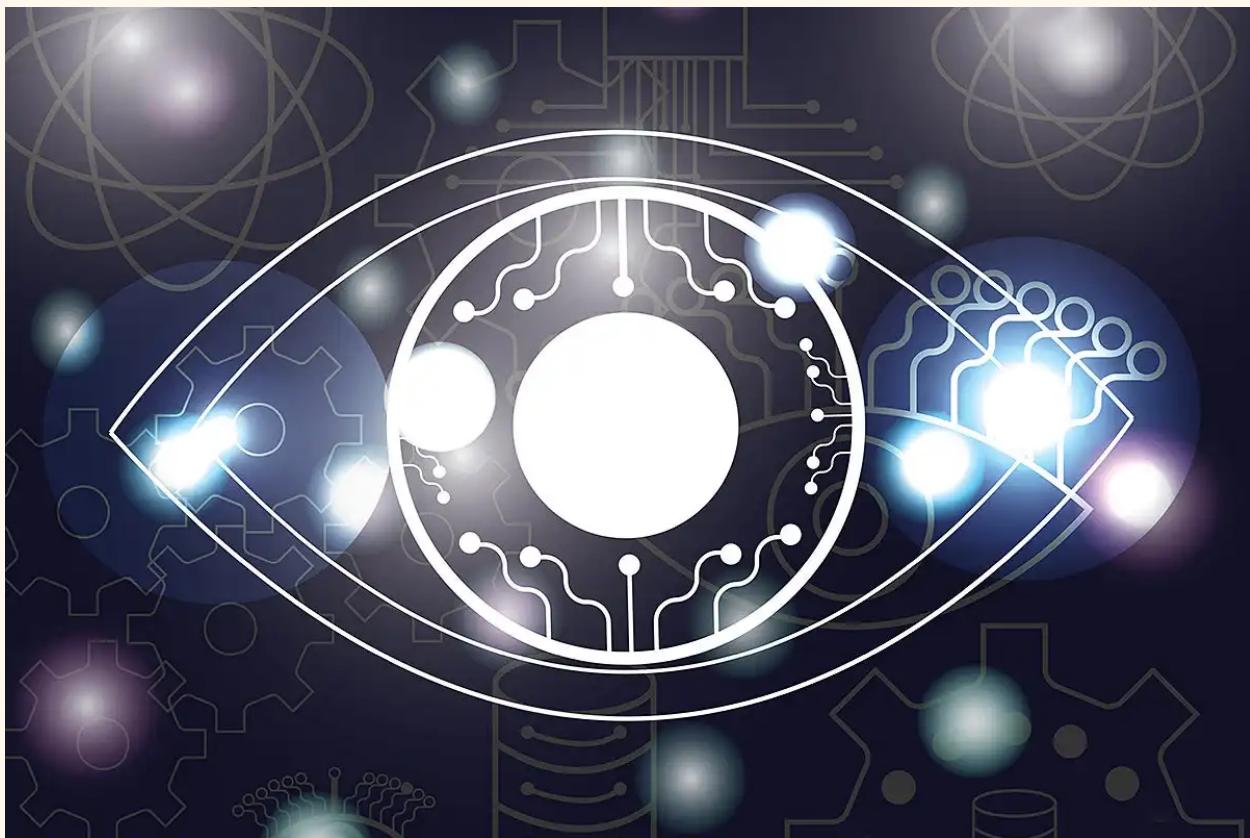


CONCEPTION D'UNE VOITURE AUTONOME GRÂCE AU DEEP LEARNING

Par Sofiane DERRAZ



| | |
|---|-----------|
| INTRODUCTION | 3 |
| Point sur les données | 3 |
| AUGMENTATION DES DONNÉES | 4 |
| Qu'est-ce que l'augmentation des données ? | 4 |
| Les techniques d'augmentation des données | 4 |
| MÉTRIQUES | 5 |
| IoU Score | 5 |
| Loss | 5 |
| Temps | 6 |
| MODÈLE DE RÉFÉRENCE UNET | 6 |
| Architecture | 6 |
| L'encodeur | 7 |
| Le décodeur | 7 |
| Résultats | 7 |
| MODÈLE UNET VGG16 | 7 |
| Architecture | 7 |
| Résultats | 8 |
| MODÈLE FPN | 9 |
| Architecture | 9 |
| Résultats | 9 |
| SYNTHÈSE | 10 |
| Comparaison du val_loss | 10 |
| Comparaison du val_mean_IoU | 10 |
| Comparaison du temps d'entraînement (en secondes) | 11 |
| CONCLUSION | 12 |

