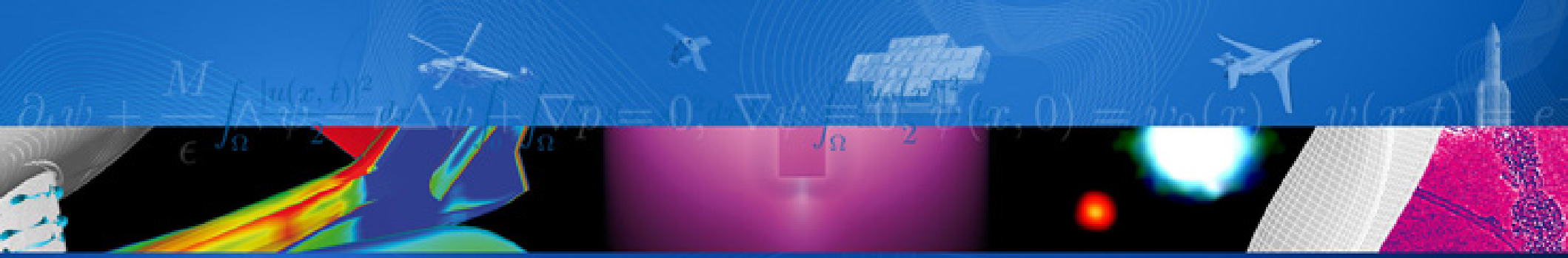


APPRENTISSAGE MACHINE & DEEP LEARNING

Deep Learning

A. Boulch, A. Chan Hon Tong, S. Herbin, B. Le Saux

ONERA

THE FRENCH AEROSPACE LAB

retour sur innovation

Deep Learning

- Réseaux de neurones convolutifs
- **Principales difficultés**
- Deep learning
  - Couches de régularisation
  - Initialisation
  - Optimiseurs
  - Les données
- Réseaux et applications

Pricipaux problèmes

ONERA
THE FRENCH AEROSPACE LAB

Vitesse d'apprentissage

Gradients exponentiels ou évanescents

Structure du réseau

Matériel

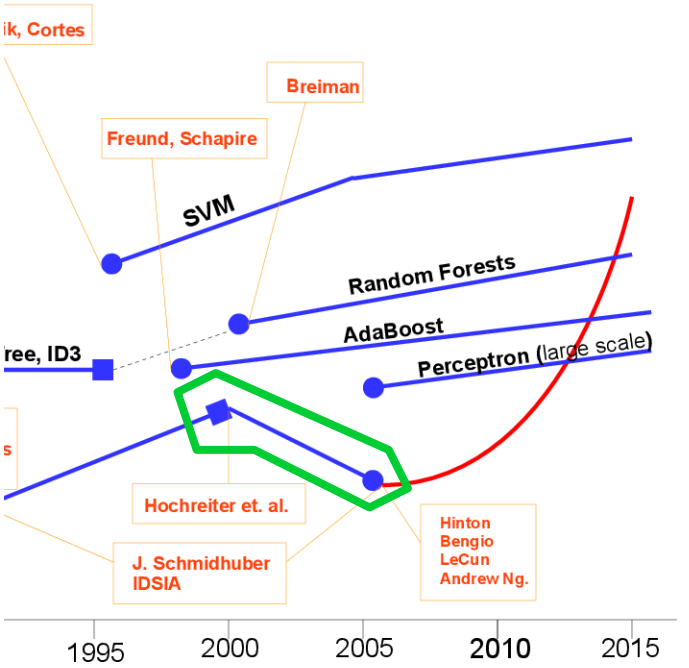
Optimiseur

Données

Initialisation des poids

Overfitting

Minima locaux

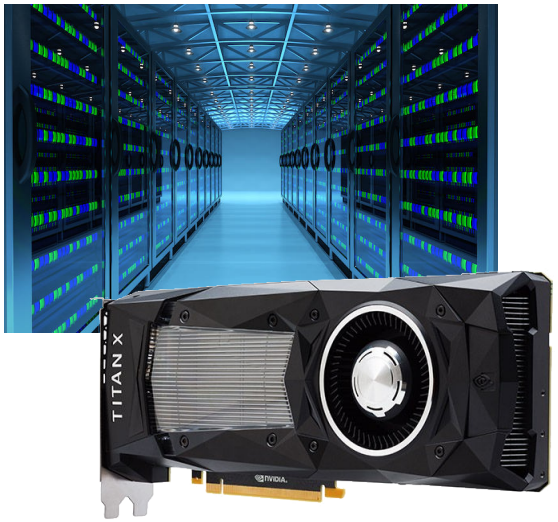


Deep Learning

- Réseaux de neurones convolutifs
- Principales difficultés
- **Deep learning**
  - Couches de régularisation
  - Initialisation
  - Optimiseurs
  - Les données
- Réseaux et applications

Deep learning : massively data driven apporaches

ONERA
THE FRENCH AEROSPACE LAB



Calcul parallèle

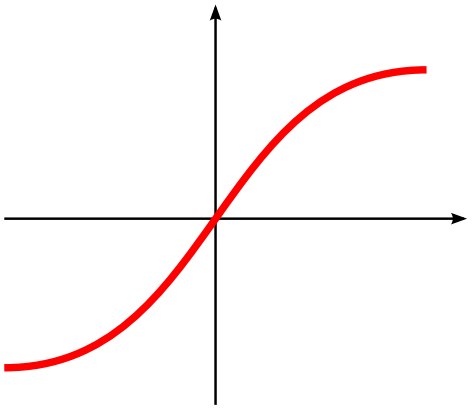
Moyens d'acquisition

Partage

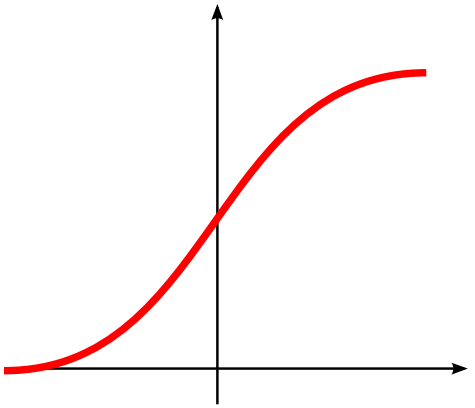
Performances

Intérêt

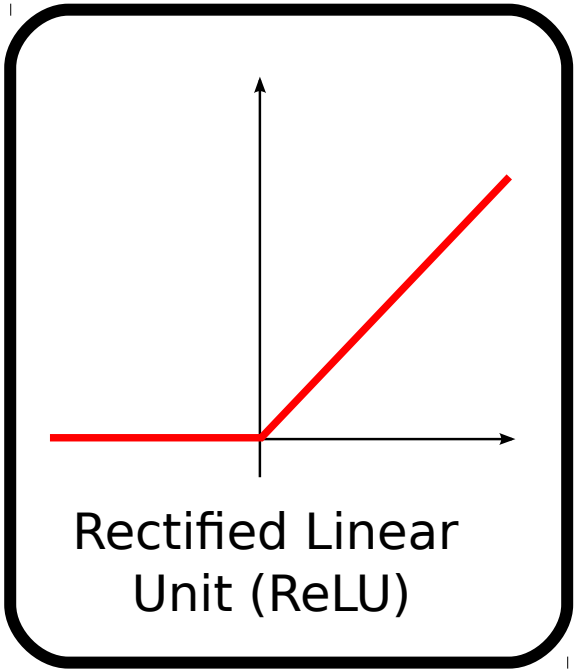
Activations



Tangente hypebolique



Sigmoïde

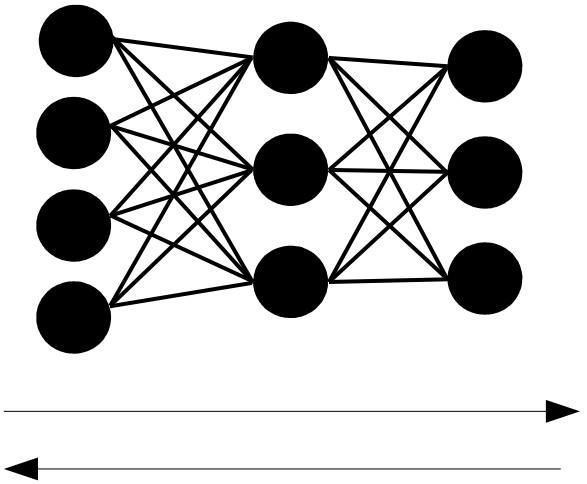


Rectified Linear Unit (ReLU)

