Participez à la conception d'une voiture autonome

Hourdin Charlène - Décembre 2023

Agenda

Présentation du projet

Pré-traitement des données

Présentation des modèles

Choix du modèle et déploiement de l'API

PRÉSENTATION DU PROJET

Appel à projet

01	Le projet	**concevoir un premier modèle** de segmentation d'images
02	L'objectif	**Réaliser un prototype** permettant de prédire la segmentation d'une image (mask)
03	La mission	**Mettre en production l'API :** Le modèle prend en entrée une image et renvoie le mask prédit (segments prédits de l'image).
04	La méthode	- Livrer un prototype fonctionnel du modèle - Rédiger une note technique présentant le travail effectué et les résultats obtenus

PRÉTRAITEMENT DES DONNÉES

Présentation du dataset

Dataset Cityscape

Stuttgart *Zurich*

Images segmentées et annotées de caméras embarquées dans différentes villes (50)
5000 images et 20 000 masques

8 catégories principales :

Ciel	*Ciel*
Construction	*batiment, mur, cloture, garde-corp, pont, tunnel*
Humain	*personne, cavalier*
Nature	*végétation, terrain*
Objet	*poteau, groupe de poteaux, panneau de signalisation, feu de signalisation*
Sol	*route, trottoir, stationnement, voie ferré*
Véhicule	*voiture, camion, bus, moto, train, caravane, remorque*
Néant	*sol, dynamique, statique*

Le dataset est divisée en 3 parties :

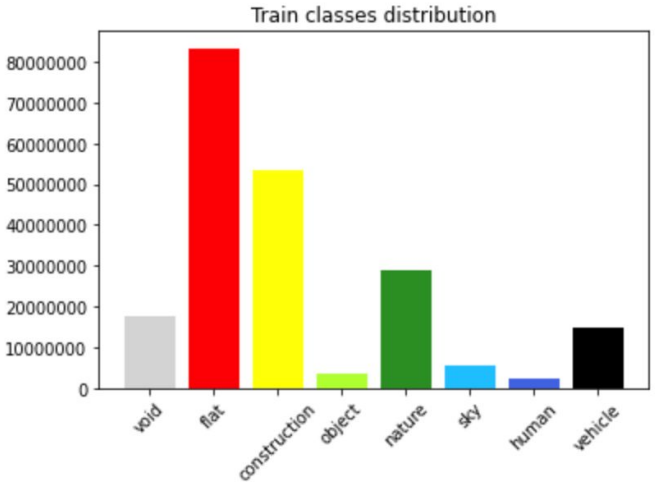
— ***Train (2975 images et masques)*** *(permet d'entraîner les modèles)*

— ***Validation (500 images et masques)*** *(évalue les modèles durant l'entraînement)*

— ***Test (500 images)***
(évalue les performances du modèle retenu)

