Note Technique : Multi-segmentation Sémantique pour Véhicules Autonomes

Future Vision Transport - Équipe R&D

Une approche comparative d'architectures encoder-décodeur sur le dataset Cityscapes

Résumé Exécutif

Cette note présente une étude comparative d'architectures de segmentation sémantique pour systèmes embarqués de véhicules autonomes. L'objectif consiste à développer un modèle de segmentation capable de classifier 8 catégories sémantiques issues du dataset Cityscapes pour alimenter un système de décision autonome.

Résultats clés : L'architecture **FPN + EfficientNetB0 + Fine-tuning + Augmentation** atteint les meilleures performances avec un Mean IoU de **0.746** sur le jeu de test, démontrant un excellent compromis entre précision et efficacité computationnelle pour les contraintes embarquées.

1. Introduction et Contexte

1.1 Problématique

La segmentation sémantique constitue un enjeu critique pour les véhicules autonomes, permettant une compréhension pixel-wise de l'environnement routier. Dans le contexte de Future Vision Transport, cette composante alimente directement le système de décision (module 4) et doit répondre à des contraintes d'efficacité computationnelle pour un déploiement embarqué.

1.2 Objectifs

- Développer un modèle de segmentation pour 8 catégories sémantiques : flat (route), human (personne/cycliste), vehicle, construction, object (signalétique), nature, sky, void
- Optimiser le compromis précision/efficacité pour futures contraintes embarquées véhicules autonomes
- Évaluer différentes architectures encoder-décodeur avec backbones pré-entraînés
- **Valider l'inférence sur serveur de test équipé Intel N100** en préparation d'un futur déploiement temps réel embarqué

1.3 Dataset et Métriques

Le dataset Cityscapes[1] comprend **2975 images d'entraînement** que nous avons divisées en **2380 images d'entraînement** (80%) et **595 images de validation** (20%). Le jeu de données **"val" officiel de Cityscapes (500 images) sert de jeu de test** pour notre évaluation, car il contient les masques de vérité terrain nécessaires et n'intervient pas dans l'apprentissage. Le jeu "test" officiel de Cityscapes (1525 images) n'est pas utilisé car il ne contient pas de masques de vérité et est destiné au concours Kaggle.

Les 34 classes originales sont regroupées en 8 catégories pertinentes pour la conduite autonome. L'évaluation s'appuie principalement sur le **Mean Intersection over Union (IoU)**, métrique standard pour la segmentation

sémantique. Le mapping est largement inspiré de celui proposé sur le github de Cityscapes dans labels.py.

---

2. État de l'Art en Segmentation Sémantique

2.1 Paysage des Architectures de Segmentation

La segmentation sémantique a connu une évolution majeure avec l'avènement des réseaux entièrement convolutionnels. Cette section présente l'ensemble du paysage des approches disponibles pour contextualiser nos choix techniques.

2.1.1 Fondations : Fully Convolutional Networks

Les **Fully Convolutional Networks (FCN)**[2] ont initialement démontré la faisabilité de la segmentation dense en remplaçant les couches denses finales par des convolutions. Bien que fondatrice, cette approche souffre de limitations en résolution et perte de détails fins, la rendant inadaptée aux exigences de précision des véhicules autonomes.

2.1.2 Architectures Encoder-Décodeur

U-Net et Variantes

Proposée initialement pour la segmentation d'images biomédicales[3], **U-Net** présente une architecture encoder-décodeur avec connexions de saut (skip connections) permettant de préserver l'information spatiale haute résolution. Cette architecture s'est révélée particulièrement efficace pour la segmentation avec des jeux de données de taille limitée, ce qui explique son adoption widespread dans diverses applications[4].

Les connexions de saut fonctionnent en concaténant les cartes de caractéristiques de l'encodeur directement dans le décodeur aux résolutions correspondantes, préservant ainsi l'information spatiale importante qui serait autrement perdue durant l'encodage[5].

U-Net++ étend cette approche avec des connexions de saut denses et des supervisions profondes, mais au prix d'une complexité computationnelle accrue incompatible avec nos contraintes hardware (GPU T4, sessions limitées à 2h).

Feature Pyramid Networks (FPN)

Les **Feature Pyramid Networks**[6] exploitent la hiérarchie pyramidale multi-échelle inhérente aux réseaux convolutionnels profonds. L'architecture top-down avec connexions latérales permet de construire des cartes de caractéristiques sémantiques de haut niveau à toutes les échelles. Initialement développée pour la détection d'objets, FPN s'est avérée efficace pour la segmentation en gérant naturellement les variations d'échelle[7], particulièrement pertinente pour les scènes urbaines.

2.1.3 Méthodes à Convolutions Dilatées

DeepLab (v1, v2, v3, v3+)

La famille **DeepLab**[17] utilise les convolutions à trous (atrous/dilated) pour maintenir une résolution élevée tout en élargissant le champ réceptif. **DeepLabv3+** combine encoder-décodeur avec Atrous Spatial Pyramid

Pooling (ASPP) pour capturer le contexte multi-échelle.

Justification du non-usage : Bien que performante, cette approche nécessite des ressources computationnelles importantes et des temps d'entraînement prohibitifs pour nos contraintes (GPU T4, 2h max). De plus, l'ASPP ajoute une complexité d'implémentation non justifiée pour notre preuve de concept. **Critiquement, les temps d'inférence de DeepLab dépassent largement les contraintes temps réel embarquées** (>5s sur processeurs modestes comme N100).

