



Université
de Rennes

FACULTÉ DES
SCIENCES ÉCONOMIQUES

Projet de Deep Learning

Segmentation sémantique
d'images aériennes

M2 MAS 2022/2023 - Manh Hung Nguyen, Yvo Le Doudic

Table des matières

- I. Le traitement d'image en Deep Learning
- II. Présentation des données traitées
- III. Méthodologie et traitement de nos données
- IV. Modélisation et résultats : U-Net et ResNet

I. Le traitement d'image en Deep Learning

Le Deep Learning appliqué à la reconnaissance d'images

Classification



CAT

Semantic Segmentation



GRASS, CAT, TREE, SKY

Object Detection



DOG, DOG, CAT

Instance Segmentation



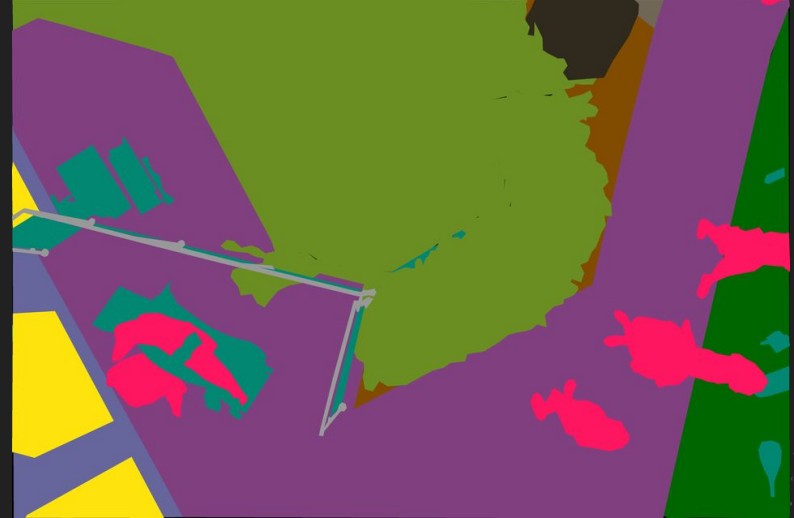
DOG, DOG, CAT

Les grandes familles de modèles de reconnaissance d'image

La segmentation sémantique dans le détail

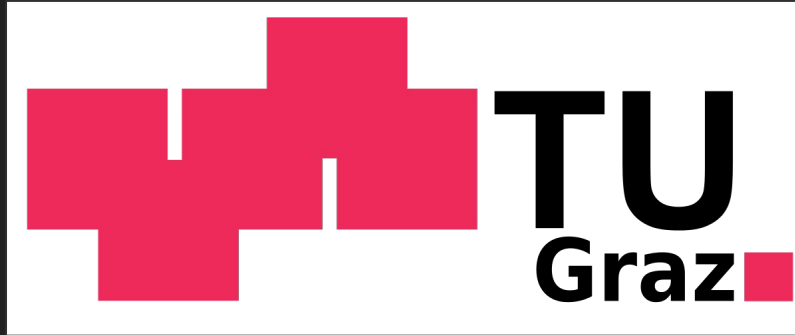


Prédiction



Un exemple de segmentation sémantique : chaque pixel de l'image est associé à une classe (par exemple : personne, végétation etc.)

II. Présentation des données traitées



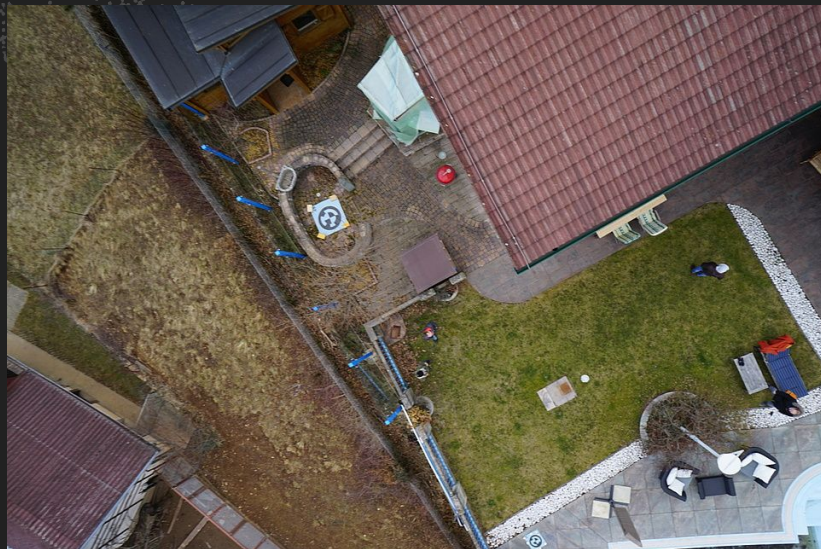
Universität Technische de Graz, Autriche