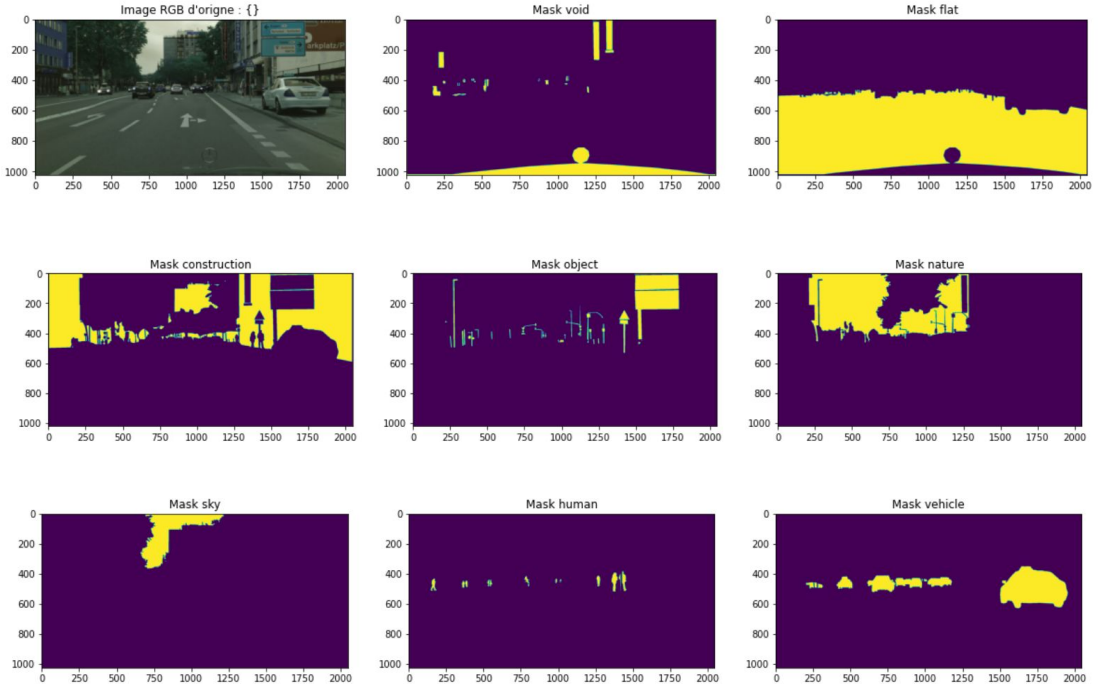
Exploration des données

Distribution du jeu de données Train

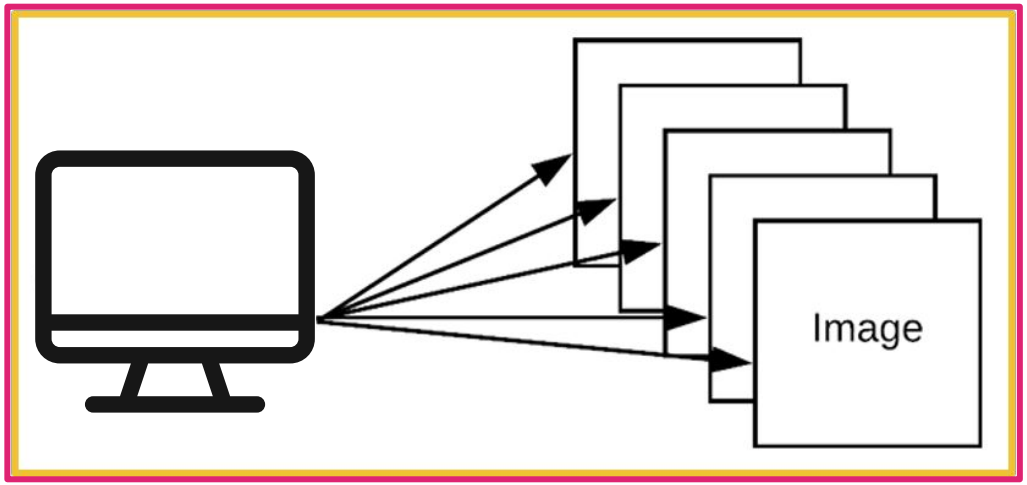


Classe déséquilibré a prendre en compte dans les mesures de performances des modèles

Matrice de segmentation mask



Générateur de données

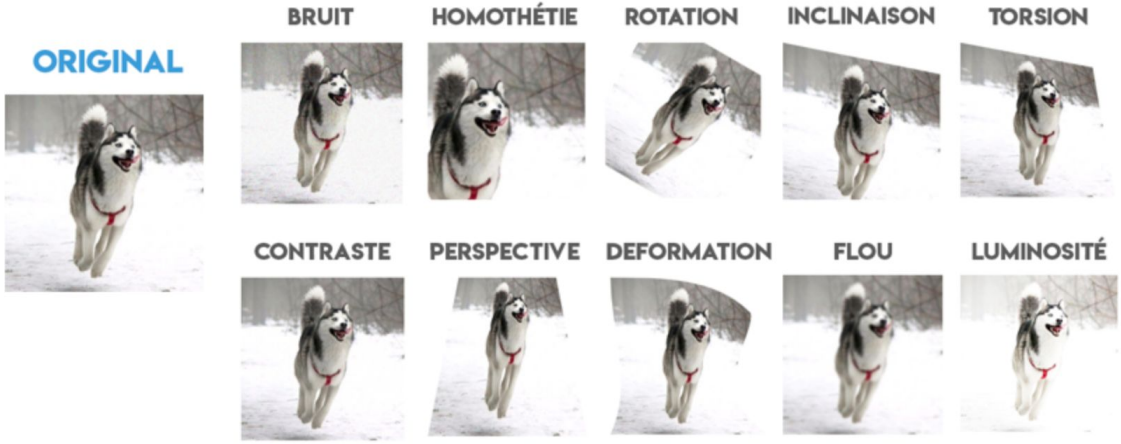


Création d'une classe
ImageSequenceGenerator

Séparation du jeu de données en batch

Augmentation de données intégrée au générateur

Augmentation des données



3 types d'augmentations :

- **Flou** *(Ajout d'un flou gaussien avec un sigma allant de 0 à 3.)*
- **Zoom** *(Zoom de l'image entre 50% et 150%.)*
- **Dropout2d** *(applique l'abandon avec la probabilité donnée à l'image, il définira aléatoirement une zone rectangulaire de l'image sur zéro.)*

PRÉSENTATION DES MODÈLES

Les modèles

Unet Mini *(baseline)*

Un modèle **Unet mini** est une version simplifiée du modèle **Unet original** qui utilise moins de couches et de paramètres pour réduire la complexité et les exigences en matière de calcul.

