# REvision

1. Premier pipeline 3D CNN
2. Retour au 2D CNN, pipeline simplifié

# Téléchargements & INSTALLATION

Deux contenus sont téléchargeables :

* les datasets,
* les scripts Python.

Tous les scripts Python, et les modèles générés, sont hébergés par GitHub et gérés en configuration.

Les datasets ne sont pas hébergés par GitHub compte tenu de la quantité de données. Pour éviter la synchronisation automatique de ces datasets sur GitHub, ils doivent être stockés dans un dossier local dédié indépendant du repo GitHub local géré en configuration.

## Téléchargement des DATASET

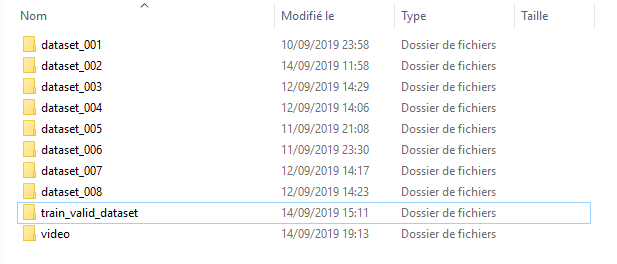
Les datasets sont disponibles en téléchargement via **OneDrive** :

<https://1drv.ms/u/s!AgPL1p8-kSaRg9RDySAEe-KTSa6sng>

Ces fichiers compressés contiennent les images, et les labelles (direction, gaz et position ligne) correspondant.

## INSTALLATION locale des datasets

1. **Créer un dossier local pour stocker les datasets** et les vidéos. Exemple : c:/datasets
2. **Décompresser les datasets téléchargés** dans ce dossier local.
3. **Créer les dossiers « train\_valid\_dataset » et « video »** dans ce même dossier local.
4. Le contenu du dossier local doit être le suivant :



## Scripts Python

Les scripts Python sont disponibles sur GitHub :

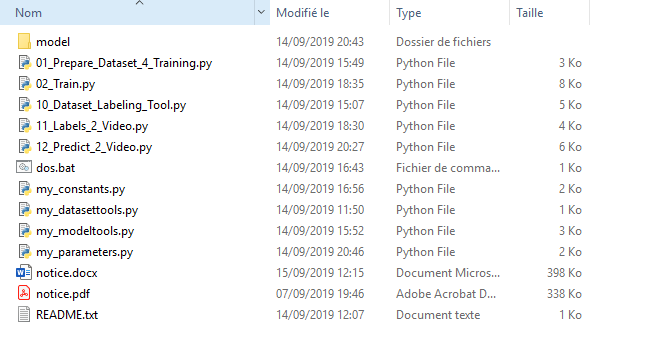
**GitHub/pat92fr/Carre92**/02 - Logiciel PC/04 - CNN Pipeline Line Detection/

<https://github.com/pat92fr/Carre92/tree/master/02%20-%20Logiciel%20PC/04%20-%20CNN%20Pipeline%20Line%20Detection>

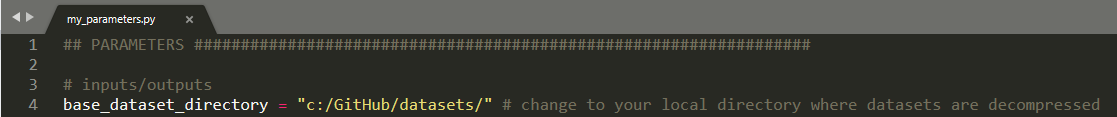
1. **Utiliser GitHub Desktop** pour télécharger et synchroniser les scripts en local.

Note : Sinon, copier tous les scripts dans un dossier local et créer le dossier « model ».

1. Le contenu du dossier local doit être le suivant :



1. **Modifier « my\_parameters.py »** pour indiquer le chemin vers le dossier local des datasets



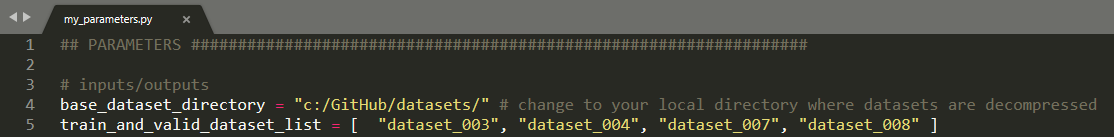
## Fichier de configuration

Le fichier « **my\_parameters.py** » contient toutes les données de configuration de toutes les étapes.

