

Projet 8

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

Segmentation sémantique d'images pour véhicules autonomes



David Scanu | juin 2025



Parcours **AI Engineer**



Sommaire

- 🚗 Contexte & Problématique
- 🖼️ Jeu de données Cityscapes
- ⚙️ Preprocessing et augmentation des données
- 🧱 Stratégie de Regroupement des Classes
- 🏗️ Architecture U-Net avec Backbone MobileNetV2
- 🚂 Stratégie d'entraînement et MLflow
- 🛠️ Mise en production : API REST (Backend) & Web App (Frontend)
- 🚀 Pistes d'optimisation
- 💻 Démonstration
- 🏁 Conclusion



Contexte et problématique métier



Contexte

- **Future Vision Transport :**
 - Entreprise spécialisée dans les systèmes embarqués de vision par ordinateur pour véhicules autonomes
- **Notre rôle :**
 - Ingénieurs IA au sein de l'équipe R&D



Architecture du système embarqué

1. Acquisition des images en temps réel
2. Traitement des images (Franck)
3. **Segmentation des images**
4. Système de décision (Laura)



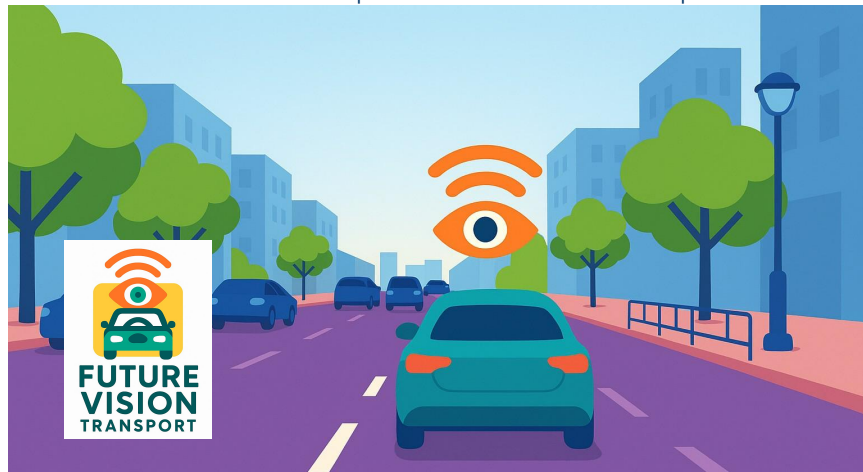
Mission

- Développer un module de **segmentation d'images** performant et industrialisable pour identifier **8 catégories d'objets urbains** en temps réel



Objectifs du Projet

- **Modèle IA :** Segmentation sémantique avec Keras/TensorFlow
- **API Production :** Déploiement FastAPI sur le cloud
- **Interface Demo :** Application Next.js de visualisation
- **Documentation :** Pipeline industrialisable et reproductible



⚡ Jeu de données Cityscapes

Cityscapes

Jeu de données orienté **segmentation sémantique de scènes urbaines**

Caractéristiques Principales

- **5 000 images annotées** avec segmentation fine
- **50 villes européennes** différentes
- **Conditions variées** : Saisons, météo, éclairage
- **Résolution haute** : 2048 × 1024 pixels
- **30+ classes détaillées** avec annotations

✓ Avantages pour Notre Cas d'Usage

1. **Réalisme** : Images capturées depuis véhicules en circulation
2. **Diversité** : Variété de scènes urbaines représentatives
3. **Qualité** : Annotations précises et validation communautaire

Image



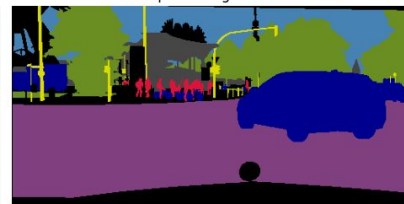
Masque de segmentation



Image



Masque de segmentation



Image



Masque de segmentation



Preprocessing et Augmentation des Données

Division des données

- **Dossier Train** (2 975 images)
 - 80 % pour l'**entraînement (2380 images)**
 - 20 % pour la **validation (595 images)**
- **Dossier Val (500 images)**
 - Utilisées pour le test final
- **Dossier Test** (1 525 images)
 - Non utilisé (masques indisponibles)