Le jeu de données vient de l'Université Technique de Graz.

Il contient 400 photos de scènes urbaines pour augmenter la sécurité des vols de drones autonomes et des procédures d'atterrissage.

Les images représentent plus de 20 maisons vues du dessus, prises à une altitude de 5 à 30 mètres au-dessus du sol avec une caméra haute résolution.

Des données propices à la segmentation sémantique



Maison de banlieue périurbaine

Des données propices à la segmentation sémantique



Scène urbaine

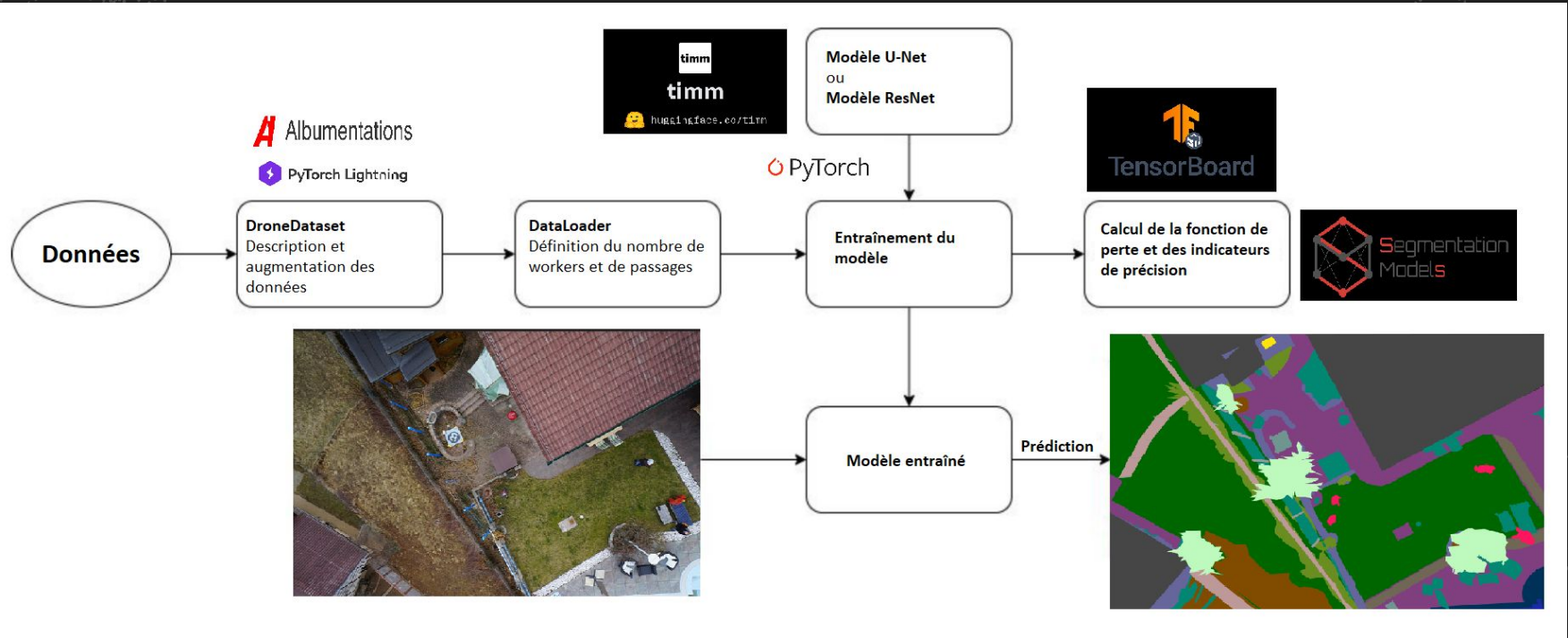


Initialement, le jeu de données utilisé prévoyait 24 classes à prévoir. Nous avons fait le choix de regrouper ces classes en 5 grandes catégories :