- Architecture simplifiée par rapport à **Unet**
- Moins de couches et de paramètres
- Utilisé pour des tâches de segmentation d'image avec moins de calculs.

Unet

Unet est un modèle de réseau de neurones convolutionnel utilisé pour les tâches de segmentation d'image, où il combine un encodage de bas niveau de l'image avec un décodage de haut niveau pour prédire la segmentation de l'image.

- Architecture en forme de U
- Utilisation de couches d'encodage et de décodage
- Conçu pour des tâches de segmentation d'image.

Les backbones

ResNet 50

ResNet50 est un modèle de réseau de neurones très profond qui utilise des raccourcis de chemin (shortcuts ou skip connections) pour améliorer la performance.

- Utilise des raccourcis de chemin (shortcuts ou skip connections)
- Performances élevées.

VGG16

VGG16 est un modèle de réseau de neurones convolutionnel qui utilise des couches de convolution profondes avec des filtres de petite taille pour extraire des caractéristiques de l'image

- Filtres de petite taille pour extraire des caractéristiques de l'image
- Performances élevées.

Les métriques

IOU score

IOU est une métrique qui mesure la similitude entre deux images de segmentation en comparant le nombre de pixels correctement classés par rapport au nombre total de pixels dans l'image.

- Mesure de la similitude : mesure la similarité entre la réalité et la prédiction en comparant les pixels corrects.
- Robuste aux différences de tailles : prend en compte les différences de tailles entre les classes

F1-score

Le **F1 score** est une métrique utilisée pour évaluer la performance des modèles de classification binaire en combinant la précision et le rappel en un seul score.

- Mesure de la performance de la classification
- Équilibre entre précision et rappel : prend en compte les erreurs de type faux positifs et faux négatifs

Les fonctions Loss

Categorical Crossentropy

La fonction de perte **categorical crossentropy** mesure la dissimilarité entre les prédictions du modèle et les valeurs cibles pour les modèles de classification multi-classes en utilisant la cross-entropie.

- Utilisée pour les modèles de classification multi-classes
- minimise la divergence entre la probabilité prédite et réelle pour chaque classe.

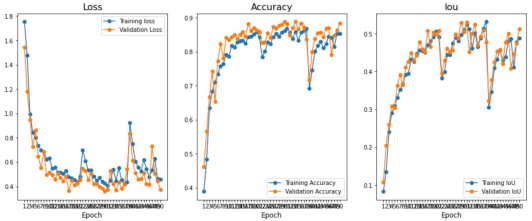
Balanced loss

La fonction de perte **balanced loss** est une méthode pour traiter les déséquilibres de classe en utilisant des poids pour compenser les classes surreprésentées dans les données d'entraînement pour améliorer la performance de la classification pour les classes sous-représentées.

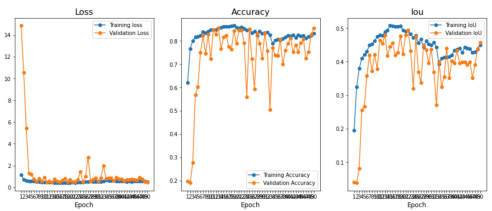
- Traitement des déséquilibres de classe
- Utilisation de poids pour les classes surreprésentées