INTRODUCTION

Future Vision Transport est une entreprise qui conçoit des systèmes embarqués de vision par ordinateur pour les véhicules autonomes. Cette note technique se concentre sur la conception d'un premier modèle de segmentation d'images qui devra s'intégrer dans la chaîne complète du système embarqué.

Différentes approches seront présentées avec une synthèse de l'état de l'art. Il sera également vu plus en détail : le modèle et l'architecture qui ont été retenus. Pour terminer, une synthèse des résultats obtenus et une conclusion avec des pistes d'amélioration envisageables.

Point sur les données

Les données utilisées dans ce projet sont les données de cityscapes, accessibles via ce [lien](#). Le jeu de données Cityscapes se concentre sur la compréhension sémantique des scènes de rue urbaines. Des images de différentes villes ont été récoltées et annotées. Exemple ci-dessous (à gauche, l'image d'entrée. À droite, l'annotation). C'est l'image de droite que l'on veut en sortie.



Il y a 35 classes dans les masques d'annotation. Dans ce projet, les 35 classes ont été réduites à leurs sous-classes. Ce qui nous donne 8 classes à prédire. Le masque en nuance de gris ci-dessous est l'image contenant les ids des 35 classes et qui est utilisé par le modèle.



AUGMENTATION DES DONNÉES

Qu'est-ce que l'augmentation des données ?

L'augmentation des données est une technique qui permet d'augmenter artificiellement la taille d'un ensemble d'entraînement en créant des données modifiées à partir de l'ensemble existant.

L'augmentation des données est une bonne pratique si on veut éviter le surentraînement, si l'ensemble de données initial est trop petit pour l'entraînement, ou même si on veut obtenir de meilleures performances sur le modèle.

Le jeu de données de cityscapes contient 2975 images d'entraînement et 500 images de validations. Grâce à l'augmentation des données, on peut avoir au-delà de 2975 images.

Les techniques d'augmentation des données

- Transformation géométrique : retourne l'image, recadre, fait pivoter ou translater.
- Transformation de l'espace couleur : modifie les canaux de couleur RGB, intensifie une couleur.
- Filtres à noyaux : rend l'image plus nette ou plus floue.
- Effacement aléatoire : supprime une partie de l'image initiale
- Mélange d'images : mélange essentiellement les images entre elles.

Voici un exemple ci-dessous d'une image augmentée (une distorsion a été appliquée) :

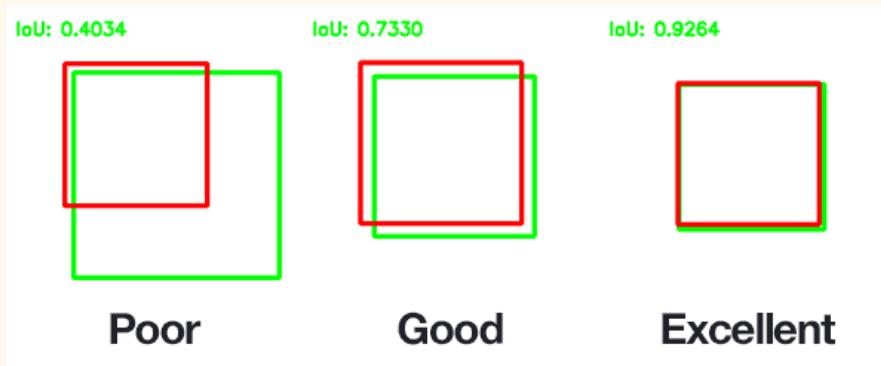


MÉTRIQUES

IoU Score

Le IoU score (Intersection over Union) est un nombre compris entre 0 et 1 qui quantifie la proportion de chevauchement entre le masque cible et le masque prédit.