Gradients plus rapide à calculer

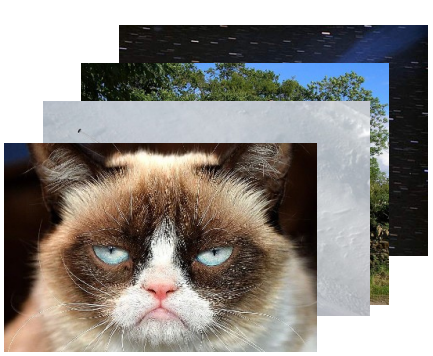
Convergence identique

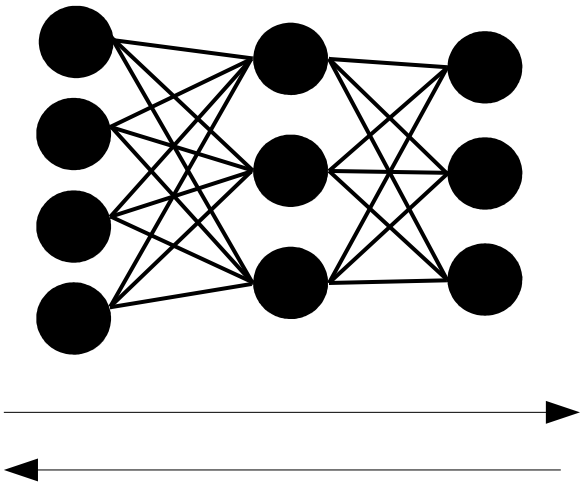
Mini-batch



Erreur image 1

Mini-batch

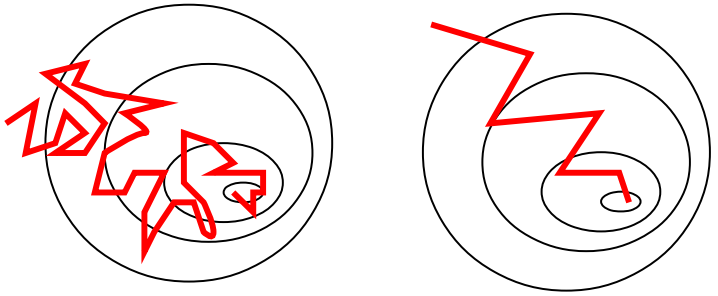




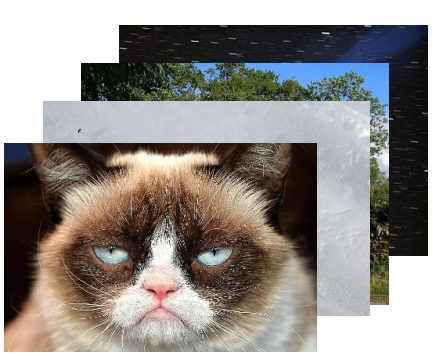
Erreur image 1
Erreur image 2
Erreur image 3
Erreur image 4
...

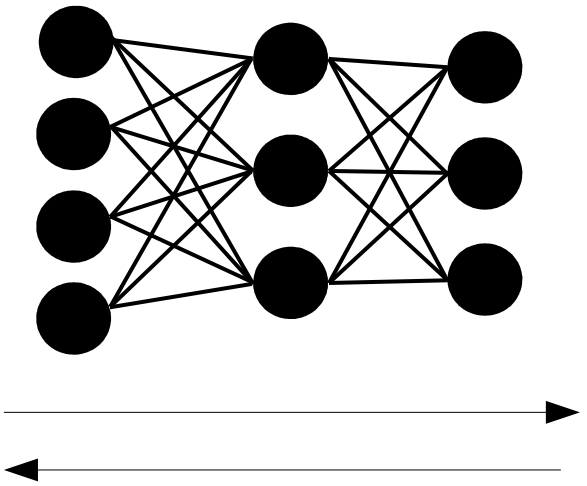
- Forward
  - Prédictions indépendantes
- Backward
  - Moyennes sur les gradients

- **Lissage des gradients**
  Réduction de la variance
  Possibilité d'utilisation d'un learning rate plus grand



Mini-batch





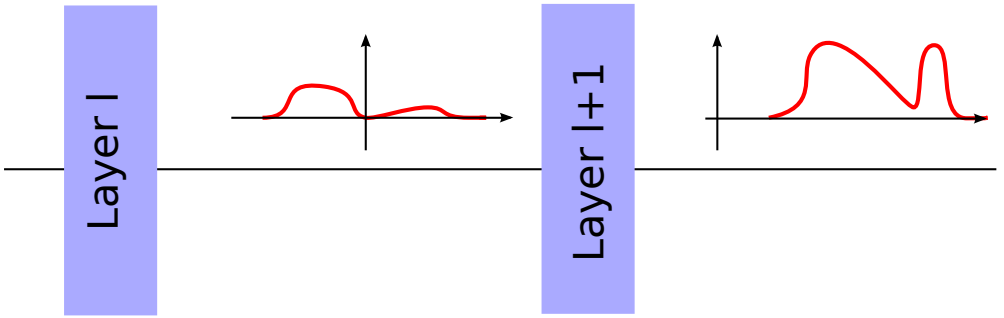
Erreur image 1
Erreur image 2
Erreur image 3
Erreur image 4
...

- Forward
  - Prédictions indépendantes
- Backward
  - Moyennes sur les gradients

- **Lissage des gradients**
  Réduction de la variance
  Possibilité d'utilisation d'un learning rate plus grand

- **Accélération matérielle (GPU)**
  Appliquer les même opérations

Couches : Batch Norm



Changements dans la distribution ⇒ variance et moyenne
Apprentissage plus difficile

Contrôler la distribution des données avec une couche pour maîtriser la distribution

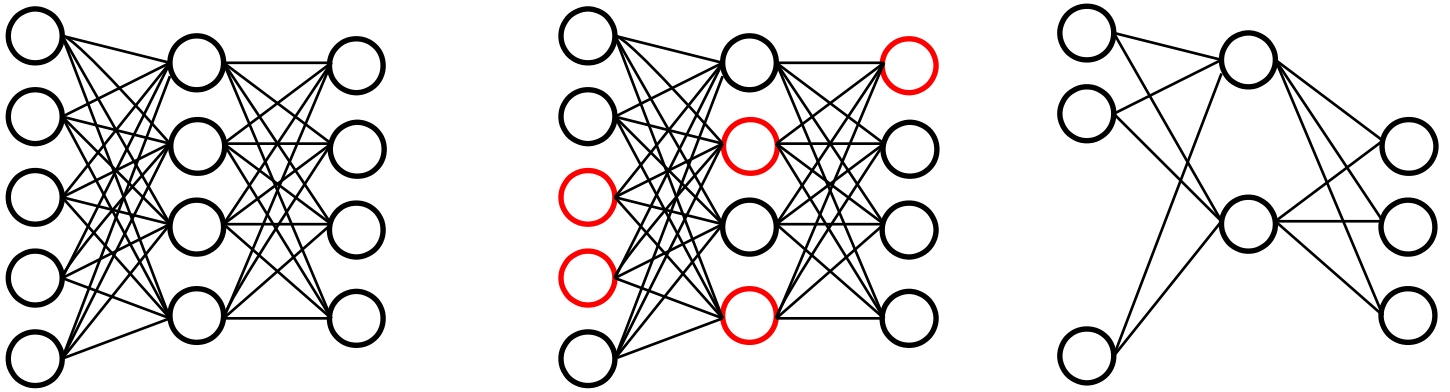
$$y^{*l} = \frac{y^{l} - \mu}{\sqrt{\sigma^2 + \epsilon}} \gamma + \beta$$

Les β et ɣ sont appris.
μ et σ sont calculés sur les batchs.

Apprentissage plus rapide (en nombre d'itération), mais en général plus lent (calcul des statistiques)

Batch normalization: Accelerating deep network training by reducing internal covariate shift, Ioffe and Szegedy, 2015

Dropout



Entraînement :

- Éteindre certains neurones avec probabilité $1\text{-}p$

⇒ Entraîner une multitude de sous-réseaux

Test :

- Utiliser tous les neurones pondérés par p

Les fully connected concentrent les poids des réseaux ⇒ overfitting
Le dropout apporte plus de robustesse et une meilleure généralisation

Moins de nœuds ⇒ plus rapide

Initialisation des poids

ONERA
THE FRENCH AEROSPACE LAB

Les poids des réseaux sont initialisés aléatoirement.
L'initialisation a une grande influence sur la convergence et sur la vitesse d'apprentissage.

