

# Apprentissage Profond - Partie 2

Redouane Lguensat

Master Data Engineering de l'EHTP

Avril 2020

# Table des matières

- 1 Introduction
- 2 Les réseaux de neurones convolutionnels
- 3 Les réseaux de neurones récurrents
- 4 Les réseaux antagonistes génératifs

# Notions élémentaires en vision par ordinateur

- **Les images digitales en couleurs:** Les études sur l'oeil humain ont conclus que toute couleur discernable par l'homme est caractérisée par un point tri-dimensionnel (c.f. les cônes dans le système visuel).

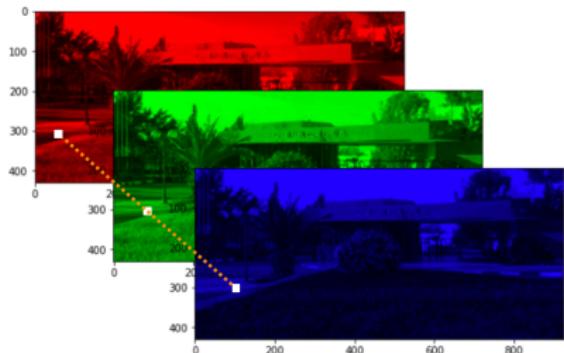
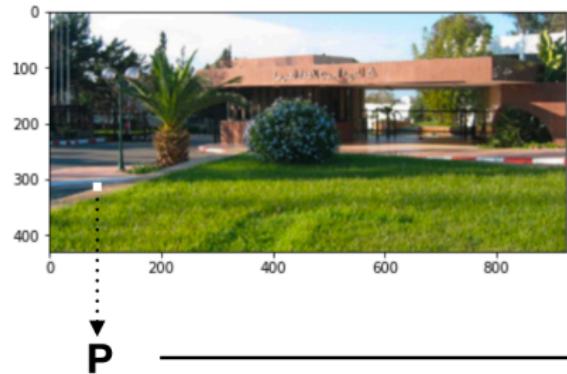
# Notions élémentaires en vision par ordinateur

- **Les images digitales en couleurs:** Les études sur l'oeil humain ont conclus que toute couleur discernable par l'homme est caractérisée par un point tri-dimensionnel (c.f. les cônes dans le système visuel).
- Les couleurs d'un écran d'ordinateur peuvent être projetés sur un espace colorimétrique 3D, où les dimensions sont décrites par les intensités du rouge, du vert et du bleu (**RGB**). C'est le système le plus classiquement utilisé pour les images digitales (mais pas le seul!).

# Notions élémentaires en vision par ordinateur

- **Les images digitales en couleurs:** Les études sur l'oeil humain ont conclus que toute couleur discernable par l'homme est caractérisée par un point tri-dimensionnel (c.f. les cônes dans le système visuel).
- Les couleurs d'un écran d'ordinateur peuvent être projetées sur un espace colorimétrique 3D, où les dimensions sont décrites par les intensités du rouge, du vert et du bleu (**RGB**). C'est le système le plus classiquement utilisé pour les images digitales (mais pas le seul!).
- L'implémentation en 24 bits (8 bits par canal) est très utilisée et traduit les couleurs sur 256 niveaux.

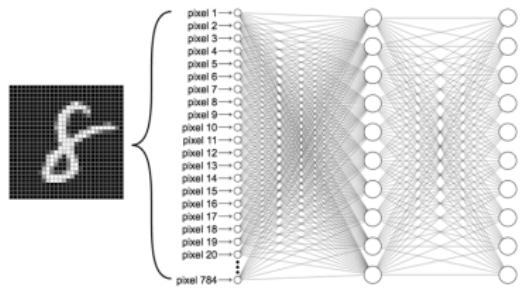
## Exemple avec une image de l'EHTP



**Exercice 1:** En utilisant le module preprocessing de Keras, affichez les composantes RGB d'une image donnée.

# Les limitations des MLPs

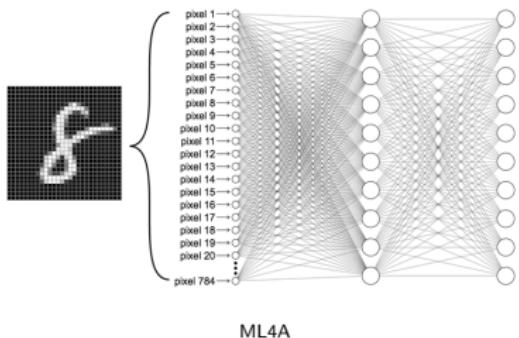
Au terme du cours précédent nous avons entraîné un MLP sur des imagettes de 28x28 pixels, avec deux couches cachées de 512 neurones chacune → total de **669,706** paramètres pour un réseau basique utilisé pour une classification à 10 classes.