Un IoU score de 0 signifie qu'il n'y a pas de chevauchement, un IoU score de 1 signifie que l'union des masques est complète. Voici une image qui résume clairement de quoi il s'agit :



Loss

La fonction de perte qui a été choisie est le multi-class cross-entropy loss (entropie croisée). C'est la fonction de perte par défaut à utiliser pour les problèmes de segmentation multi-classes.

L'entropie croisée calcule un score qui résume la différence moyenne entre les distributions de probabilité réelles et prédites pour toutes les classes. Le score est minimisé et une valeur parfaite d'entropie croisée est 0. L'entropie croisée peut être spécifiée comme fonction de perte dans Keras en spécifiant "categorical_crossentropy" lors de la compilation du modèle :

```
model.compile(loss='categorical_crossentropy', metrics=['accuracy', 'IoU'])
```

Temps

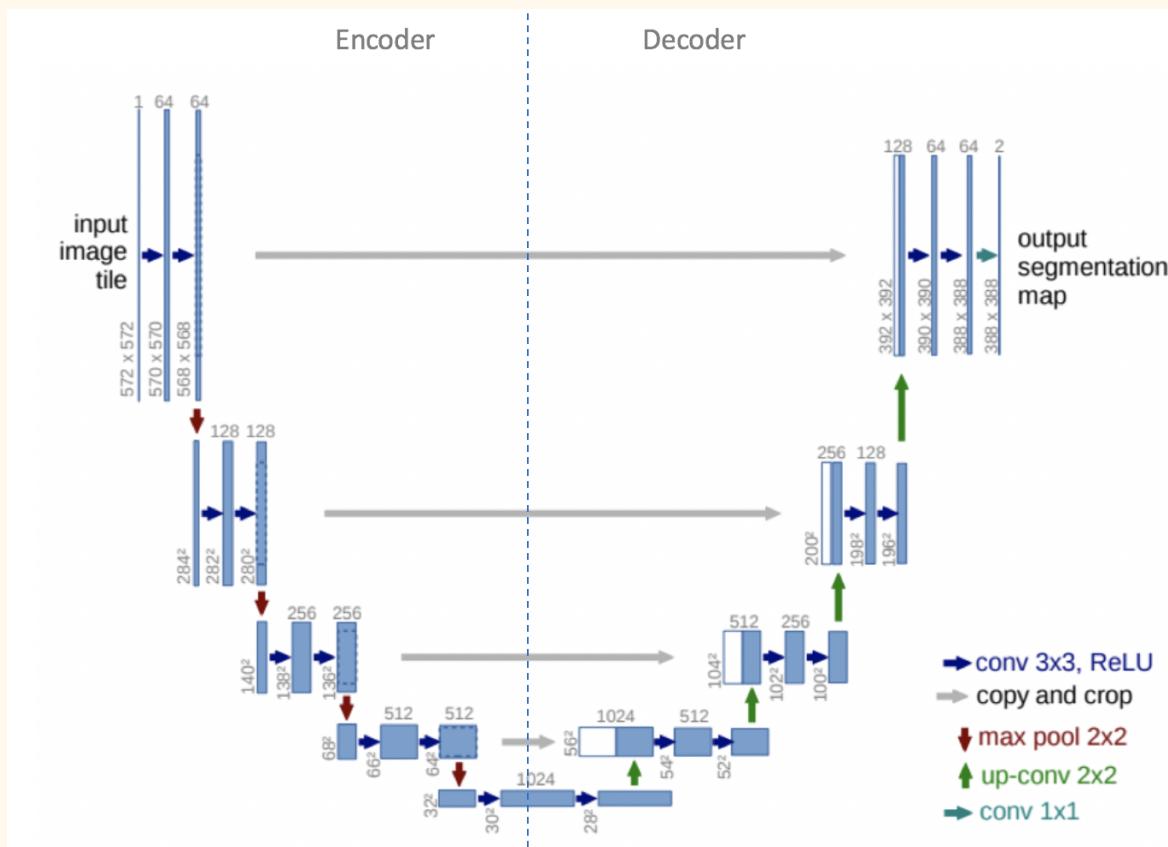
Le temps d'entraînement d'un modèle est également une métrique à prendre en compte. Surtout lorsqu'il s'agit de modèles de deep learning complexes et de gros volumes de données, les temps d'entraînement peuvent rapidement devenir très élevés. Le temps permet d'avoir un œil sur le coût en ressource de l'entraînement.