- Poids trop faibles ⇒ le signal peut décroître jusqu'à s'annuler
- Poids trop grands ⇒ le signal peut devenir trop gros pour être utilisable

Exemple : Initialisation Xavier

X de dimension n, un neurone linéaire avec poids W, la sortie est Y

$$Y = W_1 X_1 + W_2 X_2 + \ldots + W_n X_n$$

Or

$$Var(W_i X_i) = E[X_i]^2 Var(W_i) + Var(W_i) Var(X_i) + Var(X_i) E[W_i]^2 = Var(W_i) Var(X_i)$$

Ainsi $Var(Y) = n Var(W_i) Var(X_i)$ et $Var(W_i) = \frac{1}{n} = \frac{1}{n_{in}}$

Glorot et Bengio () :
idem avec la back prop. $Var(W_i) = \frac{1}{n_{out}}$ Moyenne des deux $Var(W_i) = \frac{1}{n_{in} + n_{out}}$

He, Rang, Zen et Sun () : $Var(W_i) = \frac{2}{n_{in}}$ (un ReLU aura la moitié de ses activations nulles)

Optimiseurs

ONERA
THE FRENCH AEROSPACE LAB

- **Variation du learning rate**
  - Constant par morceau
  - Décroissance exponentielle
- **Lissage du gradient**
  - Momentum

$$\boldsymbol{w} = \boldsymbol{w} - \alpha * \Delta \boldsymbol{w}$$

$$v_t = \gamma\, v_{t-1} + \alpha * \Delta \boldsymbol{w}$$
$$\boldsymbol{w} = \boldsymbol{w} - v_t$$

Optimiseurs

ONERA
THE FRENCH AEROSPACE LAB

- **Méthodes adaptatives**
  - Adagrad : adaptation du learning rate à la fréquence

$$w_{t+1,i} = w_{t,i} - \frac{\alpha}{\sqrt{G_{t,i} + \epsilon}} \Delta w_{t,i}$$

$G_{t,i}$ la somme des gradients au carré pour $w_{t,i}$

  - Adam (adaptive momentum estimation)

    Adaptation du learning rate et utilisation du momentum

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) \Delta w_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) \Delta w_t^2$$

Estimation de la moyenne et de la variance

$$\hat{m}_t = m_t / (1-\beta_1^t)$$
$$\hat{v}_t = v_t / (1-\beta_2^t)$$

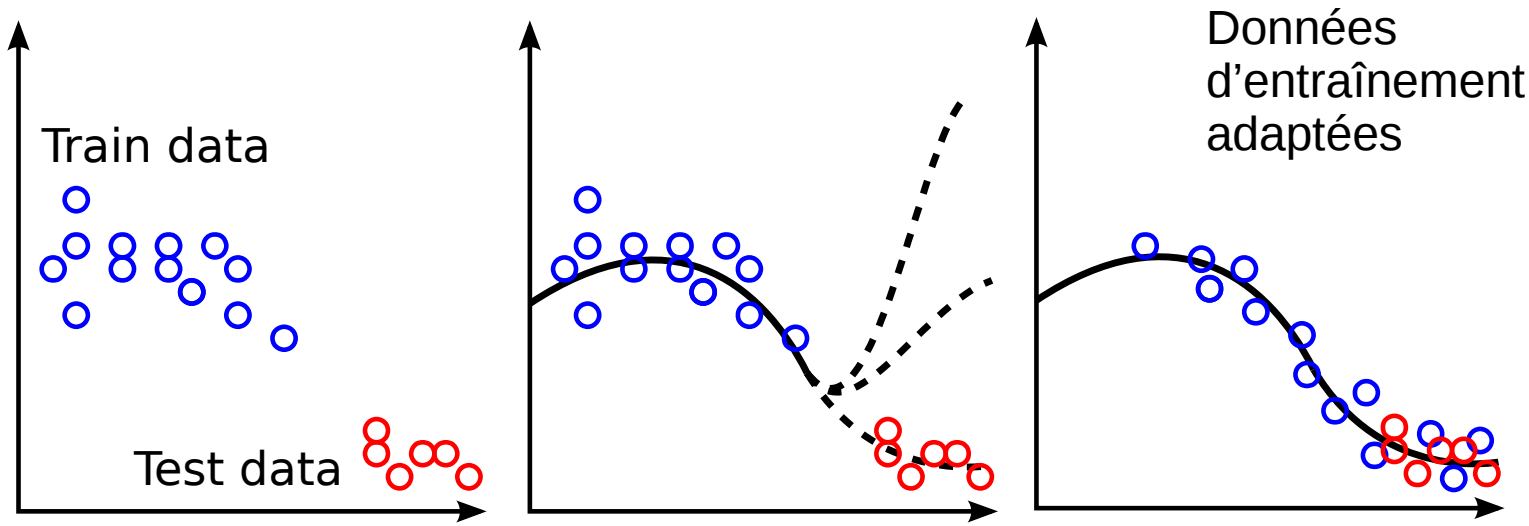
Moments biaisés vers 0.

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

Règle de mise à jour

Les données

- Données adaptées



Aucune garantie sur la généralisation à l'ensemble de test

Les données

ONERA
THE FRENCH AEROSPACE LAB

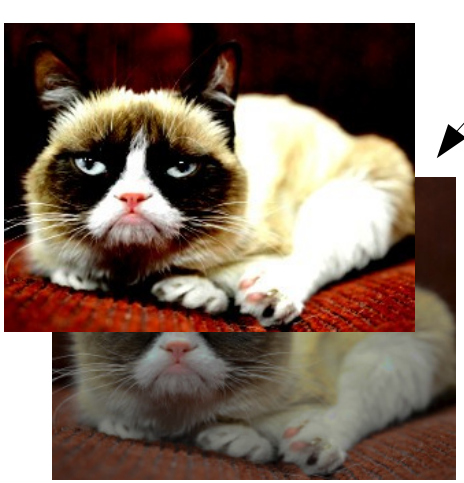
- Données adaptées
- Normalisation

  Moyenne nulle, variance 1

  ⇒ Éviter les problème de dynamique

Les données

- Données adaptées
- Normalisation
- Augmentation de données
  - Faire varier les paramètre qu'on ne veut pas apprendre.



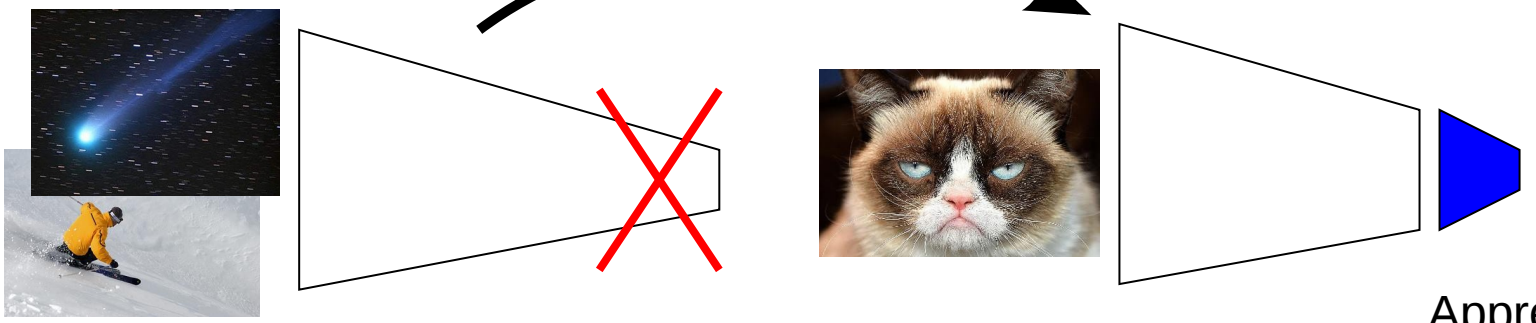
Illumination / contraste

Imagettes

Symétries

Les données

- Données adaptées
- Normalisation
- Augmentation de données
- Adaptation de domaine