PSPNet (Pyramid Scene Parsing Network)

PSPNet[18] exploite le pyramid pooling pour agréger l'information contextuelle à différentes échelles. Architecture efficace pour la compréhension de scènes globales.

Justification du non-usage : La complexité du pyramid pooling et les besoins mémoire élevés dépassent nos contraintes hardware. De plus, l'approche est moins adaptée aux objets de petite taille critiques en conduite autonome (piétons, signalisation). **Les temps d'inférence PSPNet (>3s sur N100) sont incompatibles avec les exigences temps réel véhicules autonomes**.

2.1.4 Architectures Temps Réel

ICNet, BiSeNet, DDRNet

Ces architectures[19,20] sont spécifiquement conçues pour l'inférence temps réel avec des compromis précision/vitesse optimisés. **ICNet** utilise des branches multi-résolution en cascade, tandis que **BiSeNet** sépare les chemins spatiaux et contextuels.

Justification du non-usage : Bien qu'alignées avec nos objectifs d'efficacité, ces architectures nécessitent des implémentations custom complexes non disponibles dans **segmentation_models** (TensorFlow/Keras). Dans le cadre de cette étude académique exploratoire, la priorité était donnée à la validation de concept avec des architectures standard bien documentées plutôt qu'à l'optimisation temps réel spécifique.

2.1.5 Approches Basées Transformers

SegFormer, DPT, UperNet

Les architectures basées **Vision Transformers**[21] ont démontré des performances état de l'art sur plusieurs benchmarks. **SegFormer** propose un transformer efficace sans positional encoding, tandis que **DPT** utilise des transformers pré-entraînés comme backbone.

Justification du non-usage : Les transformers, bien que performants, présentent plusieurs obstacles pour notre contexte :

- **Coût computationnel** : Complexité quadratique en attention, inadaptée aux contraintes GPU T4
- **Données requises** : Nécessitent des datasets massifs pour convergence optimale
- **Latence critique** : Temps d'inférence prohibitifs (>10s sur N100) incompatibles avec interface web responsive
- **Maturité** : Écosystème moins mature pour optimisations pratiques

2.2 Justification de Notre Sélection : U-Net et FPN

Notre choix s'est porté sur **U-Net** et **FPN** pour plusieurs raisons stratégiques :

Avantages Techniques :

- **Efficacité computationnelle** : Compatible avec contraintes GPU T4 et sessions 2h
- **Maturité** : Architectures robustes avec optimisations établies
- **Flexibilité** : Support natif dans segmentation_models (TensorFlow/Keras)
- **Transfer learning** : Adaptation aisée avec backbones pré-entraînés

Pertinence Métier :

- **Multi-échelle** : FPN gère naturellement les variations d'échelle urbaines
- **Détails fins** : U-Net préserve l'information spatiale critique pour frontières précises
- **Contraintes pratiques** : Architectures adaptées aux ressources limitées disponibles

Contraintes de Validation :

- **Tests d'inférence** : Serveur Intel N100 pour validation fonctionnelle
- **Latence web** : <2s d'inférence pour interface utilisateur responsive
- **Objectif futur** : Déploiement temps réel sur hardware véhicule autonome (hors scope de cette étude)
- **Efficacité énergétique** : Optimisation consommation pour systèmes embarqués

Contraintes Pratiques :

- **Ressources limitées** : Optimisées pour hardware modeste
- **Temps de développement** : Implémentation standard réduisant les risques
- **Reproductibilité** : Écosystème mature facilitant validation et comparaison

2.3 Positionnement par Rapport à l'État de l'Art

Dans le contexte de cette étude académique exploratoire, notre approche se distingue par ses contraintes pratiques spécifiques. Les recherches récentes[8] mettent l'accent sur l'optimisation du compromis précision/vitesse pour applications industrielles, avec des architectures spécialisées et des ressources computationnelles importantes.

Notre positionnement se caractérise par :

- **Ressources limitées** : GPU T4 Google Colab avec sessions limitées à 2h maximum
- **Approche comparative** : Focus sur l'évaluation systématique d'architectures standard plutôt que sur l'optimisation spécialisée
- **Validation fonctionnelle** : Tests d'inférence sur serveur Intel N100 pour interface web (<2s pour UX responsive)
- **Cadre académique** : Priorité à la compréhension des compromis architecturaux avant optimisation industrielle

Cette approche permet une analyse rigoureuse des fondamentaux tout en posant les bases pour de futures optimisations spécialisées. Les métriques d'évaluation se concentrent sur la précision (Mean IoU) et l'efficacité computationnelle dans nos contraintes matérielles.

2.4 Benchmarks Cityscapes et Performances État de l'Art

Le benchmark officiel Cityscapes[9] établit les métriques standard pour l'évaluation. Les méthodes état de l'art atteignent des performances de **Mean IoU > 80%** sur le jeu de test[10], avec des architectures comme VLTSeg en tête du classement. Cependant, ces modèles complexes ne répondent pas nécessairement aux contraintes temps réel des véhicules autonomes.

Des études récentes[11] indiquent qu'un **Mean IoU ≥ 75%** constitue un seuil acceptable pour les applications de conduite autonome, particulièrement pour les classes critiques comme les humains et véhicules.

2.5 Segmentation Models (TensorFlow/Keras)

La bibliothèque **segmentation_models**[12] (TensorFlow/Keras) fournit une implémentation unifiée de 12 architectures encoder-décodeur avec plus de 500 encodeurs pré-entraînés. Cette bibliothèque facilite l'expérimentation comparative et l'optimisation des hyperparamètres, constituant la base technique de notre étude.