Pipeline de Préparation

- **Redimensionnement 2048×1024 → 224×224**
 - Optimisé pour **MobileNetV2**
- **Méthode bilinéaire** : Préservation des détails importants
- **Normalisation** : Division par 255 pour valeurs [0,1]

Techniques d'Augmentation

- **Augmentations Géométriques**
 - Flip horizontal (50% probabilité) - Synchronisé image/masque
- **Augmentations Photométriques**
 - Variation luminosité (± 0.1) - Image uniquement
 - Clipping automatique pour maintenir [0,1]

Objectifs

- Robustesse aux variations de conditions
- Prévention du sur-apprentissage
- Compatibilité contraintes embarquées
















Stratégie de regroupement des classes



Mapping des classes principales

De 30+ classes → **8 catégories** pertinentes

1.   **flat** : road, sidewalk, parking, rail track
2.   **human** : person, rider
3.   **vehicle** : car, truck, bus, on rails, motorcycle, bicycle, caravan, trailer
4.   **construction** : building, wall, fence, guard rail, bridge, tunnel
5.   **object** : pole, pole group, traffic sign, traffic light
6.   **nature** : vegetation, terrain
7.   **sky** : sky
8.   **void** : unlabeled, ego vehicle, rectification border, out of roi, static, dynamic, ground



Bénéfices de cette approche

- **Pertinence fonctionnelle** :
 - Classes alignées sur besoins navigation
- **Performance computationnelle** :
 - Modèle plus léger et rapide
- **Robustesse** :
 - Évite sur-spécialisation sur classes rares
- **Simplicité** :
 - Facilite l'intégration système décision

Architecture U-Net avec Backbone MobileNetV2

▼ Encodeur : MobileNetV2 (gelé)

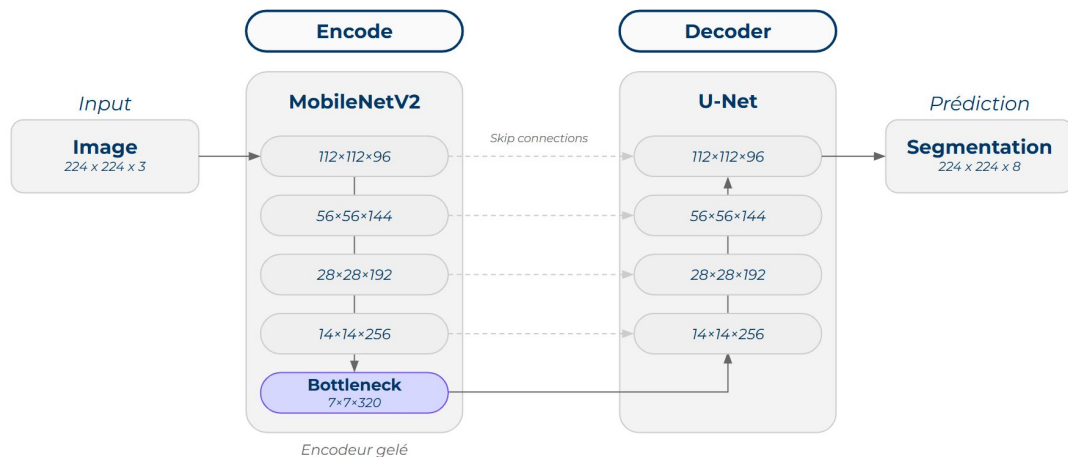
- **Backbone pré-entraîné sur ImageNet**
- **Extraction multi-échelle** : Skip connections
- **Bottleneck** : 7×7 (niveau le plus profond)
- **Avantages** :
 - Transfer learning
 - Efficacité computationnelle