Deep Learning

- Réseaux de neurones convolutifs
- Principales difficultés
- Deep learning
  - Couches de régularisation
  - Initialisation
  - Optimiseurs
  - Les données
- **Réseaux et applications**

Frameworks

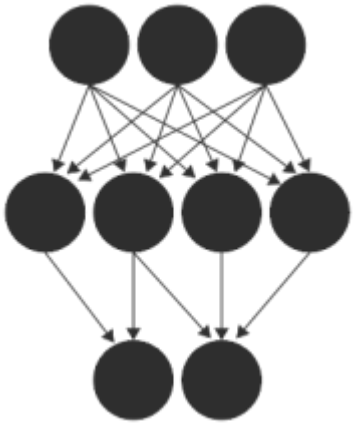
dmlc

mxnet

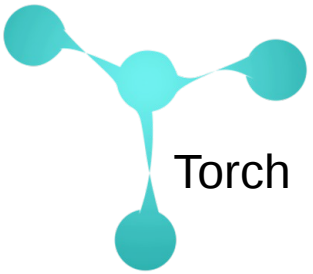
ONERA
THE FRENCH AEROSPACE LAB

TensorFlow

PYTORCH



ConvNetJs



Torch

Caffe

Keras

Microsoft
CNTK

DEEPLEARNING4J

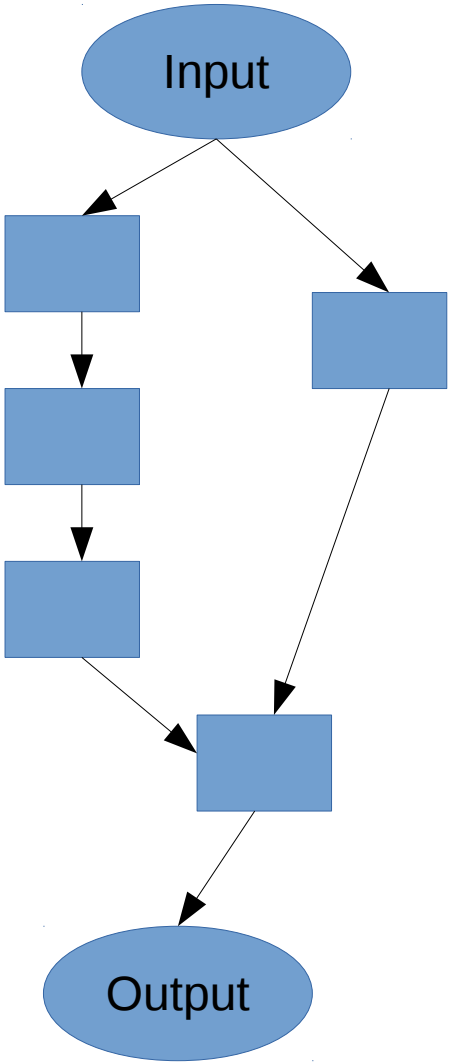
theano

MatConvNet

Frameworks

ONERA
THE FRENCH AEROSPACE LAB

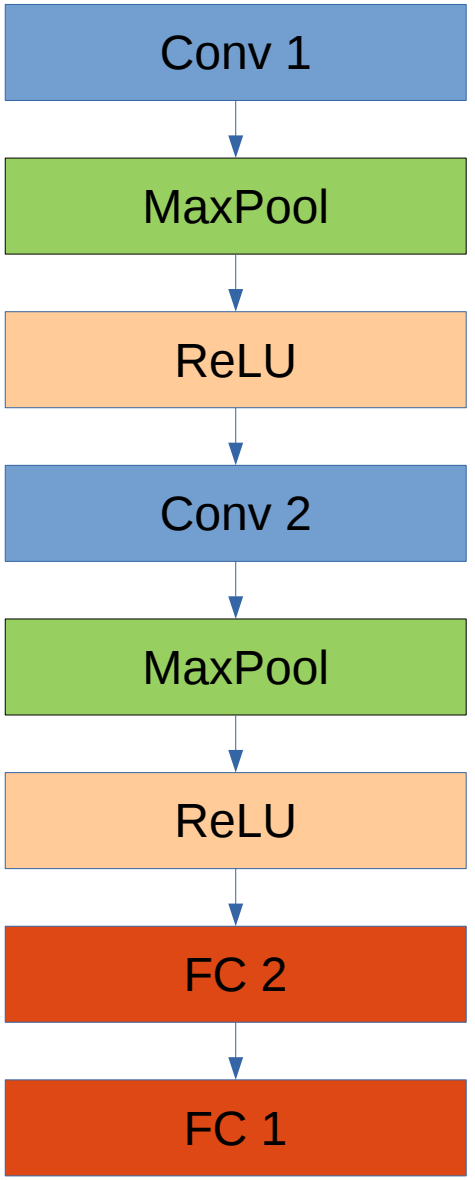
- Même principe :
  - Un graphe (les nœuds sont des couches)
  - Un optimiseur
  - Données



Exemple : LeNet (PyTorch)

ONERA
THE FRENCH AEROSPACE LAB

```
class LeNet(nn.Module):

    # Les couches avec poids
    def __init__(self):
        super(LeNet, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4*4*50, 500)
        self.fc2 = nn.Linear(500, 10)

    # structure du réseau
    def forward(self, x, target):
        x = self.conv1(x)
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(x)
        x = self.conv2(x)
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(x)
        x = x.view(-1, 4*4*50)
        x = self.fc1(x)
        x = self.fc2(x)
        return x
```



Exemple : LeNet

ONERA
THE FRENCH AEROSPACE LAB

Définition du réseau

```
conv1 = conv2d(image, 64 ,[3,3], [1,1])
relu1 = relu(conv1)
pool1 = max_pool_2x2(relu1)
conv2 = conv2d(pool1, 128, [3,3], [1,1])
relu2 = relu(conv2)
pool2 = max_pool_2x2(relu2)
flat  = flatten(pool2)
fc1   = fully_connected(flat, 256)
reluf1= relu(fc1)
fc2   = fully_connected(reluf1, 10)
```

Fonction de perte et optimiseur

```
loss =softmax_cross_entropy(fc2,labels)
optimizer = AdamOptimizer(learning_rate)
train_step = optimizer.minimize(loss)
```

Exemple : LeNet

ONERA
THE FRENCH AEROSPACE LAB

Entraînement

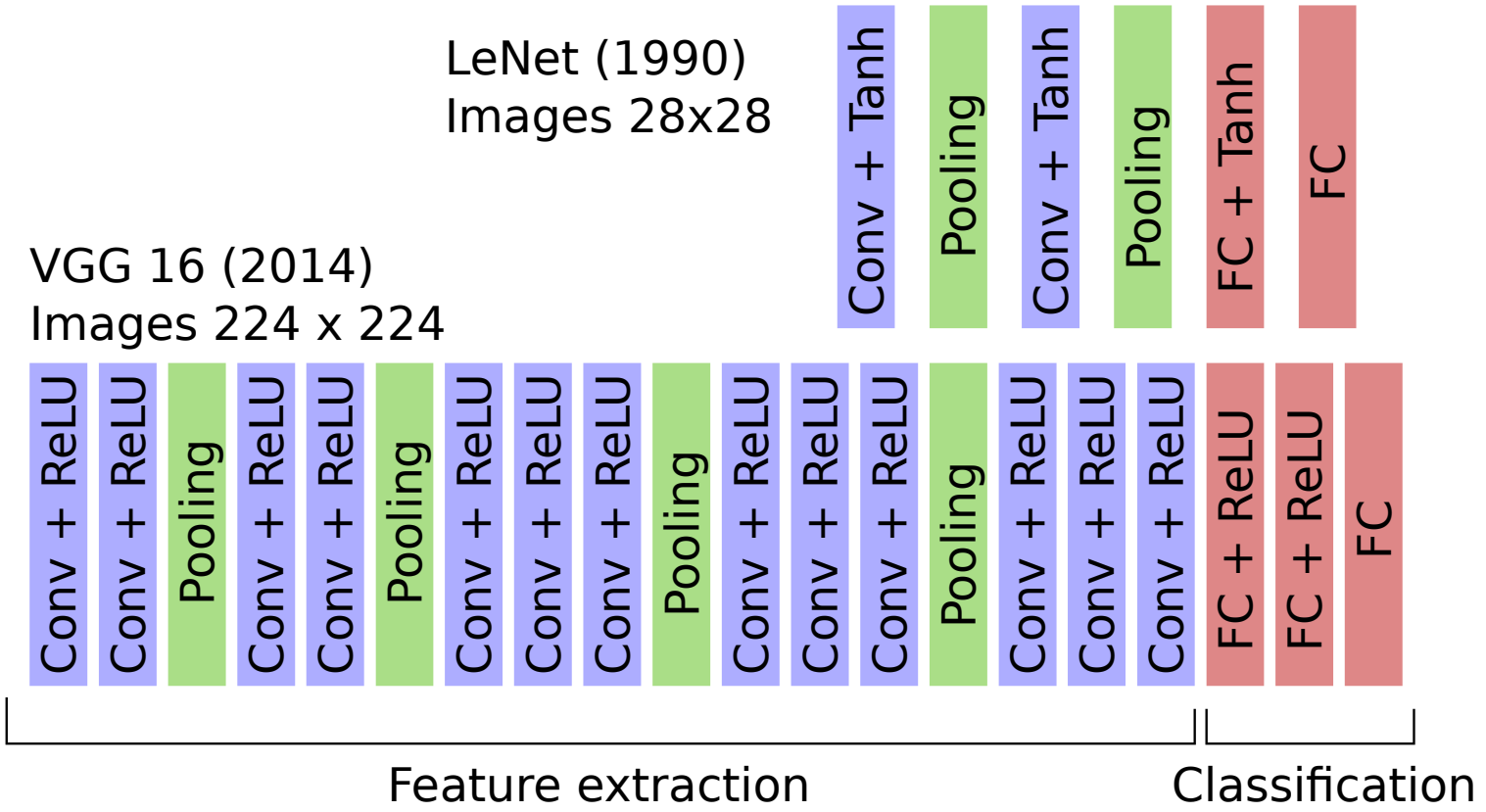
```
for epoch in range(epoch_max) :

    for batch in batches :
        # numpy tabs
        fd = {images:batch.images, labels.labels}
        sess.run(train_step, feed_dict=fd)
```

