

# Initiation YOLO

## Analyse de photographies de paysages urbains via la vision par ordinateur

Décrypter la Planification Territoriale. Échelles, Méthodes et Enjeux

Mardi 28 janvier 2025

Nicolas Lépy, Thomas Buhler

# L'intelligence artificielle dans la vision par ordinateur



# Initiation YOLO

openai/whisper



Reconnaissance  
automatique de  
la parole

Vision par ordinateur

Data Science

Robotique

INTELLIGENCE ARTIFICIELLE

IA générative

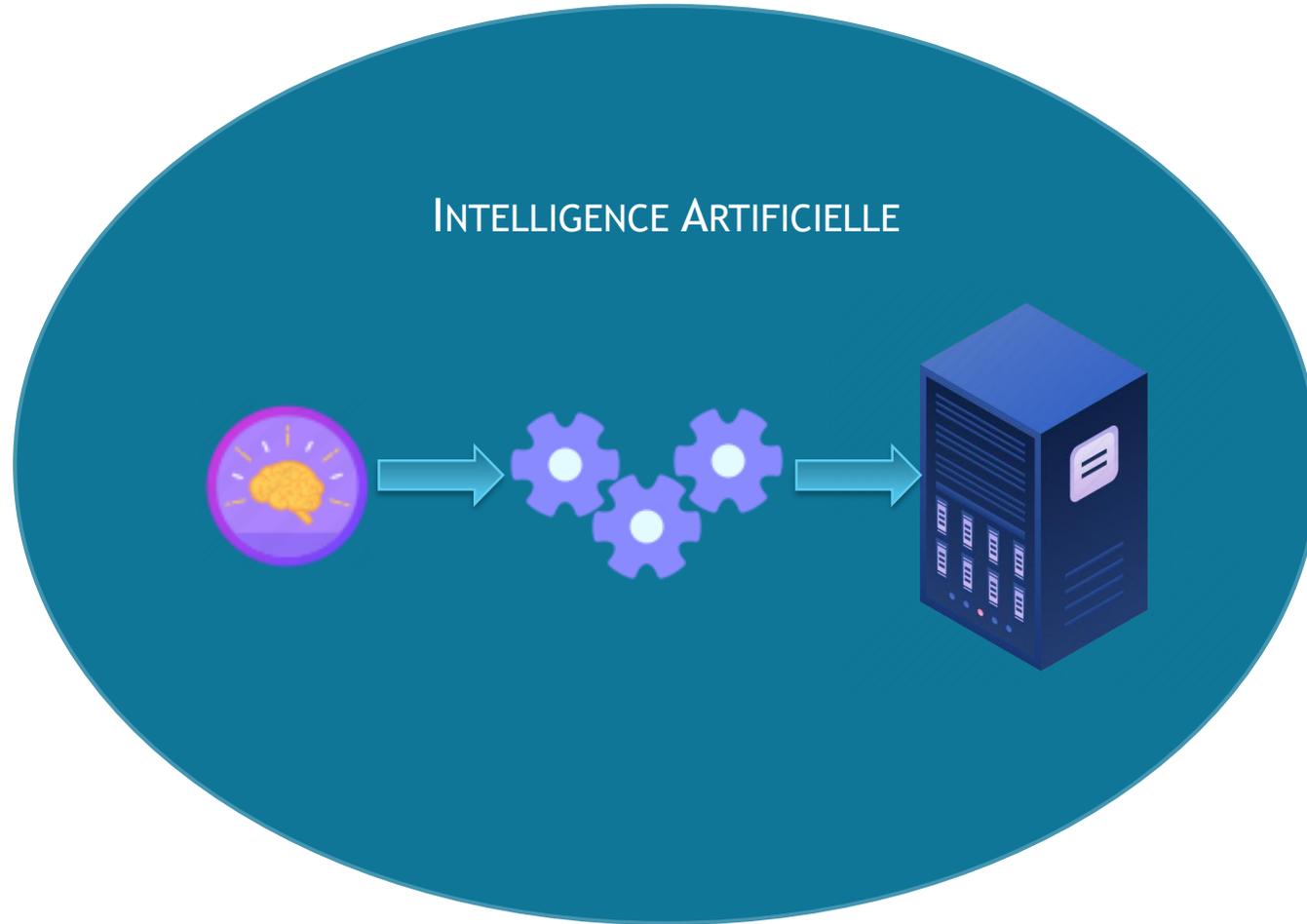
Logique  
floue

Systèmes  
experts

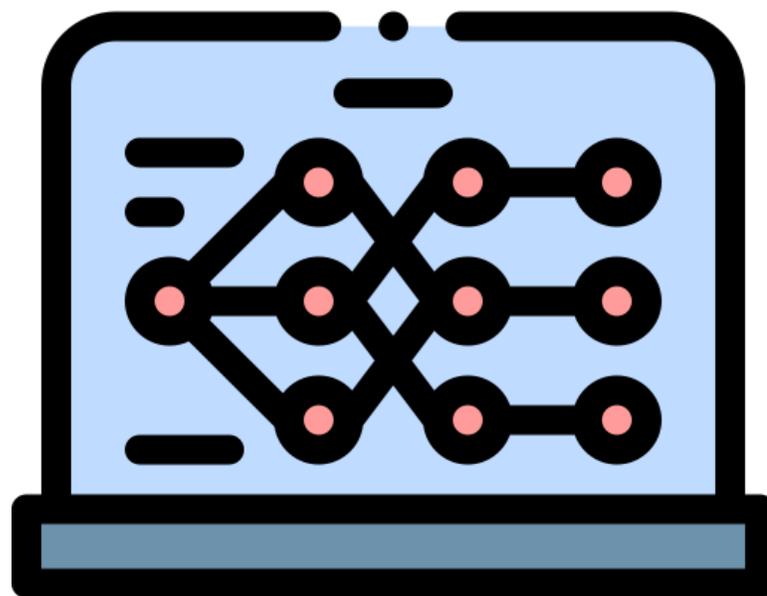
Systèmes  
multi-  
agents



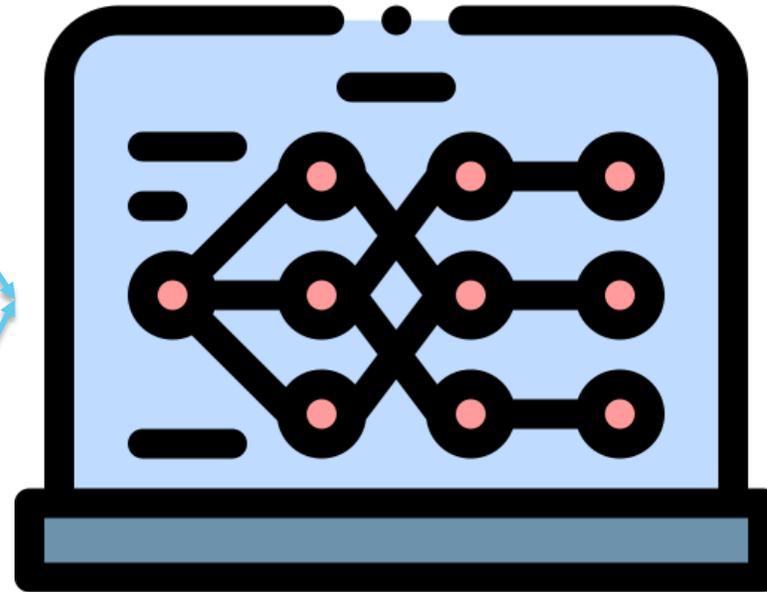
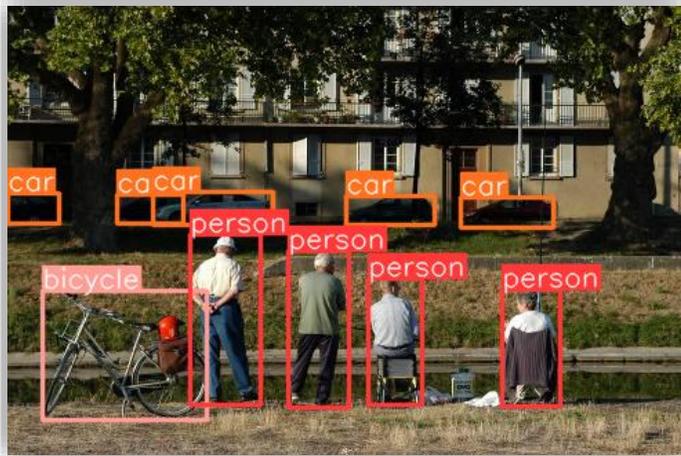
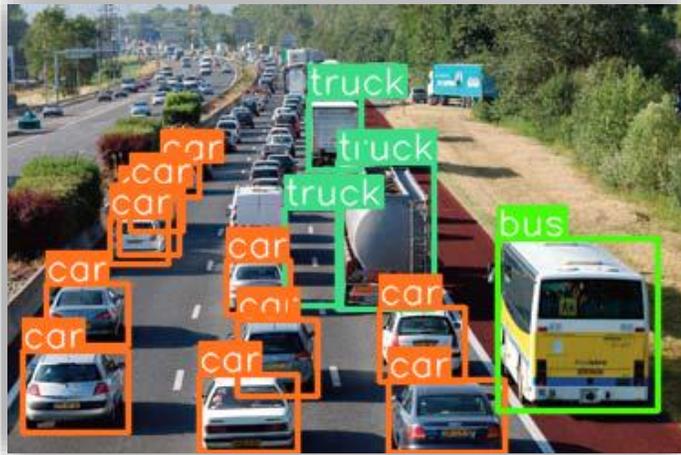
# Démystifier l'IA



