

UNIVERSITÉ DE NAMUR

IMAGE CLASSIFIER : DOCUMENTATION IA

---

# Rapport de projet

---



2<sup>e</sup> quadrimestre 2023-2024

GENTILE Donato  
MONCOMBLE Baptiste  
ALARCON Diego  
CAVRENNE Louis  
BARNICH Juliette  
DE BAETS Anaé

## Table des matières

<b>1</b>	<b>Documentation du client/webserver</b>	<b>2</b>
1.1	Utilisation du CLI . . . . .	2
1.1.1	Pré-requis . . . . .	2
1.1.2	Utilisation . . . . .	2
1.2	Utilisation du Webserver . . . . .	3
1.3	Etiquetage . . . . .	4
1.4	Entraînement . . . . .	6
1.5	Pré-requis . . . . .	10
1.5.1	Entraînement . . . . .	10
1.5.2	Utilisation . . . . .	10
1.6	Performances . . . . .	10
1.6.1	Minimale . . . . .	10
1.6.2	Recommandée . . . . .	10
1.7	Futur . . . . .	10

# 1 Documentation du client/webserver

## 1.1 Utilisation du CLI

### 1.1.1 Pré-requis

1. Télécharger la dernière release depuis [GitHub](#).
2. Dans le dossier racine du projet, installer les requirements des dossiers "IA" et "WebServer" avec les commandes suivantes :

```
pip install -r ./AI/requirements.txt
pip install -r ./AI/gpu-enable-requirements.txt
pip install -r ./WebServer/requirements.txt
```

### 1.1.2 Utilisation

On utilise les flags dans le CLI pour délimiter la multiplicité de zip et de tag.

Flags:

- -t : Sera utilisé pour les tags.
- -n : Sera utilisé pour le nom de batching.
- -z : Sera utilisé pour les noms de zips à extraire.

NB: Les flags peuvent être mis dans n'importe quel ordre.

Exemples d'utilisation du CLI:

- imageclassifier -n "Photos camp" -z zip1 -z zip2 -z zip3 -t tag1 -t tag2
- imageclassifier -z zip1 -z zip2 -z zip3 -t tag1 -t tag2 -t tag3
- imageclassifier -n "Photos camp" -z zip1 -z zip2 -z zip3
- imageclassifier -z zip1 -z zip2 -z zip3

Formats d'images acceptés :

- Windows bitmaps - \*.bmp, \*.dib
- JPEG files - \*.jpeg, \*.jpg, \*.jpe
- JPEG 2000 files - \*.jp2
- Portable Network Graphics - \*.png
- WebP - \*.webp
- AVIF - \*.avif
- Portable image format - \*.pbm, \*.pgm, \*.ppm \*.pxm, \*.pnm
- PFM files - \*.pfm
- Sun rasters - \*.sr, \*.ras
- TIFF files - \*.tiff, \*.tif
- OpenEXR Image files - \*.exr
- Radiance HDR - \*.hdr, \*.pic