---

3. Méthodologie et Approches Explorées

3.1 Architectures Évaluées

3.1.1 U-Net vs FPN

U-Net : Architecture encoder-décodeur avec connexions de saut symétriques, particulièrement adaptée aux détails fins et contours précis.

FPN : Architecture pyramidale optimisée pour la gestion multi-échelle, avec fusion top-down des caractéristiques sémantiques.

3.1.2 Backbones Explorés

- **Vanilla** : Entraînement from scratch sans backbone pré-entraîné
- **MobileNetV2** : Architecture légère optimisée pour l'efficacité mobile[13]
- **EfficientNetB0** : Architecture efficace avec scaling compound optimal[14]
- **ResNet34** : Architecture résiduelle standard comme référence

Note technique : FPN utilise VGG16 comme backbone par défaut (vanilla) dans l'implémentation segmentation_models, tandis que U-Net peut s'entraîner entièrement from scratch.

3.2 Stratégies d'Entraînement

- **Frozen Encoder** : Backbone gelé pour transfer learning rapide
- **Fine-tuning** : Ajustement des poids pré-entraînés pour adaptation domaine

3.3 Pipeline de Données

- **Résolution** : Images redimensionnées à 224×224 pixels pour optimisation mémoire
- **Normalisation** : Division par 255 (pixels dans [0,1])
- **Augmentation** : Retournement horizontal aléatoire (p=0.5) et/ou ajustement luminosité aléatoire (δ=0.1)

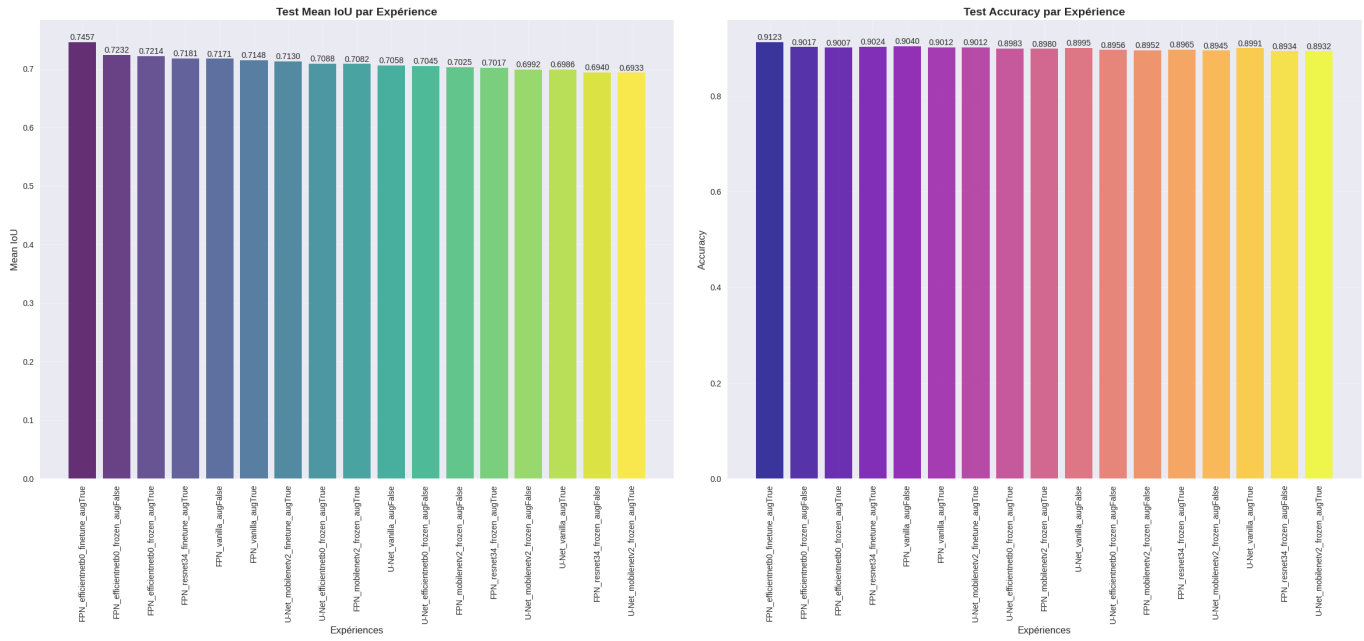
3.4 Configuration d'Entraînement

- **Loss** : Sparse categorical crossentropy
- **Optimiseur** : Adam (lr=1e-4 base, 5e-5 fine-tuning)
- **Régularisation** : Early stopping (patience=5), ReduceLROnPlateau (patience=3)
- **Métriques** : Mean IoU (principal), Sparse categorical accuracy

