

Détection automatique de la couverture du sol à partir d'images satellites

MODÈLES DE DEEP LEARNING APPLIQUÉS À LA SEGMENTATION SÉMANTIQUE

Clément Guillo

07/04/2022



PLAN DE LA PRÉSENTATION

1 CONTEXTE

- Le challenge Data
- Les données en entrée

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

- Analyse quantitative des modèles
- Analyse qualitative des résultats

4 BILAN

1 CONTEXTE

- Le challenge Data
- Les données en entrée

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

- Analyse quantitative des modèles
- Analyse qualitative des résultats

4 BILAN

DESCRIPTION DU CHALLENGE

- Le site web <https://challengedata.ens.fr/> met à disposition des challenges Data autour de l'apprentissage supervisé.
- Challenge proposé par des acteurs issus du milieu scientifique ou des entreprises
- Le challenge proposé par Preligens (ex : Earthcube) propose de travailler sur des images satellites
- Objectif : être en mesure de détecter la distribution par type d'occupation du sol pour une image satellite donnée
- 10 classes possibles

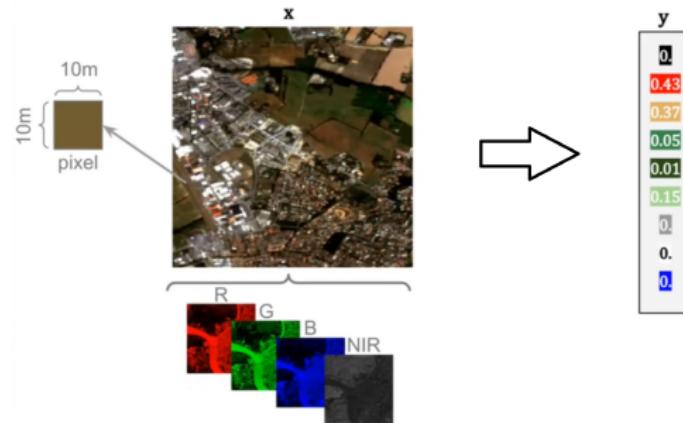
A QUOI ÇA SERT ?

- Connaître son territoire dans le temps et l'espace, capacité de production, en niveau ou évolution
- Minimiser les coûts de collecte (Exemple : enquête Teruti)

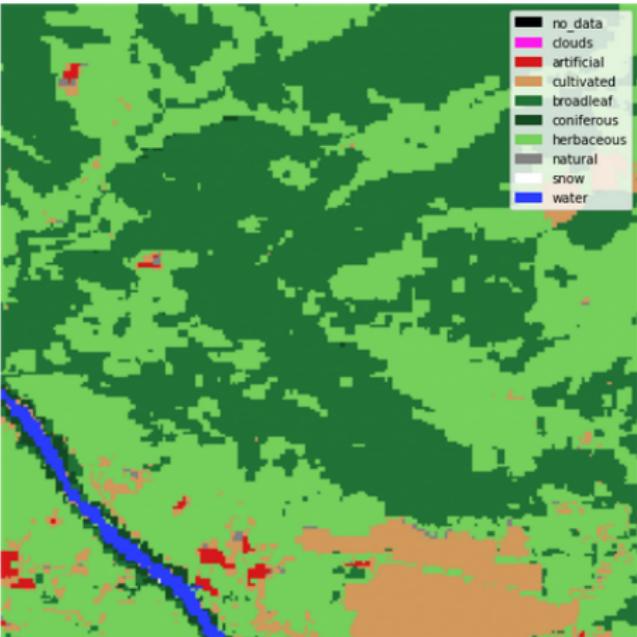
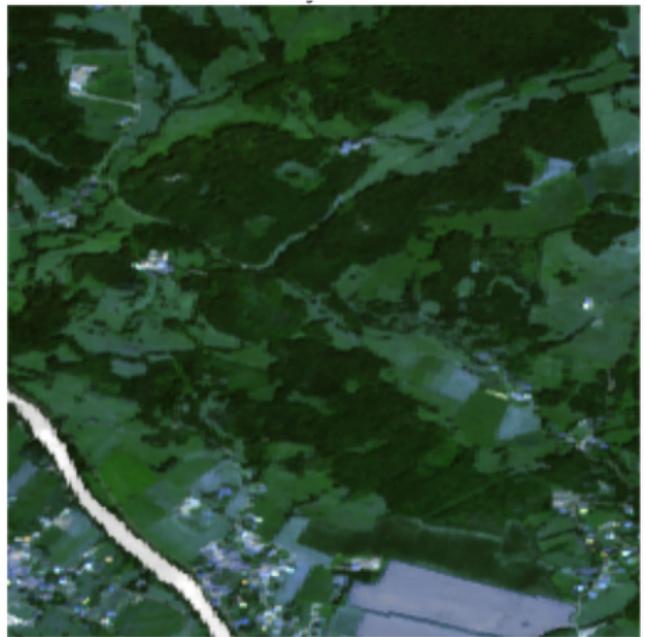


DESCRIPTION DU PROBLÈME

- On veut construire ici un algorithme qui :
 - Prend en entrée une image satellite de taille 256×256 pixels (4 channels : RGB et Infrarouge)
 - Retourne en sortie un vecteur de taille 10 $y = (y_0, \dots, y_9)$ donnant la distribution des pixels par catégorie : $\sum_{i=0}^9 y_i = 1$



LES DONNÉES EN entrée

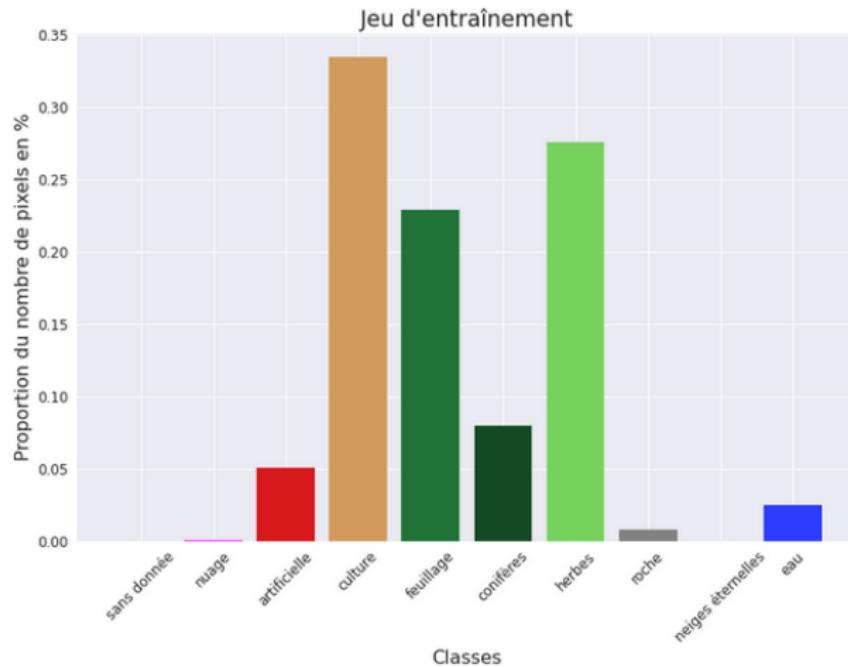


- Un ensemble de couples (image, masque) est à notre disposition :
 - ▶ L'image est une photographie du sol
 - ▶ Le masque est en fait une matrice de taille 256x256 attribuant à chaque pixel(i,j) de l'image une catégorie d'occupation du sol codée de 1 à 10

LES DONNÉES EN ENTRÉE

- Les images satellites proviennent de l'European Space Agency (ESA)
- Les masques sont issus du projet Sentinel-2 Global Land Cover S2GLC lancé par l'ESA dont le but principal est de créer des cartes d'occupation du sol de haute résolution
- On dispose :
 - ▶ D'un jeu d'entraînement de 18000 couples (image,masque)
 - ▶ d'un jeu de test de 5000 images sans les masques associés → On veut prédire une distribution des pixels par catégorie sur ces images

RÉPARTITION DES CLASSES DANS LE JEU D'ENTRAÎNEMENT



QUELLE MESURE DE PERFORMANCE ?