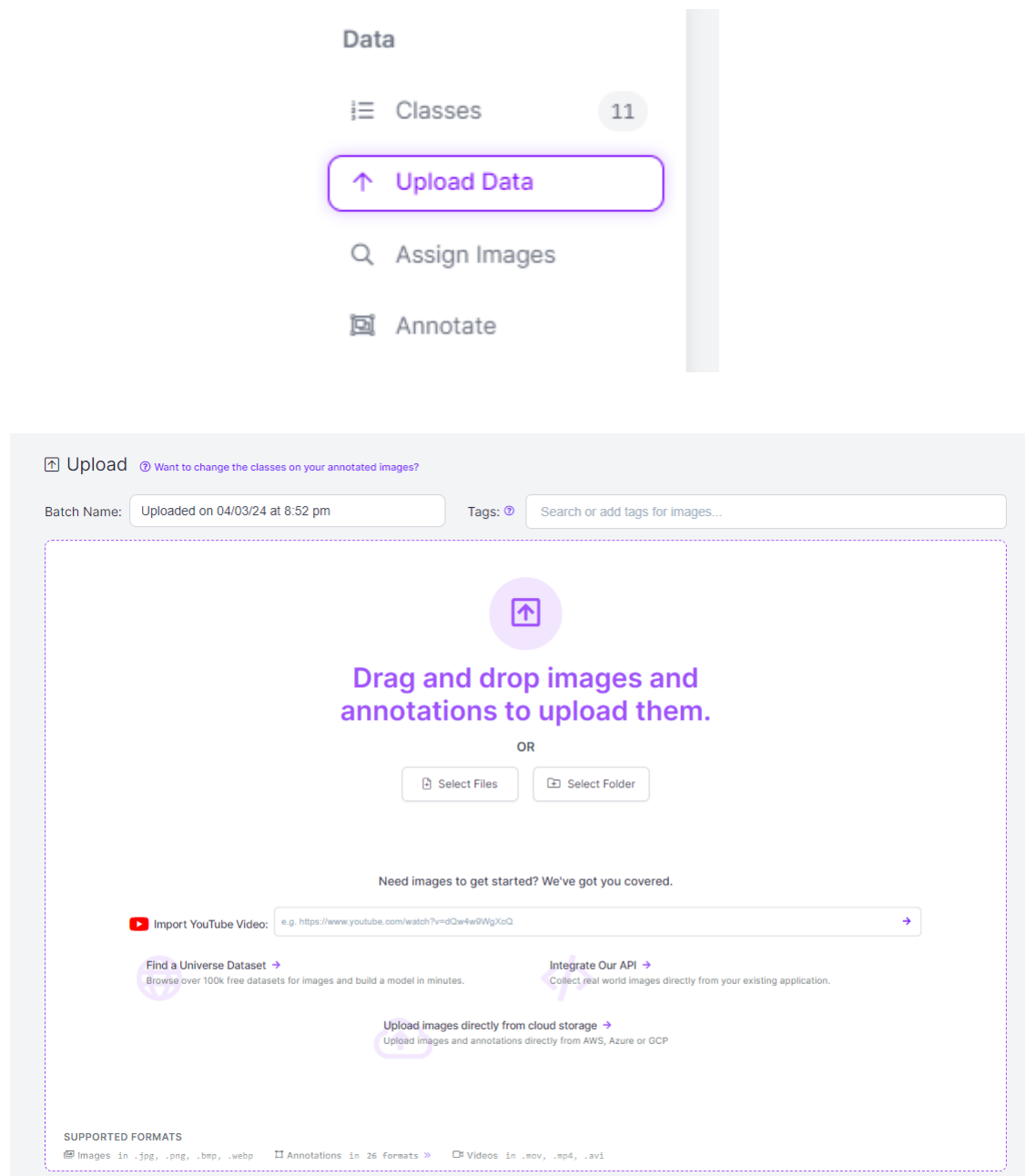
## 1.2 Utilisation du Webserver

Pour l'utilisation du WebServer, nous avons rendu cela beaucoup plus simple à l'aide des différents scripts de démarrage. Il vous suffira d'ouvrir le script de démarrage correspondant à votre système d'exploitation. Pour windows vous avez donc le .bat et pour linux le .sh. Une fois lancé, le webserver sera directement lancé.

En alternative, vous pouvez utiliser le docker compose pour créer un conteneur complet avec le webserver, l'ia et le GUI.

### 1.3 Etiquetage

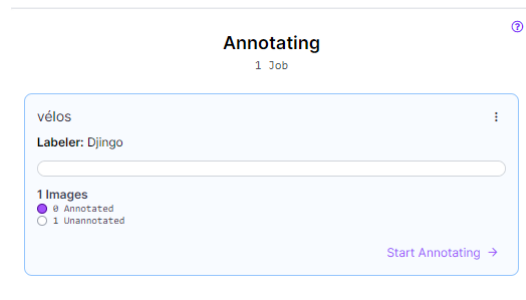
L'outil conseillé pour l'étiquetage est Roboflow. Pour étiqueter de nouvelles images avec Roboflow, voici comment procéder : Commencez par télécharger vos images dans l'un des formats acceptés, tels que JPG, PNG, WEBP ou BMP dans l'onglet "Upload Data".