Test

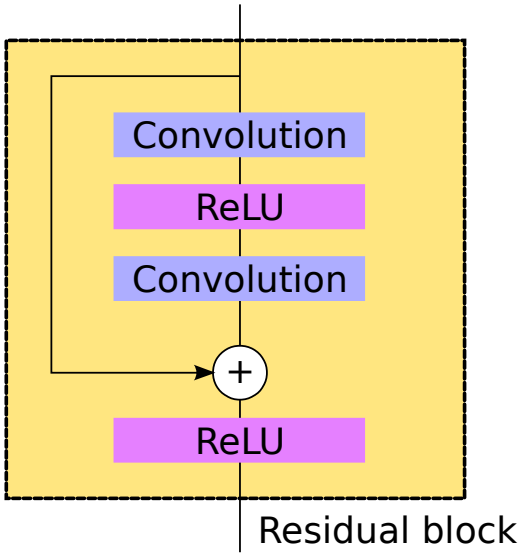
```
for test_images in test_set :
    fd = {images:test_images}
    predictions = sess.run(fc2, fd)
```

MNIST / VGG 16

- Architecture classique en vision



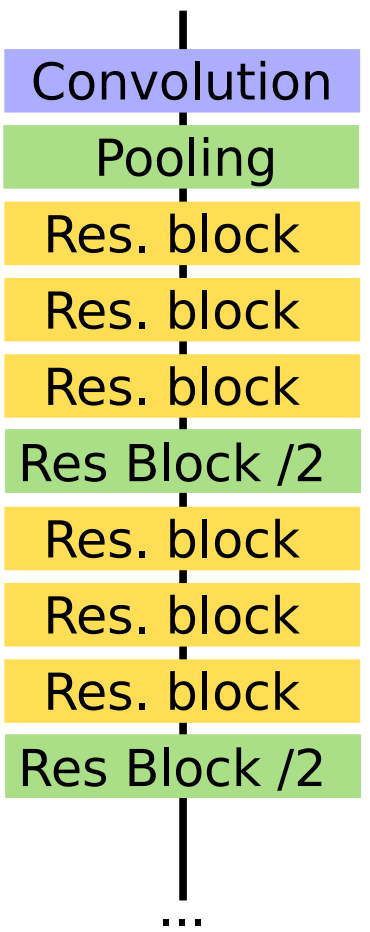
ResNet

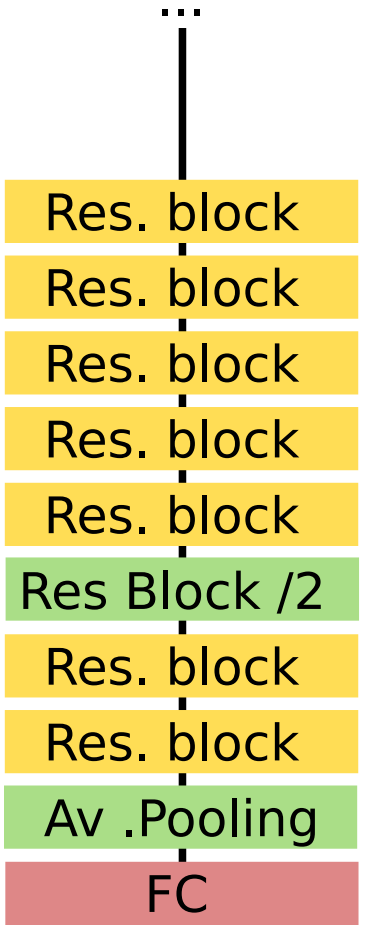


Court circuit sur les convolutions.

⇒ impose aux nouvelles convolution d'apprendre quelque chose de différent de la couche précédente

⇒ Meilleure circulation des gradients. Possibilité de réseaux plus profonds.





SegNet

ONERA
THE FRENCH AEROSPACE LAB

- Segmentation sémantique
  - Le unpooling layer pour retrouver la localité en mémorisant les emplacement des activations