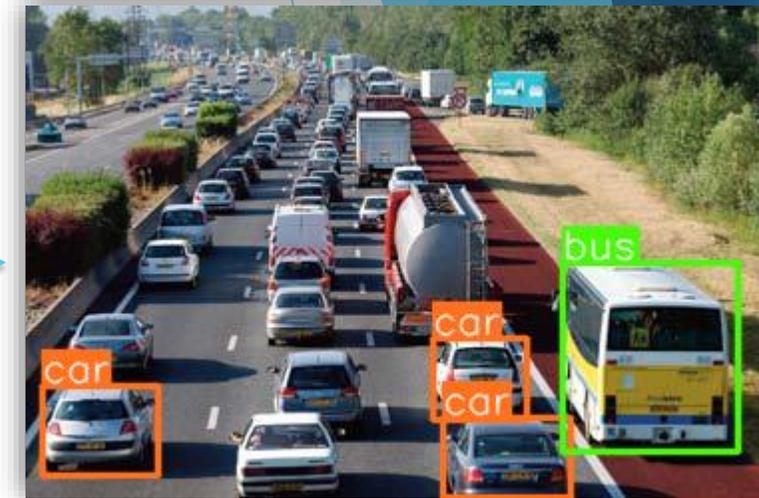
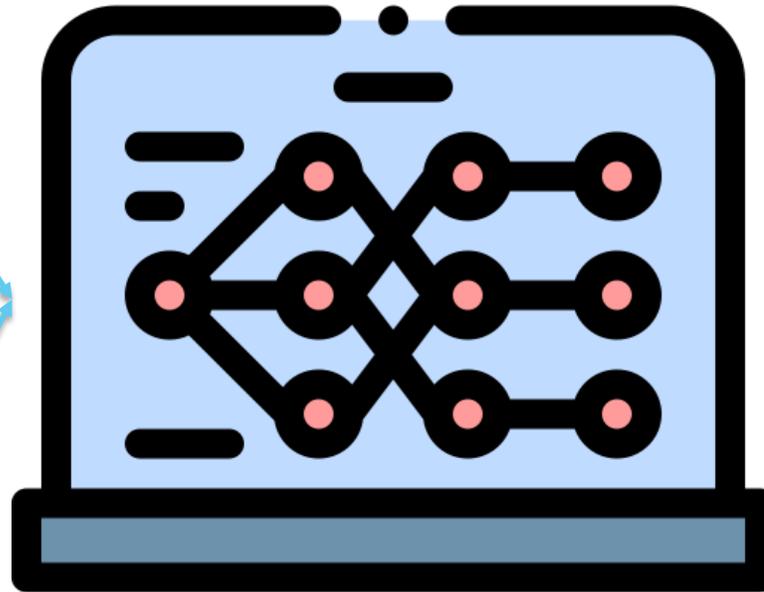
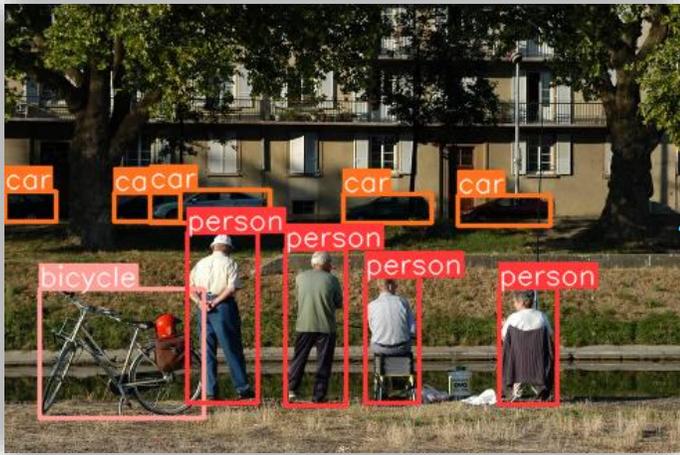
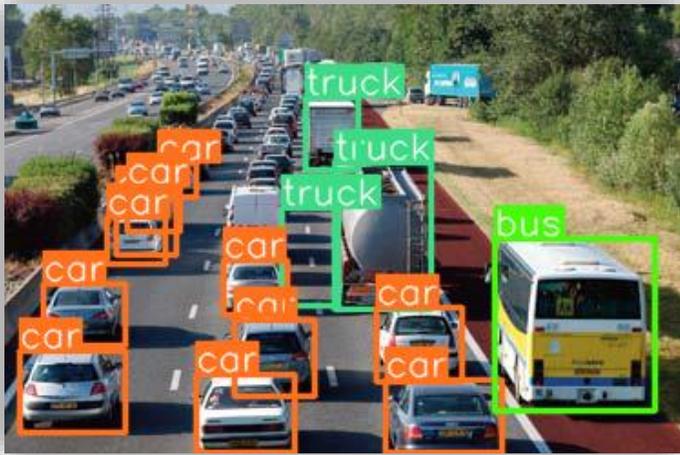
# Démystifier l'IA