- Un score de performance est mesuré par les organisateurs du challenge
- Ce score est calculé sur les distributions par catégorie estimées par l'algorithme sur les 5000 images du jeu de test
- Pour une image satellite de "vraie" distribution par catégorie $y = (y_1, \dots, y_{10})$ *connue uniquement des organisateurs du challenge* et de distribution $\hat{y} = (\hat{y}_1, \dots, \hat{y}_{10})$ prédite par l'algorithme la divergence de kullback-Leibler est calculée :

$$KL_{image} = \sum_{i=1}^{10} y_i \log \left(\frac{y_i}{\hat{y}_i} \right)$$

- Le score de l'algorithme est ensuite obtenu en appliquant la moyenne sur nos N= 5000 images du jeu de test :

$$Score_{algo} = \frac{1}{N} \sum_{image \in test} KL_{image}$$

- Le score pris tel quel est peu informatif → quelques scores associés à des algorithmes plus ou moins performants

Type Algo	Score
Benchmark	0.0593
Réseau Simple	0.23
Aléatoire	0.63

1 CONTEXTE

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

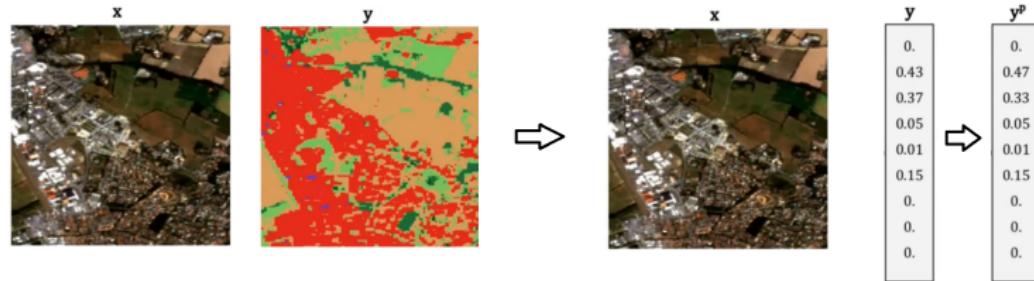
3 RÉSULTATS

4 BILAN

STRATÉGIE DE RÉSOLUTION

On va se servir des images et masques fournis par le challenge pour entraîner un algorithme de façon supervisée, **2 stratégies possibles** :

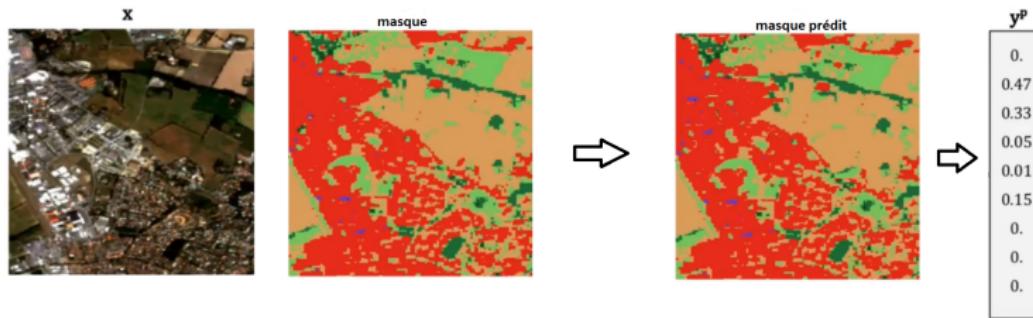
- ① En partant des masques fournis dans le jeu d'entraînement : construire le vrai vecteur de probabilité à partir du masque fourni et entraîner l'algorithme à le reproduire



En faisant cela on diminue la précision du signal en entrée de l'entraînement : on n'utilise pas l'information spatiale et l'agencement des types d'occupation du sol dans l'espace → Plus proche de la statistique cependant

STRATÉGIE DE RÉSOLUTION

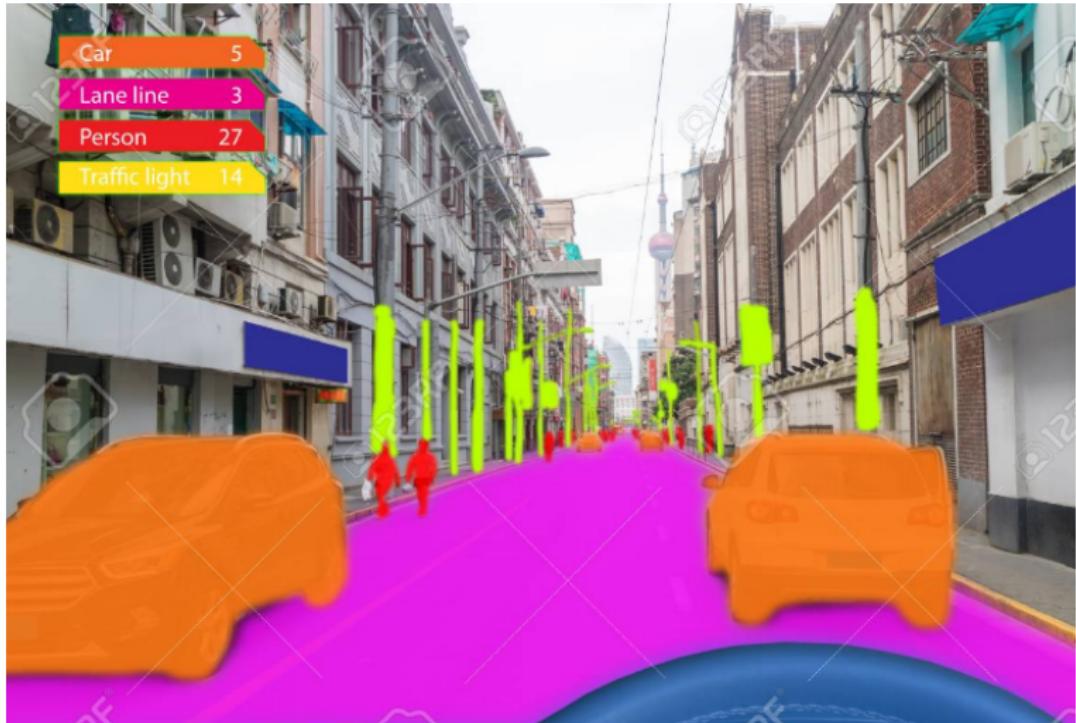
- ② En partant des masques fournis dans le jeu d'entraînement : entraîner l'algorithme à reproduire un masque puis calculer la distribution des types d'occupation du sol induite par le masque calculé



- Le problème ainsi posé tient compte de l'information spatiale contenue dans les masques
- C'est cette stratégie que nous allons suivre dans la suite

UN PROBLÈME DE SEGMENTATION SÉMANTIQUE

- Le problème qui consiste à classer des zones de l'image dans telle ou telle catégorie est un problème de *segmentation sémantique*



1 CONTEXTE

2 RÉSOLUTION

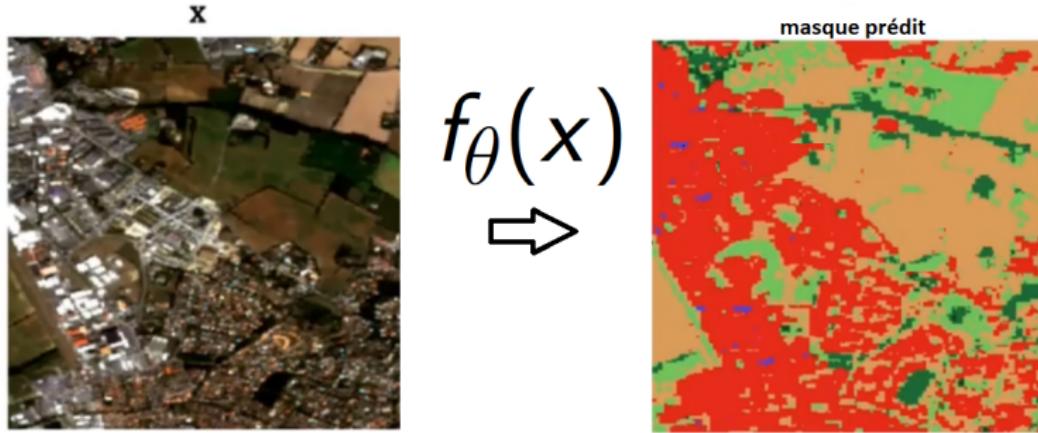
- Stratégie de Résolution
- **Formalisation du problème**
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

4 BILAN

FORMALISATION

- On peut modéliser "l'algorithme" comme une fonction de paramètres $\theta \in \mathbb{R}^d$ qui à une image x donnée associe son masque prédict $f_\theta(x)$



- Cette formalisation est classique en apprentissage statistique, l'objectif visé ici est d'entraîner notre modèle $f_\theta(x)$, i.e trouver le jeu de paramètres optimal θ^* qui conduira aux masques prédicts de meilleure qualité (qualité à définir par la suite)

FORMALISATION

Dans notre situation :

- Les modèles de deep learning que nous utilisons et détaillons par la suite sont des fonctions $f_\theta(x)$ avec une structure en réseau
- Le nombre d de paramètres contenus dans θ est extrêmement élevé
- De plus, les images x en entrée des fonctions $f_\theta(x)$ sont des tableaux de taille 256x256x4

Le calcul de la fonction $f_\theta(x)$ pour une image donnée peut donc être coûteux !