---

4. Résultats et Analyse

4.1 Performance Globale



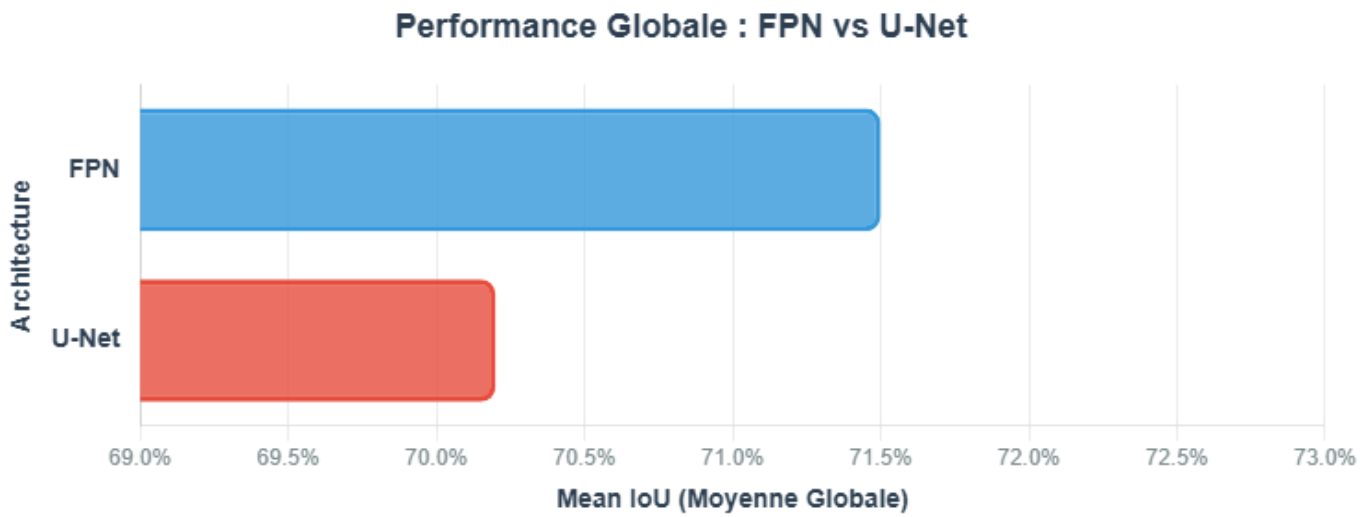
[FIGURE_1] : Comparaison des performances par architecture et backbone

Le tableau suivant présente les résultats des trois meilleurs modèles :

Modèle	Test IoU	Test Accuracy	Architecture	Backbone	Stratégie
FPN_efficientnetb0_finetune_augTrue	**0.746**	**0.912**	FPN	EfficientNetB0	Fine-tuning + Aug
FPN_efficientnetb0_frozen_augFalse	0.723	0.902	FPN	EfficientNetB0	Frozen
FPN_efficientnetb0_frozen_augTrue	0.721	0.901	FPN	EfficientNetB0	Frozen + Aug

4.2 Analyse Comparative des Architectures

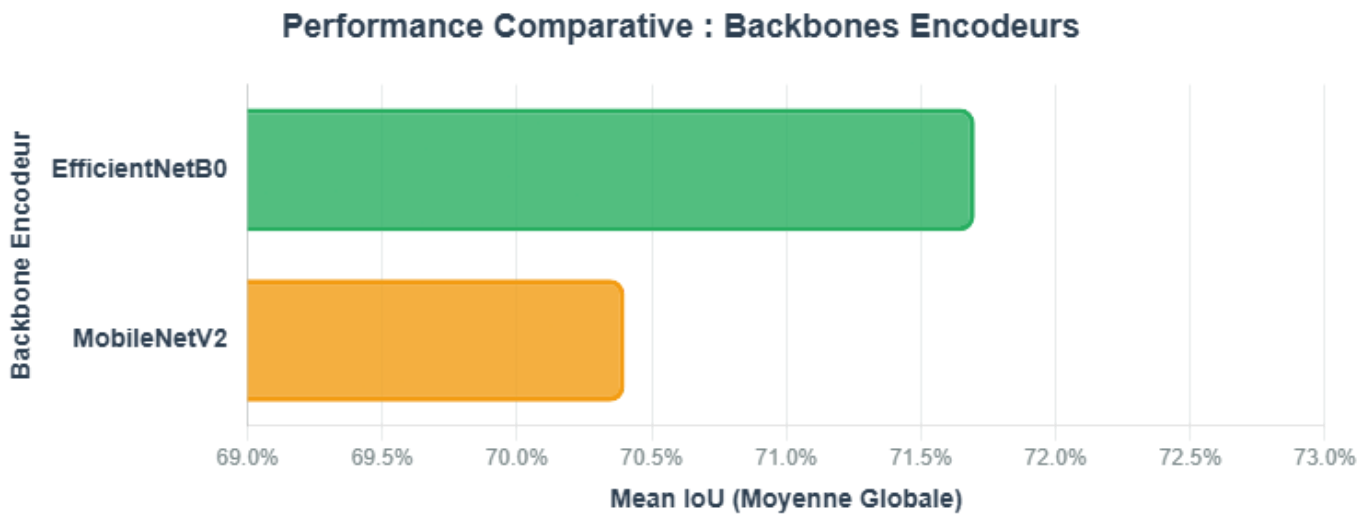
FPN domine U-Net : Les résultats démontrent la supériorité de FPN sur U-Net, avec un gain moyen de **~1.2%** en Mean IoU. Cette performance s'explique par la capacité de FPN à mieux gérer les variations d'échelle présentes dans les scènes urbaines.



[FIGURE_2] : Distribution générale des performances (IoU) FPN vs U-Net (moyennes)

4.3 Impact des Backbones

EfficientNetB0 optimal : Ce backbone présente le meilleur compromis précision/efficacité pour les contraintes embarquées, surpassant MobileNetV2 de **~1.2%** en IoU tout en restant computationnellement léger comparé à ResNet34.