ML4A

# Les limitations des MLPs

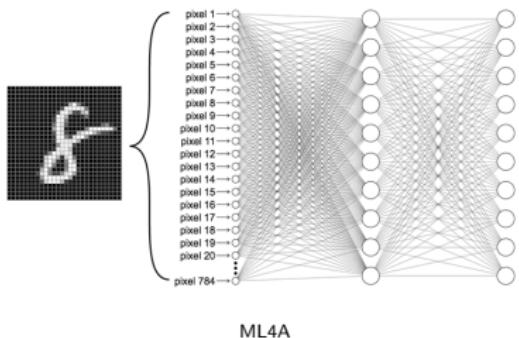
Au terme du cours précédent nous avons entraîné un MLP sur des imagettes de 28x28 pixels, avec deux couches cachées de 512 neurones chacune → total de **669,706** paramètres pour un réseau basique utilisé pour une classification à 10 classes.



Supposons maintenant que nous avons des images RGB de taille 224x224x3, le nombre de paramètres du même réseau MLP deviendra

# Les limitations des MLPs

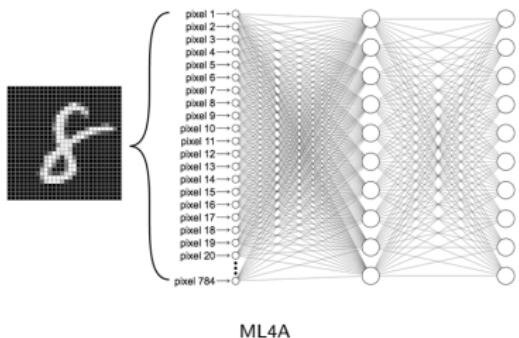
Au terme du cours précédent nous avons entraîné un MLP sur des imagettes de 28x28 pixels, avec deux couches cachées de 512 neurones chacune → total de **669,706** paramètres pour un réseau basique utilisé pour une classification à 10 classes.



Supposons maintenant que nous avons des images RGB de taille 224x224x3, le nombre de paramètres du même réseau MLP deviendra **77,338,634!!**

# Les limitations des MLPs

Au terme du cours précédent nous avons entraîné un MLP sur des imagettes de 28x28 pixels, avec deux couches cachées de 512 neurones chacune → total de **669,706** paramètres pour un réseau basique utilisé pour une classification à 10 classes.

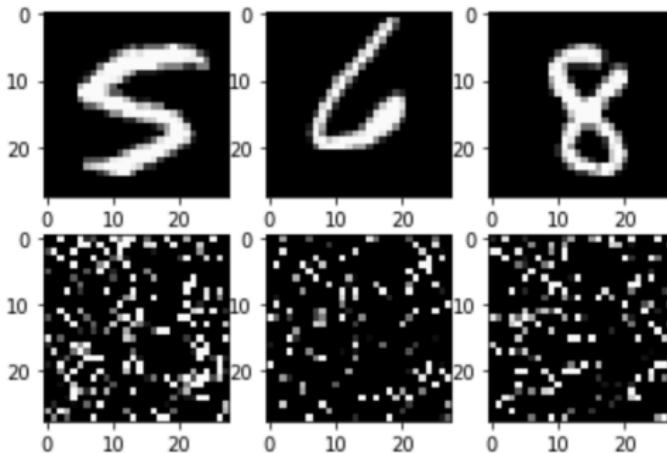


# Les limitations des MLPs

## Exercice 2:

Expérience intéressante: entraîner le même MLP précédent mais sur MNIST avec les pixels permute

MNIST



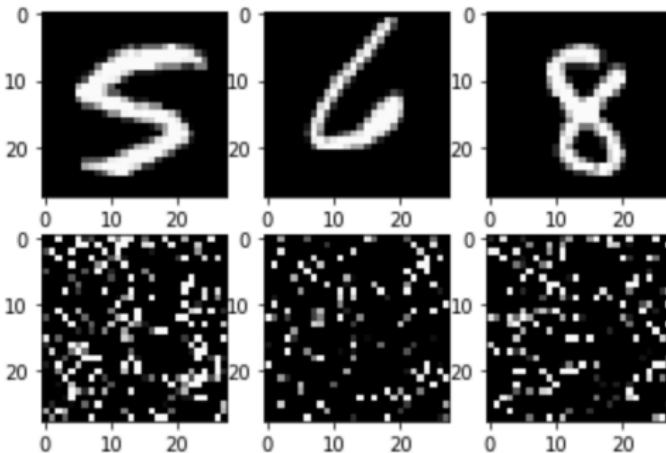
Permuted  
MNIST

# Les limitations des MLPs

## Exercice 2:

Expérience intéressante: entraîner le même MLP précédent mais sur MNIST avec les pixels permute

MNIST



Permuted  
MNIST

Dans un MLP on perd la structure locale!