→ Ce qui va induire une stratégie d'entraînement du modèle modifiée par rapport au cadre classique

1 CONTEXTE

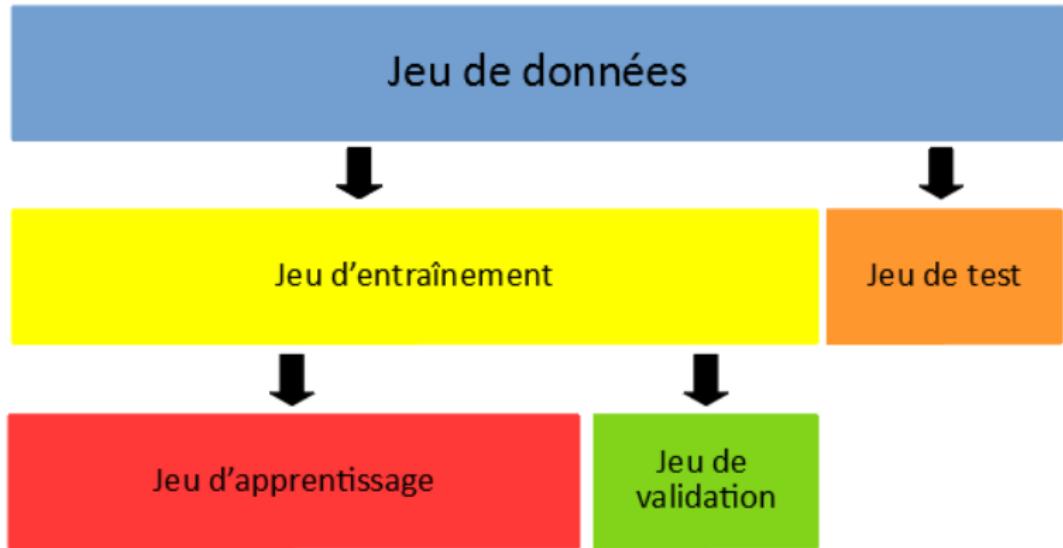
2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement**
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

4 BILAN

ORGANISATION DE L'ENTRAÎNEMENT



- Pour le jeu de test nous n'avons pas de masques à disposition, donc nous n'avons pas la vraie distribution (nous cherchons à la calculer)
- On alloue une partie du jeu d'entraînement à l'évaluation de notre modèle → jeu de validation
- L'autre partie servira à entraîner le modèle → jeu de d'apprentissage

ZOOM SUR f_θ

- La fonction $f_\theta(x)$ prend en fait une image x en entrée et associe à chacun des 256×256 pixels de l'image un vecteur de probabilité $v_{pixel} = (\hat{p}_1, \dots, \hat{p}_{10})$ tel que $\sum_{i=1}^{10} \hat{p}_i = 1$
- on attribue ensuite à chaque pixel la catégorie j telle que :

$$j = \underset{i \in (1..10)}{\operatorname{argmax}} \hat{p}_i$$

- Ceci permet de reconstituer un masque pour l'image x

DÉFINITION DE L'ERREUR

- Pour orienter la recherche du jeu de paramètre θ^* il faut avoir une idée de l'erreur commise par le modèle
- Nous allons définir pour une image du jeu d'apprentissage donnée une erreur

$$E_\theta(x) = \frac{1}{256 \times 256} \sum_{p \in \text{pixels}} e_\theta(p)$$

- $e_\theta(p)$ est l'erreur commise sur un pixel définie via la distance de Kullback-Leibler :

$$e_\theta(p) = \sum_{i=1}^{10} p_i \log \left(\frac{p_i}{\hat{p}_i} \right)$$

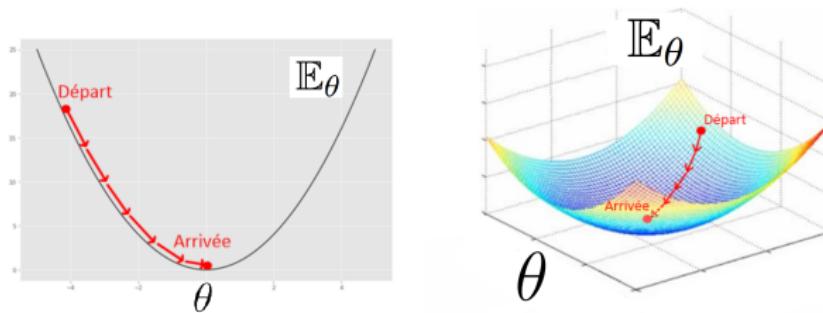
- Avec $p = (p_1, \dots, p_n)$ la vraie distribution du pixel telle que $p_j = 1$ pour la catégorie labellisée pour ce pixel et 0 partout ailleurs

DESCENTE DE GRADIENT

- On pourrait donc calculer l'erreur moyenne commise \mathbb{E}_θ sur nos N_a images du jeu d'apprentissage :

$$\mathbb{E}_\theta = \frac{1}{N_a} \sum_{x \in \text{apprentissage}} E_\theta(x)$$

- .. Et trouver le paramètre θ^* optimal par descente de gradient



- Pas réalisable en pratique, une itération sur le jeu d'apprentissage (18 000 images) peut prendre beaucoup de temps ! → **descente de gradient stochastique**

DESCENTE DE GRADIENT STOCHASTIQUE

- L'idée principale est **d'estimer** l'erreur à partir d'un nombre d'images réduit pour que le calcul d'une itération soit moins coûteux
- Pour ce faire on sélectionne aléatoirement un sous-ensemble \mathbb{S} de taille n_b du jeu d'apprentissage sur lequel on calcule l'erreur approximée :

$$\hat{\mathbb{E}}_{\theta} = \frac{1}{n_b} \sum_{x \in \mathbb{S}} E_{\theta}(x)$$

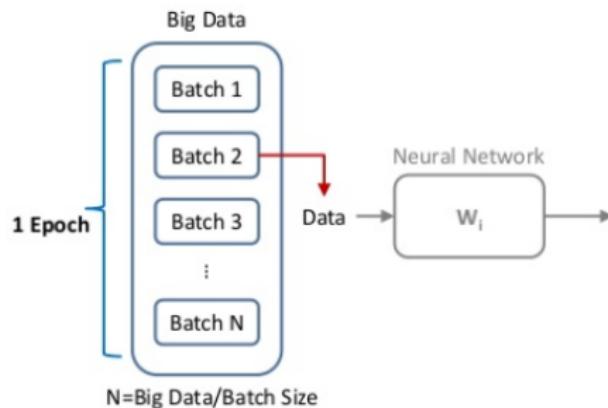
- Si le jeu d'images d'apprentissage est homogène, $\hat{\mathbb{E}}_{\theta}$ approxime correctement la vraie erreur \mathbb{E}_{θ}
- Ceci permet de réduire le cout associé à une itération

DESCENTE DE GRADIENT STOCHASTIQUE : EN PRATIQUE

- En pratique, on réalise une partition aléatoire de notre jeu d'apprentissage en différents "batchs"
- Pour chacun des batchs b , on calcule l'erreur approximée $\hat{\mathbb{E}}_\theta^b$ et on réalise une itération
- Une fois l'ensemble des batchs parcourus, on dit qu'une **epoch** est réalisée, on recommence l'opération en repartitionnant aléatoirement le jeu d'apprentissage —→ on réitère sur plusieurs epochs

DESCENTE DE GRADIENT STOCHASTIQUE : EN PRATIQUE

- En pratique, on réalise une partition aléatoire de notre jeu d'apprentissage en différents "batchs" de même taille
- Pour chacun des batchs b , on calcule l'erreur approximée \hat{E}_θ^b et on réalise une itération
- Une fois l'ensemble des batchs parcourus, on dit qu'une **epoch** est réalisée, on recommence l'opération en repartitionnant aléatoirement le jeu d'apprentissage



DESCENTE DE GRADIENT STOCHASTIQUE : L'ALGORITHME

Algorithme 1 Descente de gradient stochastique (SGD)

- 1: Initialisation : $\theta \leftarrow \theta^0$
 - 2: α , taux d'apprentissage
 - 3: N , nombre d'epochs
 - 4: **for** $i \leftarrow 1$ to N **do**
 - 5: Sélection du *batch* b
 - 6: Calcul de l'erreur du *batch* b : \hat{E}_θ^b
 - 7: Calcul du gradient $\nabla \hat{E}_\theta^b$
 - 8: $\theta \leftarrow \theta - \alpha \nabla \hat{E}_{\theta_t}^b$
 - 9: **end for**
-

DESCENTE DE GRADIENT STOCHASTIQUE : QUELLE TAILLE POUR LES BATCHS ?