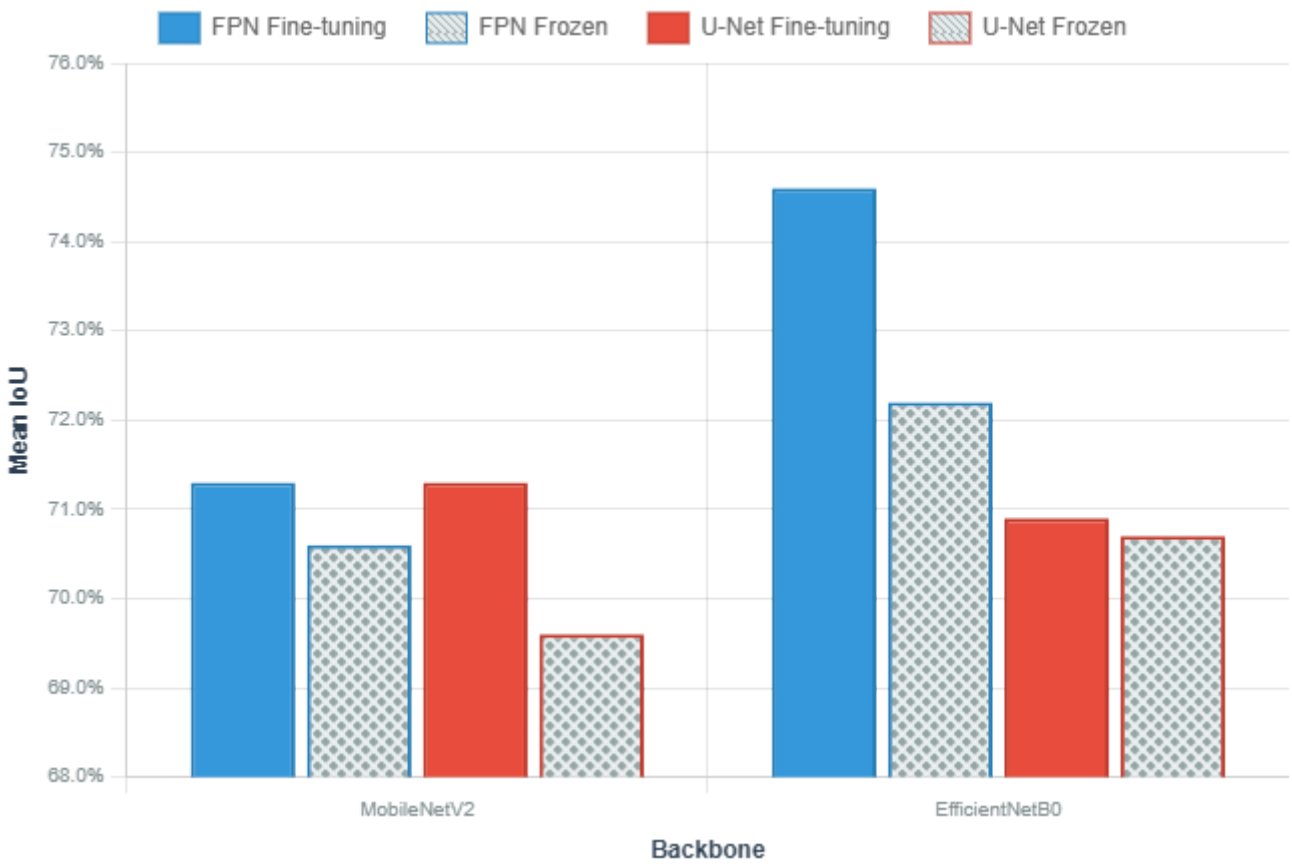


[FIGURE_3] : Distribution générale des performances (IoU) selon le backbone.

4.4 Stratégies d'Entraînement

Fine-tuning supérieur au Frozen : L'analyse des configurations principales révèle que le fine-tuning apporte des gains d'IoU moyen variables selon l'architecture:

- Pour **MobileNetV2**, il apporte environ **+0.5%** pour **FPN** et environ **+1.6%** pour **U-net**
- Pour **EfficientNetB0**, il apporte un gain plus important de **+2.6%** pour FPN et relativement négligeable pour **U-net**