- Les obstacles (roches, murs, barrières etc.)
- Les surfaces d'eau (mer, piscines etc.)
- Les surfaces molles (terre, herbe etc.)
- Objets mouvants (personnes, chiens, voitures etc.)
- Zones sur lesquelles le drone peut atterrir (terrain pavé, gravier etc.)

III. Méthodologie et traitement de nos données

Processus de modélisation ou *pipeline*



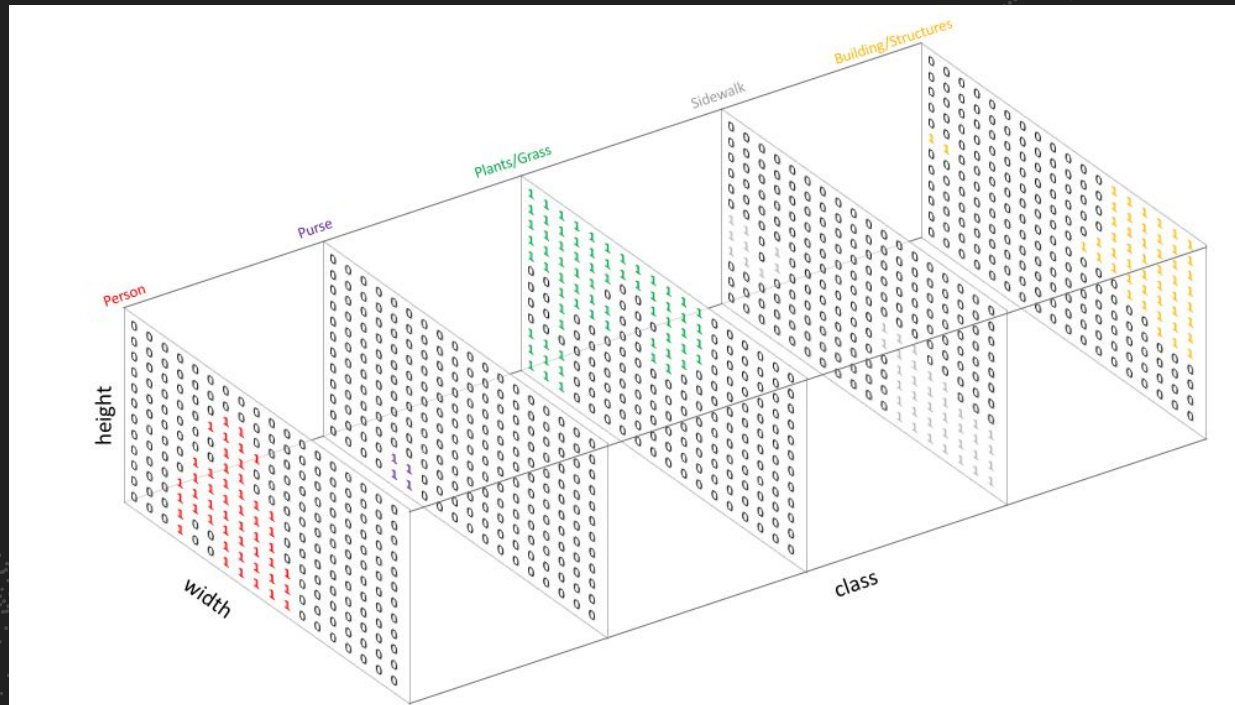
Input et output du modèle

Input & Output (NumPy array):