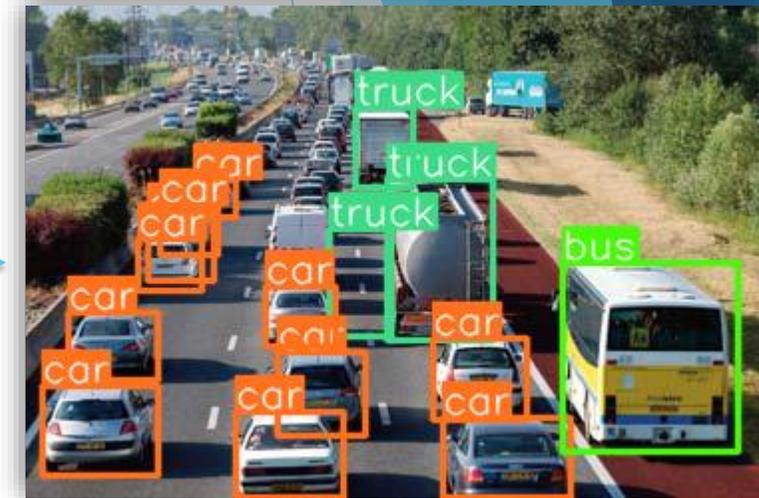
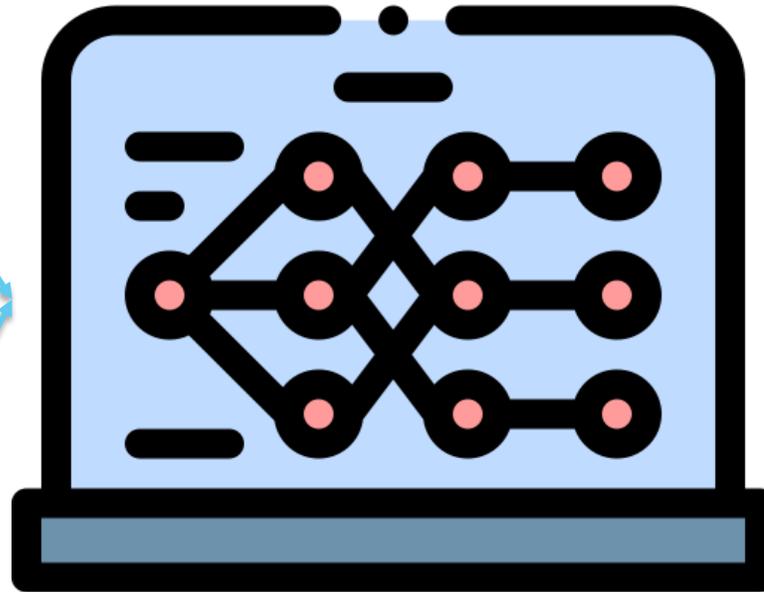
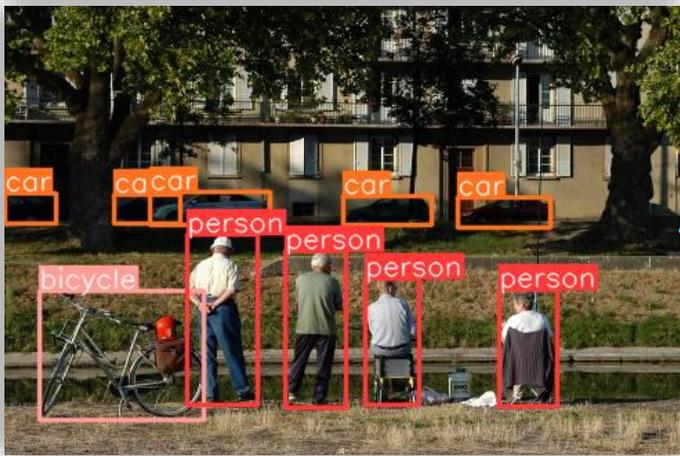
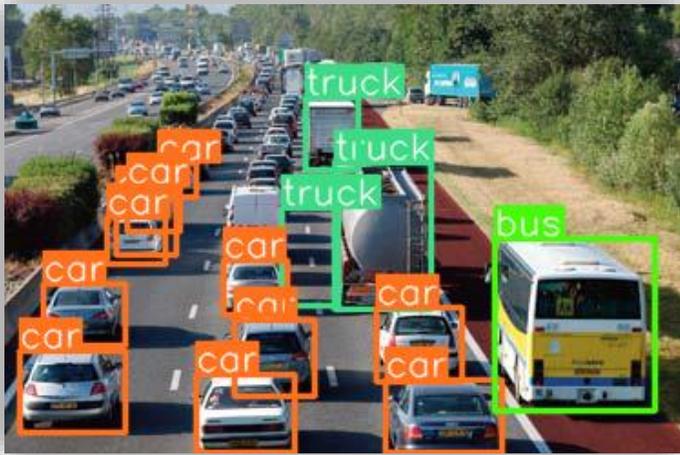
# Démystifier l'IA