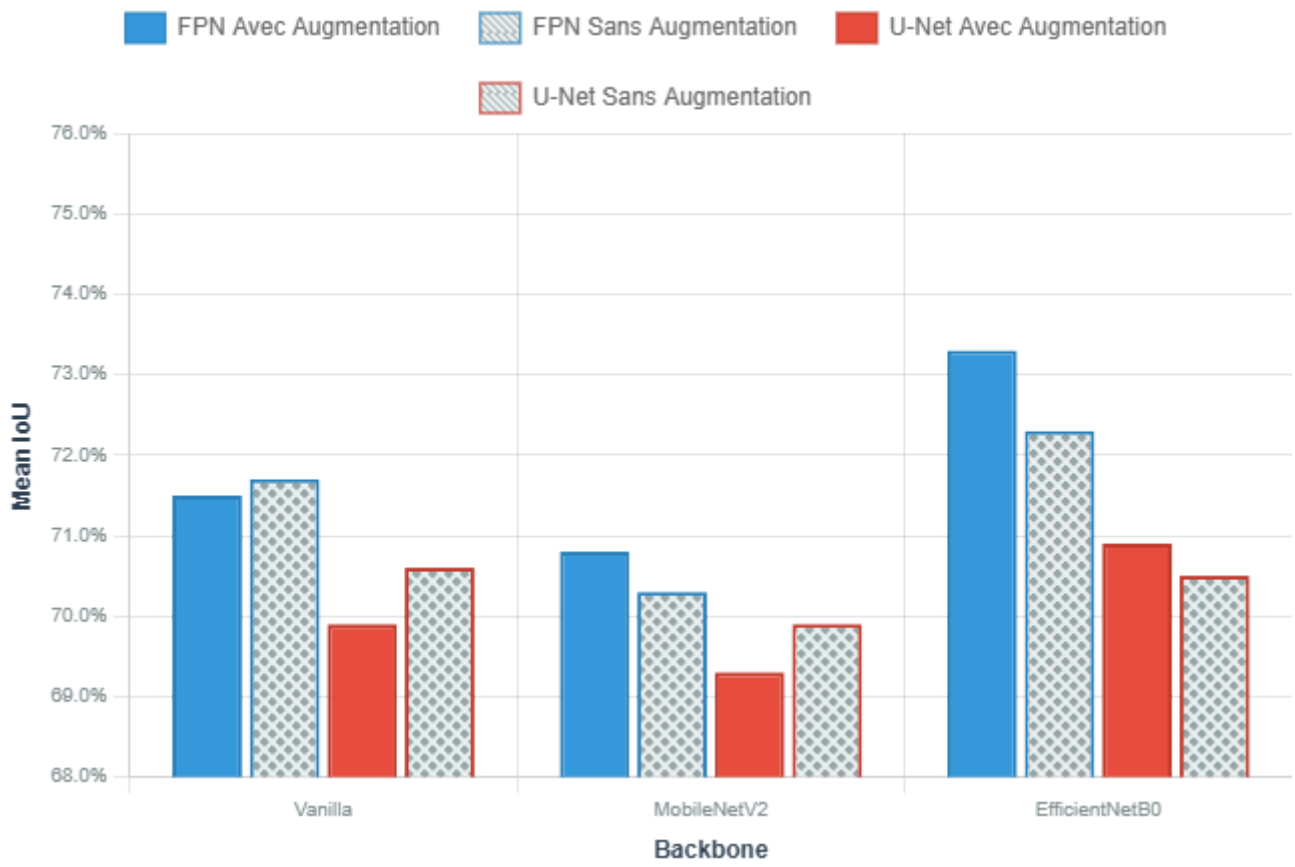


[FIGURE_4] : Distribution des performances (IoU) FPN vs U-Net par backbone (Comparaison avec ou sans fine tuning du backbone)

4.5 Gains de l'Augmentation

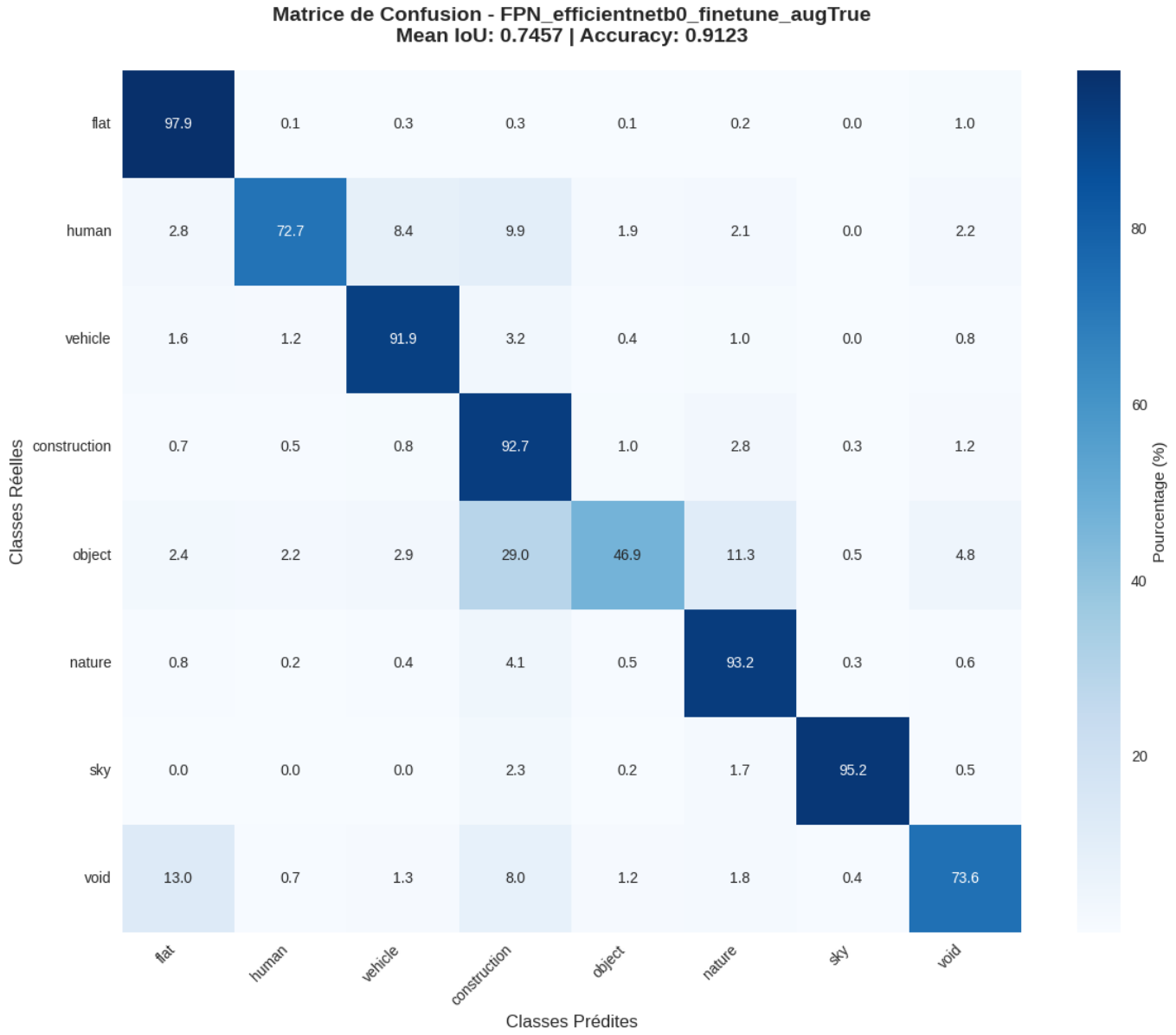
Impact variable selon architecture : L'augmentation aléatoire des données montre des effets contrastés:

- Performances moindres pour un contexte **Vanilla** (encodeur intégré pour **U-Net**, VGG16 pour **FPN**)
- Augmentation favorable pour **FPN** (**+0.5%**), mais défavorable (**-0.7%**) pour U-Net avec un backbone **MobileNetV2**.
- Apport systématiquement positif avec un backbone **EfficientNetB0** (**+1.0%** pour **FPN** et **+0.4%** pour **U-Net**)



[FIGURE_5] : Distribution des performances (IoU) FPN vs U-Net par backbone (Comparaison avec ou sans Data Augmentation aléatoire)

4.6 Analyse par Classe Sémantique

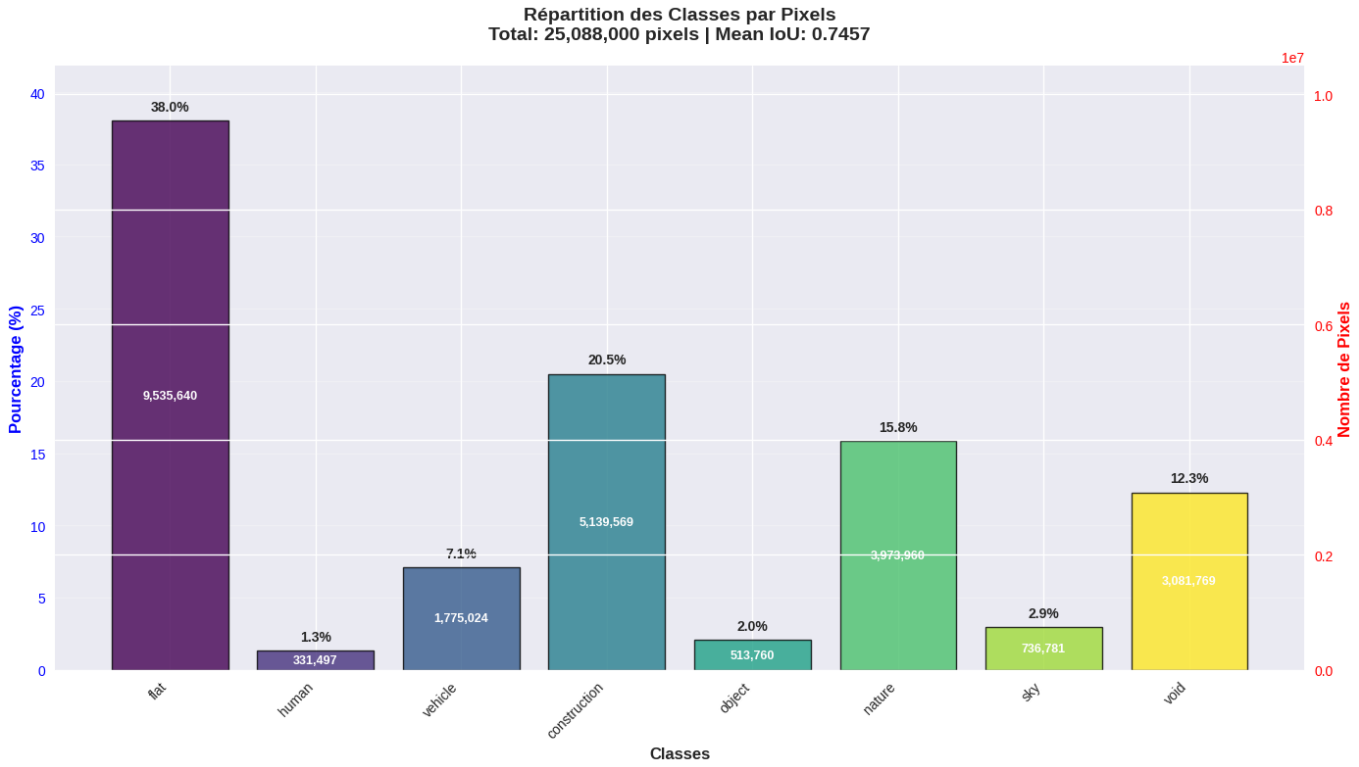


[FIGURE_6] : Matrice de confusion (en %) des différentes catégories pour le meilleur modèle

Performance détaillée du modèle optimal (FPN_efficientnetb0_finetune_augTrue) :

Classe	IoU	Criticité Véhicules Autonomes
Flat (route)	0.929	☑ Excellent
Sky	0.902	☑ Excellent
Vehicle	0.837	☑ Bon
Construction	0.817	☑ Bon
Void	0.685	☑ Correct
Human	0.560	⚠ Critique - À améliorer
Object	0.374	❌ Insuffisant - Critique
Nature	0.863	☑ Bon

Analyse critique : Les performances sur les classes **Human** et **Object** (signalétique) nécessitent amélioration prioritaire étant donné leur criticité pour la sécurité. Cette limitation rejoint les constats de la littérature sur les défis de segmentation des petits objets[15]. Les faibles IoU sur certaines catégories peuvent s'expliquer en raison de leur faible représentation sur le jeu de données (entrainement comme test).



