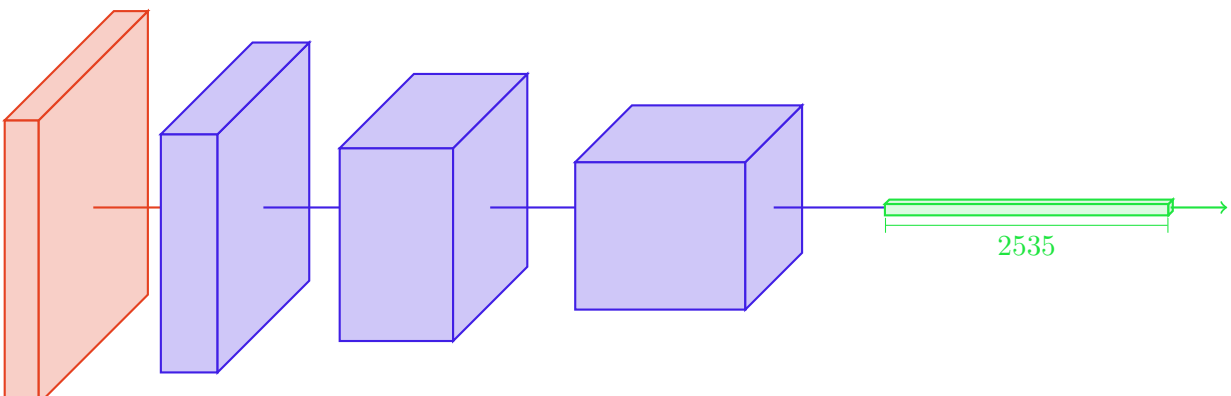


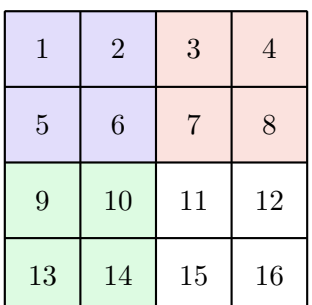
Figure – Exemple d'architecture d'un réseau convolutionnel

On commence par une matrice image qui après l'application de 5, 10 puis 15 filtres est *aplati* pour former un vecteur qui servira d'input à un réseau de neurone dense.

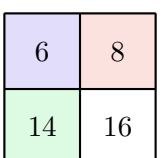
2. Pour le moment...

Couche de pooling

La dimension des images peut parfois être très importante. Dans l'objectif de réduire la dimension sans pour autant perdre trop d'information, la couche de pooling permet de réduire drastiquement la taille d'une image.



(a) Input



(b) Après couche de pooling

Figure – Couche de max-pooling 2×2 avec stride de 2

La couche de pooling la plus communément utilisée est *max-pooling* qui consiste à prendre la valeur maximale de la sous-image. Avant la couche *average-pooling* était préférée.

LeNet 5 - 1998

> *The main message of this paper is that better pattern recognition systems can be built by relying more on automatic learning, and less on hand-designed heyristics. This is made possible by recent progress in machine learning and computer technology.*
> — Yann Le Cun, Léon Bottou, Yoshua Bengio, Patrick Haffner (1998)

L'article *Gradient-based learning applied to document recognition* [LeCun et al., 1998] est l'aboutissement de 9 ans de travail avec le premier modèle LeNet publié en 1989. C'est la première fois qu'un réseau de neurone avec plusieurs couches cachées est présenté. C'est également le développement des couches de convolution et de pooling [3].

Modèle	Année	Paramètres	Taux d'erreur (%)
LeNet	1989	2600	1.70
LeNet-4		17000	1.10
LeNet-5	1998	60000	0.95