# Struture locale dans différents types de données

Signal 1D

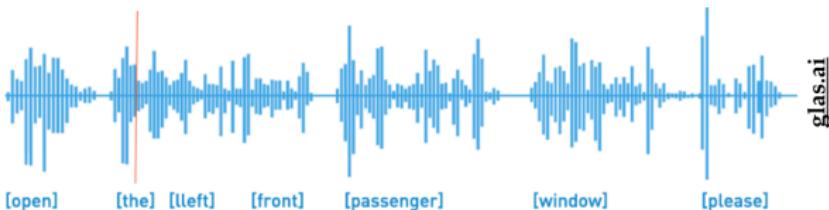
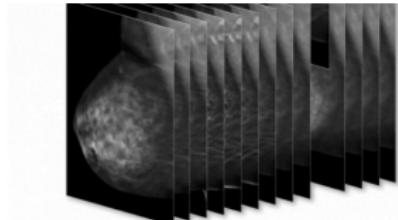


Image 2D

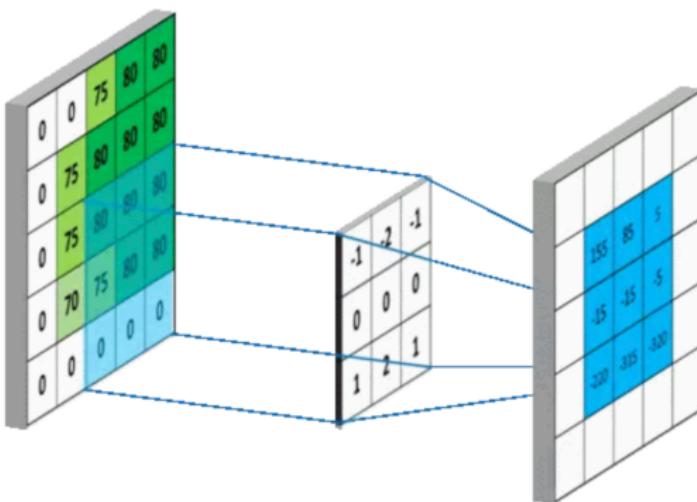


Données 3D



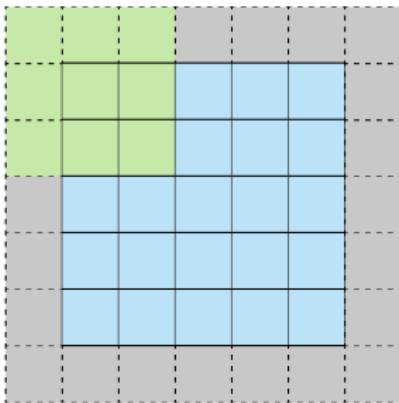
# Solution: localiser les connexions

Utiliser des **convolutions** pour réduire le nombre de connexion des neurones et prendre en compte l'information locale. Idée inspirée du système visuel biologique.

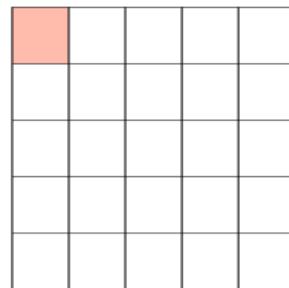


## Comment gérer les frontières de l'image?

## Comment gérer les frontières de l'image? le *padding*



Stride 1 with Padding



Feature Map

Arden Dertat

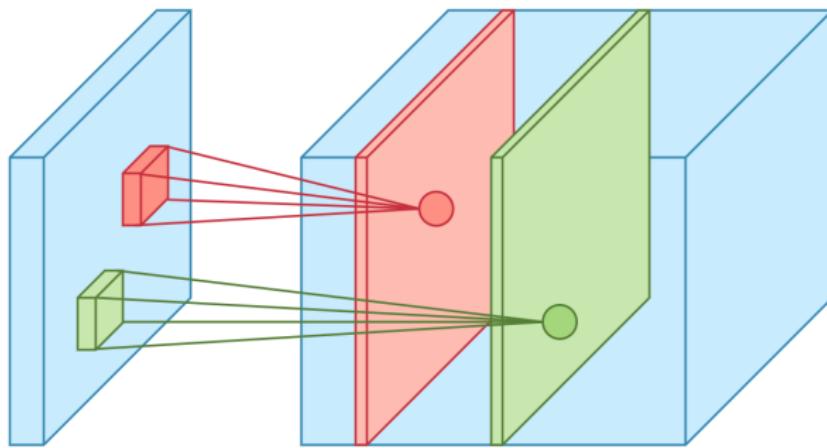
Cliquez ici

# Table des matières

- 1 Introduction
- 2 Les réseaux de neurones convolutionnels
- 3 Les réseaux de neurones récurrents
- 4 Les réseaux antagonistes génératifs

# Couches convolutionels (Conv layers)

- Utiliser les convolutions dans des réseaux de neurones revient à paramétriser des **filtres**, ce sont les poids à optimiser.
- En utilisant plusieurs filtres on construit des tenseurs où on combine les résultats de chaque application de filtre, c'est ce qu'on appelle des *feature maps*



# Autres couches élémentaires

Le *pooling* sert à réduire la dimension des feature maps mais aussi à réduire le nombre de paramètres du réseau (ce qui aide à combattre l'overfitting).