- Le choix de la taille du batch repose sur un antagonisme entre :
 - ▶ **Le temps de calcul** alloué à une itération : plus la taille du batch est grande plus le temps de calcul nécessaire à une itération de la descente de gradient stochastique est élevé
 - ▶ **La précision** associée à une itération : plus la taille du batch est petite plus l'erreur estimée risque d'être de mauvaise qualité (variabilité élevée due au faible taux d'échantillonnage)
- La bonne taille de batch est celle qui respecte la règle du juste milieu !

1 CONTEXTE

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

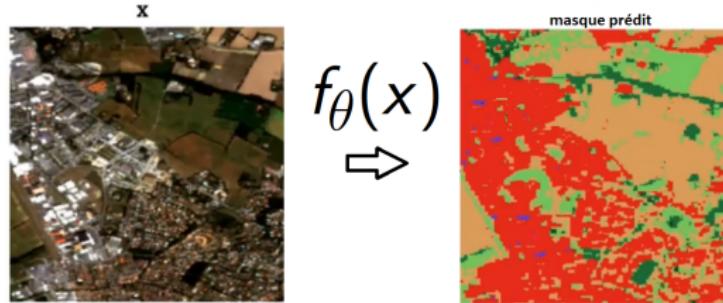
4 BILAN

OUVERTURE DU CAPOT : LES MODÈLES DE DEEP LEARNING

- A ce stade la stratégie d'entraînement du modèle à partir du jeu d'apprentissage est explicitée :

$$f_{\theta} \longrightarrow \mathbb{E}_{\theta} \longrightarrow \theta_t \longrightarrow \theta_{t+1} \longrightarrow \theta^*$$

- Il est temps de préciser la forme de la fonction $f_{\theta}()$!



LES RÉSEAUX DE NEURONES

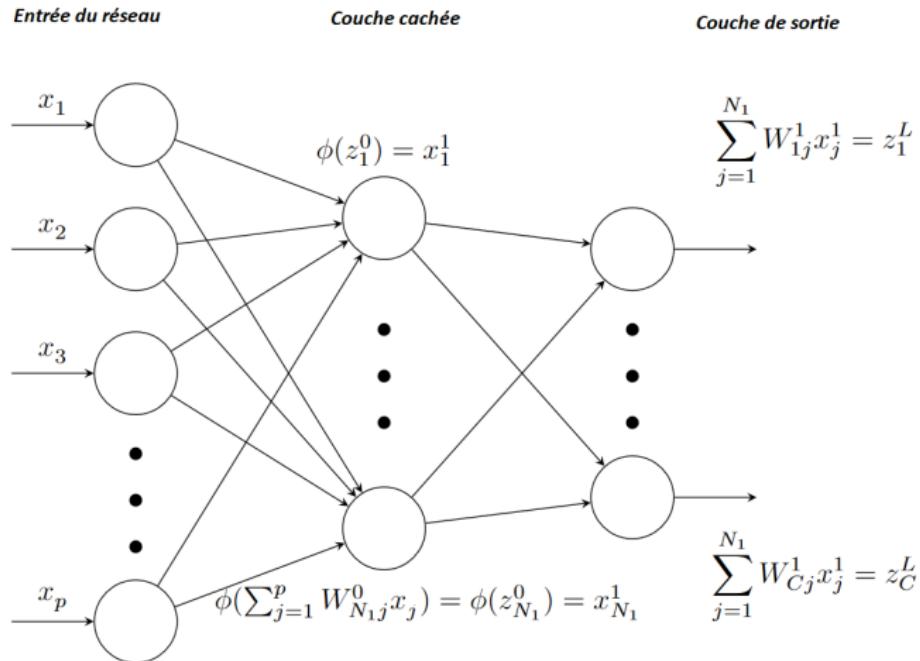
- Nous nous intéressons ici seulement aux modèles de deep learning, et ensuite une sous-catégorie de modèle particulièrement adapté au traitement de l'image, les réseaux de neurones convolutifs
- Formellement un réseau de neurones prend en entrée un vecteur $X = (x_1, \dots, x_p)$ et retourne

$$f_{\theta}(X) = W_N \phi(W_{N-1}(\dots \phi(W_1(\phi(W_0 X + b_0)) + b_1) \dots) + b_{N-1}) + b_N$$

Avec :

- ▶ $\theta = (W_0, \dots, W_N, b_0, \dots, b_n)$
- ▶ $(W_i)_{0 \leq i \leq N}$ les *matrices de poids*
- ▶ $(b_i)_{0 \leq i \leq N}$ les *biais*
- ▶ ϕ une non-linéarité appliquée composante par composante sur le vecteur de sortie : $\phi(x_1, \dots, x_p) = (\phi(x_1), \dots, \phi(x_p)) \longrightarrow$ ce n'est pas un paramètre mais un hyper-paramètre !
- On comprend rapidement pourquoi le nombre de paramètres peut exploser (Stanford, 11,2 milliards de paramètres)

LES RÉSEAUX DE NEURONES : SCHÉMATIQUEMENT



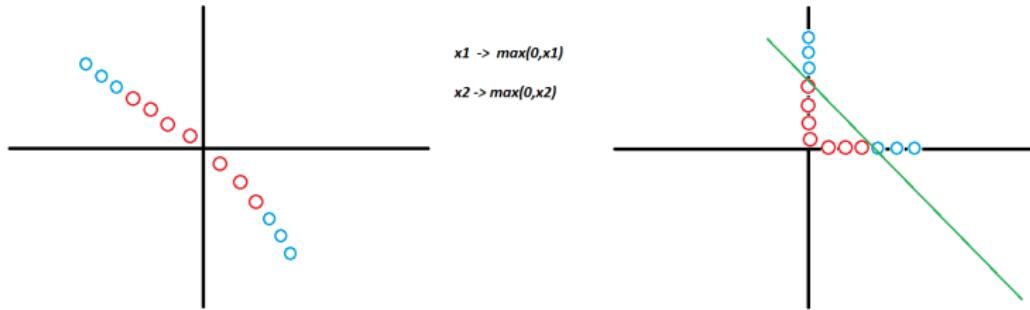
Si l'on veut s'amuser avec des réseaux de neurone → [ici](#)

LES RÉSEAUX DE NEURONES : MOTIVATION

- le théorème d'approximation universelle stipule que pour n'importe quelle fonction $g : X \mapsto g(X)$ continue, il existe toujours un réseau de neurones à une seule couche cachée $f_\theta()$ qui approxime $g()$
- Le théorème ne dit rien sur la façon de le trouver..
- Etant donné la forme de la fonction, le calcul des gradients est très simple, *backpropagation* ou plus sobrement : dérivé composée
- Calculs de gradient parallélisables —> marche bien avec des GPU !