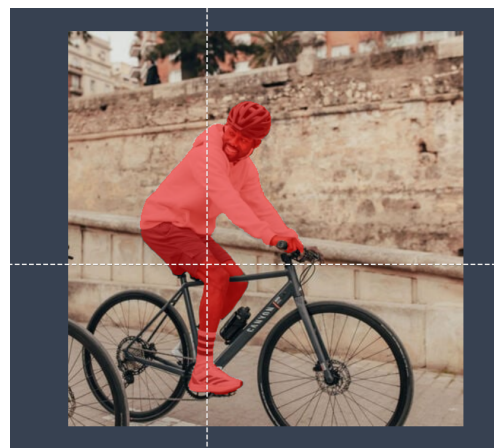
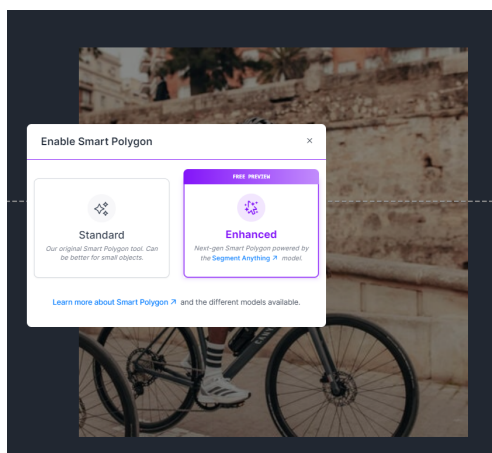
Roboflow offre différentes méthodes d'annotation. La fonctionnalité Label Assist permet une annotation automatique en utilisant soit une version précédente du modèle, soit des modèles disponibles publiquement sur la plateforme. Cette fonctionnalité détecte automatiquement les objets, les personnes, ou autres éléments présents dans les images et applique les étiquettes correspondantes.

En cas de dysfonctionnement de l'outil automatique, il est toujours possible d'annoter manuellement avec l'outil polygonal pour annoter les images. C'est d'ailleurs la méthode que nous recommandons.

Une fois vos images téléchargées, vous devez accéder à l'onglet "Annotate" et cliquer sur le bouton "Commencer annotation".



Cela vous mènera à un menu où vous pourrez commencer à étiqueter vos images. Dans ce menu, vous devrez choisir l'option "Smart Polygon", "Enhanced" et cliquez sur l'objet que vous souhaitez étiqueter. Assurez-vous de sélectionner avec précision les contours de l'objet pour une annotation correcte et détaillée.

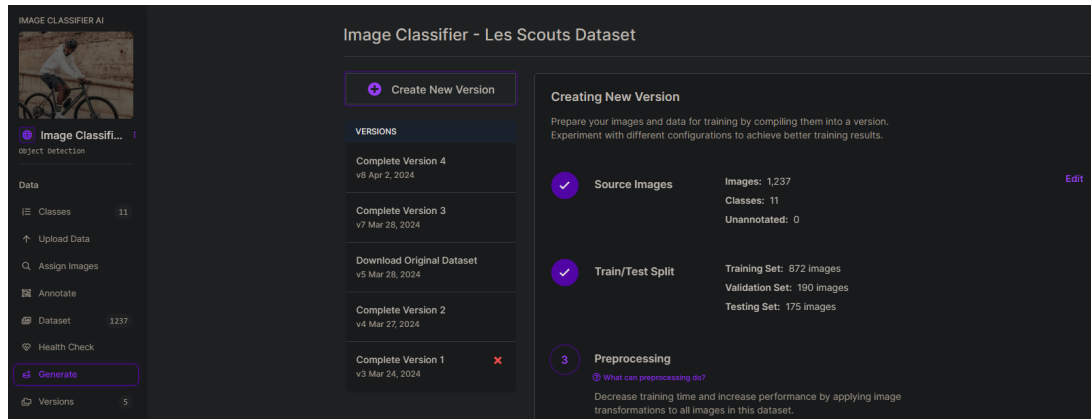


Veuillez sélectionner la classe appropriée chaque fois que vous effectuez une annotation.