## Autres couches élémentaires

Le *pooling* sert à réduire la dimension des feature maps mais aussi à réduire le nombre de paramètres du réseau (ce qui aide à combattre l'overfitting). Exemple assez courant: le **MaxPooling**

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

Feature map



6	8
3	4

Pooled  
Feature map

Cliquez ici



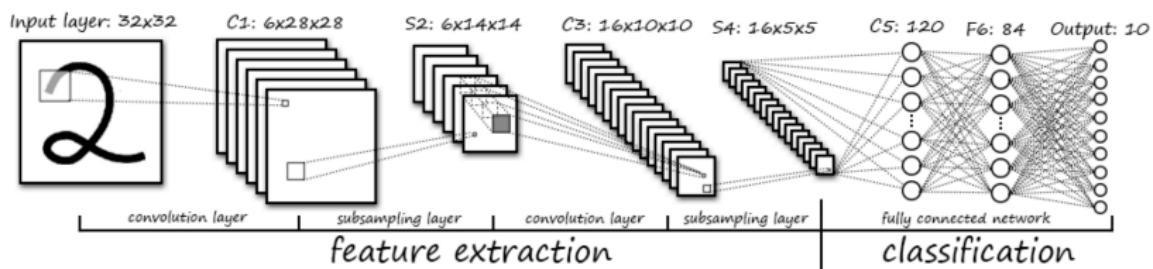
# Les premiers réseaux de neurones convolutionnels: LeNet5

Yann LeCun en 1989 a développé le tout premier réseau de neurones convolutionnel: LeNet

# Les premiers réseaux de neurones convolutionnels: LeNet5

Yann LeCun en 1989 a développé le tout premier réseau de neurones convolutionnel: LeNet

Un réseaux de neurones convolutionnels est la combinaison de plusieurs types de couches (conv, pool, fully connected etc..)

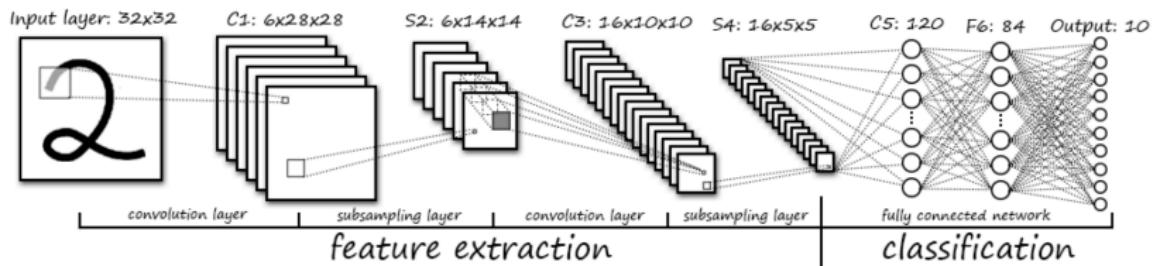


Cliquez ici

# Les premiers réseaux de neurones convolutionnels: LeNet5

Yann LeCun en 1989 a développé le tout premier réseau de neurones convolutionnel: LeNet

Un réseaux de neurones convolutionnels est la combinaison de plusieurs types de couches (conv, pool, fully connected etc..)

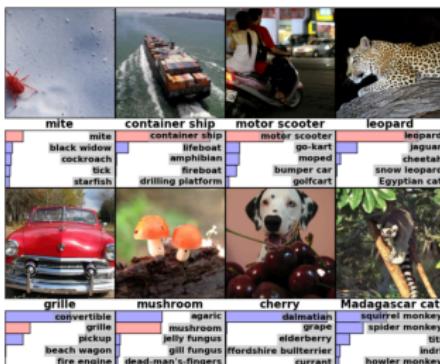


Cliquez ici

**Exercice 3:** utilisez Keras pour entraîner LeNet5 sur MNIST

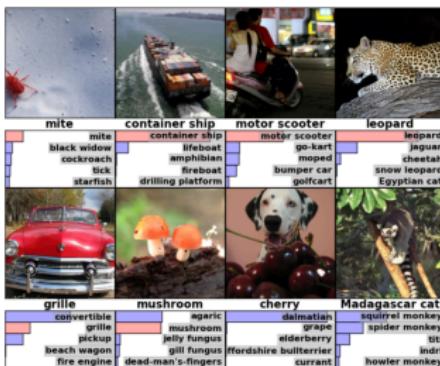
# Le deep learning frappe fort: ImageNet Challenge

- La base de données ImageNet: Base de données de 15 millions d'images et 22.000 classes.