Modèle avec Crossentropy

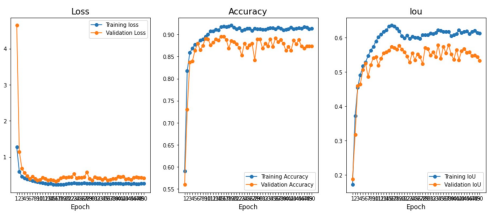
Unet mini



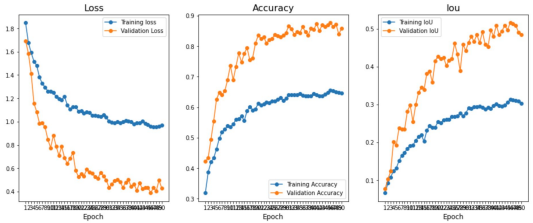
Unet



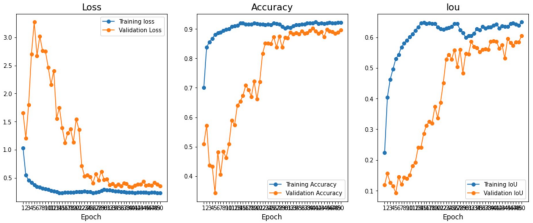
Unet-Resnet50



Unet mini avec augmentation



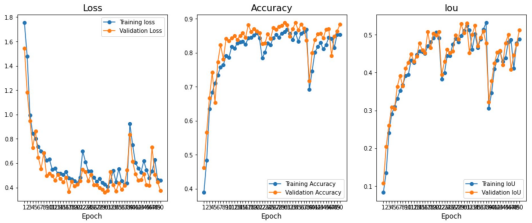
Unet - VGG16



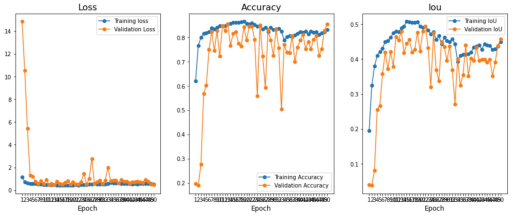
	Model	Loss Function	Loss	Accuracy	F1-score	Iou score	Time
1	Unet_mini	categorical_crossentropy	0.520794	0.841938	0.597809	0.486344	4863.944373
2	Unet_mini_aug	categorical_crossentropy	0.521012	0.829878	0.598681	0.484836	4774.845998
3	Unet	categorical_crossentropy	0.635241	0.796512	0.543981	0.417973	4247.406577
4	Unet_resnet_CC	categorical_crossentropy	0.543803	0.847009	0.679934	0.560435	3708.879752
5	Unet_vgg16_CC	categorical_crossentropy	0.520273	0.838212	0.642198	0.512760	3675.355234

Modèle avec Balanced loss

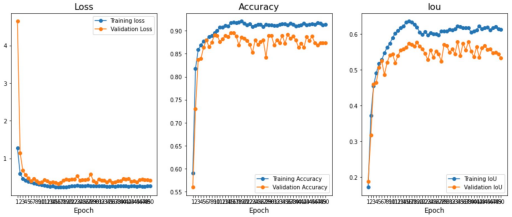
Unet mini



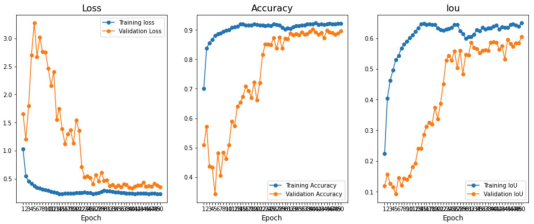
Unet



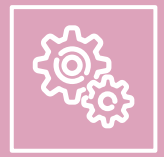
Unet-Resnet50



Unet - VGG16



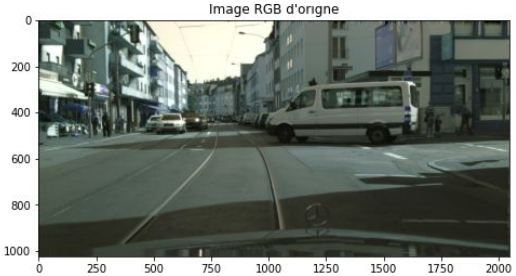
	Model	Loss Function	Loss	Accuracy	F1-score	Iou score	Time
6	Unet_mini_BL	balanced_loss	0.478510	0.385911	0.069537	0.048239	4467.612423
7	Unet_BL	balanced_loss	0.448479	0.822848	0.619009	0.534574	3858.366940
8	Unet_resnet_BL	balanced_loss	0.444933	0.875154	0.723645	0.632965	3832.857036
9	Unet_VGG16_BL	balanced_loss	0.446515	0.851046	0.700407	0.600885	3747.001836