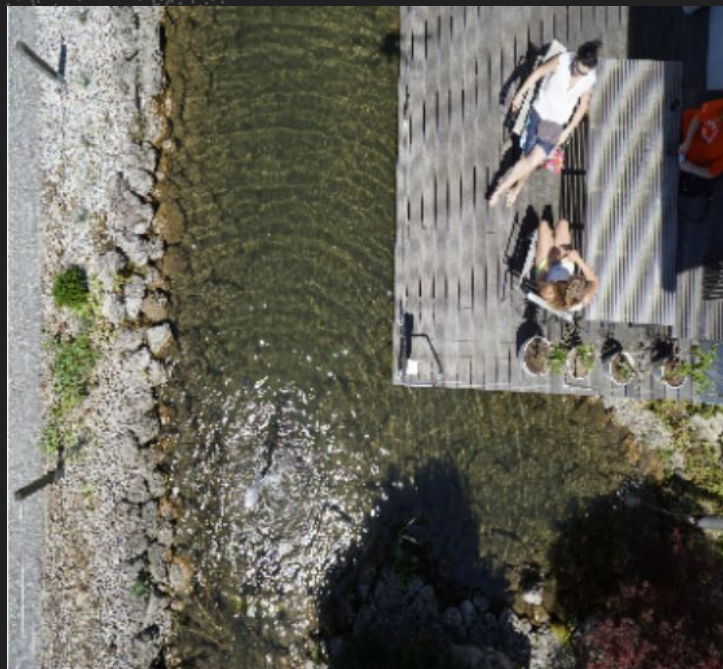
- Input :
Image (384, 384, 3)
- Output :
Masque (384, 384, 5)

Tensor (PyTorch) :

- Input :
Image (B, 3, 384, 384)
- Output :
Masque (B, 5, 384, 384)



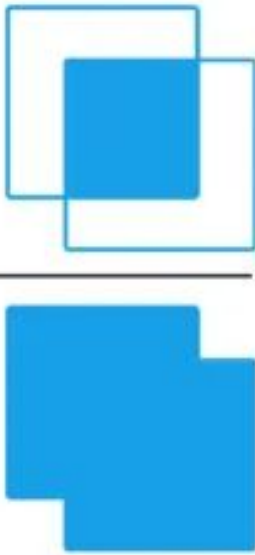
Augmentation de données ou *data augmentation*

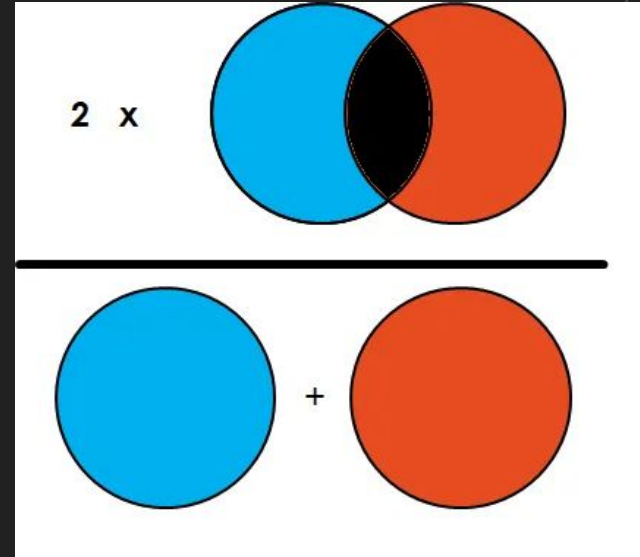


Altération



Critère d'évaluation de la performance de la prédiction

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$




Indicateurs IoU et Dice, plus pertinents que l'accuracy dans un contexte de segmentation sémantique

Illustration du calcul des indicateurs IoU et Dice

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Ground Truth

0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0
0	0	0	0	1	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