Table – Performance des modèles LeNet pour la classification d'MNIST

3. Initialement *average* mais remplacé dans certaine implémentation moderne par du *max*

Architecture célèbre

LeNet 5 - 1998

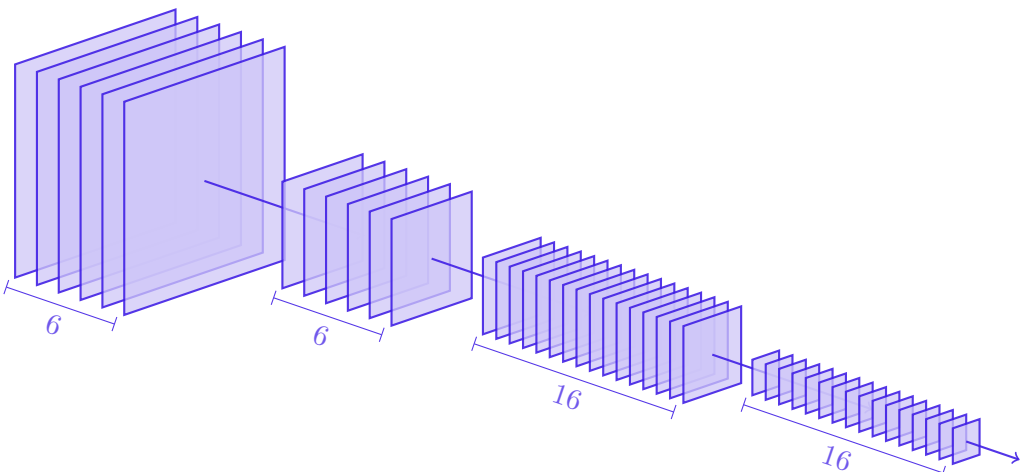


Figure – Partie convolutionnelle de LeNet-5 avec comme input une image 28×28

L'architecture se poursuit avec un réseau dense de 120, 84 puis 10 neurones.

Architecture célèbre

AlexNet

ImageNet est un dataset de référence qui contient des millions d'images labellisées de manière participative. Ce dataset a été utilisé pour de nombreux challenges de 2010 à 2017. AlexNet [Krizhevsky et al., 2012] est le réseau vainqueur de la compétition de 2012 en atteignant une performance de presque 11 points inférieure au second !

> *Suddenly people started to pay attention, not just within the AI community but across the technology industry as a whole.*
>
> — The Economist - *From not working to neural networking* (2016)

L'architecture générale est similaire à celle de LeNet-5 : c'est une succession de couche de convolution et de pooling avec différents hyper paramètres mais on peut noter les différences suivantes :

- **Taille** : AlexNet contient environs 60 millions de paramètres contre environs 60 milles pour LeNet-5. Pour réussir à apprendre l'ensemble de ces paramètres, c'est la première fois que l'on utilise un GPU
- **Fonction d'activation** : c'est ReLU qui est retenu pour ce réseau, contrairement à sigmoid
- **Max Pooling** : l'average pooling est abandonné au profit du max pooling
- **Dropout** [4] : une technique de régularisation que l'on verra à la prochaine séance

4. Technique introduite en 2012 [Hinton et al., 2012]

Architecture célèbre

Inception v1

Deux ans après AlexNet, c'est Google qui propose un nouvel modèle baptisé Inception [Szegedy et al., 2015]. Cette version du modèle est vainqueur de la même compétition ImageNet en 2014 [5]. A nouveau, les couches de convolutions et de pooling sont très présente, mais agencées différemment. On peut noter quelques points de comparaisons :

- **Taille** : Inception v1 contient environs 6.9 millions de paramètres, soit presque 9 fois moins qu'AlexNet
- **Profondeur** : Inception est un réseau de neurone avec 22 couches, alors qu'AlexNet en avaient 8 et LeNet-5 7.
- **Échelles** : des couches de convolutions avec différentes tailles sont utilisées en parallèle
- **Classifier auxiliaire** : pour s'assurer que les features apprises dans les couches profondes du réseaux soient déjà discriminante, des classifiers auxiliaires sont utilisés pour l'entraînement

5. Sous le nom d'équipe GoogLeNet en référence aux travaux pionners de LeNet.

Architecture célèbre

Inception v1 - Module Inception

La remarque qui motive la création du module Inception est la suivante : des patterns importants pour reconnaître un objet peuvent être présent à différentes échelles selon les images. Ainsi, il faudrait être capable d'avoir plusieurs manière d'observer l'image : c'est l'objectif des différentes taille de convolution.