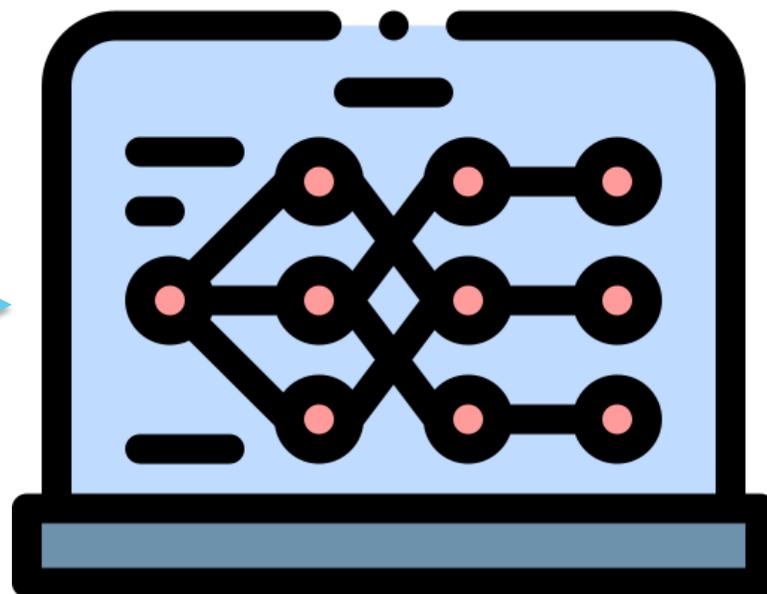
# Démystifier l'IA



# Démystifier l'IA

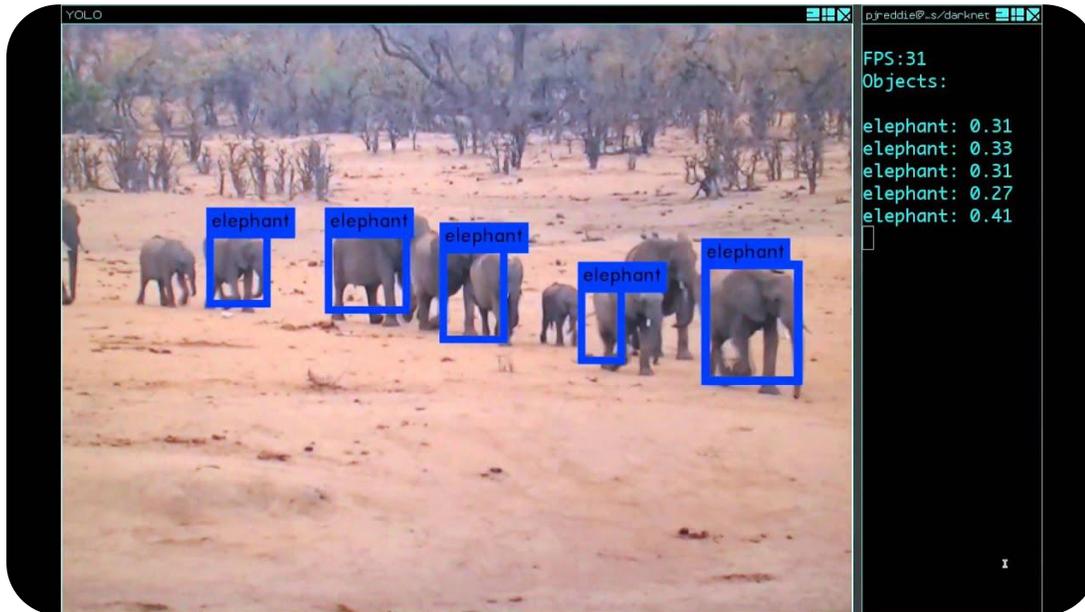


# Démystifier l'IA

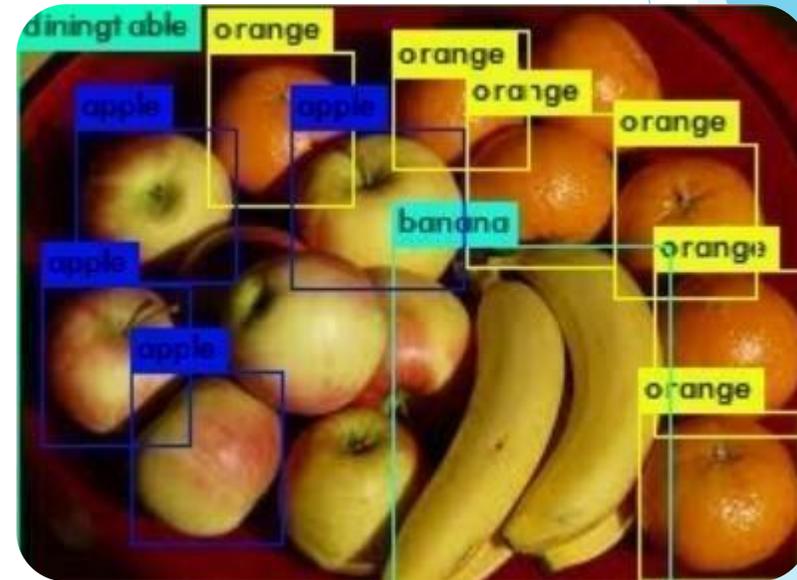


# YOLO (You Only Look Once)

- Classification et localisation d'objets



pjreddie.com



A Review of Fruits Image Analysis Using Computer Vision and Deep Learning Techniques

