PROJET 8

NOTE TECHNIQUE

Traiter les images pour le système embarqué d'une voiture autonome

APPLICATION DE SEGMENTATION D'IMAGES

https://future-vision-transport.streamlit.app

SOMMAIRE

1. INTRODUCTION

Contexte du projet

Dans le domaine des véhicules autonomes, la segmentation d'images joue un rôle crucial en permettant aux systèmes embarqués de comprendre et d'interpréter l'environnement en temps réel. Cette compréhension repose sur l'identification précise des différents éléments de l'environnement, comme les routes, les véhicules, les piétons, et la végétation. La segmentation sémantique des images est ainsi un maillon essentiel pour garantir la sécurité, la prise de décision, et la navigation autonome.

Le projet s'inscrit dans cette dynamique en visant à développer et à optimiser un modèle de segmentation d'images performant, tout en garantissant une intégration fluide dans un pipeline de production pour les véhicules autonomes.

Objectifs spécifiques

Ce travail vise à relever les défis de la segmentation d'images en :

- Identifiant la meilleure fonction de perte pour optimiser les performances.
- Comparant des stratégies de data augmentation pour améliorer la généralisation des modèles.
- Évaluant plusieurs architectures de segmentation (U-Net, SegNet, DeepLabV3+, PSPNet, FPN) avec des backbones modernes (EfficientNetB0, ResNet34, ResNet50, VGG16).
- Développant une API robuste pour déployer le modèle retenu.
- Assurant la reproductibilité grâce à une documentation et un versioning rigoureux.

Problématique

- Importance des métriques IoU, Dice et de la fonction de perte.

2. METHODOLOGIE

Choix des métriques d'évaluation

La qualité des modèles a été mesurée principalement avec l'IoU (Intersection over Union), qui évalue la précision des prédictions en comparant l'intersection et l'union des zones prédites et réelles, et le Dice Score, qui met l'accent sur l'équilibre entre précision et sensibilité, utile pour les classes sous-représentées.
En parallèle, différentes fonctions de perte ont été testées, notamment la Dice Loss, IoU Loss, Tversky Loss, Focal Tversky Loss, et Total Loss (combinaison de Binary Crossentropy et Dice Loss), pour guider l'entraînement des modèles.

Pipeline général

Un pipeline structuré a été établi pour garantir la reproductibilité. Le prétraitement a adapté les images et masques selon les exigences des modèles : résolution de 256x256 pixels pour la plupart, 288x288 pour SegNet, et 512x512 pour FPN afin de capturer davantage de détails.

Les modèles, allant du U-Net Mini (baseline) à des architectures avancées comme FPN et PSPNet, ont été entraînés avec différentes fonctions de perte et stratégies de data augmentation. Leurs performances ont été évaluées à l'aide des métriques IoU, Dice, et Loss pour identifier le modèle optimal.

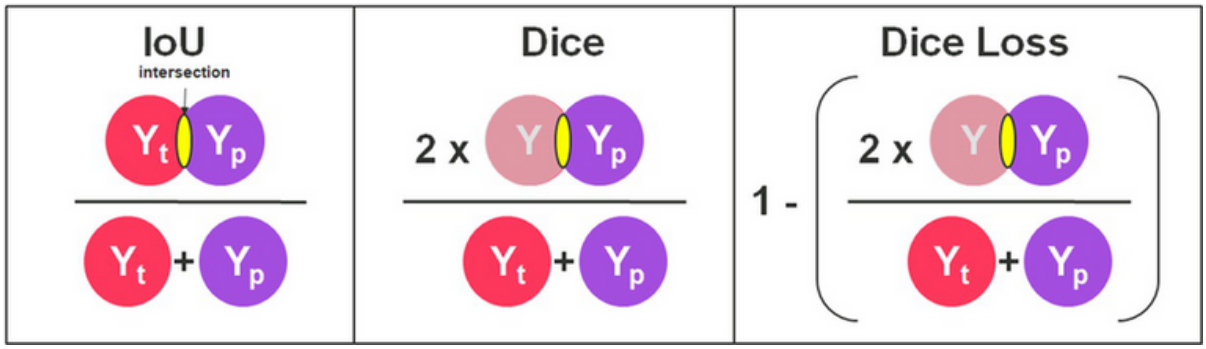
Ce modèle a ensuite été intégré dans une API FastAPI capable de traiter les images et de renvoyer les masques segmentés au format PNG. Le déploiement, réalisé avec Docker sur Google Cloud Run, inclut une interface Streamlit pour une interaction conviviale.