# Le deep learning frappe fort: ImageNet Challenge

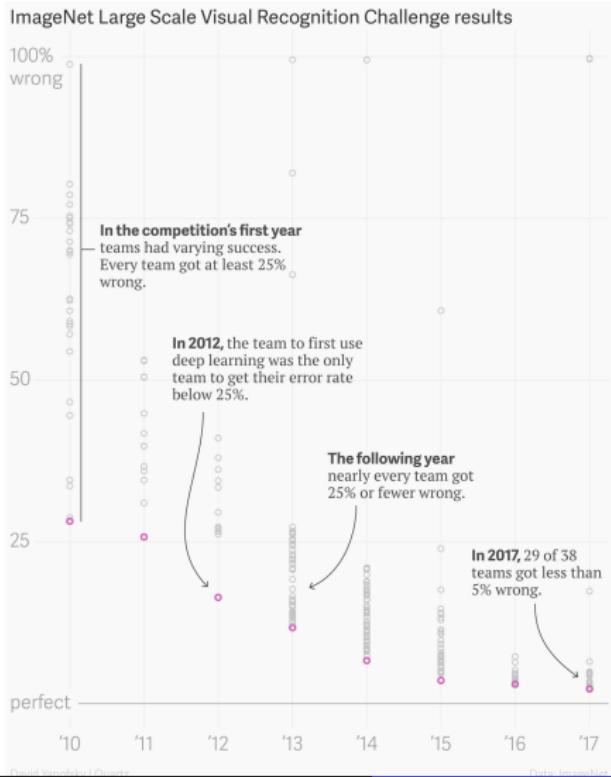
- La base de données ImageNet: Base de données de 15 millions d'images et 22.000 classes.



- Un subset du dataset (1.2M images de train, 50,000 validation , et 150,000 test, 1000 classes) est utilisé pour la compétition annuelle ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)