▲ Décodeur : Reconstruction progressive

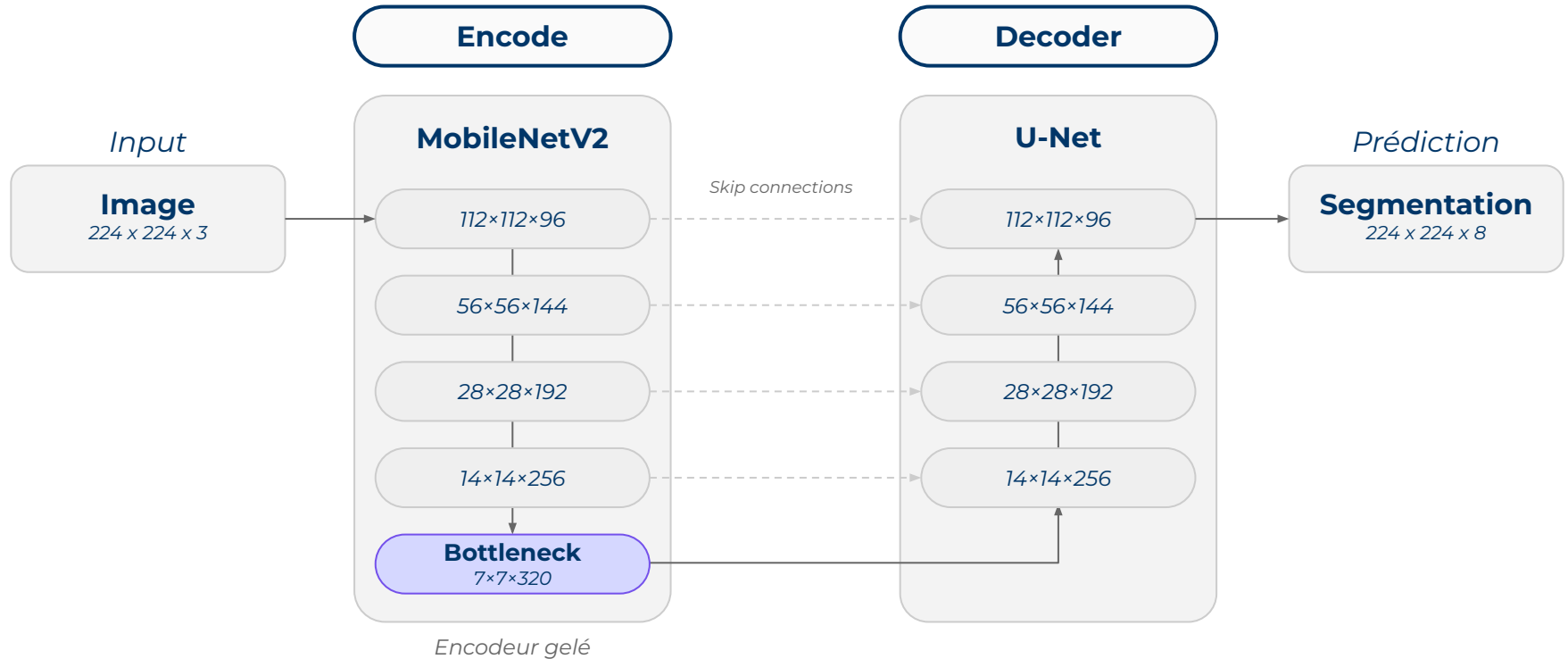
- **Convolutions** transposées pour **upsampling**
- **Skip connections** : Préservation détails spatiaux

🎯 Sortie Finale

- Conv2D (1×1) + **Softmax** → **8 classes par pixel**
- Format : **Masque de segmentation 224×224**



⚡ **Paramètres du Modèle** : 5,4 millions | **Entraînables** : 3,6 millions | **Gelés** : 1,8 millions



Architecture U-Net avec Backbone MobileNetV2 pour Segmentation Sémantique



Stratégie d'entraînement et MLflow



Configuration des hyperparamètres



```
EXPERIMENT_CONFIG = {  
    "learning_rate": 0.0001,  
    "batch_size": 8,  
    "epochs": 20,  
    "base_model_trainable": False, # Encoder gelé  
    "validation_split": 0.2,  
    "augmentation": True  
}
```



Callbacks d'optimisation

- **ReduceLROnPlateau** :
 - Réduction adaptative du learning rate
- **EarlyStopping** :
 - Arrêt anticipé (patience=5)
- **ModelCheckpoint** :
 - Sauvegarde du meilleur modèle