MODÈLE DE RÉFÉRENCE UNET

Le modèle UNET servira de modèle de base pour comparer les résultats d'autres modèles.

Architecture

Le modèle UNET a une architecture qui contient deux réseaux. Le premier réseau est un réseau d'encodeurs, suivi d'un réseau de décodeurs. Dans le document original, l'architecture est décrit comme suit :



L'encodeur

Il s'agit généralement d'un réseau de classification pré-entraîné comme VGG/ResNet, dans lequel on applique des blocs de convolution suivis d'un sous-échantillonnage "maxpool" pour encoder l'image d'entrée en caractéristiques.

Le décodeur

Le but est de projeter sémantiquement les caractéristiques discriminantes (basse résolution) apprises par l'encodeur sur l'espace des pixels (haute résolution) pour obtenir une classification dense. Le décodeur consiste en un suréchantillonnage et une concaténation suivis d'opérations de convolution régulières.

Résultats

| model_name | training_time | val_loss | val_mean_IoU |
|-------------------|---------------|----------|--------------|
| baseline_unet_aug | 6574.661471 | 1.749396 | 0.357545 |
| baseline_unet | 3864.345063 | 5.297047 | 0.210689 |

Le tableau ci-dessus montre un meilleur résultat pour la baseline avec augmentation des données. On a un score IoU de **0.211** pour la baseline contre **0.358** pour la baseline augmentée.