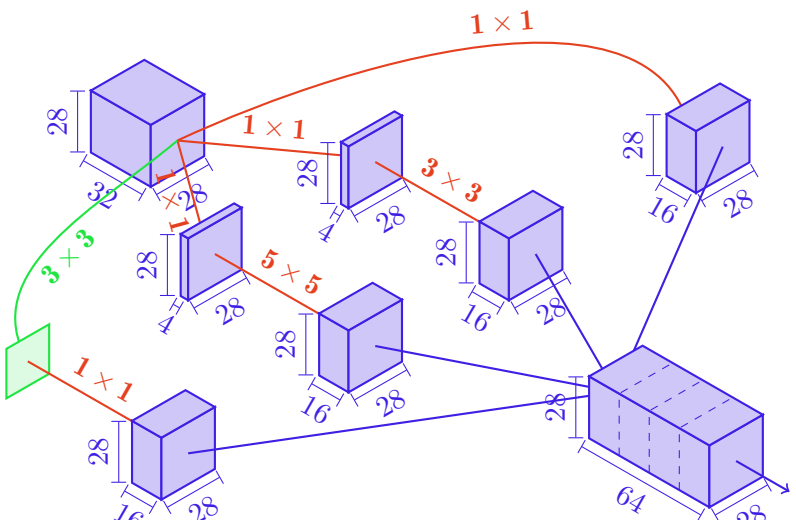


Figure – Module inception avec max-pooling, convolution et concaténation

Architecture célèbre

Modèles VGG

Les modèles VGG [6] [Simonyan and Zisserman, 2014] talonne le modèle Inceptionv1 en 2014 à la même compétition ImageNet. L'objectif de ces modèles est de réduire le nombre de paramètres dans les couches de convolutions en remarquant, par exemple [7], qu'une couche de convolution de taille 5×5 peut être répliquée avec deux couches de convolutions de taille 3×3. En terme de nombre de paramètres :

- Une couche de convolution 5×5 : 26 paramètres
- Deux couches de convolution 3×3 : 20 paramètres

L'ensemble des couches de convolution des VGGNet on des tailles de convolutions de 3×3 et des couches de max-pooling de taille 2×2 de stride 2. Cependant, VGG16 a un total de 138 millions de paramètres. Le nom vient du groupe *Visual Geometry Group* de l'université d'Oxford qui a travaillé sur ces modèles.

6. Par exemple VGG16 contient 16 couches, VGG19 en contient 19...
7. De même, pour une convolution de taille 7×7, trois convolutions de taille 3×3 sont équivalentes.

Architecture célèbre

ResNet

Un an plus tard en 2015, c'est un nouveau type de réseau qui gagne la compétition : les réseaux résiduels [He et al., 2016]. Ce sont des réseaux très profonds qui utilise des *ResBlock* :

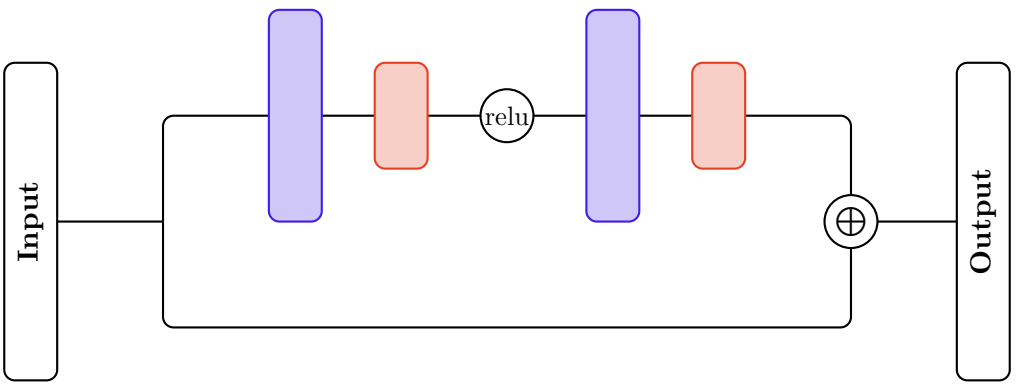


Figure – ResBlock dans un ResNet avec des couches de convolutions et de Batch Normalisation

L'input passe deux couches de convolution et le résultat s'ajoute à l'input pour produire l'output. De cette manière on peut apprendre des features plus intéressantes plus profondément sans pour autant tomber dans les difficultés d'effondrement des réseaux de neurones profonds. Ainsi, le ResNet50 utilise 50 couches !

En résumé

Bibliographie I

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020).
Language models are few-shot learners.
Advances in Neural Information Processing Systems.

He, K., Zhang, X., Ren, S., and Sun, J. (2016).
Deep residual learning for image recognition.
In *Proceedings of the IEEE conference on computer vision and pattern recognition.*

Hendrycks, D. and Gimpel, K. (2016).
Gaussian error linear units (gelus).
arXiv preprint arXiv :1606.08415.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012).
Improving neural networks by preventing co-adaptation of feature detectors.
arXiv preprint arXiv :1207.0580.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).
Imagenet classification with deep convolutional neural networks.
Advances in neural information processing systems.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998).
Gradient-based learning applied to document recognition.
Proceedings of the IEEE.

Bibliographie II

LeCun, Y., Cortes, C., and Burges, C. (2010).
Mnist handwritten digit database.
ATT Labs [Online]. Available : http ://yann.lecun.com/exdb/mnist, 2.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. (2022).
A convnet for the 2020s.
In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021).
Zero-shot text-to-image generation.
In *International Conference on Machine Learning*.