Gestion des Expériences avec MLflow

- **Versioning automatique** des modèles et hyperparamètres
- **Traçabilité complète** des expériences
- **Comparaison des performances**
- **Reproductibilité** garantie



Stratégie Transfer Learning

- **Phase 1** : Encoder **MobileNetV2** gelé
- **Bénéfices** :
 - Convergence rapide
 - Stabilité d'entraînement
 - Réduction du sur-apprentissage
 - Compatible ressources limitées



Perspectives d'Amélioration

- **Phase 2** : Fine-tuning avec **encoder dégelé + learning rate réduit**



Résultats et performance











Performance Globale

- **Mean IoU** : 63.25%
- **Précision pixel-wise** : 87.95%
- **Convergence** : 18 époques / ~2h d'entraînement
- **Paramètres** : 5.4M (3.6M entraînaibles)



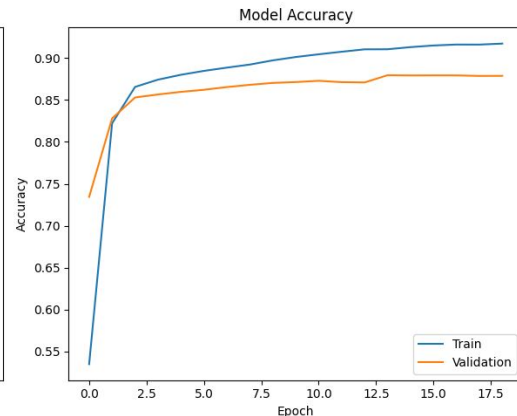
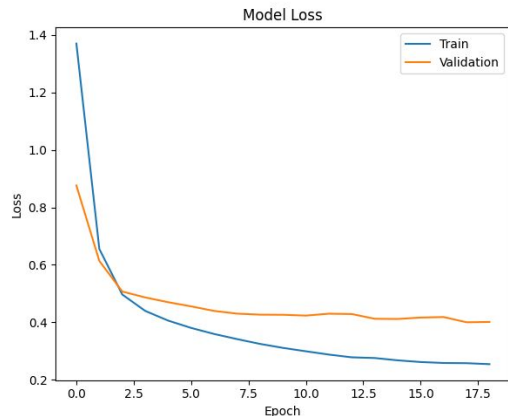
Performance par Catégorie (IoU)

-  **flat** : 90.8% - Excellent
-  **sky** : 83.4% - Excellent
-  **nature** : 79.8% - Bon
-  **vehicle** : 74.7% - Bon
-  **construction** : 74.5% - Bon
-  **void** : 64.3% - Acceptable
-  **human** : 32.0% - Difficile
-  **object** : 6.5% - Très difficile



Évolution de l'Entraînement

- **Convergence rapide** : 5 premières époques critiques
- **Transfer learning efficace** : Exploitation des features ImageNet
- **Stabilisation** : Plateau à 91% (train) / 87% (validation)
- **Sur-apprentissage maîtrisé** : Écart acceptable de 4%








Matrice de Confusion (en millions de pixels)