# YOLO (You Only Look Once)



- Vélo
- Voiture
- Moto
- Avion
- Bus
- Train
- Camion
- Bateau
- Feu de signalisation
- Borne d'incendie
- Panneau stop
- Parcmètre

# YOLO (You Only Look Once)

► Qui a développé YOLO



Joseph Redmon  
Université de Washington  
2015



YOLOv4  
2020

Licence Apache 2.0



YOLOv5  
2020



Licence AGPL-3.0  
Licence commerciale

Licence GPL-3.0

YOLOv7

Chien-Yao Wang, Alexey Bochkovskiy et al  
2022

YOLOv11  
2024

# Limites de YOLO

- ▶ Quelques erreurs de classifications
- ▶ Petits objets ne sont pas détectés
- ▶ Des détections très générales
  - ▶ Pour les personnes détectées, pas d'information sur le sexe, l'âge...

# Semantic Segmentation (semseg) CSAIL, MIT (2018)

- ▶ Segmentation d'images
  - ▶ Attribuer à chaque pixel de l'image une catégorie



[PDU Besançon 2015 p.30]

Semantic Understanding of Scenes through ADE20K Dataset. B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso and A. Torralba. International Journal on Computer Vision (IJCV), 2018.

# Semantic Segmentation (semseg)



ANTHROPISÉ

VÉGÉTATION

POINT D'EAU

car (87%)

car (87%)

# Comment accéder à ces modèles

- ▶ Le plus souvent, un modèle d'IA est un fichier avec une extension particulière



- ▶ On peut y accéder :

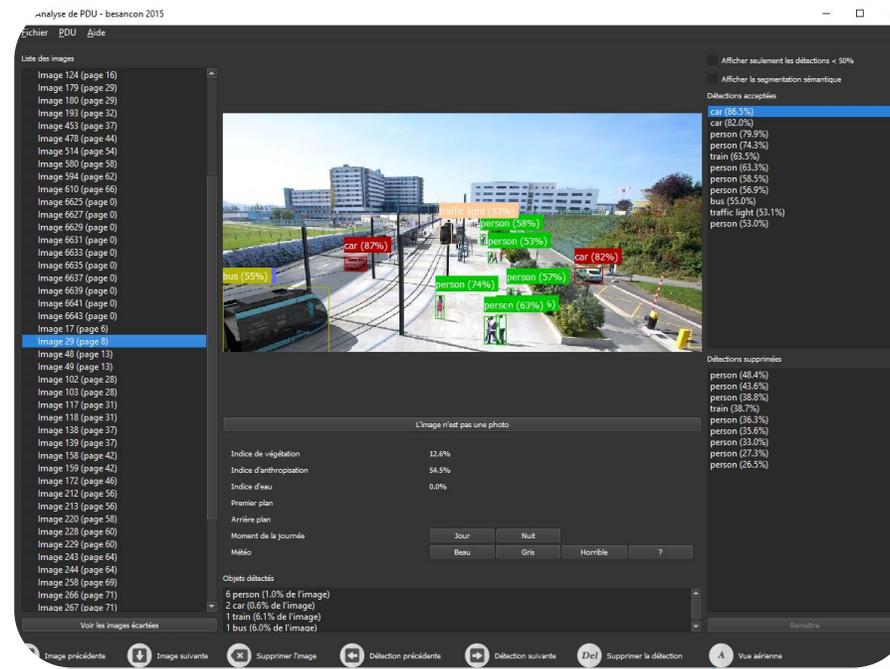
## Par un langage de script

Langage Python

```
from ultralytics import YOLO

model = YOLO(model="yolo11l.pt")
results = model("images\\image_1.png")
results[0].save(filename="export_1.jpg")
print(results[0].pandas())
```

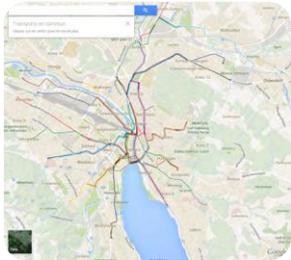
## Par une interface graphique



# Reconnaissance d'objets dans les PDU

## ► Développement d'une interface graphique

- Faciliter l'utilisation des modèles
- Extraction automatique des images et détection des photographies



VS



## ► Métriques (végétation, ...)



Végétation



Anthropisé

Analyse de PDU - besancon 2015

Erchier PDU Aide

Liste des images

- Image 124 (page 16)
- Image 179 (page 29)
- Image 180 (page 29)
- Image 193 (page 32)
- Image 453 (page 37)
- Image 478 (page 44)
- Image 514 (page 54)
- Image 580 (page 58)
- Image 594 (page 62)
- Image 610 (page 66)
- Image 6625 (page 0)
- Image 6627 (page 0)
- Image 6629 (page 0)
- Image 6631 (page 0)
- Image 6633 (page 0)
- Image 6635 (page 0)
- Image 6637 (page 0)
- Image 6639 (page 0)
- Image 6641 (page 0)
- Image 6643 (page 0)
- Image 17 (page 6)
- Image 29 (page 8)**
- Image 48 (page 13)
- Image 49 (page 13)
- Image 102 (page 28)
- Image 103 (page 28)
- Image 117 (page 31)
- Image 118 (page 31)
- Image 138 (page 37)
- Image 139 (page 37)
- Image 158 (page 42)
- Image 159 (page 42)
- Image 172 (page 46)
- Image 212 (page 56)
- Image 213 (page 56)
- Image 220 (page 58)
- Image 228 (page 60)
- Image 229 (page 60)
- Image 243 (page 64)
- Image 244 (page 64)
- Image 258 (page 69)
- Image 266 (page 71)
- Image 267 (page 71)