3. TESTS DE LA MEILLEURE FONCTION DE PERTE

Présentation des fonctions de perte testées

Cinq fonctions de perte adaptées à la segmentation d'images ont été testées à l'aide du modèle U-Net Mini, choisi pour sa rapidité d'entraînement.

- **Dice Loss** : Optimise la similarité entre les masques prédits et les vérités terrain. Idéale pour gérer les classes déséquilibrées, elle pénalise efficacement les erreurs de segmentation.
- **Total Loss** : Combine la Binary Crossentropy pour une convergence rapide et la Dice Loss pour améliorer la précision, particulièrement sur les bordures des objets.



- **IoU Loss** : Directement liée à la métrique IoU, elle minimise les erreurs sur les grandes régions mal prédites, renforçant ainsi la précision globale.
- **Tversky Loss** : Variante pondérée de la Dice Loss, elle ajuste l'importance des faux positifs et faux négatifs, ce qui améliore les performances sur les classes minoritaires.
- **Focal Tversky Loss** : Extension de la Tversky Loss, elle accentue l'impact des erreurs grâce à un paramètre exponentiel, rendant cette fonction particulièrement adaptée aux classes rares.

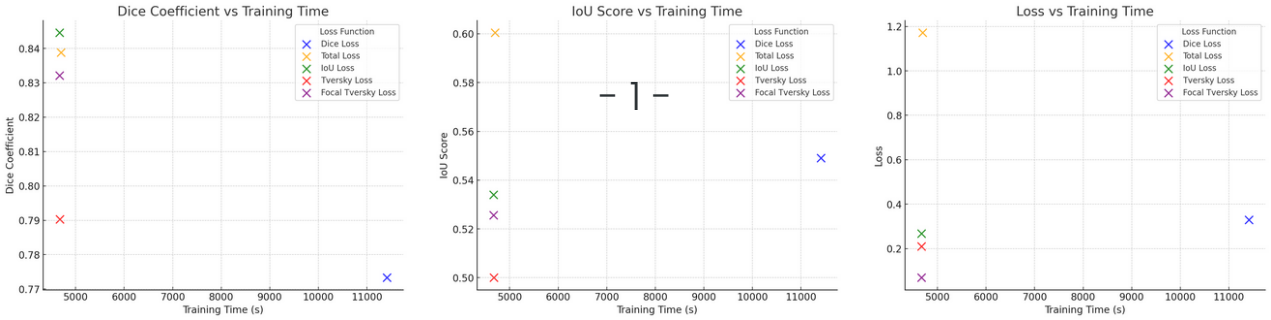
Résultats obtenus avec U-Net Mini

Chaque fonction de perte a été évaluée en termes de trois métriques principales : IoU, Dice Score, et Loss. Ces tests ont permis de comparer leurs impacts respectifs sur la qualité des prédictions.

	Loss Function	Loss	IoU Score	Dice Coefficient	Training Time (s)
0	Dice Loss	0.329876	0.549028	0.773327	11412.677246
1	Total Loss	1.171774	0.600455	0.838799	4696.565830
2	IoU Loss	0.268082	0.533995	0.844572	4667.411883
3	Tversky Loss	0.209705	0.500018	0.790295	4672.325828
4	Focal Tversky Loss	0.069539	0.525667	0.832158	4668.709084

Retenue de la fonction Total Loss

La **Total Loss** semble être la meilleure option dans ce contexte. Elle offre les meilleures performances (IoU et Dice Coefficient) tout en maintenant un temps d'entraînement raisonnable. Si les ressources sont une contrainte majeure, une alternative serait la IoU Loss, qui est rapide et produit des résultats acceptables.



4. TESTS DES DIFFÉRENTES DATA AUGMENTATIONS

Stratégies d'augmentation testées

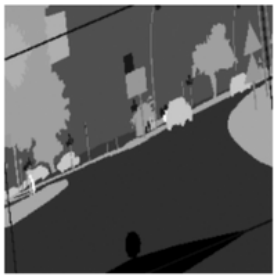
Pour améliorer la robustesse des modèles, plusieurs niveaux de data augmentation ont été testés, chacun introduisant des variations croissantes dans les données d'entraînement.