Performances Excellentes

-  flat,  construction,  nature,  sky

Confusions identifiées

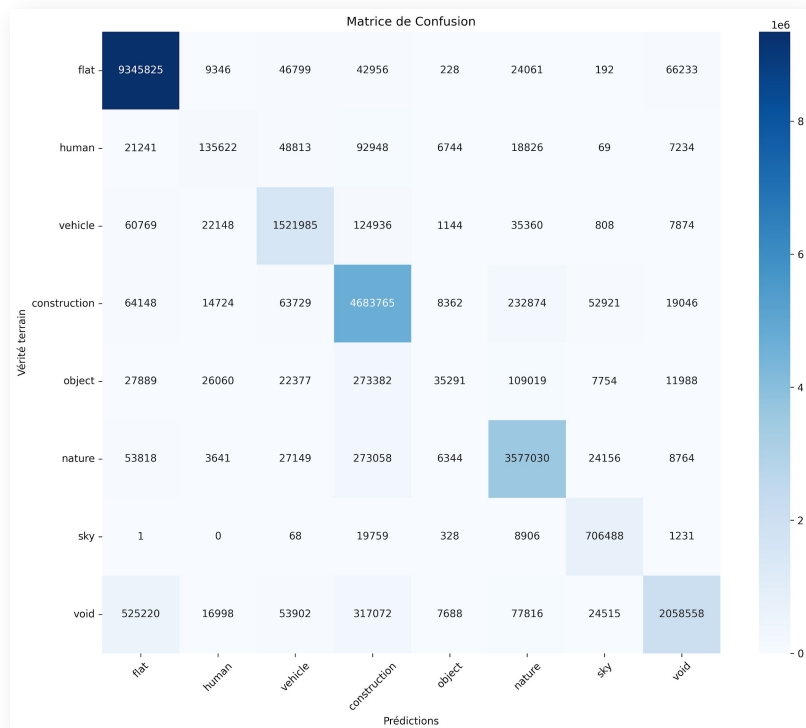
-  Human →  Construction : 93K pixels (personnes devant bâtiments)
-  Vehicle →  Flat : 61K pixels (véhicules sur routes)
-  Void dispersé : Répartition naturelle sur toutes classes

Points d'attention critiques

-  **Object sous-performant** : Seulement 35K/514K pixels détectés
- **Objets fins (panneaux, poteaux) mal segmentés**

Validation du Modèle

Le modèle excelle sur les **éléments structurants essentiels à la navigation** tout en révélant des axes d'amélioration précis pour les itérations futures.



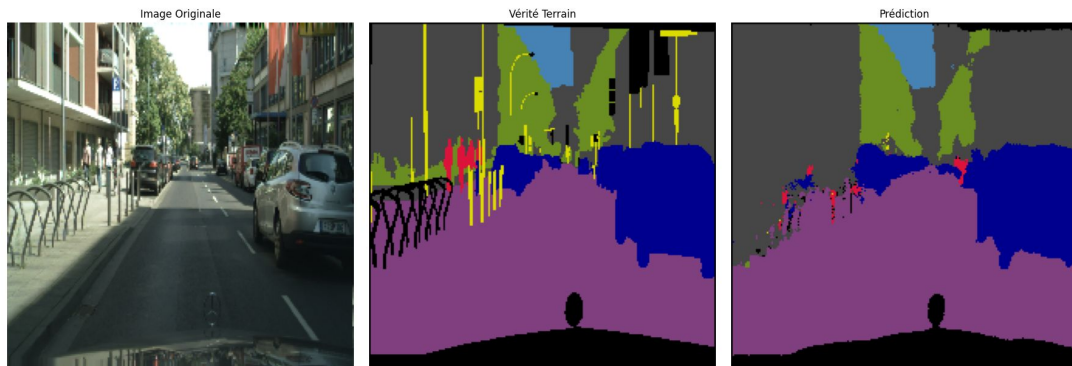
Exemple de prédiction sur une image de test

Forces du modèle









- Segmentation précise des éléments structurants (routes, bâtiments)
- Contours nets et classification cohérente des surfaces planes
- Gestion intelligente des zones ambiguës via la classe "void"

Défis identifiés

- Objets fins (panneaux, poteaux) sous-représentés
- Petits objets difficiles à détecter à résolution 224×224
- Variabilité sur la détection des humains selon contexte



Distribution des Classes Détectées

- | | |
|--|--|
| •  flat: | 19,536 pixels (38.9%) - Routes et circulation |
| •  construction: | 14,510 pixels (28.9%) - Bâtiments et infrastructures |
| •  nature: | 8,285 pixels (16.5%) - Végétation et terrain |
| •  void: | 5,459 pixels (10.9%) - Zones non classifiées |
| •  vehicle: | 1,630 pixels (3.2%) - Véhicules en circulation |
| •  human: | 594 pixels (1.2%) - Piétons et cyclistes |
| •  sky: | 98 pixels (0.2%) - Ciel visible |
| •  object: | 64 pixels (0.1%) - Signalisation urbaine |