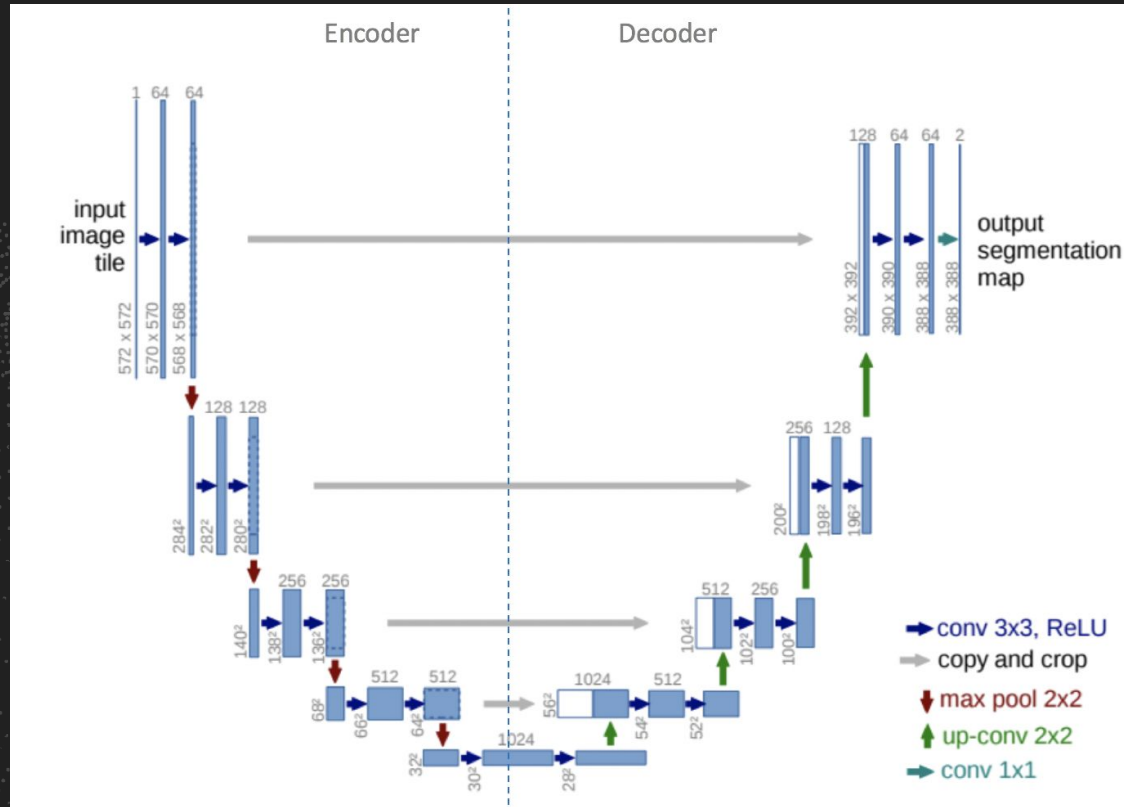
Predict

$$\text{Dice} = (2 \times 11) / (17 + 11) = 78,5\%$$

$$\text{IoU} = 11 / 17 = 64,7\%$$

IV. Modélisation et résultats : U-Net et ResNet

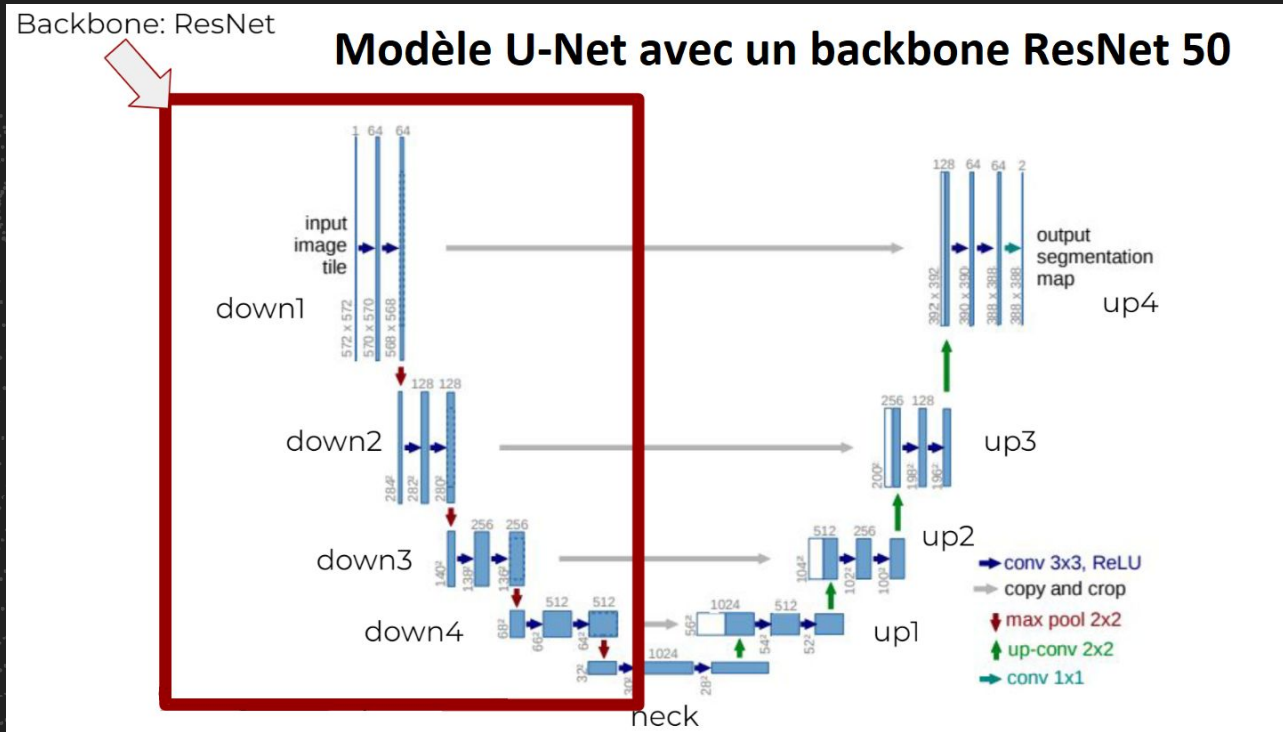
Représentation visuelle du modèle U-Net



Modalité d'entraînement des modèles

- Le jeu de données d'entraînement est composé de 80% des données (i.e. 320 images), le jeu de données de validation est composé de 20% des données (i.e. 80 images)
- Batch-size : 16
- Nombre de passages par époque : $320/16 = 20$
- Nombre d'époques : 50
- Le critère de perte (ou *loss*) est la *CrossEntropyLoss*
- L'optimiseur est l'*Adam Optimizer* avec un *learning rate* de $1e^{-4}$

Représentation visuelle du modèle ResNet



ResNet 50 est entraîné sur des images de taille 224x224 et contient plus de 25 millions de poids

Récapitulatif des performances des modèles

		Loss	Dice	IoU
Train	Unet	0,73	0,38	0,29
	Resnet50	0,22	0,83	0,73
Test	Unet	/	0,42	0,34
	Resnet50	/	0,83	0,73

ResNet50 vs U-Net

Backend Google Compute Engine Python 3 (GPU)

Affichage des ressources de 16:27 à 16:46

RAM du système
3.3 / 12.7 GB



RAM du GPU
14.4 / 15.0 GB



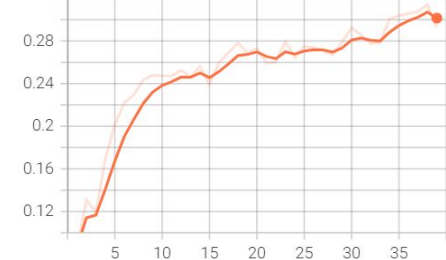
Disque
26.4 / 78.2 GB



Unet_Dice
tag: Train_Unet/Unet_Dice



Unet_IoU
tag: Train_Unet/Unet_IoU



Backend Google Compute Engine Python 3 (GPU)