- Sans augmentation : Les données originales ont été utilisées sans transformation, servant de référence pour évaluer l'impact des augmentations.
- Augmentation légère : Transformations simples telles que rotations mineures, retournements horizontaux et ajustements de contraste pour introduire de légères variations.
- Augmentation moyenne : Ajout de bruit gaussien, flou léger et rotations modérées pour enrichir la diversité des exemples.
- Augmentation modérée : Techniques plus avancées, comme les déformations élastiques et les ajustements de saturation, pour simuler des conditions variées.

- Augmentation forte : Transformations intensives incluant des zooms, des distorsions géométriques et des transformations de perspective pour évaluer les limites du modèle.
- Augmentation avancée : Mélange de toutes les techniques, avec des ajouts comme la simulation de pluie ou de neige, et le masquage aléatoire, visant à préparer le modèle pour des scénarios complexes et imprévisibles.

*image originale

*image augmentée



*masque augmenté

Résultats obtenus

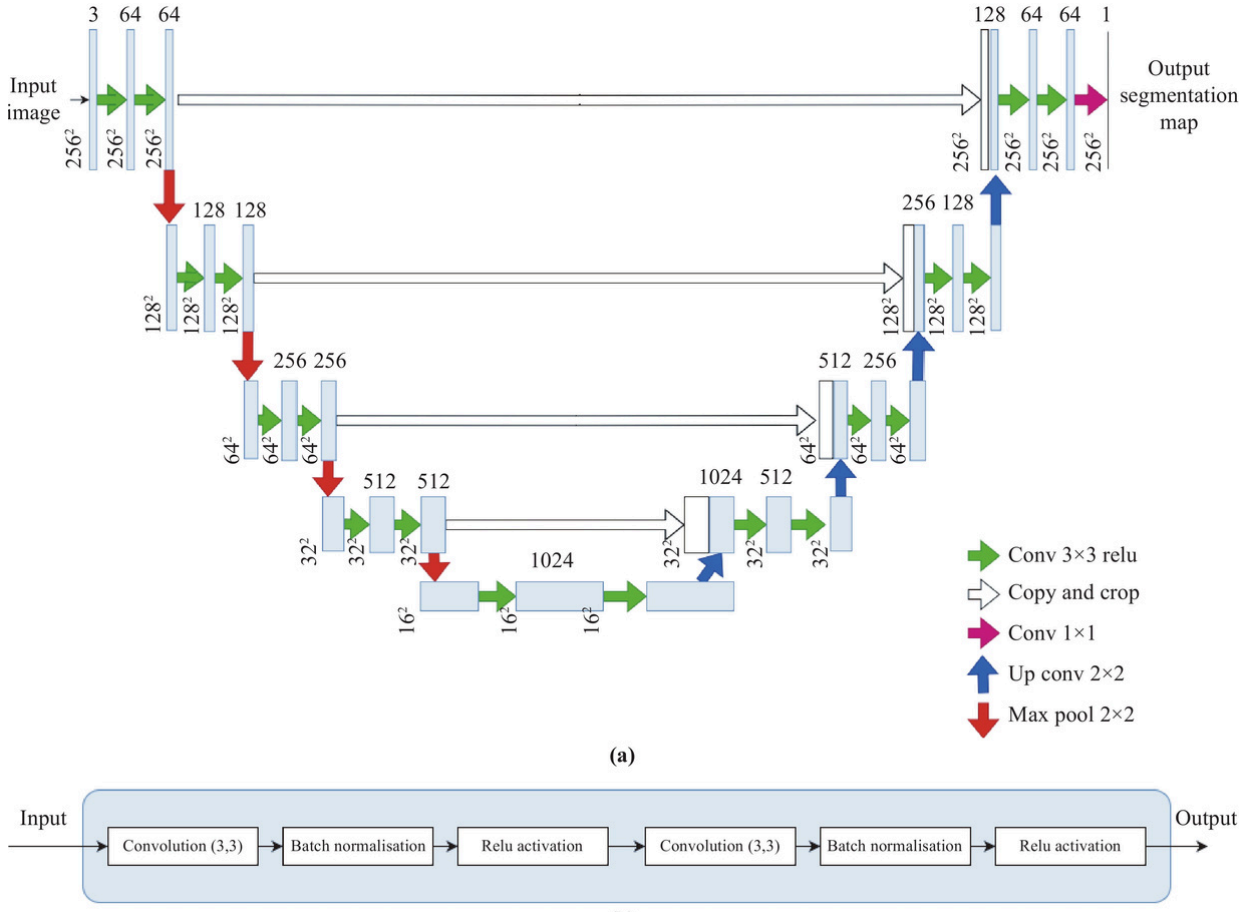
L'augmentation légère a offert la meilleure configuration, permettant au modèle de généraliser efficacement tout en conservant une perte et des métriques optimales.