API REST : Segmentation d'images



Technologies Utilisées

- **FastAPI** : Framework web asynchrone haute performance
- **TensorFlow 2.18 (CPU)** : Inférence du modèle (~250MB)
- **MLflow** : Chargement dynamique du modèle
- **Pillow** : Manipulation et traitement d'images
- **Gunicorn + Uvicorn** : Serveur ASGI pour la production



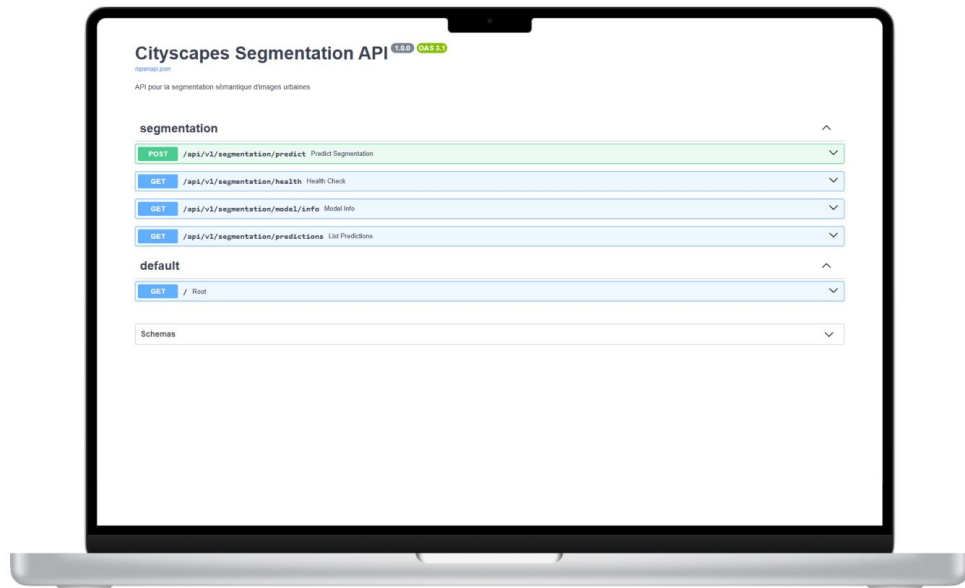
Fonctionnalités Clés

- **Chargement automatique du modèle** depuis MLflow
- **Génération d'artefacts** : Masques, overlays, statistiques
- **Validation robuste** : Formats, taille, qualité des images
- **Base64 encoding** pour transport images



Déploiement en production sur Railway

- **API déployée en production** sur [Railway](https://future-vision-api-production.up.railway.app/docs) avec URL publique sécurisée



<https://future-vision-api-production.up.railway.app/docs>





API REST : Endpoints disponibles



FastAPI



POST /api/v1/segmentation/predict

- **Endpoint principal de segmentation d'images**
- **Entrée** : Image (PNG, JPEG) via **multipart/form-data**
- **Sortie** : JSON avec masques, statistiques et visualisations
- **Artefacts générés** :
 - **Image avec overlay semi-transparent**
 - **Masque de segmentation** pixel par pixel
 - **Comparaison côte à côte** (original vs prédiction)
 - **Statistiques détaillées** par classe



GET /api/v1/segmentation/health

- Vérification état de santé de l'API



GET /api/v1/segmentation/model/info

- Informations détaillées sur le modèle



GET /docs

- Documentation interactive Swagger UI



Exemple de réponse JSON

```
{
  "class_statistics": [
    {"class_name": "flat", "pixel_count": 19536, "percentage": 38.9},
    {"class_name": "construction", "pixel_count": 14510, "percentage": 28.9},
    {"class_name": "nature", "pixel_count": 8285, "percentage": 16.5}
  ],
  "dominant_class": "flat",
  "dominant_class_percentage": 38.9,
  "image_size": [224, 224],
  "images": {
    "original": "data:image/png;base64,iVBORw0KGgoAAAA...",
    "prediction_mask": "data:image/png;base64,iVBORw0KGgoAAAA...",
    "overlay": "data:image/png;base64,iVBORw0KGgoAAAA...",
    "side_by_side": "data:image/png;base64,iVBORw0KGgoAAAA..."
  }
}
```