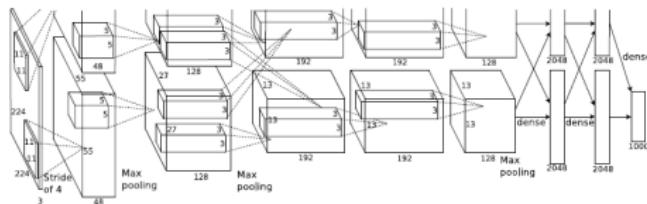


# Le deep learning frappe fort: AlexNet (2012)



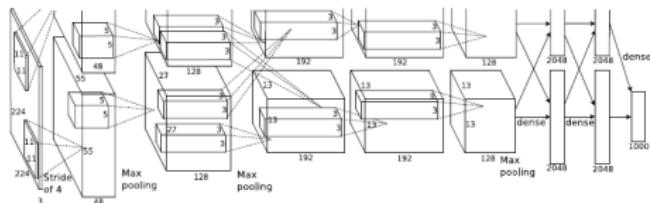
# Des architectures classiques

AlexNet (2012): 8 layers

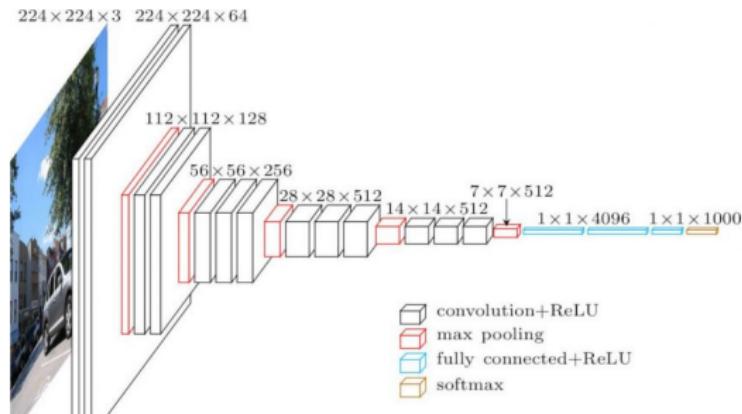


# Des architectures classiques

AlexNet (2012): 8 layers

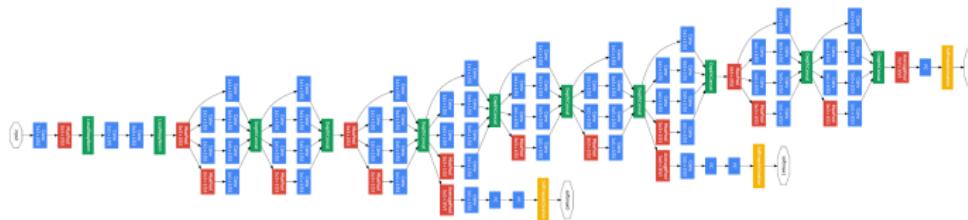


VGG (2014): 19 couches



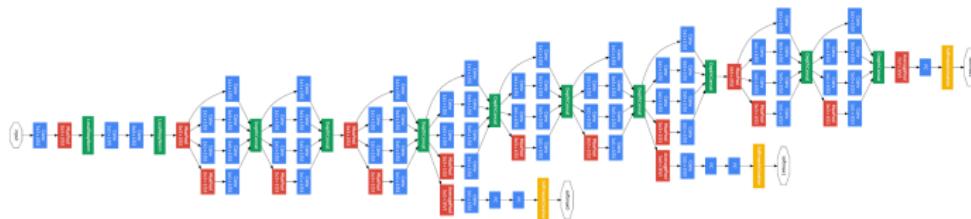
## Des architectures classiques

GoogleNet (Inception) (2015): 22 couches

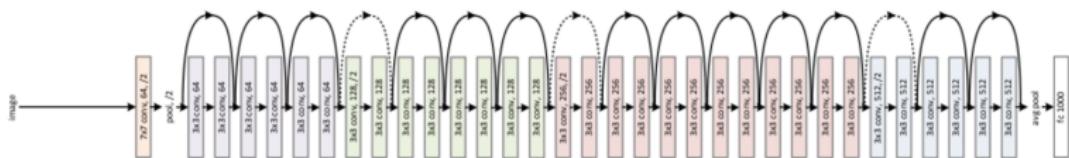


# Des architectures classiques

GoogleNet (Inception) (2015): 22 couches

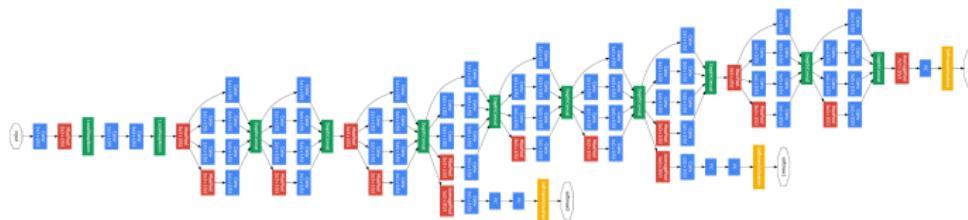


ResNet (2015): 152 couches!!!

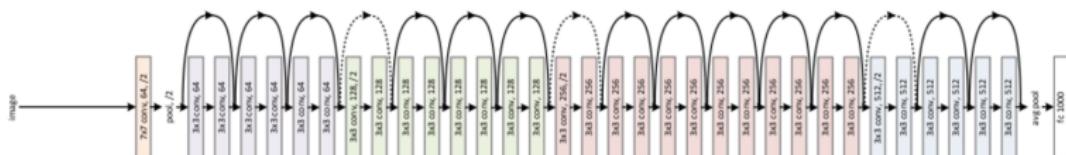


# Des architectures classiques

GoogleNet (Inception) (2015): 22 couches



ResNet (2015): 152 couches!!!



La liste est longue et toujours en pleine expansion...

# Les modèles pré-entraînés

Sur Keras par exemple, plusieurs des architectures classiques sont disponibles pour utilisation sans besoin de les entraîner from scratch

Keras Documentation

Search docs

Home

Why use Keras

GETTING STARTED

Guide to the Sequential model

Guide to the Functional API

FAQ

MODELS

About Keras models

Sequential

Model (functional API)

LAYERS

About Keras layers

Core Layers

Convolutional Layers

Pooling Layers

Locally-connected Layers

Recurrent Layers

Embedding Layers

Docs » Applications

Edit on GitHub

## Applications

Keras Applications are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning.

Weights are downloaded automatically when instantiating a model. They are stored at `~/.keras/models/`.

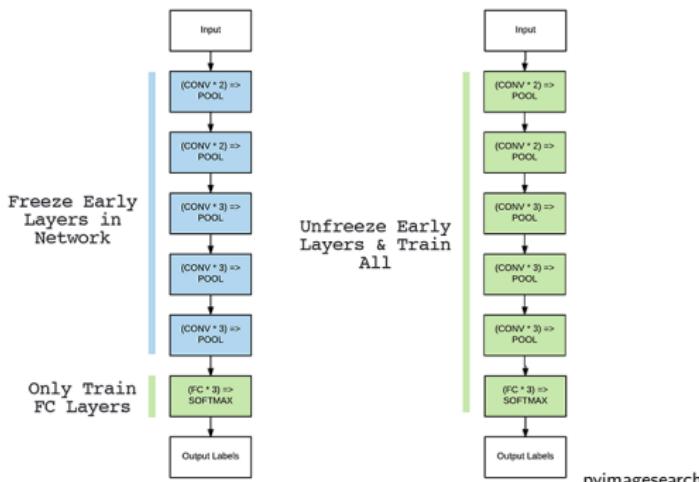
## Available models

**Models for image classification with weights trained on ImageNet:**

- Xception
- VGG16
- VGG19
- ResNet, ResNetV2
- InceptionV3
- InceptionResNetV2
- MobileNet
- MobileNetV2
- DenseNet
- NASNet

# Le fine-tuning

L'idée du fine-tuning est de garder une partie du réseau fixe (ne pas entraîner ses paramètres) et concentrer l'entraînement sur les dernières couches ou nouvelles couches adaptées à notre application.