MODÈLE UNET VGG16

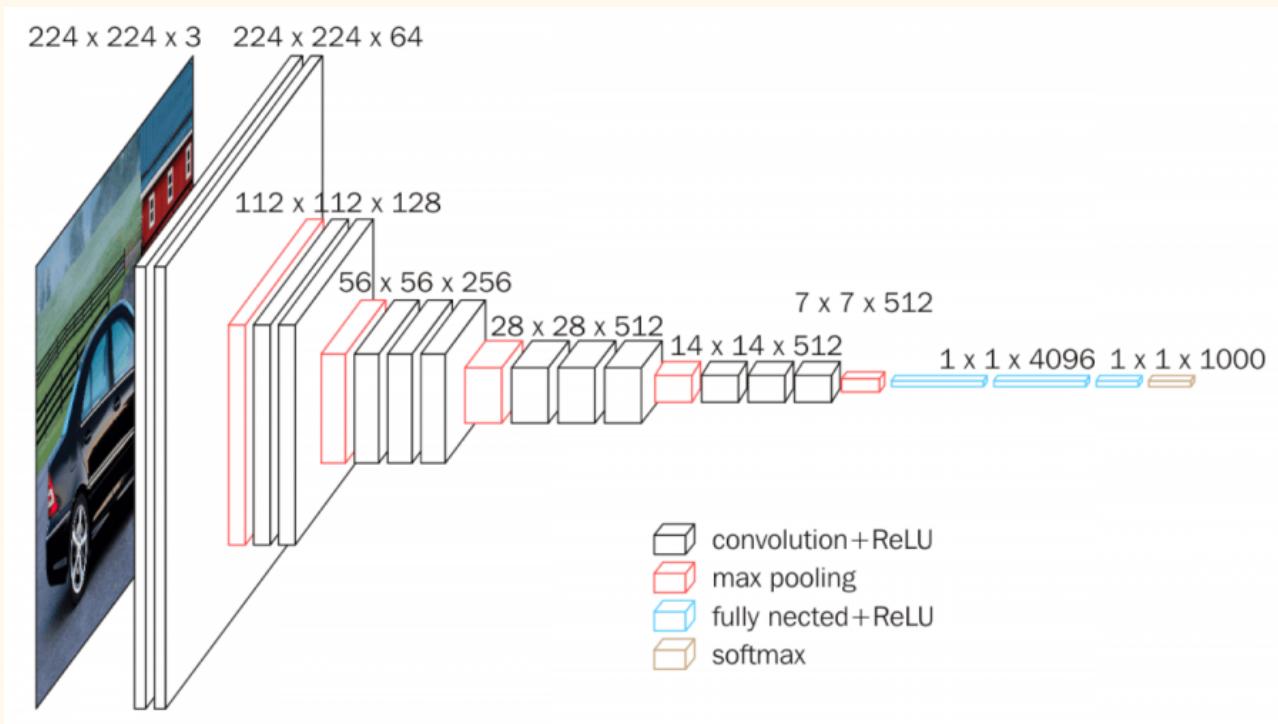
Architecture

L'architecture vgg16 est un des différents "backbones" du modèle UNET. C'est une architecture alternative du modèle UNET. Lorsque cette architecture a été conçue, l'objectif était de montrer que la profondeur du réseau est un élément critique pour de bonnes performances ([source](#)).

Ce réseau de neurones comprend 16 couches profondes. Pour toutes les couches de convolution, le noyau de convolution est de taille 3x3. La plus petite dimension capture les notions de haut, bas, gauche, droite et centre. Les couches de convolution s'accompagnent de couche "max

pooling”, chacune de taille 2x2, pour réduire la taille des filtres au cours de l’apprentissage. En sortie, nous avons 3 couches de neurones “fully connected”. Les deux premières couches sont composées de 4096 neurones et la dernière de 1000 neurones avec une fonction d’activation softmax pour déterminer la classe de l’image.

L’image ci-dessous représente l’architecture du modèle UNET VGG16.



Résultats

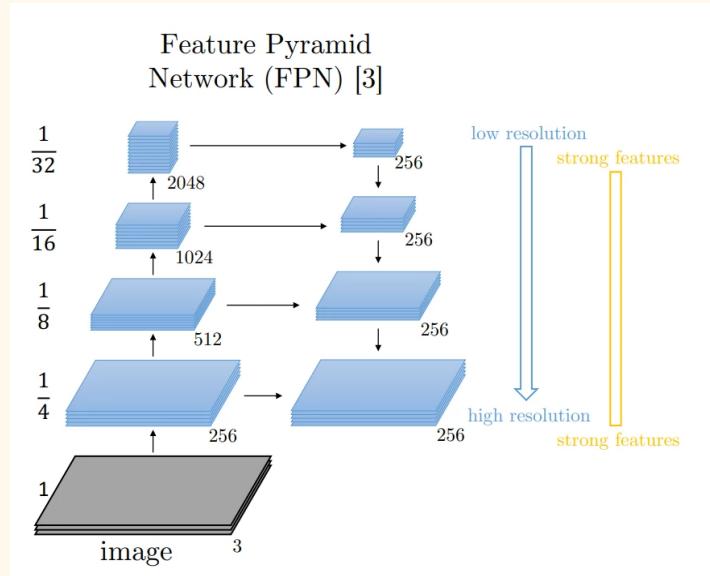
| | ◆ training_time ◆ | ◆ val_loss ◆ | ◆ val_mean_IoU ◆ |
|----------------|-------------------|--------------|------------------|
| model_name | ◆ | ◆ | ◆ |
| unet_vgg16_aug | 13613.636321 | 0.558326 | 0.583861 |
| unet_vgg16 | 12200.882865 | 0.942366 | 0.563299 |

Le score IoU est meilleur pour la version augmentée. Le score loss également. Le résultat reste toutefois bien supérieur à la baseline, avec augmentation ou sans, avec un IoU score de **0.563** contre **0.211** pour la baseline.

MODÈLE FPN

Le modèle FPN (Feature Pyramid Network) est un modèle conçu par Facebook.