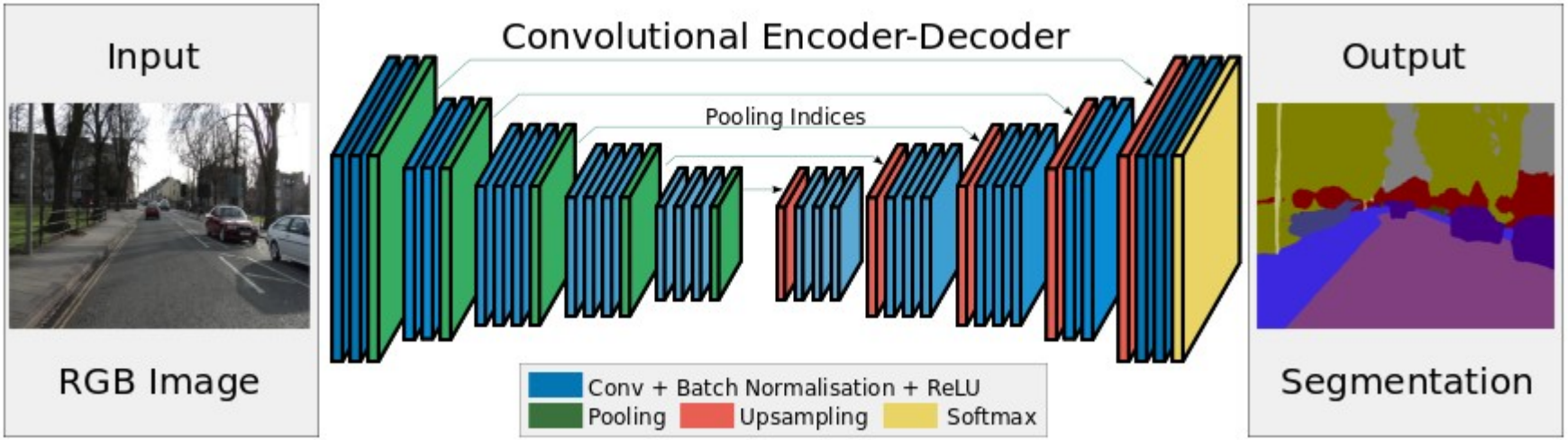


Image : http://mi.eng.cam.ac.uk/projects/segnet/

FasterRCNN

ONERA
THE FRENCH AEROSPACE LAB

- Détection d'objets

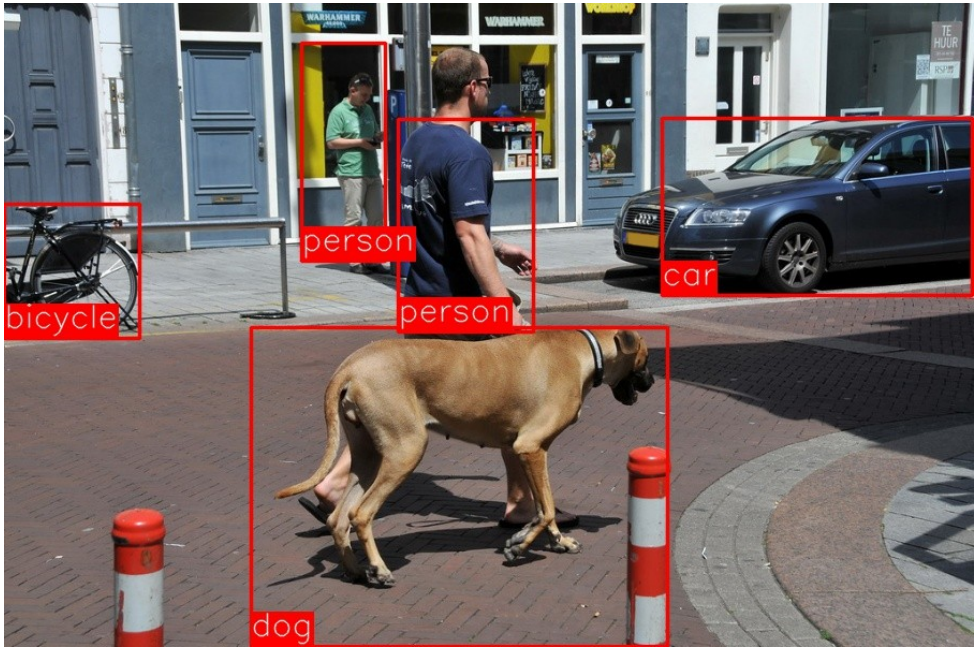


Image : https://github.com/mitmul/chainer-fast-rcnn

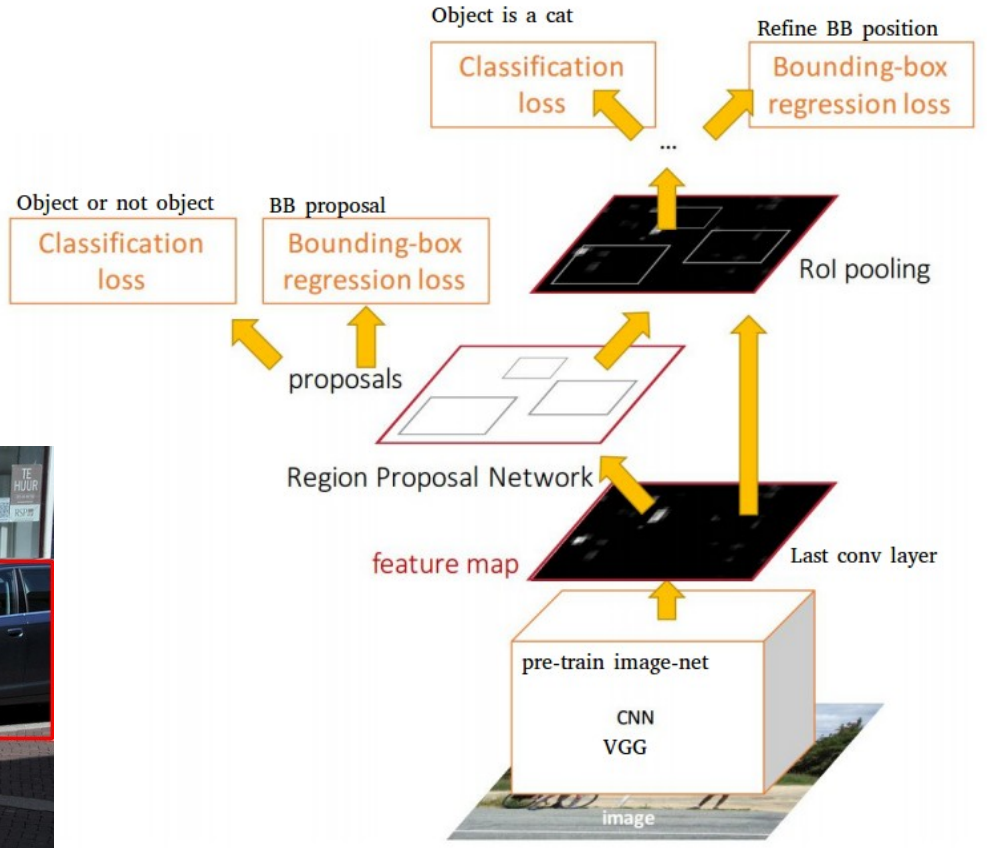
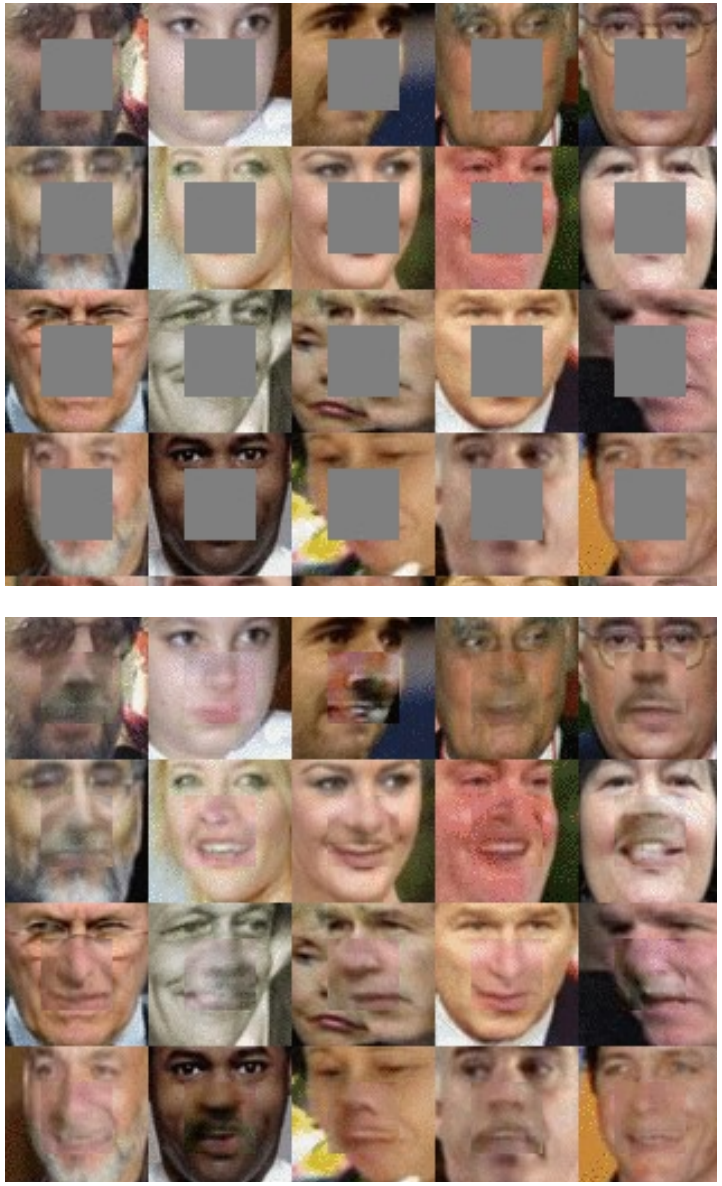


Image : https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/object_localization_and_detection.html