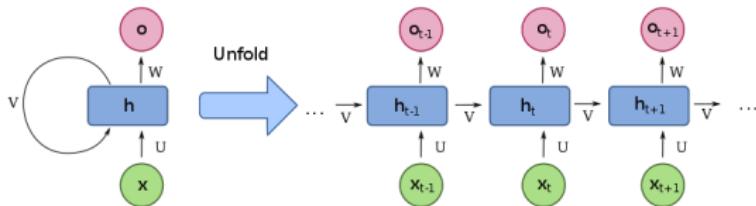


pyimagesearch

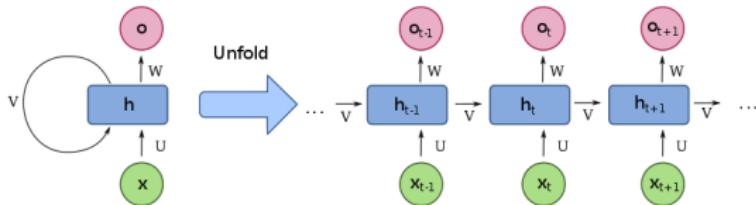
# Table des matières

- 1 Introduction
- 2 Les réseaux de neurones convolutionnels
- 3 Les réseaux de neurones récurrents
- 4 Les réseaux antagonistes génératifs

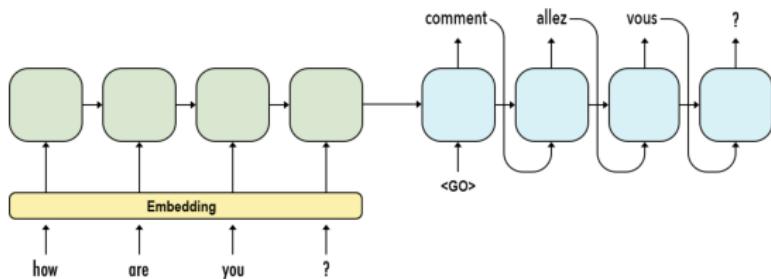
Les réseaux de neurones récurrents sont une classe importante de réseaux de neurones. Ils conviennent en particulier pour l'analyse de séries temporelles.



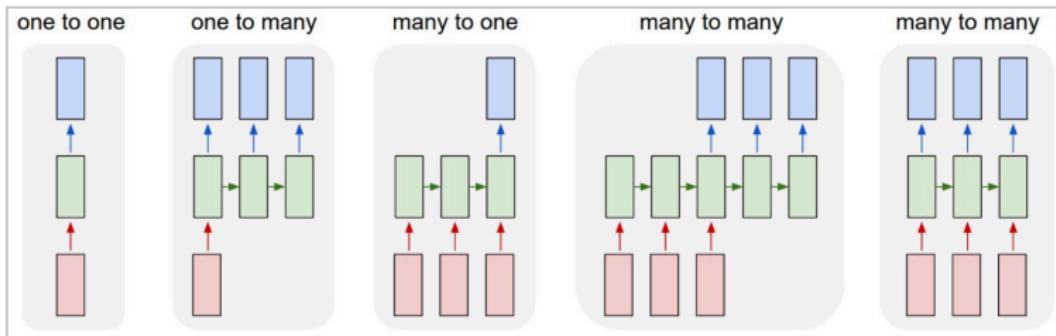
Les réseaux de neurones récurrents sont une classe importante de réseaux de neurones. Ils conviennent en particulier pour l'analyse de séries temporelles.



Exemples d'application: Natural Language Processing (NLP)



# Des exemples de RNNs



Each rectangle is a vector and arrows represent functions (e.g. matrix multiply). Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state (more on this soon). From left to right: (1) Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification). (2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words). (3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths of sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like.

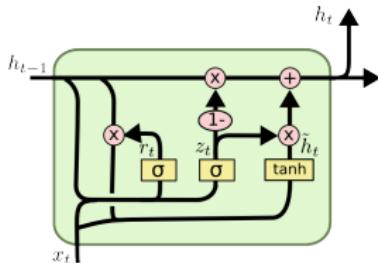
Andrej Karpathy

# Le problème du gradient en voie de disparition

- En pratique l'entraînement des réseaux de neurones récurrents classiques souffre du problème du gradient en voie de disparition (*vanishing gradient problem*). Cela provient de la multiplication de beaucoup de valeurs très petites jusqu'à arriver à des situations où la descente de gradient n'arrive pas à mettre à jour les paramètres.