Simonyan, K. and Zisserman, A. (2014).
Very deep convolutional networks for large-scale image recognition.
arXiv preprint arXiv :1409.1556.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015).
Going deeper with convolutions.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017).
Attention is all you need.
Advances in neural information processing systems, 30.

Bibliographie III

Xiao, H., Rasul, K., and Vollgraf, R. (2017).
Fashion-mnist : a novel image dataset for benchmarking machine learning algorithms.
CoRR.

Annexe : Vision Transformers

Du langage à l'image

Suite à l'article *Attention is all you need* [Vaswani et al., 2017], une nouvelle architecture[8] de réseau de neurones appelée transformers a démontré des performances impressionnantes dans le traitement du langage. C'est cette innovation qui a permit a DALL-E [Ramesh et al., 2021] et GPT-3 [Brown et al., 2020] d'exister par exemple.

Cette architecture initialement utilisée pour le traitement du language a été modifiée pour pouvoir être utilisée pour le traitement d'image, on parle de *Vision Transformers* (ViT). Le langage est représenté comme une séquence de mot. Pourtant une image n'est pas une séquence !
L'astuce utilisée est de séparer l'image en plusieurs petites images distinctes[9] et seront ensuite mise côte à côte en *séquence*. Chaque patch est ensuite projeté
dans un espace de plus petite dimension puis l'architecture Transformers (très peu modifié) est utilisée.

Un des avantages majeurs des ViT est que l'attention pour une partie de l'image peut se porter beaucoup plus loin que le filtre dans une couche de convolution classique. Ainsi, on considère que les ViT ont une vision plus globale de l'image.

8. Elle n'est pas au programme du cours
9. Appelées *patches*

Annexe : Fonction d'activation GELU

Les *Large Language Models* sont un concentré d'innovations dans les architectures Deep Learning, et une d'entre-elles est la fonction d'activation GELU [Hendrycks and Gimpel, 2016] introduite en 2016.

$$\mathrm{GELU}(x) = x \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} \mathrm{d}t$$

Le comportement asymptotique des deux fonctions est similaire, mais la différentiabilité partout de GELU est un plus. Elle n'est également pas monotones et n'a pas le phénomène de mort de neurone que peut avoir ReLU. Une explication précise de sa meilleure performance pour certain type d'architecture n'est pas encore parfaitement compris.

Son introduction dans les modèles d'OpenAI a été un tournant dans son utilisation par la communauté : elle est depuis presque systématique [a] dans les nouveaux modèles.

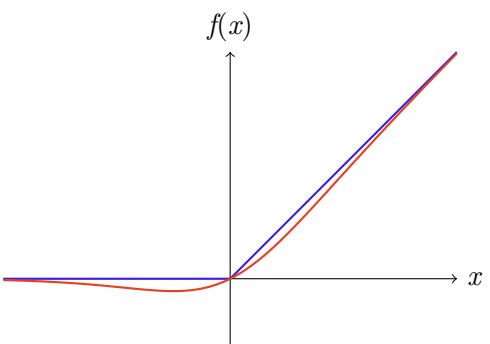


Figure – Fonctions ReLU et GELU

a. La fonction SiLU, et les couches variantes GEGLU et SwiGLU sont en concurrence.

Annexe : ConvNext

A ConvNet for the 2020s

L'article [Liu et al., 2022] propose de moderniser les architectures convolutionnel à la lumière des enseignement des ViT. Cela donne lieu à une nouvelle architecture ResNet avec quelques modifications dont nous verrons certaines à la séance prochaine :

- **Fonction d'activation** : ReLU est remplacée par GELU et est également beaucoup moins présente dans le réseau
- **Taille de filtre** : après avoir modifié des éléments d'architectures, la taille de filtre dans l'ensemble du réseau est 7×7
- **Normalisation** : Layer Normalization est préférée à BatchNormalisation, et est également moins présente dans le réseau

Il y a de nombreuses autres modifications et on se réfère à l'article pour avoir l'ensemble des détails.

Modèle	Images traitées par secondes	Accuracy
Swin-T (ViT)	1325.6	81.3
ConvNeXt-T	1943.5	**82.1**
Swin-S (ViT)	857.3	83.0
ConvNeXt-S	1275.3	**83.1**
Swin-B (ViT)	662.8	83.5
ConvNeXt-B	969.0	**83.8**

Table – Comparaison des performances en vitesse de traitement et accuracy entraîné sur ImageNet 1K