Afficher seulement les détections < 50%

Afficher la segmentation sémantique

Détections acceptées

- car (86.5%)
- car (82.0%)
- person (79.9%)
- person (74.3%)
- train (63.5%)
- person (63.3%)
- person (58.5%)
- person (56.9%)
- bus (55.0%)
- traffic light (53.1%)
- person (53.0%)

Détections supprimées

- person (48.4%)
- person (43.6%)
- person (38.8%)
- train (38.7%)
- person (36.3%)
- person (35.6%)
- person (33.0%)
- person (27.3%)
- person (26.5%)

L'image n'est pas une photo

Indice de végétation: 12.6%

Indice d'anthropisation: 54.5%

Indice d'eau: 0.0%

Premier plan

Arrière plan

Moment de la journée: Jour, Nuit

Météo: Beau, Gris, Horrible, ?

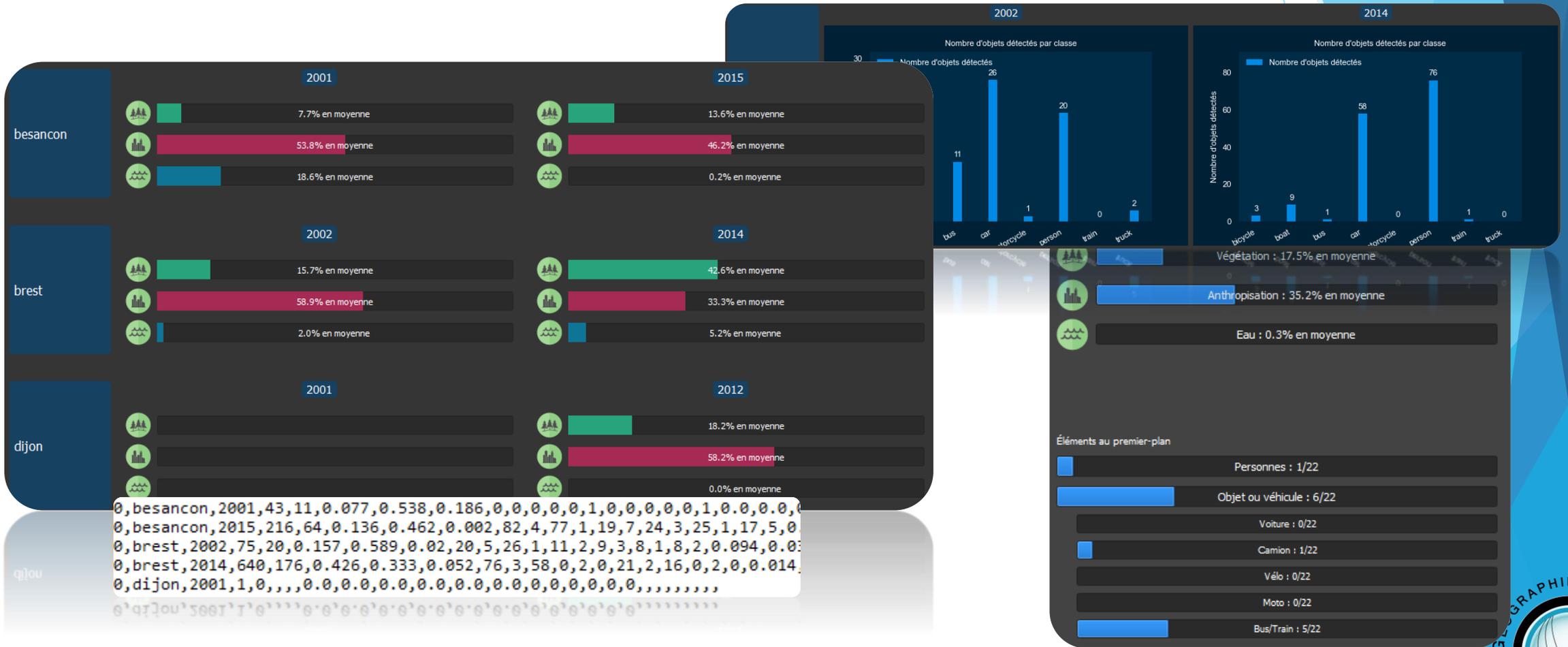
Objets détectés

- 6 person (1.0% de l'image)
- 2 car (0.6% de l'image)
- 1 train (6.1% de l'image)
- 1 bus (6.0% de l'image)