Generative Adversarial networks

ONERA
THE FRENCH AEROSPACE LAB



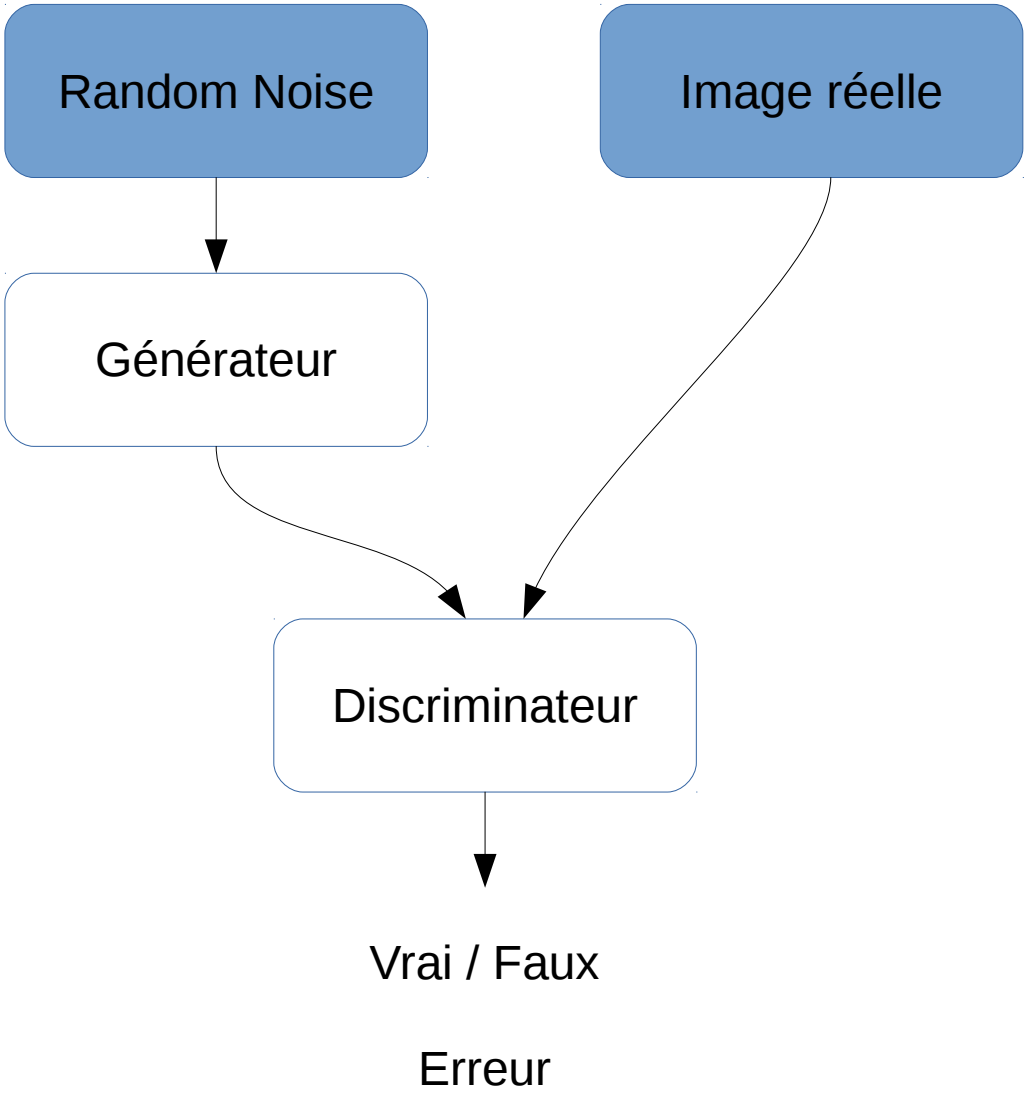
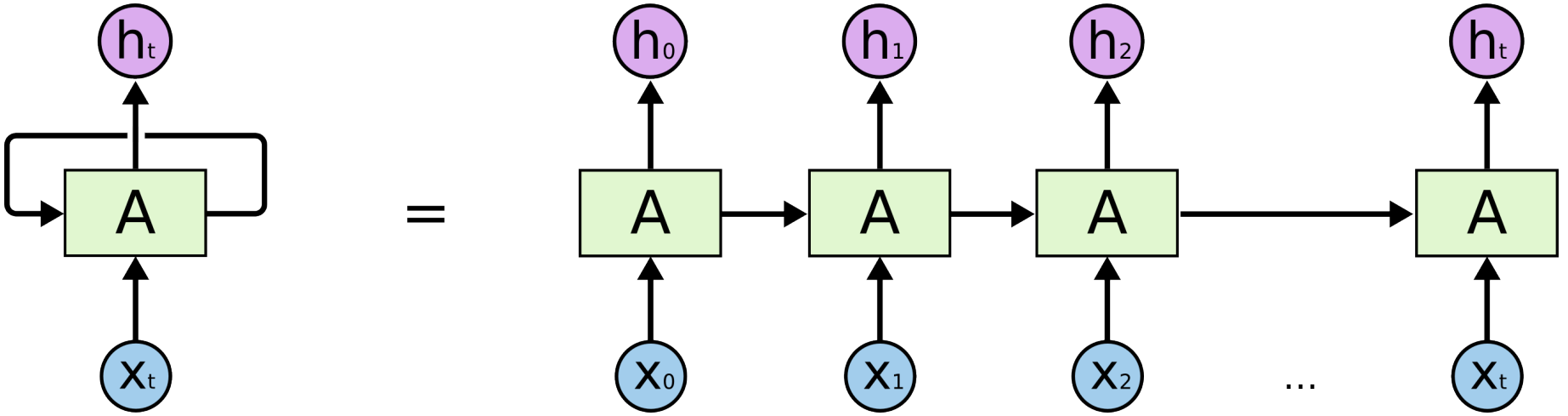


Image : https://bamos.github.io/2016/08/09/deep-completion/

Réseaux récurrents



Traitement de données temporelles :
La sortie de l'état T+1 dépend de T

Analyse de textes, de vidéos, captioning (description d'images avec du texte) ...

Image : http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Autoencoders

ONERA
THE FRENCH AEROSPACE LAB

- Que faire quand on n’a pas de données labelisées ?