## Le problème du gradient en voie de disparition

- En pratique l'entraînement des réseaux de neurones récurrents classiques souffre du problème du gradient en voie de disparition (*vanishing gradient problem*). Cela provient de la multiplication de beaucoup de valeurs très petites jusqu'à arriver à des situations où la descente de gradient n'arrive pas à mettre à jour les paramètres.
  - Pour palier à ce problème Sepp Hochreiter et Jürgen Schmidhuber (1997) ont proposé le **Long Short Term Memory**



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

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

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

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

Colah's blog



# LSTM avec Keras

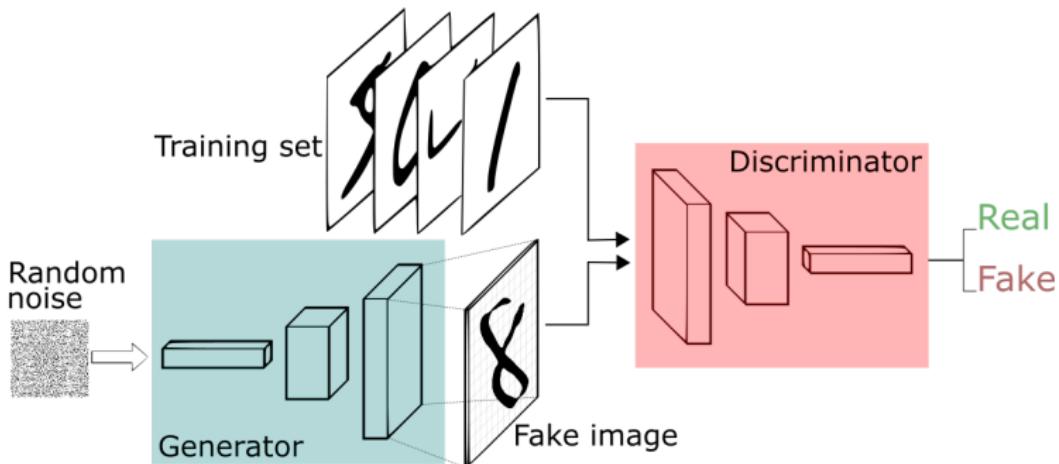
Regardons un exemple de l'utilisation des LSTMs pour la prédiction de séries temporelles

(cliquez ici pour accéder au notebook)

# Table des matières

- 1 Introduction
- 2 Les réseaux de neurones convolutionnels
- 3 Les réseaux de neurones récurrents
- 4 Les réseaux antagonistes génératifs

Un réseaux antagoniste génératif *Generative Adversarial Network* est un modèle génératif où deux réseaux sont placés en compétition. Le premier réseau est le générateur, il génère un échantillon proche d'une base de données, tandis que son adversaire, le discriminateur essaie de détecter si cet échantillon est bien réel ou s'il est le résultat du générateur.



# Exemples d'utilisation des GANs

Des performances impressionnantes en génération de visages humains



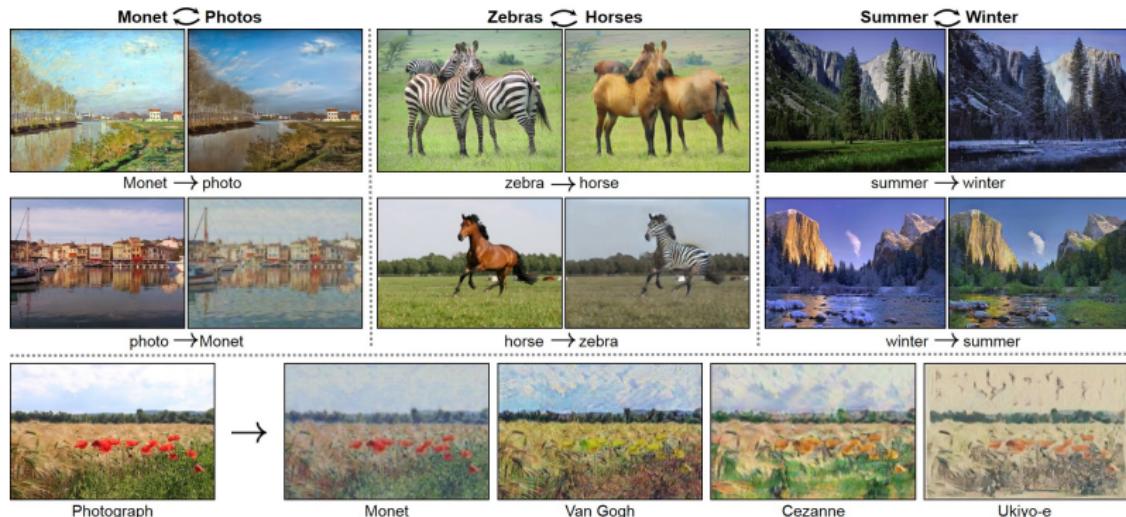
# Exemples d'utilisation des GANs

Des performances impressionnantes en génération de visages humains



# Exemples d'utilisation des GANs

## Traduction image vers image (CycleGANs)



# Exemples d'utilisation des GANs

Des applications dans l'industrie aussi. Ex: chez Zalando, génération de mannequins avec vêtements personnalisés



# Prochain cours: Tutoriel

## Classification de signes de trafic avec des ConvNets

La prochaine séance sera consacrée à une mise en pratique des connaissances acquises à l'issue des deux cours. Nous allons travailler sur un exemple classique de vision par ordinateur (*Computer Vision*): la classification d'images. Comme exemple nous allons considérer le problème de classification de signes de trafic.

**Des questions?**