LES RÉSEAUX DE NEURONES : LA NON-LINÉARITÉ

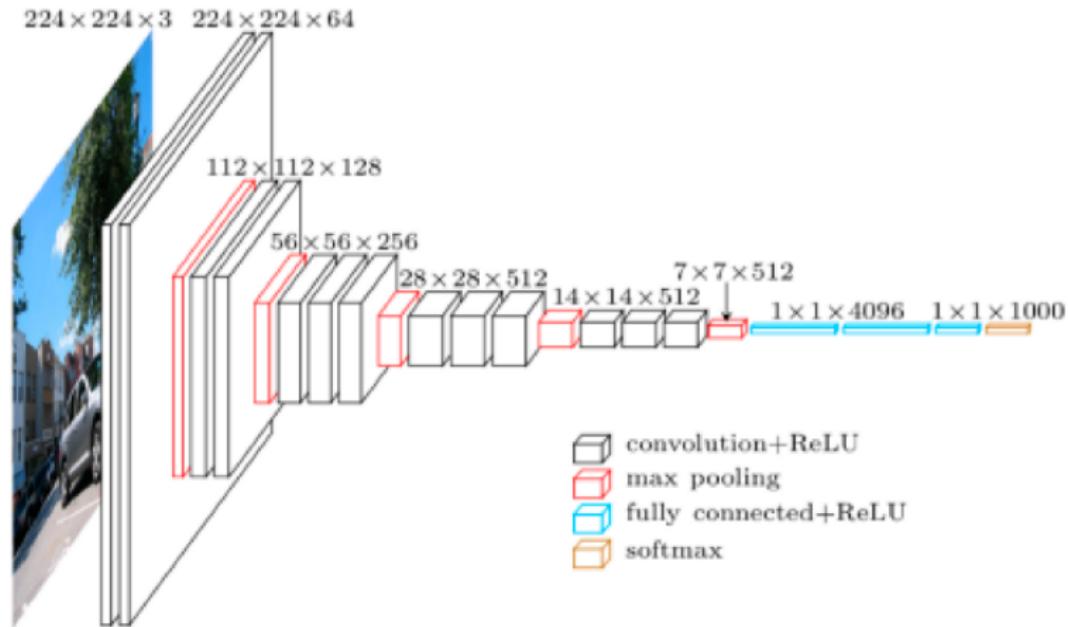
- Lors d'un exercice de classification (exemple classification d'une image dans la catégorie "chat" ou "chien"), on veut en général séparer des points entre eux.
- Si l'ensemble des points n'est pas séparable linéairement, on ne pourra pas faire grand chose sans introduire de la non linéarité !



UN PEU DE STRUCTURE

- A ce stade seules les dimensions des données en entrée et celles du vecteur de sortie sont contraintes (vecteur 0,1 dans le cas d'une classification, masque dans le cas du problème de segmentation sémantique)
- Le nombre de matrices intermédiaires et leur dimension ne sont pas paramétrés, ce sont des hyper-paramètres également !
- Empiriquement plus on met de la profondeur, plus on est performant (cf Resnet etc..) d'où l'appellation *Deep Learning*
- Aucune contrainte n'est donnée sur les coefficients des matrices W_i → un peu brutal car lorsqu'on travaille sur des images on a de l'information a priori, notamment l'invariance par translation

LE RÉSEAU DE NEURONE CONVOLUTIF CLASSIQUE



- Notre objectif est de comprendre cette représentation classique des réseaux de neurones convolutifs (CNN)

QU'EST-CE QU'UNE COUCHE CONVOLUTIVE ?

- Une couche convulsive est composée de plusieurs filtres convolutifs
- Un filtre prend une image en entrée de dimension $256 \times 256 \times C$ avec C le nombre de channels (4 pour les images d'origine correspondant à R, G, B, InfraRouge)
- Et renvoie une image résultante de dimension 256×256

L'OPÉRATION DE CONVOLUTION

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

*

7	1
1	8

=

106		

L'OPÉRATION DE CONVOLUTION

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

*

7	1
1	8

=

106	70	

L'OPÉRATION DE CONVOLUTION

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

*

7	1
1	8

=

106	70	23

L'OPÉRATION DE CONVOLUTION

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

*

7	1
1	8

=

106	70	23
32		

L'OPÉRATION DE CONVOLUTION

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

*

7	1
1	8

=

106	70	23
32	59	51
40	41	42

LES RÉSEAUX DE NEURONE CONVOLUTIFS

- Passer une convolution sur une image revient à structurer la matrice de poids correspondante
- Le noyau de convolution passe par tous les pixels de l'image ainsi certains coefficients de la matrice de poids sont identiques
- notion de parcimonie → On peut ajouter plus de couches
- Bien noter ici que les noyaux de convolution font partie du jeu de paramètres θ dans $f_\theta()$!

L'OPÉRATION MAXPOOL

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

6	

L'OPÉRATION MAXPOOL

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

6	8

L'OPÉRATION MAXPOOL

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

6	8
3	

L'OPÉRATION MAXPOOL

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

6	8
3	4

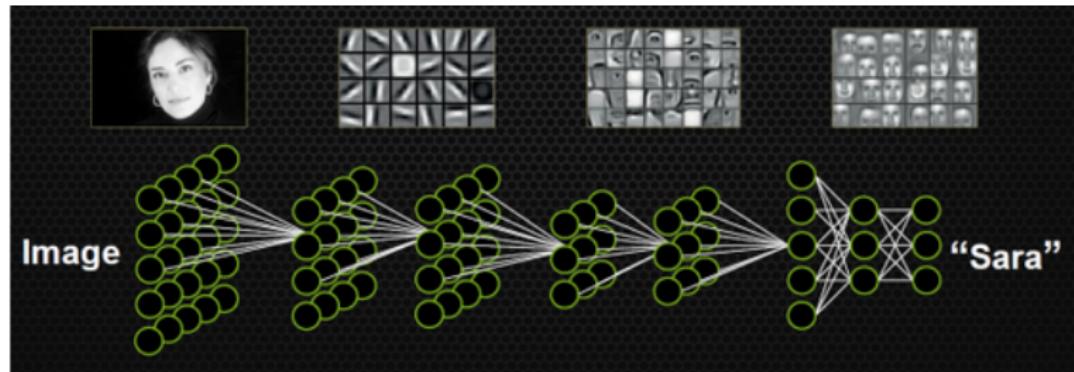
L'OPÉRATION MAXPOOL

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

6	8
3	4

- Ici aussi on réduit la dimension de la sortie, tout en retenant les coefficients les plus saillants, donc moins de coefficients sur les couches suivantes → on peut rajouter des couches !

POURQUOI EMPILER LES COUCHES ?

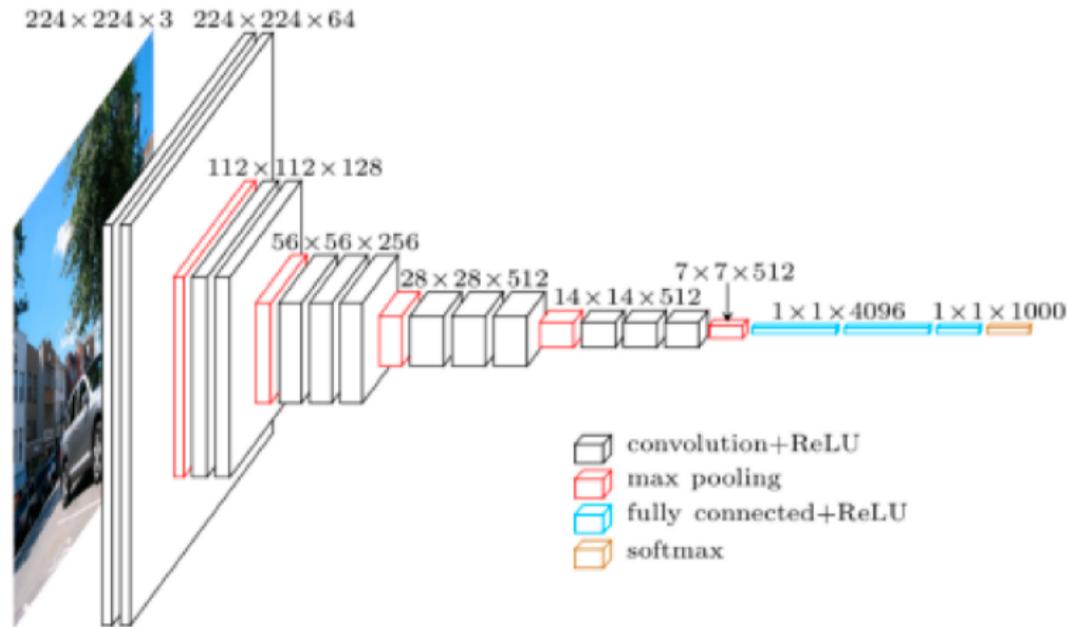


- On combine des briques élémentaires entre elles pour obtenir un résultat complexe
- Plus les couches sont profondes plus elles sont spécialisées, moins elles sont profondes plus elles sont abstraites (fine tuning)

QUELLE DIFFÉRENCE MAJEURE AVEC L'APPROCHE CLASSIQUE D'APPRENTISSAGE ?