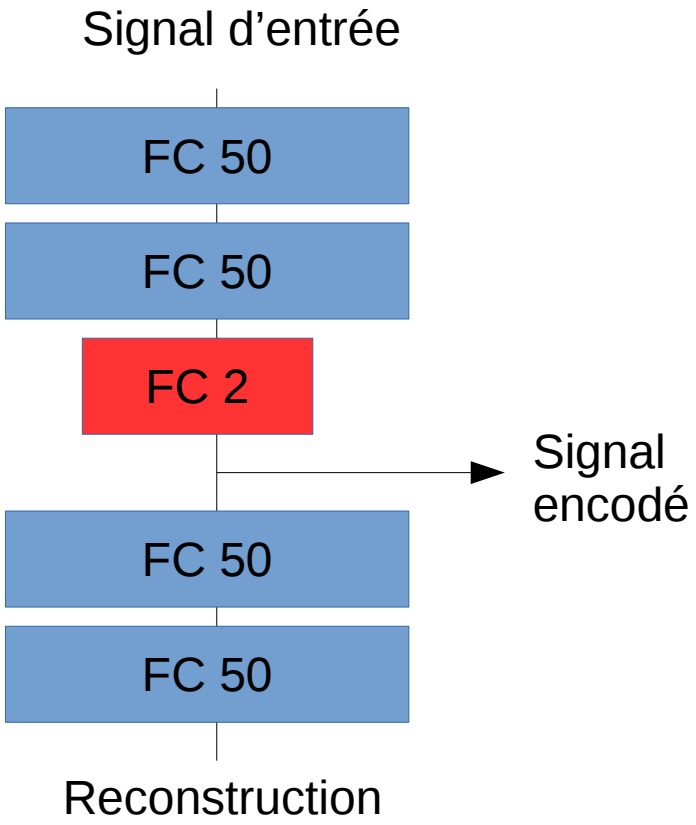


Image
http://cs.stanford.edu/people/karpathy/convnetjs/

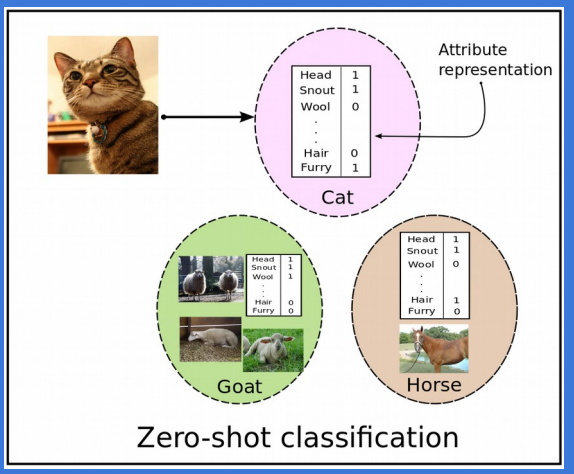
Le Deep learning à l'ONERA

ONERA
THE FRENCH AEROSPACE LAB

DTIM

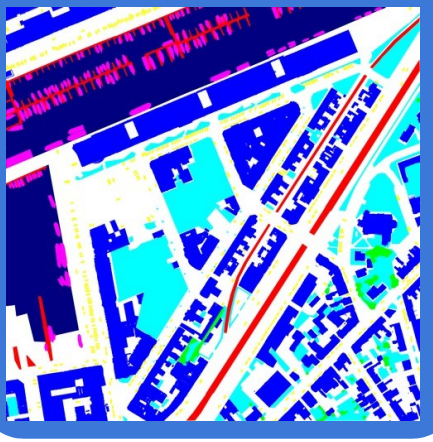
Détection

Zero Shot learning

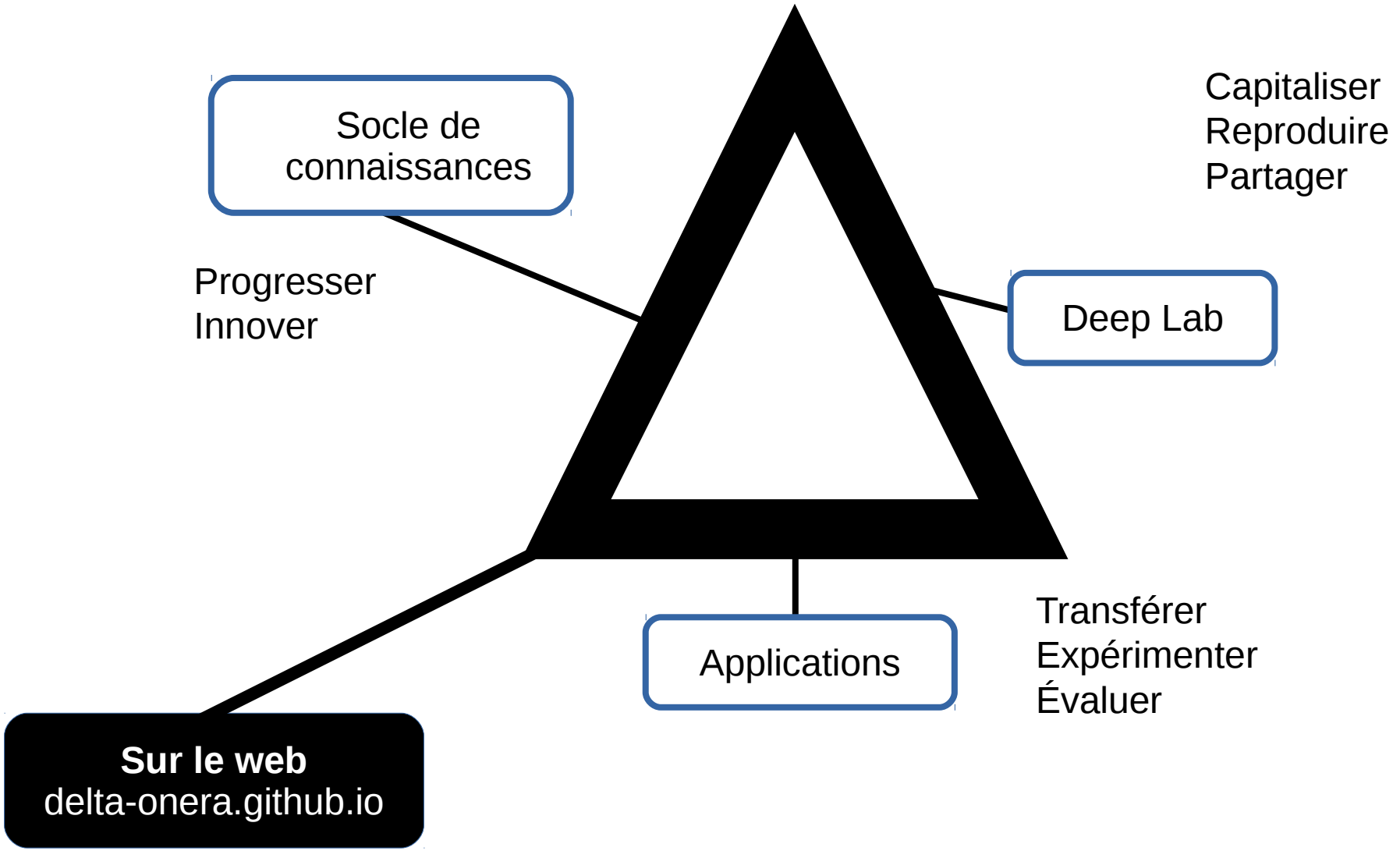


Nuages de points

Segmentation



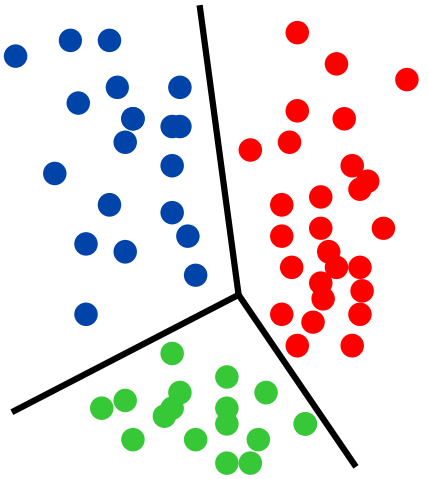
DeLTA


Socle de connaissances
Capitaliser
Reproduire
Partager
Progresser
Innover
Deep Lab
Applications
Transférer
Expérimenter
Évaluer
Sur le web
delta-onera.github.io

Conclusion

Données

Réseaux de neurones profonds



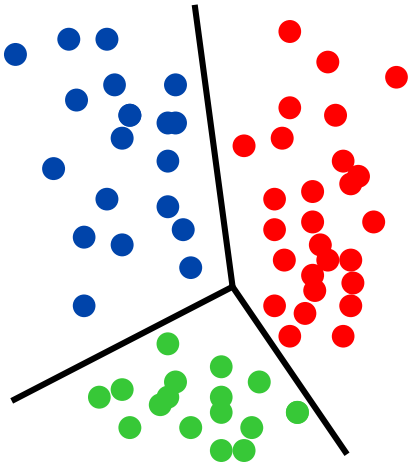
Objectif

Recherche sur les architectures, les stratégies, la mise en forme des données

Conclusion

Données

Réseaux de neurones profonds



Objectif

Recherche sur les architectures, les stratégies, la mise en forme des données

Ressources

ONERA
THE FRENCH AEROSPACE LAB

Gradient-based learning applied to document recognition,LeCun,Bottou, Bengio and Haffner, 1998

Batch normalization: Accelerating deep network training by reducing internal covariate shift, Ioffe and Szegedy, 2015

Dropout: a simple way to prevent neural networks from overfitting, Srivastava, Hinton, Krizhevsky, Sutskever and Salakhutdinov,2014

Understanding the difficulty of training deep feedforward neural networks, Glorot and Bengio

Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Duchi, Hazan and Singer, 2011

Adam: a Method for Stochastic Optimization, Kingma and Ba, 2015

Very Deep Convolutional Networks for Large-Scale Image Recognition, Simonyan and Zisserman

Deep residual learning for image recognition, He, Zhang, Ren, and Sun, 2016

Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding, Kendall, Badrinarayanan and Cipolla, 2015

Faster r-cnn: Towards real-time object detection with region proposal networks,Ren He, Girshick and Sun, 2015

Improving semantic embedding consistency by metric learning for zero-shot classiffication,Bucher and Herbin and Jurie, 2016

Processing of Extremely High-Resolution LiDAR and RGB Data: Outcome of the 2015 IEEE GRSS Data Fusion Contest--Part A: 2-D Contest, multiple authors, 2016

Semantic Image Inpainting with Perceptual and Contextual Losses, Yeh and al

Ressources

ONERA
THE FRENCH AEROSPACE LAB

Principaux frameworks

Tensorflow : tensorflow.org

PyTorch et Torch : pytorch.org torch.ch

Caffe : caffe.berkeleyvision.org/

Tutoriaux

Classification de chiffres sur MNIST :

tensorflow.org/tutorials/mnist/pros/

Classification d’images sur CIFAR 10

https://github.com/pytorch/tutorials