	Augmentation Level	Loss	IoU Score	Dice Coefficient	Training Time (s)
0	sans	1.202294	0.593455	0.840556	4674.127475
1	léger	1.188901	0.601575	0.837261	4687.173068
2	moyen	1.193900	0.598940	0.841340	4708.765272
3	fort	1.280387	0.569966	0.821273	4792.935228
4	modérée	1.260652	0.572788	0.827710	4767.465860
5	très fort	1.369935	0.542283	0.804376	4849.268191

5. COMPARAISON DES MODÈLES

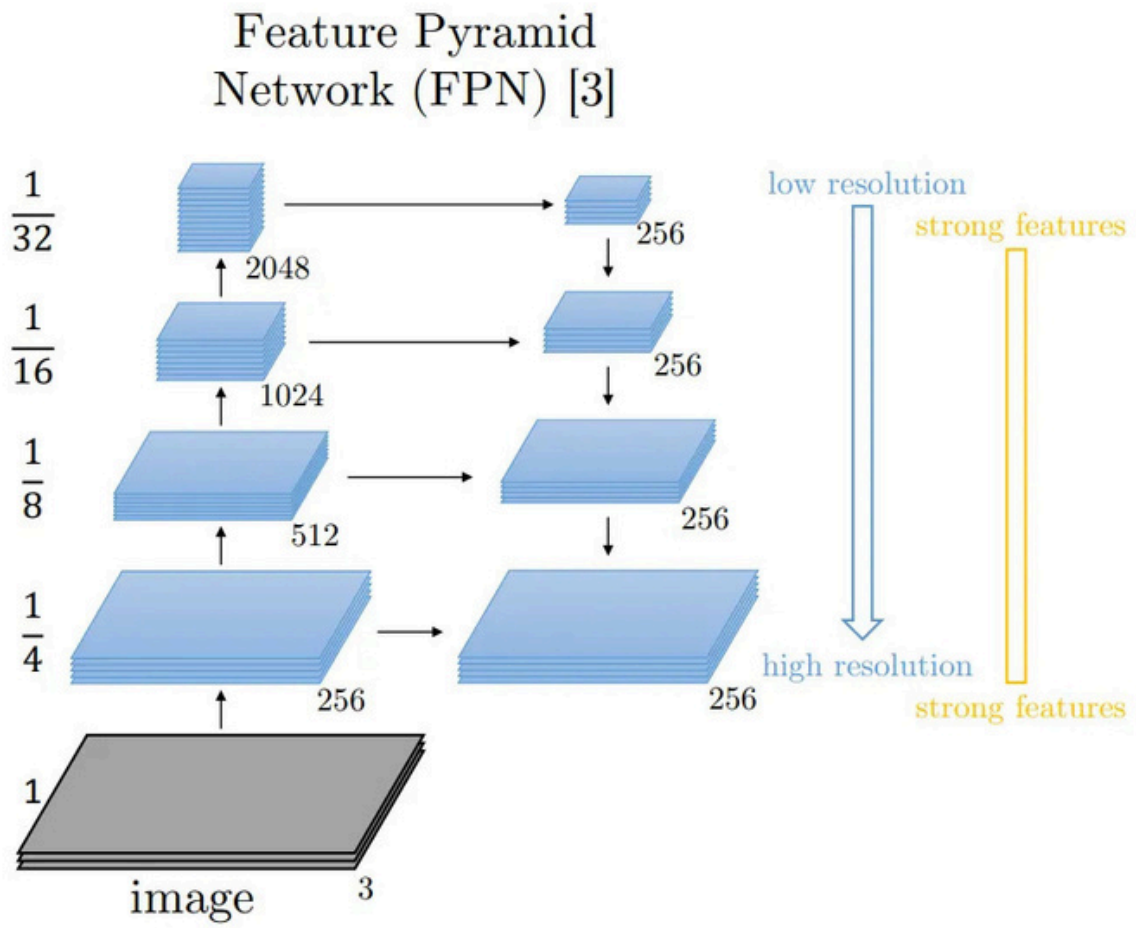
Modèles classiques de segmentation

- ***U-Net*** : *Combinant encodeur, décodeur et connexions directes, il offre une segmentation précise avec des ressources modérées.*



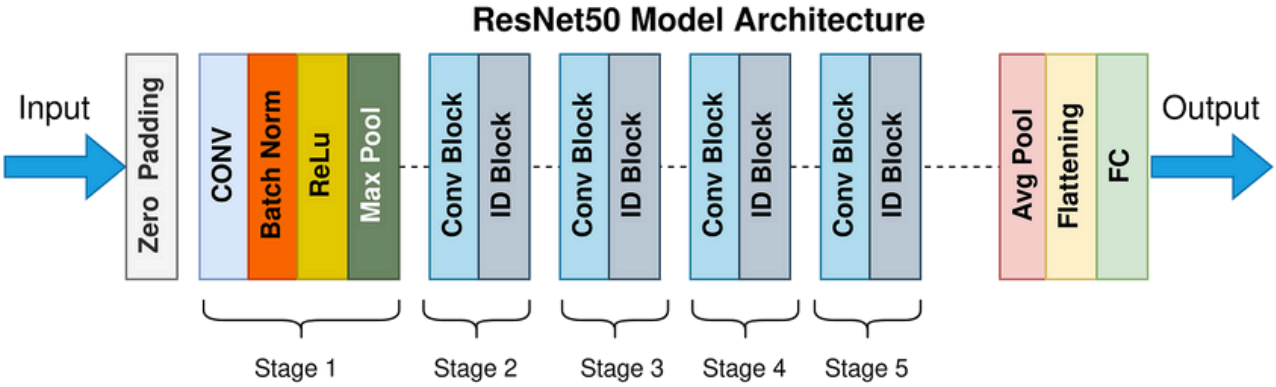
- ***SegNet*** : *Utilise des indices de pooling pour le décodeur, mais l'absence de connexions directes peut limiter la qualité des détails segmentés.*
- ***DeepLabV3+*** : *Intègre des convolutions atrous et des modules ASPP pour capturer des contextes globaux, idéal pour les scènes complexes.*
- ***PSPNet*** : *Le pooling pyramidal gère efficacement les objets de tailles variées, équilibrant précision locale et globale.*

- ***FPN*** : *Exploite une pyramide de caractéristiques pour combiner détails fins et contexte global, performant pour les scènes urbaines.*



Backbones utilisés

- **VGG16** : Référence classique, simple et efficace, pour capturer les détails structuraux des images.
- **ResNet34** : Grâce à ses connexions résiduelles, il combine stabilité et légèreté, idéal pour extraire des contextes globaux.
- **EfficientNetB0** : Moderne et optimisé pour l'efficacité, il offre un excellent équilibre entre précision et ressources.
- **ResNet50** : Version plus profonde de ResNet, il excelle dans l'extraction de caractéristiques complexes, idéal pour des modèles exigeants comme DeepLabV3+ et FPN.



Étapes suivies pour chaque modèle

Tous les modèles ont été configurés de manière standard : architecture validée par un résumé (summary), compilation avec la fonction de perte Total Loss, entraînement avec des augmentations optimales, et évaluation via les métriques IoU et Dice Score. Les modèles ont été sauvegardés au format H5 pour un usage ultérieur.

Résultats obtenus

Le FPN avec ResNet50 s'est distingué par des métriques supérieures (IoU, Dice Score) et des courbes d'apprentissage stables. Ses masques segmentés préservent mieux les détails tout en maintenant une cohérence globale, surpassant ainsi les autres modèles en précision et robustesse.