Architecture



La voie montante est la partie de l'encodage où l'image est convertie en une carte de caractéristiques à basse résolution. Pour la partie décodage, la carte de caractéristique combine ces cartes de caractéristiques à basse résolution qui ont des caractéristiques sémantiquement fortes, avec l'image précédemment suréchantillonnée qui a des caractéristiques sémantiquement faibles.

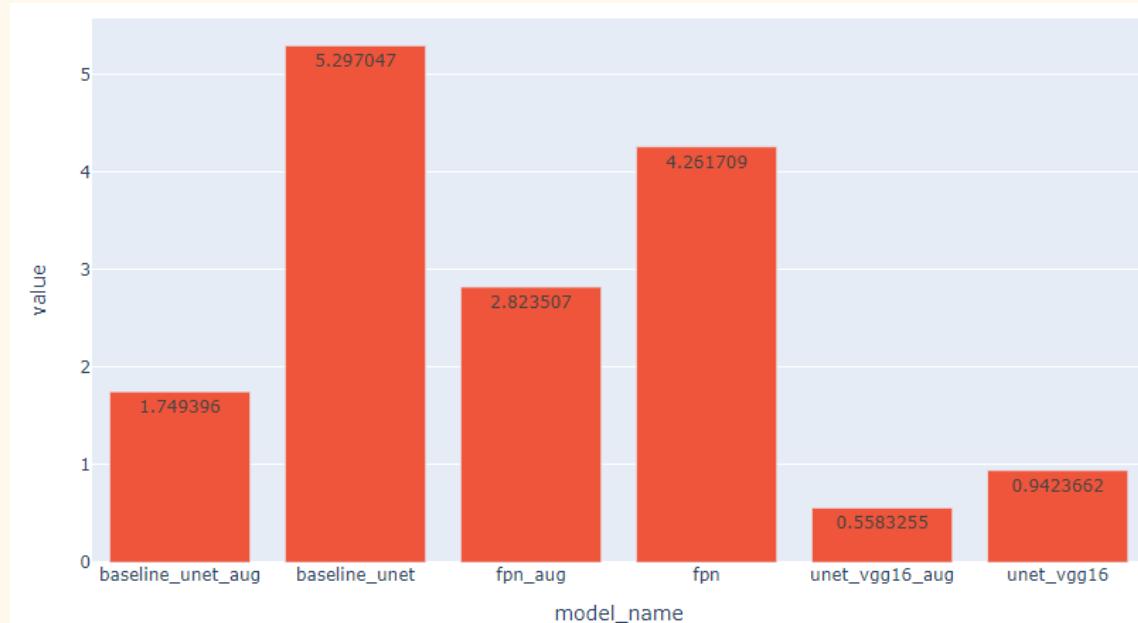
Résultats

| model_name | training_time | val_loss | val_mean_IoU |
|------------|---------------|----------|--------------|
| fpn_aug | 42049.913810 | 2.823507 | 0.190748 |
| fpn | 36727.288630 | 4.261709 | 0.099557 |