- En apprentissage classique, le schéma de travail est le suivant :
 - ➊ Extraction de variables d'intérêts basées sur l'image (éventuellement à partir de convolution dont le noyau est fixé par l'expert)
 - ➋ Classifieur construit à partir des variables d'intérêt extraites ! (qui correspondrait à la dernière couche du réseau)
- Dans les réseaux de neurones convolutifs les noyaux de convolution sont des paramètres du modèle, ils sont donc modifiés à chaque itération du paramètre global θ —> **Le processus d'extraction des variables d'intérêt est donc automatisé !**

LE RÉSEAU DE NEURONE CONVOLUTIF CLASSIQUE

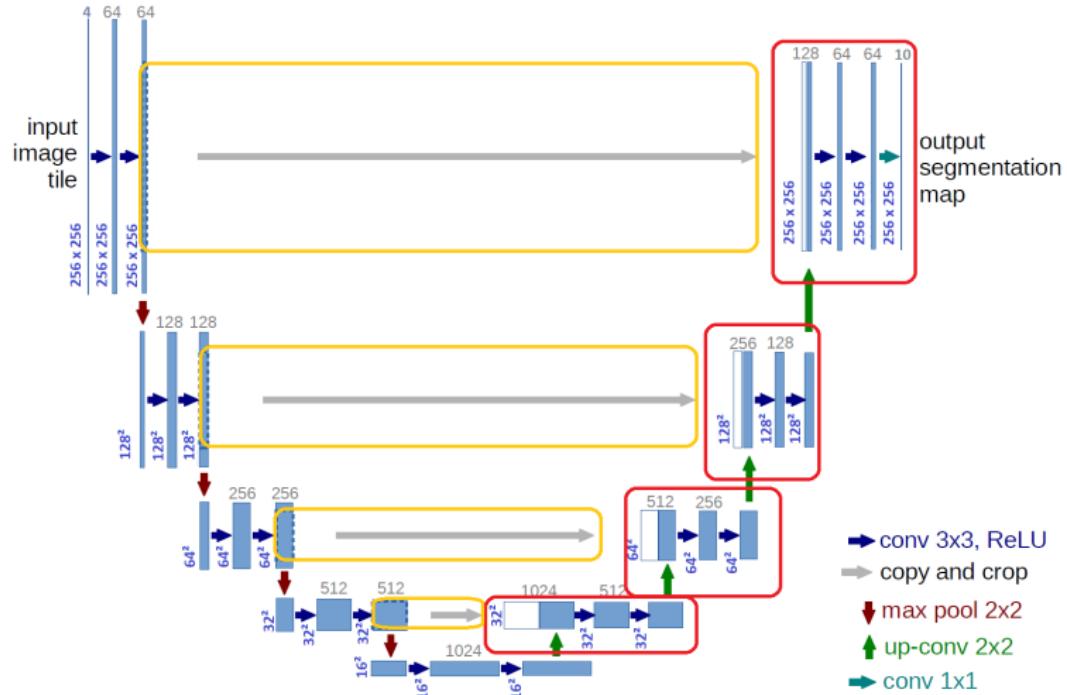


- On ne gère ici qu'un problème de classification, comment "remonter" vers une sortie au format 256×256 ?

LE U-NET

- Architecture de réseau initialement utilisé pour l'imagerie médicale
- Deux parties dans le réseau :
 - ▶ Une phase descendante qui va avoir la même structure qu'un réseau convolutif classique
 - ▶ une phase ascendante qui va dilater l'information produite par la phase descendante pour revenir au niveau pixel
- Une jonction entre les différents niveaux pour se prémunir de la dissipation de l'information à travers les couches

LE U-NET



CONVOLUTION INVERSE

$$\begin{array}{|c|c|} \hline 1 & 6 \\ \hline 3 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 7 & 1 \\ \hline 1 & 8 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 7 & 1 & \\ \hline 1 & 8 & \\ \hline & & \\ \hline \end{array}$$

CONVOLUTION INVERSE

$$\begin{array}{|c|c|} \hline 1 & 6 \\ \hline 3 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 7 & 1 \\ \hline 1 & 8 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & 42 & 6 \\ \hline & 6 & 48 \\ \hline & & \\ \hline \end{array}$$

CONVOLUTION INVERSE

$$\begin{array}{|c|c|} \hline 1 & 6 \\ \hline 3 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 7 & 1 \\ \hline 1 & 8 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & & \\ \hline 21 & 3 & \\ \hline 3 & 24 & \\ \hline \end{array}$$

CONVOLUTION INVERSE

$$\begin{array}{|c|c|} \hline 1 & 6 \\ \hline 3 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 7 & 1 \\ \hline 1 & 8 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & 14 & 2 \\ \hline & 2 & 16 \\ \hline \end{array}$$

CONVOLUTION INVERSE

$$\begin{array}{|c|c|c|} \hline 7 & 1 & \\ \hline 1 & 8 & \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline & 42 & 6 \\ \hline & 6 & 48 \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline & & \\ \hline 21 & 3 & \\ \hline 3 & 24 & \\ \hline \end{array} \boxed{+} \begin{array}{|c|c|c|} \hline & & \\ \hline & 14 & 2 \\ \hline & 2 & 16 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 7 & 43 & 6 \\ \hline 22 & 31 & 2 \\ \hline 3 & 26 & 16 \\ \hline \end{array}$$

1 CONTEXTE

- Le challenge Data
- Les données en entrée

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

- Analyse quantitative des modèles
- Analyse qualitative des résultats

4 BILAN

1 CONTEXTE

2 RÉSOLUTION

3 RÉSULTATS

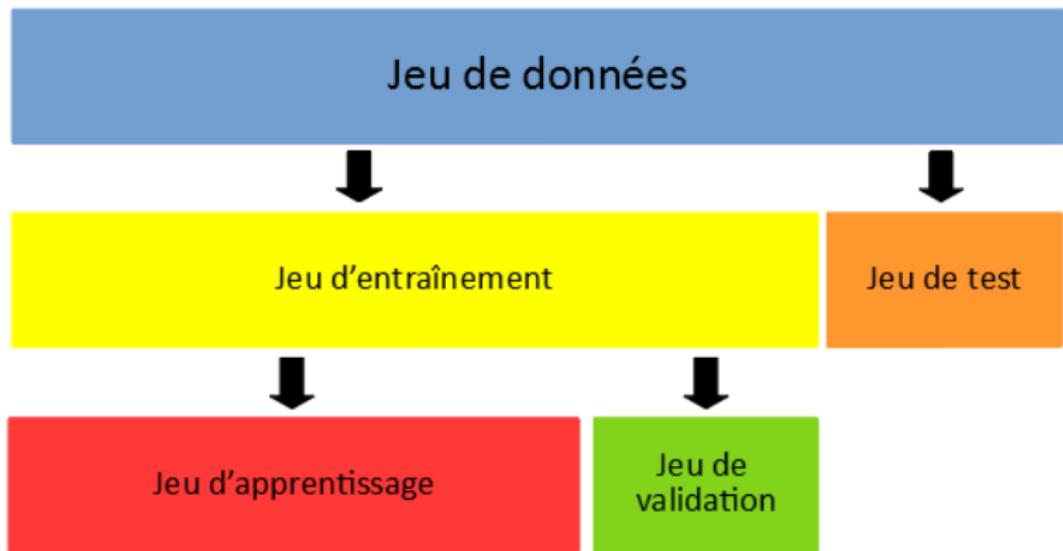
- Analyse quantitative des modèles
- Analyse qualitative des résultats

4 BILAN

LES OUTILS UTILISÉS

- Travail en Python sur le module Pytorch :
 - ▶ Outil accessible et performant
 - ▶ Peu de lignes de codes sont nécessaires en pratique
- Travail sur la plateforme Onxia du sspcloud
- Outil de monitoring [wandb](#)

EVALUATION DES MODÈLES



- On va tester nos modèles entraînés sur les 5000 images du jeu de validation

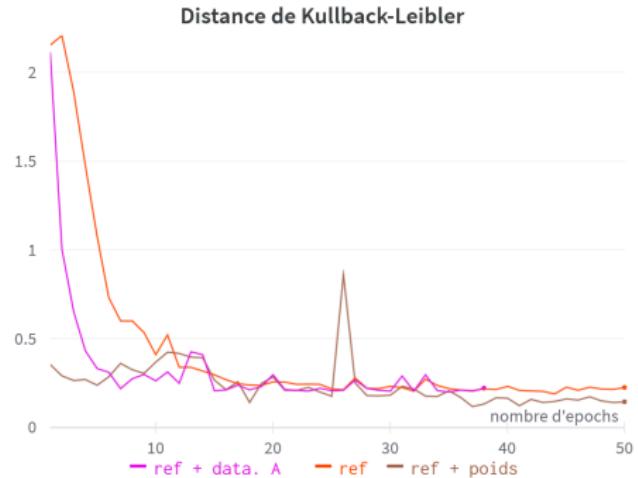
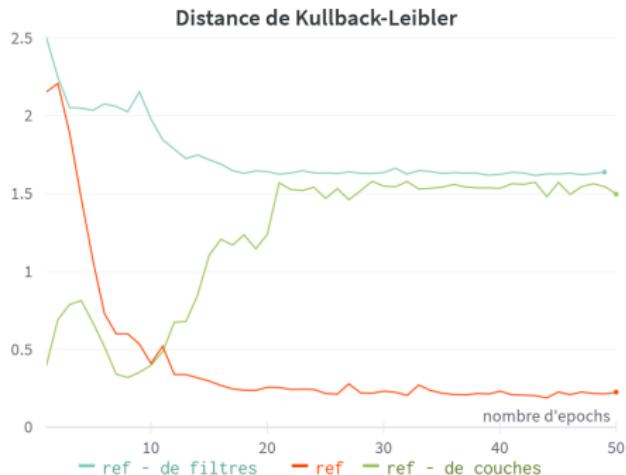
IMPACT DES HYPERPARAMÈTRES