Pour finaliser, intégrez les images traitées au dataset en ajustant si nécessaire les pourcentages d'entraînement, de validation et de test. Nous vous conseillons de conserver un ratio de 60-70% pour le train set, 15-20% pour le validation set et 15-20% pour le test set.

## 1.4 Entraînement

Tout d'abord, il va falloir que vous créiez le dataset à partir de votre lots d'images étiquetées. Pour cela, il vous suffit d'aller dans l'onglet "Generate" sur Roboflow et de créer une nouvelle version de votre dataset.



Comme vous pourrez le voir, il sera possible d'appliquer plusieurs transformations à vos images d'origines pour créer un dataset plus fourni. Voici les transformations que nous recommandons dans le cadre de ce type de dataset de détection d'objets:

### Preprocessing:

Auto-Orient: Applied

Resize: Stretch to 640x640

Filter Null: Require all images to contain annotations.

### Augmentation:

Flip: Horizontal, Vertical

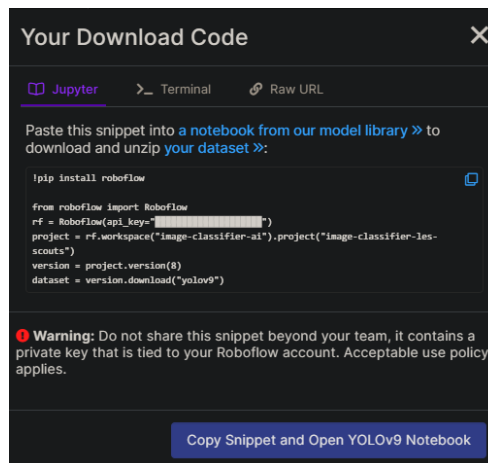
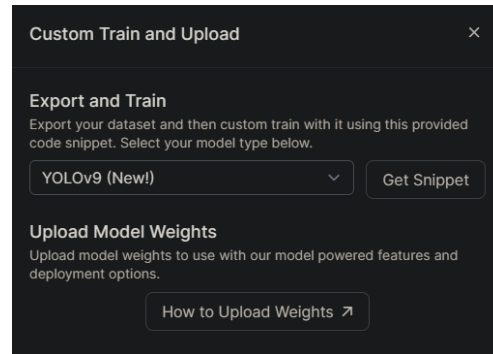
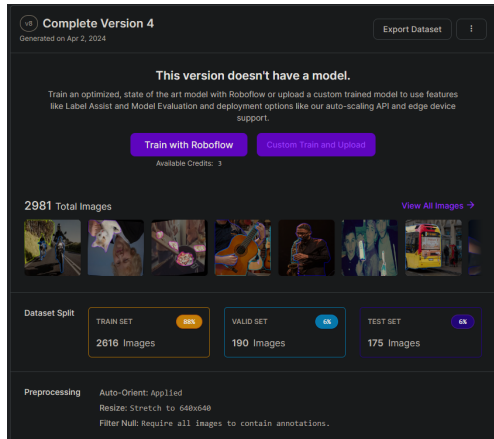
Crop: 0% Minimum Zoom, 20% Maximum Zoom