Interface utilisateur Next.js

NEXT.JS



Frontend Next.js

- **Next.js 15.3** : Framework React pour développement frontend
- **Bootstrap 5.3** : Design responsive et professionnel



Fonctionnalités principales

- **Prédiction interactive** : Feedback visuel de progression
- **Visualisations multiples** :
 - **Image avec overlay** de segmentation semi-transparent
 - **Comparaison côte à côte** (originale vs prédiction)
 - **Masque de segmentation isolé**
- **Statistiques détaillées** : Distribution des classes, classe dominante



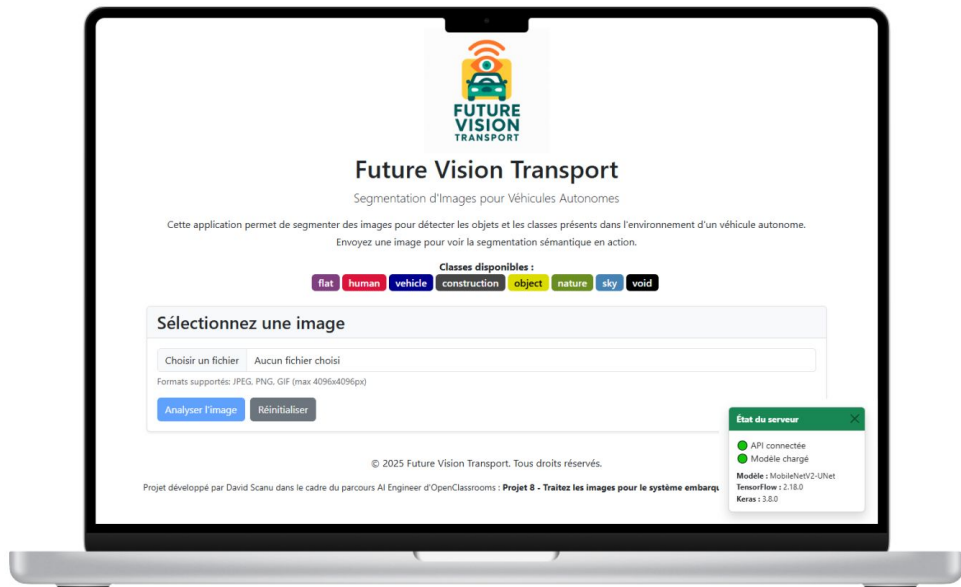
Intégration avec l'API

- **Fetch API** : Communication avec l'API FastAPI
- **Base64** : Gestion des images encodées
- **Statut de connexion** : Affichage en temps réel



Déploiement Vercel

- **Intégration continue** : déploiement automatique depuis Git
- **URL de production**



<https://oc-ai-engineer-p08-images-systeme-v.vercel.app/>



Démonstration de l'API et de l'interface utilisateur

future-vision-api | Railway

Cityscapes Segmentation API - cc-ai-engineer-p08-images-sy - Future Vision Transport - S

railway.com/project/75347ce1-1669-4dc8-a11d-ebff81b9d535/service/47b78bb2-d8d0-4274-a2f6-6ec347747a93?environmentId=a91ed4...

Applications Google OpenClassrooms Social Mes sites Piano Sport Basse Guitare Drums Audio Arthur ISEN AI Tools Learning ML/AI Datasets Python Tools Tous les favoris

future-vision-api / production Architecture Observability Logs Settings 30 days or \$4.98 left Share

future-vision-api
future-vision-api-production...

33 minutes ago via CLI

future-vision-api

Deployments Variables Metrics Settings

future-vision-api-production.up.railway.app us-west2 1 Replica

ACTIVE railway up 33 minutes ago via CLI View logs
Deployment successful

HISTORY

REMOVED railway up 43 minutes ago via CLI

REMOVED railway up 88 minutes ago via CLI

REMOVED railway up 2 hours ago via CLI

REMOVED railway up 2 hours ago via CLI



Pistes d'optimisation



Améliorations Modèle

- **Entraînement progressif** : Phase 2 avec encoder dégelé + learning rate réduit
- **Augmentation de données avancée** : Simulation météo (pluie, brouillard, nuit)
- **Attention mechanisms** : Amélioration détection petits objets (panneaux, poteaux)