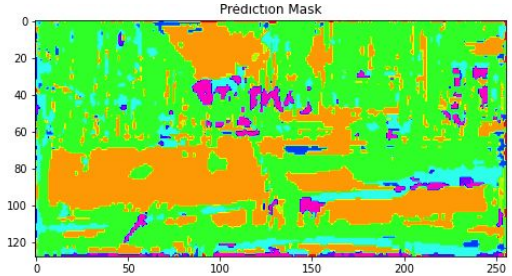


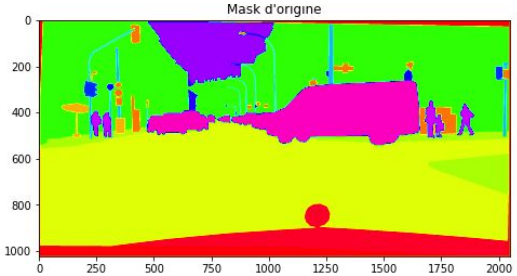
CHOIX DU MODÈLE ET DÉPLOIEMENT DE L'API

Comparaison des modèles

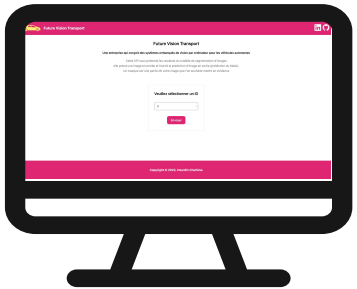
	Model	Loss Function	Loss	Accuracy	F1-score	Iou score	Time
1	Unet_mini	categorical_crossentropy	0.520794	0.841938	0.597809	0.486344	4863.944373
2	Unet_mini_aug	categorical_crossentropy	0.521012	0.829878	0.598681	0.484836	4774.845998
3	Unet	categorical_crossentropy	0.635241	0.796512	0.543981	0.417973	4247.406577
4	Unet_resnet_CC	categorical_crossentropy	0.543803	0.847009	0.679934	0.560435	3708.879752
5	Unet_vgg16_CC	categorical_crossentropy	0.520273	0.838212	0.642198	0.512760	3675.355234
6	Unet_mini_BL	balanced_loss	0.478510	0.385911	0.069537	0.048239	4467.612423
7	Unet_BL	balanced_loss	0.448479	0.822848	0.619009	0.534574	3858.366940
8	Unet_resnet_BL	balanced_loss	0.444933	0.875154	0.723645	0.632965	3832.857036
9	Unet_VGG16_BL	balanced_loss	0.446515	0.851046	0.700407	0.600885	3747.001836



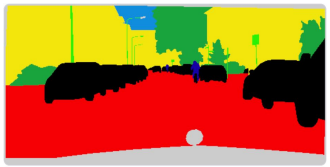


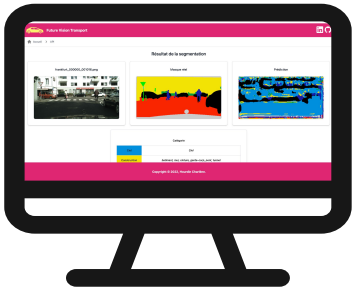


Architecture API

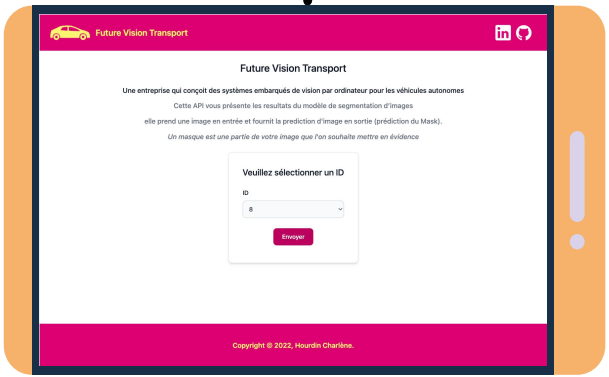


Prédiction





Déploiement du modèle



Flask

Flask est un framework open-source, qui permet de créer des applications web en python.

GIT

Git est un système de contrôle de version open source, permet ainsi de garder une trace de chaque version de votre projet et de mettre à jour une application de manière automatisée

heroku

Heroku est une plate-forme d'applications cloud, qui permet le déploiement d'application

Demonstration

Conclusion

Résultat

- ✓ Nous obtenons un modèle de qui donne de bons résultats tout en respectant les contraintes imposées.
- ✓ L'augmentation de données a été testée sur le modèle de base **Unet mini** et n'a pas permis d'améliorer les performances du modèle.
- ✓ L'api est fonctionnelle

Axe d'amélioration

- ➤ Optimiser d'autres **hyper-paramètres** comme l'optimiser *(SGD, RMSprop,...)*
- ➤ Tester d'autres versions d'augmentation des images
- ➤ Tester d'autres fonctions de perte
- ➤ Tester une approche de type **Transformer**. On pourrait pour cela utiliser la classe **AutoModelForSemanticSegmentation** de la librairie Hugging Face
- ➤ Tester d'autres modèles comme **DeepLabV3** (pré-entraîné sur le jeu de données Cityscapes)

MERCI

Avez-vous des questions?