Rotation: Between  $-15^\circ$  and  $+15^\circ$

Shear:  $\pm 10^\circ$  Horizontal,  $\pm 10^\circ$  Vertical

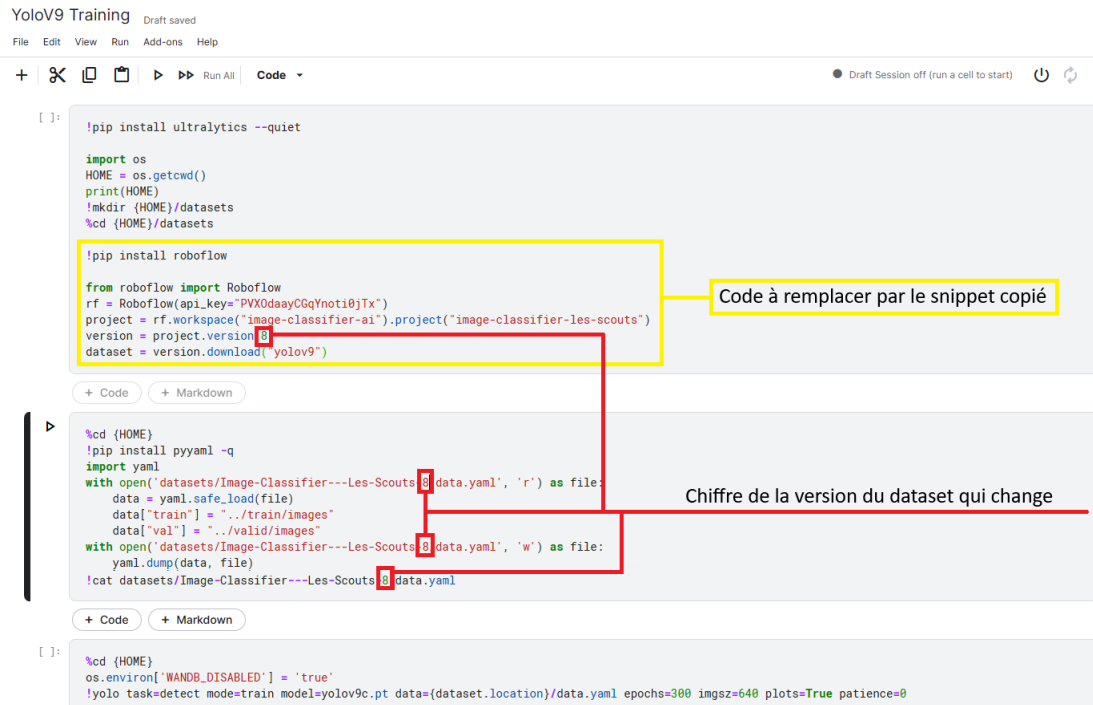
Une fois votre dataset créé, il faut entraîner le modèle avec celui-ci. Pour cela, nous avons créé un petit notebook python qui vous facilitera la tâche, vous pourrez le retrouver dans la section [documentation du repository GitHub](#). Le modèle que nous utiliserons ici est "YoloV9", un modèle à la pointe en terme de détection d'objets et qui fonctionne rapidement grâce à l'accélération matérielle GPU.

Tout d'abord rendez-vous dans l'onglet **Versions** sur Roboflow, là vous devriez y trouver votre nouveau dataset fraîchement créé. Ensuite, vous pourrez cliquer sur **Custom Train and Upload**, à ce moment là sélectionnez **YoloV9** et appuyez sur **Get Snippet**. Vous serez ensuite devant un code qui vous permettra d'exporter le dataset avec le Notebook que l'on vous a fourni.





Enfin, vous pouvez donc remplacer le code, que vous venez de copier sur Roboflow, dans le notebook comme le montre la figure suivante :



```
[ ]: !pip install ultralytics --quiet

import os
HOME = os.getcwd()
print(HOME)
!mkdir {HOME}/datasets
!cd {HOME}/datasets

!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="PVX0daayCGqYnoti0jTx")
project = rf.workspace("image-classifier-ai").project("image-classifier-les-scouts")
version = project.version[8]
dataset = version.download("yolov9")

+ Code + Markdown

!cd {HOME}
!pip install pyyaml -q
import yaml
with open('datasets/Image-Classifier---Les-Scouts[8].data.yaml', 'r') as file:
    data = yaml.safe_load(file)
    data["train"] = "../train/images"
    data["val"] = "../valid/images"
with open('datasets/Image-Classifier---Les-Scouts[8].data.yaml', 'w') as file:
    yaml.dump(data, file)
!cat datasets/Image-Classifier---Les-Scouts[8].data.yaml

+ Code + Markdown

[ ]: !cd {HOME}
os.environ['WANDB_DISABLED'] = 'true'
!yolo task=detect mode=train model=yolov9c.pt data={dataset.location}/data.yaml epochs=300 imgs=640 plots=True patience=0
```