Optimisations Architecture

- **Ensembling de modèles** : Combinaison de prédictions pour robustesse
- **Optimisation embarquée** : Quantization, pruning pour réduction latence
- **Pipeline temps réel** : Traitement vidéo frame par frame



Améliorations Système

- **Authentification API** : Sécurisation des endpoints de production
- **Monitoring avancé** : Métriques de performance et alertes
- **Batch processing** : Traitement de lots d'images



Évaluation Continue

- **A/B Testing** : Comparaison versions modèle en production
- **Feedback loop** : Amélioration continue basée sur des retours des utilisateurs
- **Métriques métier** : KPI spécifiques navigation autonome



Prochaine Étape

- **Intégration avec systèmes complémentaires (radar, lidar)** pour fusion multi-sensorielle



Conclusion et perspectives



Objectifs Atteints

- **Modèle performant** : Mean IoU de 63,25% sur 8 catégories urbaines
- **Architecture industrialisable** : Pipeline MLOps avec MLflow
- **API production** : FastAPI déployée et documentée
- **Interface utilisateur** : Application Next.js déployée
- **Documentation complète** : Note technique et présentation



Impact pour le Système Embarqué

- Module de segmentation **prêt pour intégration chaîne complète**
- **Robustesse sur éléments critiques navigation** : routes, véhicules, construction
- **Scalabilité** : Architecture cloud pour déploiement à grande échelle



Contribution Technique

- **Architecture hybride MobileNetV2-UNet**
- **Pipeline reproductible** avec versioning et traçabilité
- **Base solide pour développements futurs** et améliorations continues



Perspectives d'Évolution

- **Intégration multi-sensorielle** : Fusion avec radar/lidar
- **Optimisations embarquées** : Quantization pour performance
- **Déploiement véhicule** : Tests en conditions réelles



Résultat

Solution complète et industrialisable répondant :

- aux contraintes de Franck (traitement des images)
- aux besoins de Laura (système de décision)



Récapitulatif des livrables



Dépôt GitHub

- [Code source complet disponible sur GitHub](#)



Notebook de développement

- **Pipeline complet** : Exploration → Modélisation → Évaluation
- **Expérimentations MLflow** : Traçabilité et reproductibilité
- **Notebook** : [Disponible sur Colab](#)



Modèle de segmentation

- **Architecture** : MobileNetV2-UNet hybride
- **Performance** : 63.25% Mean IoU
- **Format** : Keras 3.x compatible production
- **Registry** : Versioning MLflow pour déploiement



API FastAPI de production

- **Endpoints REST** : Prédiction + utilitaires de monitoring
- **Génération d'artefacts** : Masques, overlays, statistiques
- **Documentation** : Swagger UI intégrée
- **Déployé sur Railway** : [URL publique accessible](#)



Interface Next.js

- **Application web responsive** : Interface utilisateur intuitive
- **Fonctionnalités** : Upload + visualisation résultats
- **Intégration API** : Communication temps réel avec backend
- **Visualisations multiples** : Comparaisons et statistiques
- **Déployée sur Vercel** : [URL publique accessible](#)



Documentation complète

- [Rapport technique](#)
- [Article complet disponible sur dev.to](#)
- [Support de présentation](#) pour l'équipe de Laura

Merci 🙏

Avez-vous des questions ?



Annexes

A propos

Projet développé par **David Scanu** dans le cadre du parcours AI Engineer d'[OpenClassrooms](#) :

Projet 8 - Traitez les images pour le système embarqué d'une voiture autonome.

- [David Scanu - Développeur en intelligence artificielle - Carrefour | LinkedIn](#)
- [Formation AI Engineer - OpenClassrooms](#)

Documentation

- [U-Net Architecture For Image Segmentation | DigitalOcean](#)
- [MobileNet, MobileNetV2, and MobileNetV3](#)
- [U-NET : le réseau de neurones de Computer Vision](#)
- [MLflow: A Tool for Managing the Machine Learning Lifecycle](#)
- [FastAPI](#)
- [Next.js Documentation](#)