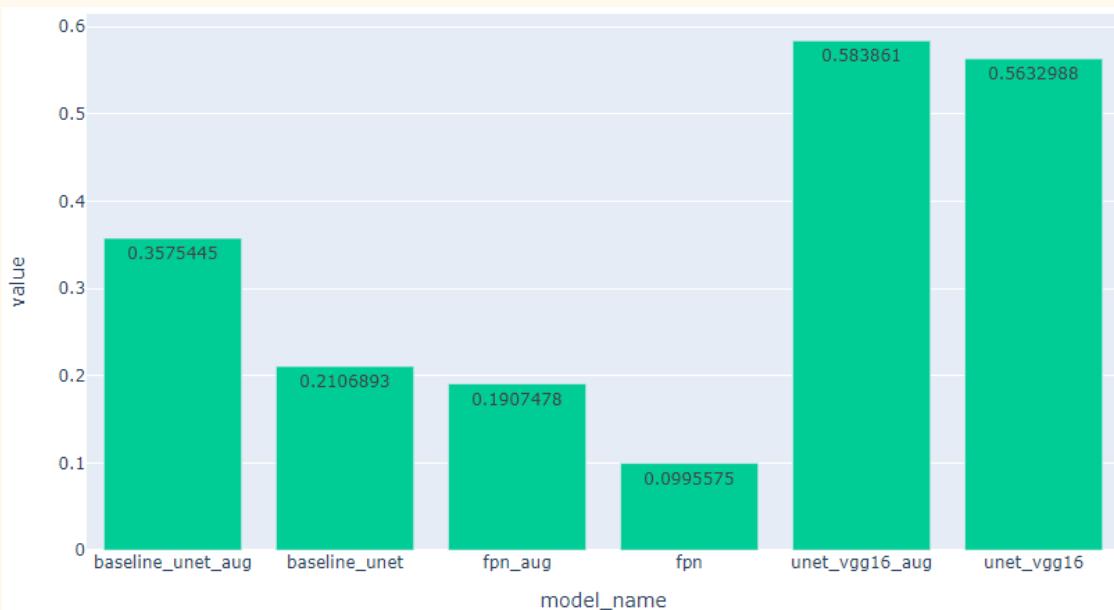
Comparé à la baseline et au modèle UNET VGG16, ce modèle ne présente pas de meilleures performances. De plus, le temps d'entraînement est très élevé (~10h). On peut cependant noter une nette amélioration grâce à l'augmentation des données.

SYNTHÈSE

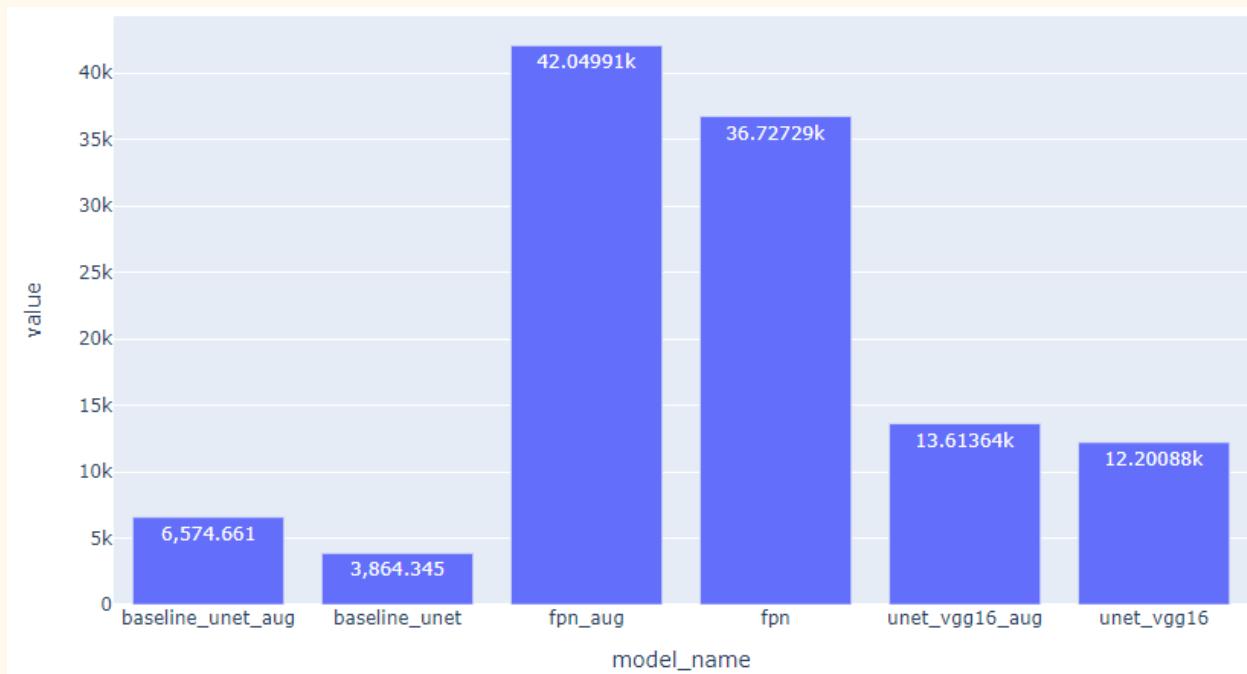
Comparaison du val_loss



Comparaison du val_mean_IoU



Comparaison du temps d'entraînement (en secondes)



Pour rappel, plus le score de perte tend vers 0, plus le score est bon. Le modèle UNET VGG16 est celui qui possède un meilleur score.