Plusieurs hyperparamètres à considérer :

- La structure du réseau
- La profondeur du réseau
- le nombre de filtres en sortie de chaque couche
- le taux d'apprentissage
- la taille des batchs
- les pondérations des erreurs associées à chaque catégorie d'occupation du sol
- Et bien d'autres encore ..(batchnorm, dropout..)

nom scénario	n filtres	n couches	n param (millions)
référence	64	8	31,1
ref, - de filtres	6	8	4,4
ref, - de couches	64	4	1,7
ref + poids	64	8	31,1
ref + data. A	64	8	31,1

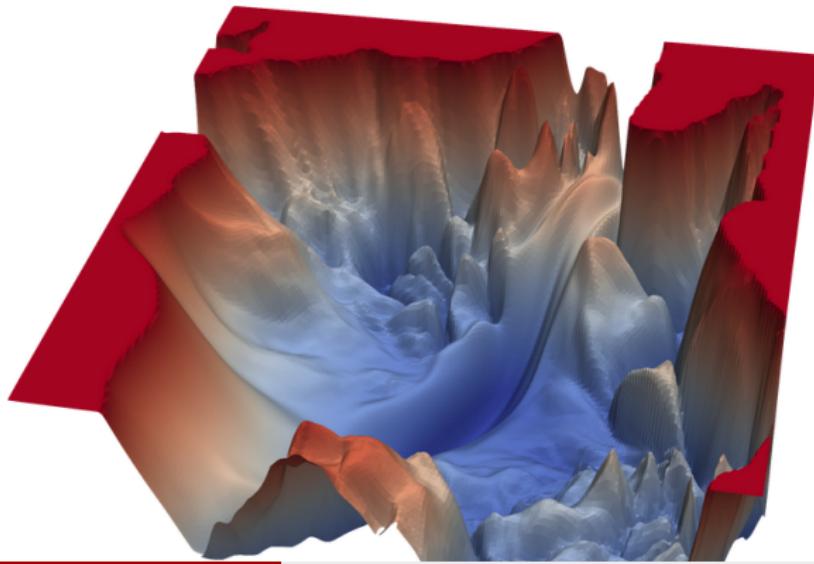
IMPACT DES HYPERPARAMÈTRES



- Fort effet de la profondeur et de la largeur du réseau sur le score
- Faible effet de l'augmentation de données

UN PETIT DÉTOUR SUR L'INSTABILITÉ OBSERVÉE

- On observe une forme d'instabilité assez forte de la fonction d'erreur suivant les epochs
- Et pour cause le problème est hautement non convexe et les minimas locaux sont omniprésents !
- [Plus de détails ici](#)



UN PETIT DÉTOUR SUR L'INSTABILITÉ OBSERVÉE

- Beaucoup de solutions sont proposées pour éviter les mauvais creux, elles ne sont pas abordées ici (sauts de taux d'apprentissage, taux d'apprentissage décroissant..)
- Etant donné le nombre de paramètres, on trouvera toujours une direction pour descendre !
- Comment aller plus loin que le score dans l'observation des performances ?

INTERSECTION OVER UNION

- On voudrait connaître la capacité du réseau à détecter certaines catégories plutôt que d'autres
- Plutôt que le score d'exactitude pour chaque classe qui ne prend pas en compte faux positifs et faux négatifs, on va utiliser l'IOU (Intersection Over Union)



$\text{iou} = 0$



$\text{iou} = 1/7$

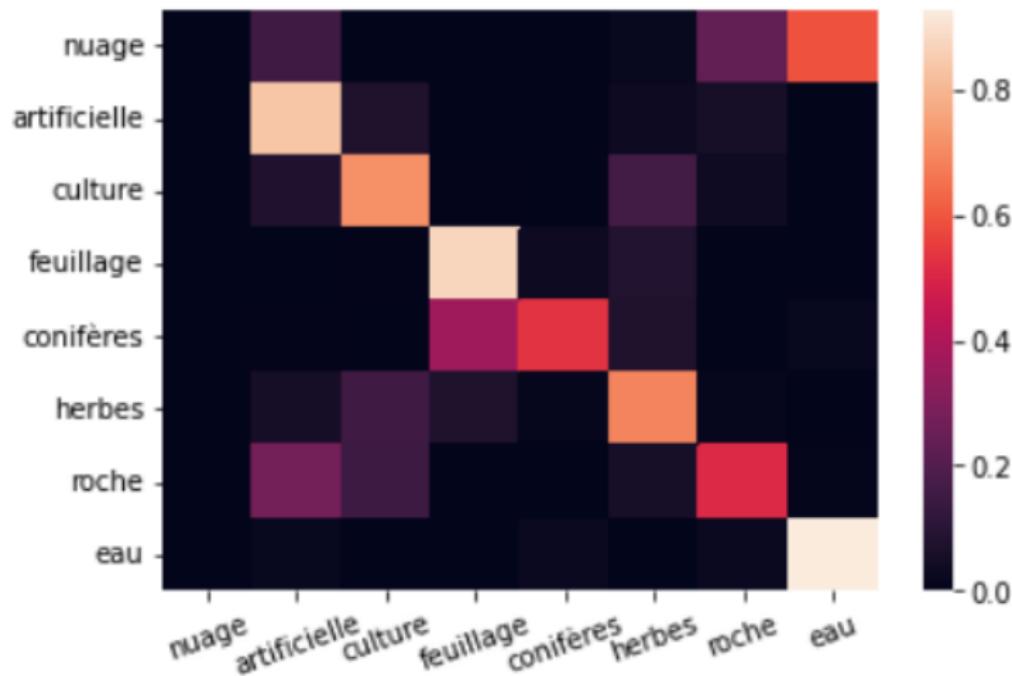


$\text{iou} = 1$

wandb!

MATRICES DE CONFUSION

- On cherche à savoir ici à quelles classes profitent les erreurs de classification



1 CONTEXTE

2 RÉSOLUTION

3 RÉSULTATS

- Analyse quantitative des modèles
- Analyse qualitative des résultats

4 BILAN

UN PREMIER EXEMPLE

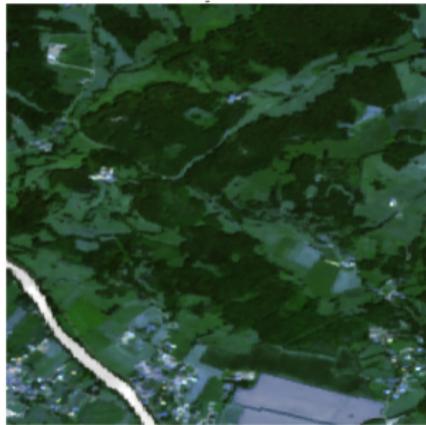
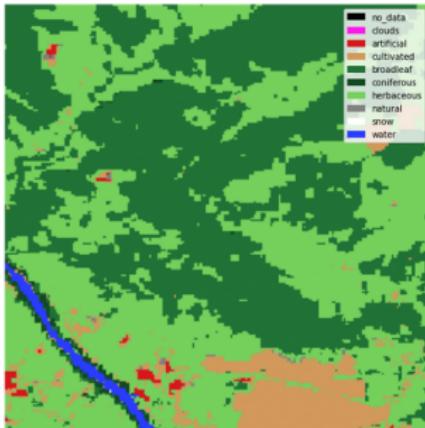
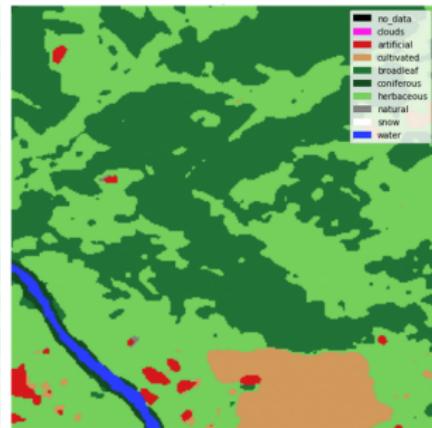