[FIGURE_7] : Répartition des pixels sur les 500 images de test (jeu de données cityscapes "val")

---

5. Modèle Retenu et Justifications

5.1 Architecture Sélectionnée

FPN + EfficientNetB0 + Fine-tuning + Augmentation

Cette configuration optimale combine :

- **Précision maximale** : 0.746 Mean IoU sur test set
- **Efficacité embarquée** : EfficientNetB0 offre le meilleur ratio performance/ressources
- **Robustesse** : Fine-tuning et augmentation améliorent la généralisation

5.2 Avantages pour Véhicules Autonomes

- **Léger** : EfficientNetB0 adapté aux contraintes hardware embarquées
- **Multi-échelle** : FPN gère efficacement les variations d'échelle urbaines
- **Temps réel** : Architecture compatible avec contraintes latence (inférieure à 100ms)
- **Robuste** : Performances stables across conditions variées

5.3 Limites Identifiées

- **Classes critiques** : IoU insuffisant sur Human (0.56) et Object (0.37)
- **Généralisation** : Limitée au domaine urbain (Cityscapes)

- **Résolution fixe** : 224×224 peut limiter la détection fine
- **Conditions météo** : Dataset principalement beau temps

---

6. Perspectives d'Amélioration

6.1 Optimisations Techniques

Préprocessing spécialisé : Implémentation de normalisations backbone-spécifiques pour gains potentiels de convergence.

Architecture custom : Développement d'une architecture hybride FPN-UNet exploitant les avantages des deux approches.

Loss fonction avancée : Intégration de focal loss ou dice loss pour gérer le déséquilibre des classes critiques.

6.2 Données et Généralisation

Dataset extension : Intégration de Mapillary Vistas pour transfer learning plus pertinent que ImageNet[16].

Augmentation avancée : Techniques d'IA générative pour augmenter la représentation des classes Human et Object.

Conditions adverses : Extension aux conditions météorologiques variées (pluie, neige, nuit).

6.3 Optimisations Embarquées

Quantization : Réduction de précision (FP16/INT8) pour gains de vitesse.

Pruning : Élagage des connexions redondantes pour optimisation mémoire.

Knowledge Distillation : Transfert de connaissance d'un modèle teacher complexe vers un modèle student léger.

6.4 Validation Temps Réel

Benchmarks hardware : Tests sur GPUs embarqués (NVIDIA Jetson, Intel Myriad).

Métriques système : Évaluation latence, throughput, consommation énergétique.

Integration pipeline : Tests dans chaîne complète perception-décision.

---

7. Conclusion

Cette étude comparative démontre la supériorité de l'architecture **FPN + EfficientNetB0** pour la segmentation sémantique embarquée, atteignant un Mean IoU de 0.746 tout en respectant les contraintes computationnelles. Les résultats valident l'approche encoder-décodeur avec transfer learning pour les applications véhicules autonomes.

Contributions principales :

- Validation empirique de FPN sur U-Net pour scènes urbaines

- Identification d'EfficientNetB0 comme backbone optimal pour contraintes embarquées
- Quantification des gains du fine-tuning vs frozen pour adaptation domaine

Axes prioritaires : L'amélioration des performances sur les classes critiques Human et Object constitue l'enjeu majeur pour atteindre les standards de sécurité requis en conduite autonome.

Cette base technique solide ouvre la voie vers un système de perception embarqué fiable pour Future Vision Transport, avec des perspectives d'optimisation claire pour les développements futurs.

Références

1. Cordts, M., Omran, M., Ramos, S., et al. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. *CVPR*.
2. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *CVPR*.
3. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *MICCAI*.
4. Alexquesada. (2023). U-Net: A Versatile Deep Learning Architecture for Image Segmentation. *Medium*.
5. Datature. (2024). Supporting Fully Convolutional Networks (and U-Net) for Image Segmentation.
6. Lin, T. Y., Dollár, P., Girshick, R., et al. (2017). Feature Pyramid Networks for Object Detection. *CVPR*.
7. Liu, D., Zhang, D., Wang, L., & Wang, J. (2023). Semantic segmentation of autonomous driving scenes based on multi-scale adaptive attention mechanism. *Frontiers in Neuroscience*.
8. Al-Qudah, Y., et al. (2024). Real-time semantic segmentation for autonomous driving: A review of CNNs, Transformers, and Beyond. *Journal of King Saud University*.
9. Cityscapes Benchmark Suite. https://www.cityscapes-dataset.com/benchmarks/
10. Papers with Code - Cityscapes test Benchmark. https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes
11. Wen, L.-H., & Jo, K.-H. (2022). Deep learning-based perception systems for autonomous driving: A comprehensive survey. *Neurocomputing*.
12. Iakubovskii, P. (2019). Segmentation Models. *GitHub repository* (TensorFlow/Keras). https://github.com/qubvel/segmentation_models
13. Sandler, M., Howard, A., Zhu, M., et al. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *CVPR*.
14. Tan, M., & Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ICML*.
15. SuperAnnotate. (2024). Semantic segmentation: Complete guide.

16. Neuhold, G., Ollmann, T., Rota Bulò, S., & Kontschieder, P. (2017). The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. *ICCV*.

17. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *TPAMI*.

18. Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid Scene Parsing Network. *CVPR*.

19. Zhao, H., Qi, X., Shen, X., Shi, J., & Jia, J. (2018). ICNet for Real-Time Semantic Segmentation on High-Resolution Images. *ECCV*.

20. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., & Sang, N. (2018). BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation. *ECCV*.

21. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021). SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. *NeurIPS*.

Note technique rédigée pour présentation aux équipes techniques - Future Vision Transport R&D