FIGURE 1 – Code du Notebook Python sur Kaggle

Sur cette figure, vous pouvez donc voir en jaune le code à remplacer avec celui que vous avez copié sur Roboflow (Get Snippet). Et en rouge vous pouvez voir qu’une variable change à chaque itération de votre dataset. Dans le cas de l’image, on serait sur la 8ème itération de notre dataset.

Il vous suffira de lancer le notebook python avec votre environnement pour que le téléchargement et l’entraînement de votre dataset commence. Attention cependant, l’entraînement peut prendre plusieurs heures, voir plusieurs jours selon la puissance de la machine qui est utilisée pour l’entraînement du modèle.

Pour récupérer le modèle entraîné à la fin, il vous suffit d'aller dans l'arborescence des dossiers comme suit dans l'image, et de récupérer le fichier "best.pt" qui vous donnera le meilleur entraînement de la session.

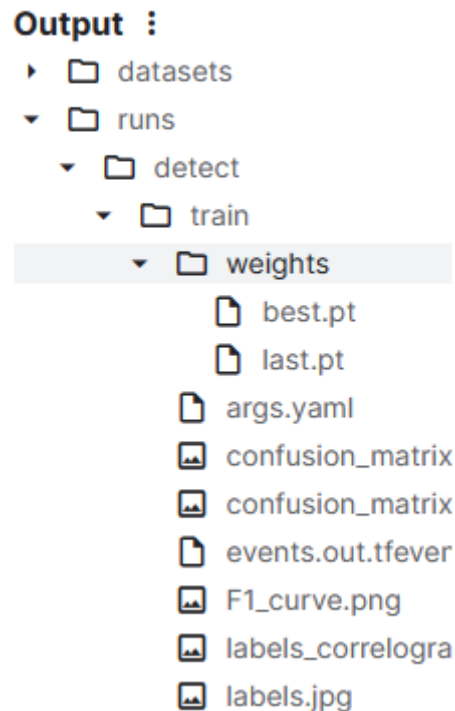


FIGURE 2 – Modèle à récupérer après l'entraînement sur Kaggle

Il vous suffira ensuite de placer ce fichier dans le dossier "models" du dossier "IA".

## 1.5 Pré-requis

### 1.5.1 Entraînement

En ce qui concerne les pré-requis pour faire tourner notre application nous vous conseillons ceci :

- Carte graphique NVIDIA avec beaucoup de Cuda Cores (3000+ cores)

### 1.5.2 Utilisation

Nous conseillons de posséder une machine puissante comme spécifiée dans la section 1.6, mais l'application fonctionne sur n'importe quelle machine en x86. La variance se situe dans le temps de traitement des images.

## 1.6 Performances

### 1.6.1 Minimale

CPU : Dual Core architecture x86

HDD : 15 GB + espace nécessaire pour stocker et traiter les images.

Software : Installer Python et NodeJs (ou utiliser Docker)

### 1.6.2 Recommandée

CPU : 8 Core architecture x86

GPU : RTX 2060

HDD : 50 GB

Software : Installer Python et NodeJs (ou utiliser Docker)

## 1.7 Futur

Concernant le futur de l'application, il est envisageable d'enrichir à la fois l'intelligence artificielle et l'interface utilisateur. Les détails concernant les améliorations de l'IA sont précisés dans les sections 1.3 et 1.4 du document. Quant à l'interface utilisateur, elle pourrait être adaptée selon les préférences des utilisateurs, tout en maintenant la cohérence actuelle de la communication entre l'affichage et le programme en ligne de commande (CLI). Veuillez consulter la documentation (1.1) pour savoir comment utiliser le CLI.