Image en entrée



Masque connu

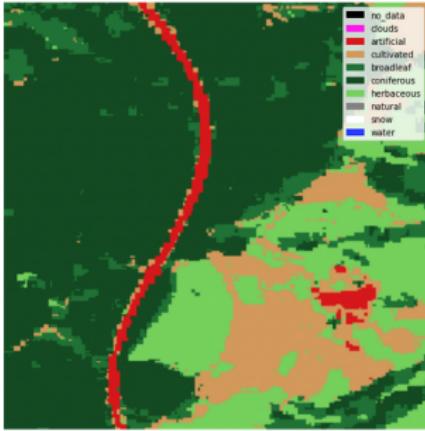


Masque estimé

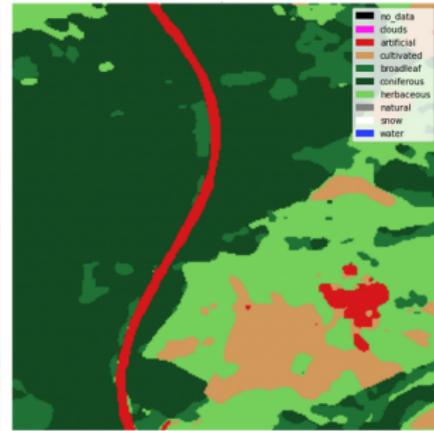
UN DEUXIÈME EXEMPLE



Image en entrée



Masque connu

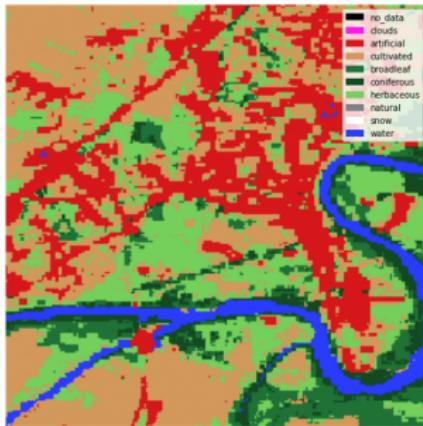


Masque estimé

UN TROISIÈME EXEMPLE



Image en entrée



Masque connu



Masque estimé

1 CONTEXTE

- Le challenge Data
- Les données en entrée

2 RÉSOLUTION

- Stratégie de Résolution
- Formalisation du problème
- Stratégie d'entraînement
- Les modèles de Deep Learning utilisés

3 RÉSULTATS

- Analyse quantitative des modèles
- Analyse qualitative des résultats

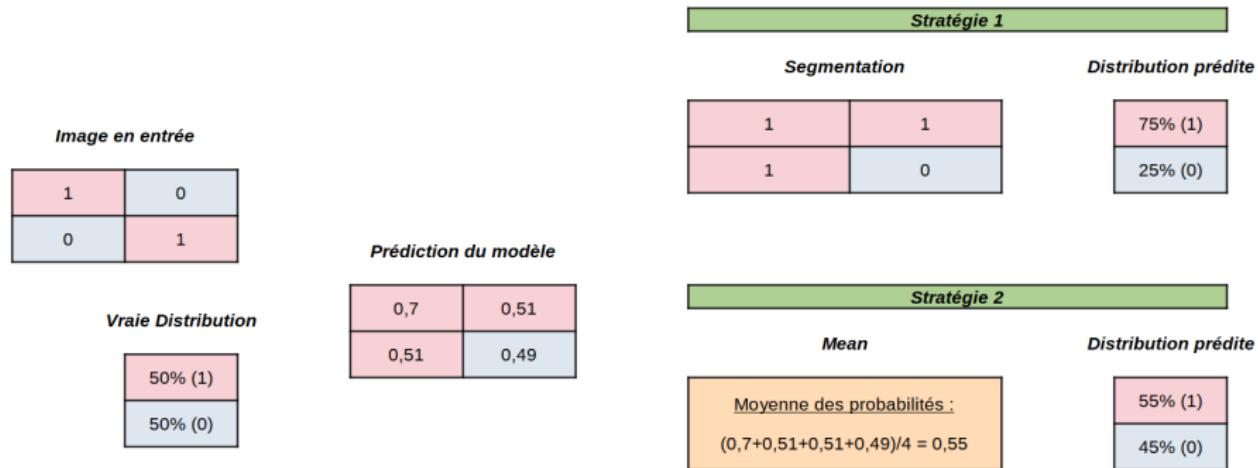
4 BILAN

BILAN

- Au bout du compte on arrive à réaliser un algorithme qui classifie correctement les pixels d'une image donnée !
- Les méthodes de *Deep Learning* sont particulièrement adaptées à ce genre de travaux

COMMENT FALLAIT-IL FAIRE POUR GAGNER ?

- Un couple de candidats a dominé le challenge avec une idée simple



- De l'importance de la bonne compréhension du problème avant de foncer dans la technique

RETOUR SUR LA FLEXIBILITÉ DES OUTILS DISPONIBLES

- Travail rendu possible par l'agilité permise par l'utilisation de la plateforme (et par ses créateurs)
- Totalement adaptée aux travaux sur données non confidentielles
- GPU disponibles !
- Possibilité d'aller plus loin avec du calcul distribué et l'entraînement de réseaux en parallèle (système de vote)
- On s'est bien amusé, mais il ne faut pas oublier que le projet était cadré et que nous n'avons pas eu à nous interroger sur les données en entrées

QUESTIONS BRÛLANTES

- Memorandum de Varsovie à l'issue de la conférence DGINS d'Octobre 2021 "Earth observation for official statistics"
- Une stratégie d'utilisation des données satellites en tant que système d'information semble se dessiner au niveau des INS Européens
- L'intégration de cette stratégie nécessite de développer une conscience sur la charge induite
- Positionnement en ce qui concerne les infrastructures informatiques :
 - ▶ Stockage des données
 - ▶ Création de jeux de données millésimés (à l'instar du COG)
 - ▶ Actualisation du jeu de données (séries temporelles d'images à haute résolution)
 - ▶ Est-ce à l'INSEE de gérer cet aspect, où s'inscrire dans les infrastructures déjà existantes ?
- La labellisation est une opération très coûteuse et les performances des réseaux de neurones sont liées à la qualité de cette dernière
- La détermination des catégories de label l'est aussi, dépend des usages → A réfléchir en amont, nécessite une vision stratégique !

QUESTIONS BRÛLANTES

- Ces questions sont importantes et dépassent les questions techniques développées dans les slides précédentes
- Ces questions ne sont pas à considérer uniquement du point de vue INSEE, il faut prendre en compte d'autres acteurs (SSP, IGN, etc.), leur aide et leur expertise peuvent être précieuses.
- Il faut également prendre le temps de réfléchir à des cas d'usages qui seront utiles → l'idée n'est pas de se substituer à un système de données déjà très complet

CAS D'USAGES

- Cas d'usages évidents en ce qui concerne l'agriculture et l'observation des changements climatiques (exemple de la pollution lumineuse)
- En statistique démographique et sociales, l'idée n'est pas de se substituer à un système de données solide et déjà bien rôdé
- Par contre on pourrait se servir de ces méthodes pour combler des défaillances du système d'information dans certaines zones ! Idées :
 - ▶ Comparer la BDTOPO de l'IGN aux prédictions faites par un réseau entraîné à détecter des bâtiments ?
 - ▶ Utiliser un réseau entraîné sur des zones bien couvertes statistiquement pour combler le déficit de données à Mayotte ?

MERCI

Merci pour votre attention !

BIBLIOGRAPHIE

-  Albawi Mohammed Al Zawi (2017), Understanding of a convolutional neural network
-  Constantin Ding Lee (2018), Accurate Road Detection from Satellite Images Using Modified U-net
-  Garcia Orts Escolano, Oprea Villena-Martinez Garcia-Rodriguez (2017), A Review on Deep Learning Techniques Applied to Semantic Segmentation
-  Jégou Drozdzal Vazquez Romero Bengio (2017), The One Hundred Layers Tiramisu : Fully Convolutional DenseNets for Semantic Segmentation
-  Ronneberger Fischer Brox 2015b, U-Net : Convolutional Networks for Biomedical Image Segmentation
-  Malinowski al., Automated Production of a Land Cover/Use Map of Europe Based on Sentinel-2 Imagery