Pour le score IoU également, le modèle UNET VGG16 est celui qui possède le meilleur score avec **0.563** contre **0.211** pour la baseline et **0.100** pour le modèle FPN.

Le modèle FPN est celui qui est le plus gourmand en ressource avec un temps d'entraînement de **~10h**. Le modèle UNET VGG16 a pris **~3h30** seulement pour l'entraînement.

Grâce à l'augmentation des données pour le modèle UNET VGG16, on est passé d'un score IoU de **0.563** à **0.584**.

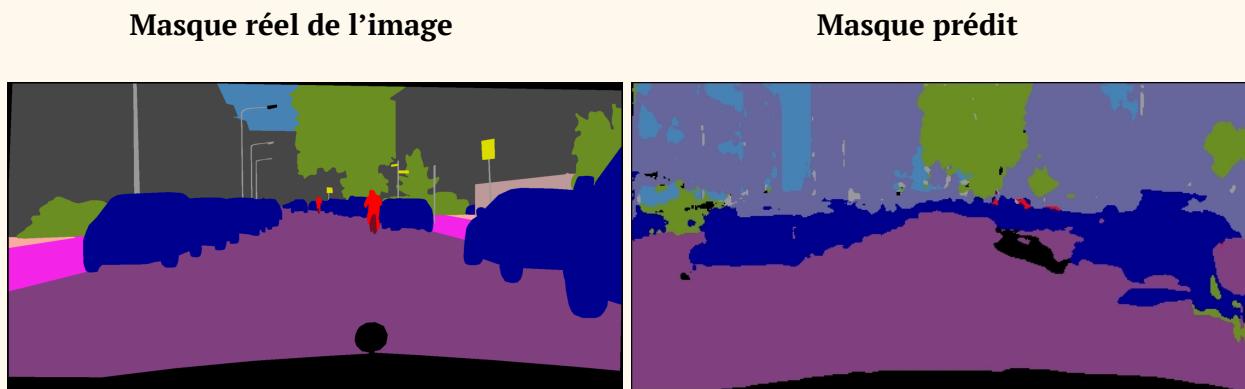
CONCLUSION

| model_name | training_time | val_loss | val_mean_IoU | val_accuracy |
|-------------------|---------------|----------|--------------|--------------|
| unet_vgg16_aug | 13613.636321 | 0.558326 | 0.583861 | 0.859886 |
| unet_vgg16 | 12200.882865 | 0.942366 | 0.563299 | 0.840755 |
| baseline_unet_aug | 6574.661471 | 1.749396 | 0.357545 | 0.641116 |
| baseline_unet | 3864.345063 | 5.297047 | 0.210689 | 0.462864 |
| fpn_aug | 42049.913810 | 2.823507 | 0.190748 | 0.535531 |
| fpn | 36727.288630 | 4.261709 | 0.099557 | 0.408541 |

Même si le score “accuracy” n'a pas été pris en compte dans les métriques, il permet tout de même d'avoir une idée générale des performances des modèles. On observe une amélioration de la majorité des modèles grâce à l'utilisation de l'augmentation de données.

Le modèle UNET VGG16 est meilleur que les autres modèles sur tous les scores avec un temps d'entraînement raisonnable. C'est donc le modèle UNET VGG16 qui a été retenue pour la suite du projet. Néanmoins, des améliorations sont encore possibles afin d'augmenter les performances du modèle. Par exemple, en effectuant une meilleure hyper-optimisation ou en modifiant l'augmentation des données, de sorte à obtenir une amélioration significative.

Voici un exemple de prédiction du modèle ci-dessous :



Pour aller plus loin, ce [lien](#) répertorie tous les modèles qui ont été testés sur le jeu de données de cityscape et effectue une comparaison.