	Modèles + Backbones	Dice Score	IoU score	Loss	Training Time (s)
0	Unet-Mini	0,851476	0,625975	0,932589	27477
1	Unet - Resnet34	0,886292	0,694511	0,717363	17070
2	Unet - EfficientNetB0	0,895946	0,709106	0,675505	20533
3	Segnet - VGG16	0,860424	0,635287	0,866438	44150
4	PSPnet - ResNet50	0,859045	0,627893	0,921813	14337
5	DeepLabV3+ - ResNet50	0,874207	0,660959	0,826846	21856
6	FPN - ResNet50	0,916729	0,770361	0,506423	180049

6. DÉVELOPPEMENT ET DÉPLOIEMENT DE L'API

Développement et déploiement de l'API

L'API FastAPI inclut des fonctionnalités clés : chargement automatique des modèles depuis Google Cloud, validation des images soumises, post-traitement des masques (colorisation, classes discrètes), sélection du modèle via une liste déroulante, et retour des masques segmentés au format PNG.

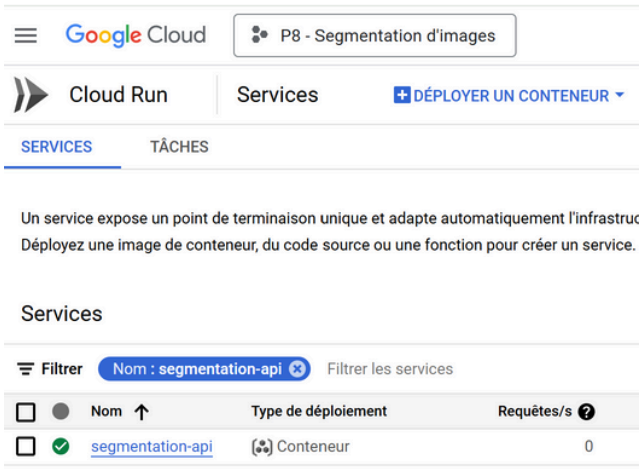


*image docker sur Google Cloud Run

Le déploiement s'appuie sur Docker pour encapsuler les dépendances, et Google Cloud Run pour un environnement serverless, garantissant scalabilité et simplicité de gestion.

Interface utilisateur et tests

Une interface interactive, développée avec Streamlit Cloud, permet de soumettre des images, choisir un modèle et visualiser les résultats. La communication avec FastAPI repose sur des requêtes HTTP POST. L'API a été testée localement avec Uvicorn, avec validation visuelle des résultats sur des cas réels.

*Interface de tests locale

7. RÉSULTATS ET ANALYSE GLOBALE

Synthèse des performances

Les choix méthodologiques ont eu un impact notable sur les performances des modèles. La **Total Loss**, en équilibrant précision et robustesse, s'est distinguée comme la fonction de perte optimale, particulièrement efficace pour les classes sous-représentées.
L'**augmentation légère des données** a offert le meilleur compromis, améliorant les métriques tout en maintenant une stabilité d'entraînement, contrairement aux niveaux plus élevés d'augmentation qui ont dégradé les performances.
Le **FPN avec ResNet50** a affiché les meilleurs résultats, atteignant un IoU de 0.77, un Dice Score de 0.91, et une perte réduite à 0.50. Ces résultats, combinés à des courbes d'apprentissage stables, confirment sa robustesse pour la segmentation d'images.

8. CONCLUSION ET PERSPECTIVES

Résumé des choix méthodologiques et des résultats obtenus

Le projet a adopté une approche méthodique pour développer un modèle performant de segmentation d'images. La Total Loss et une augmentation légère ont été les choix les plus efficaces, tandis que le modèle FPN avec ResNet50 a affiché des performances robustes. L'intégration d'une API FastAPI et d'une interface Streamlit a démontré la praticité des résultats en temps réel.

Limites

- Dataset limité : Les 2975 images d'entraînement du dataset Cityscapes restreignent la généralisation des modèles.
- Classes sous-représentées : Les catégories rares, comme les piétons, sont pénalisées par la dominance des classes majoritaires.

Améliorations futures

- Backbones avancés : Tester des architectures comme EfficientNetB7 pour capturer davantage de détails dans des scènes complexes.
- Résolution native : Utiliser des images haute résolution (1024x2048 pixels) pour des segmentations plus précises, malgré le coût matériel.