Voir les images écartées

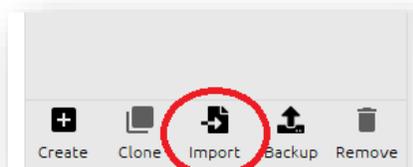
Image précédente, Image suivante, Supprimer l'image, Détection précédente, Détection suivante, Supprimer la détection, Vue aérienne

# Reconnaissance d'objets dans les PDU

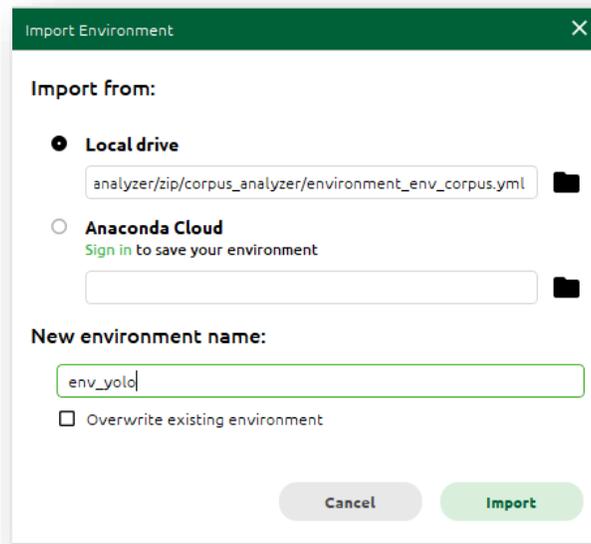


# Reconnaissance d'objets dans les PDU

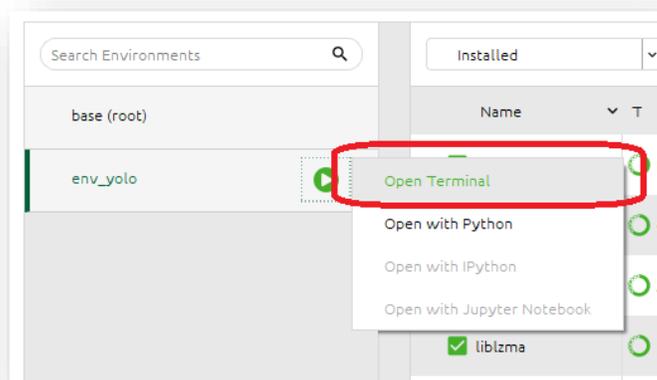
1.



2.



3.



4.

```
(env_yolo) C:\Users\nlepy>d:  
(env_yolo) D:\>cd path/to/main.py  
(env_yolo) D:\>python main.py
```

-> Uniquement si le logiciel est sur le disque D:

-> Se positionner sur le dossier qui contient le logiciel

-> Exécuter le logiciel

19

# Accéder à YOLO par un langage de script

- ▶ Les modèles d'IA sont généralement disponibles sous forme de fichiers
- ▶ Des outils sont développés pour utiliser ces modèles sur n'importe quelle image
- ▶ Le plus souvent en langage **Python**, mais possible aussi sur **R** ...

# Accéder à YOLO par un langage de script

- ▶ Exécuter YOLO sur une image et exporter l'image du résultat

```
from ultralytics import YOLO

# == Exécuter YOLO sur une image et exporter l'image du résultat ==

model = YOLO(model="yolo11l.pt")           # Charger le fichier yolo11l.pt
results = model("images\\image_1.png")    # Utiliser YOLO sur une photo sur le disque
results[0].show()                          # Afficher le résultat à l'écran
results[0].save(filename="export_1.jpg")    # Exporter le résultat sur le disque
```

# Accéder à YOLO par un langage de script

- ▶ Exécuter YOLO sur une image et exporter le résultat au format texte

```
# == Exécuter YOLO sur une image et exporter le résultat au format texte ==  
  
model = YOLO(model="yolo11l.pt")           # Charger le fichier yolo11l.pt  
results = model("images\\image_1.png")    # Utiliser YOLO sur une photo sur le disque  
print(results[0].boxes)                   # Afficher le résultat sous forme textuelle à l'écran  
results[0].save_txt(txt_file="export_1.txt", save_conf=True) # Exporter la liste des objets détectés au format .TXT (plus brut)  
results[0].to_df().to_csv("export_1.csv")  # Exporter la liste des objets détectés au format .CSV (plus lisible)
```

# Accéder à YOLO par un langage de script

- ▶ Exécuter YOLO sur toutes les images d'un dossier et exporter le résultat au format texte

```
# == Exécuter YOLO sur toutes les images d'un dossier et exporter le résultat au format texte ==  
  
model = YOLO(model="yolo11l.pt")           # Charger le fichier yolo11l.pt  
results = model( source="images", conf=0.5) # On donne le chemin d'un dossier et non d'une image ; toutes les images du dossier sont traitées par YOLO.  
for result in results:                     # Boucle pour sauvegarder les résultats dans le fichier export.txt  
    result.save_txt(txt_file="export.txt", save_conf=True)
```