Affichage des ressources de 16:58 à 17:07

RAM du système
3.5 / 12.7 GB



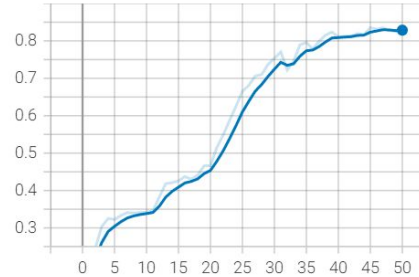
RAM du GPU
8.8 / 15.0 GB



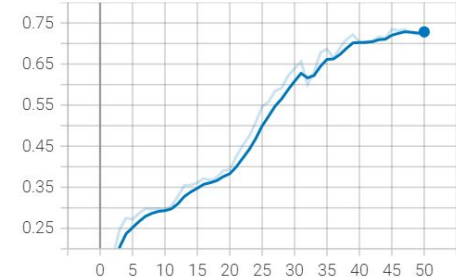
Disque
26.5 / 78.2 GB



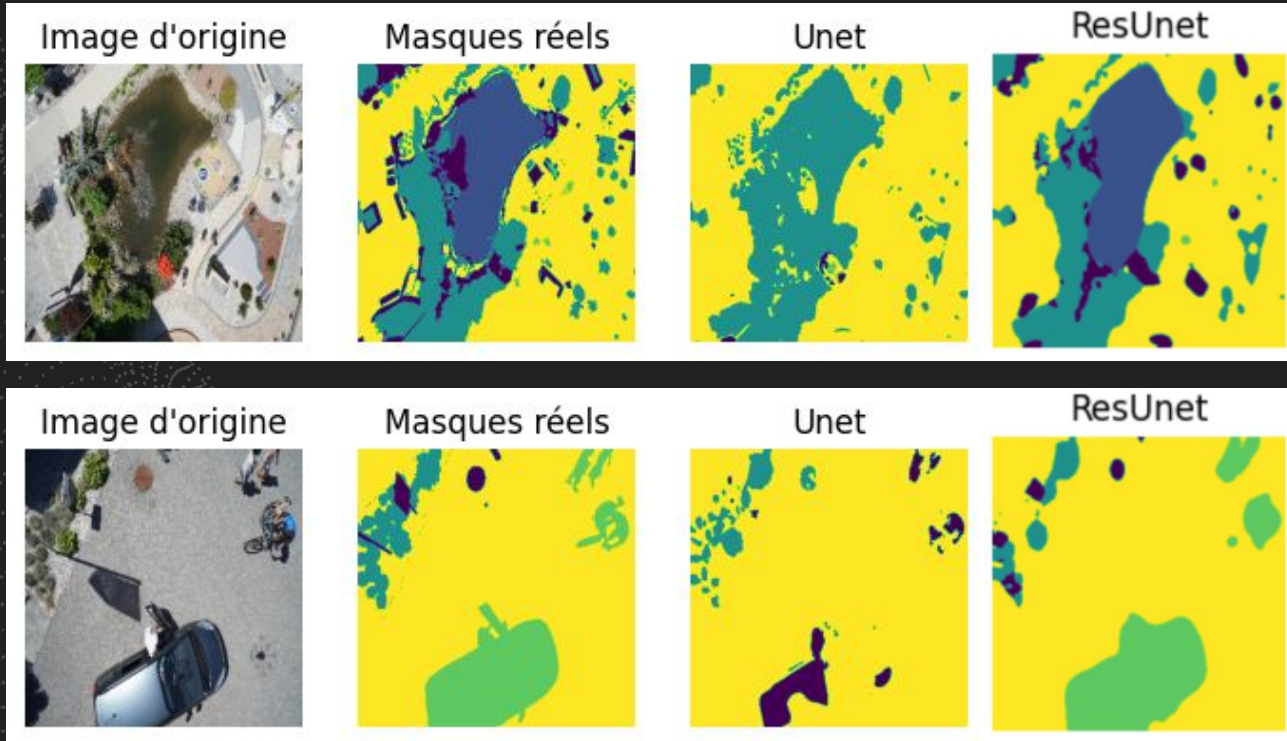
ResUnet_Dice
tag: Train_ResUnet/ResUnet_Dice



ResUnet_IoU
tag: Train_ResUnet/ResUnet_IoU



Comparaison de la qualité de prédiction des différents modèles



Bibliographie

- Cours de Deep Learning, M2 MAS, Stéphane Tufféry
- <https://www.tugraz.at/index.php?id=22387>
- <https://www.jeremyjordan.me/evaluating-image-segmentation-models/>
- <https://www.jeremyjordan.me/semantic-segmentation/>