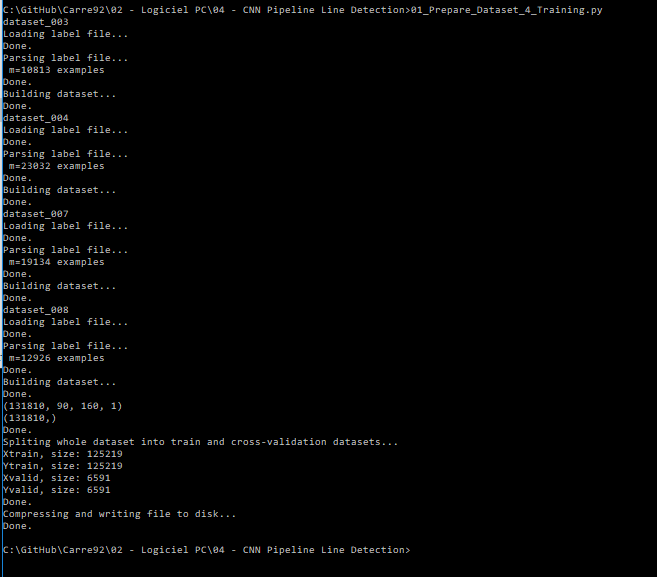
# Pipeline (2 étapes)

## Etape 1 : Synthèse du DATASET d’apprentissage

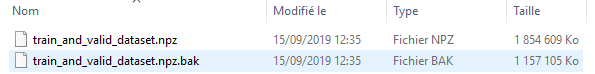
1. Sélectionner les datasets à utiliser pour l’apprentissage, en modifiant la variable **« train\_valid\_dataset\_list »** du fichier de configuration « **my\_parameters.py** » :



1. Lancer **« 01\_Prepare\_Dataset\_4\_Training.py »** pour générer le dataset d’apprentissage.

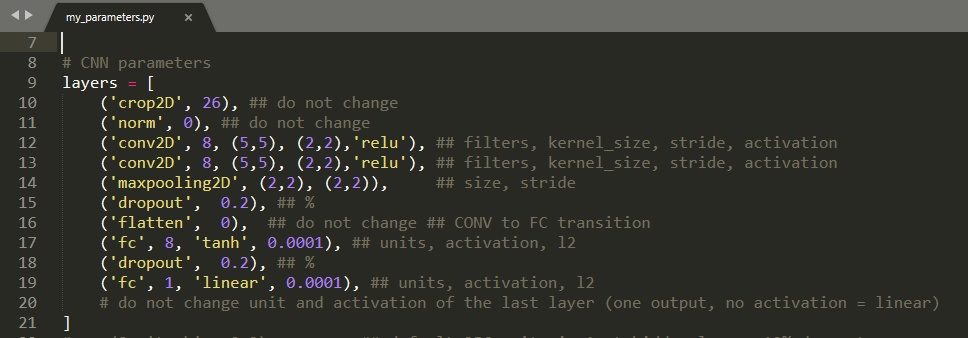


1. Le fichier généré est stocké localement dans le dossier « **train\_valid\_dataset** », du dossier local où se trouve tous les datasets. Note : le précédent dataset d’apprentissage est automatiquement sauvegardé (.bak).



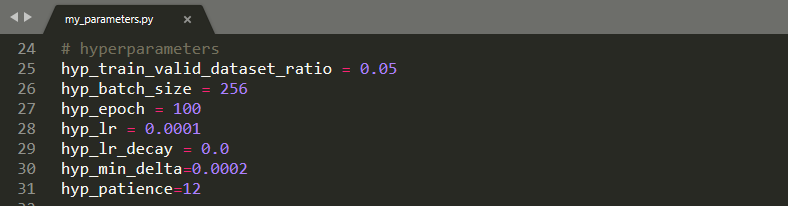
## ETAPE 2 : Apprentissage DU CNN

1. **Définir l’architecture du CNN**, en modifiant la variable **« layers »** du fichier de configuration « **my\_parameters.py** » :



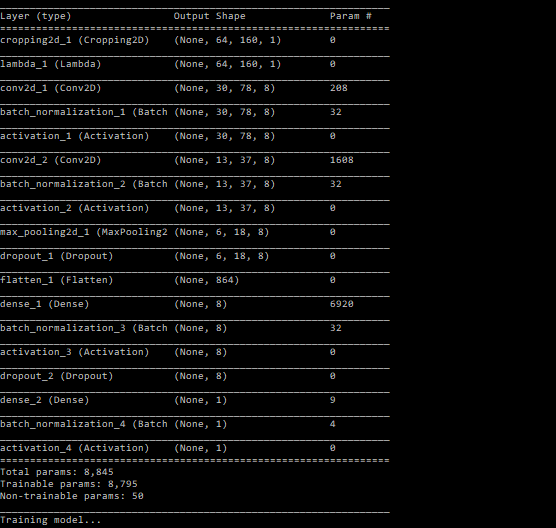
* La couche **crop2D** supprime les 26 premières lignes de l’image (160x90) de telle manière que le CNN traite une image tronquée (160x64).
* Le couche **norm** normalise les valeurs RGB de l’image dans la plage [0.0 , 1.0]
* Une couche de filtres de convolution **conv2D** prend en argument le nombre de filtres, la taille des filtres (h,w), le déplacement des filtres (h,w) et la fonction d’activation.
* La couche **maxpooling** prend en argument la taille (h,w), et le déplacement (h,w),
* La couche **dropout** prend en argument le % de suppression.
* La couche de neurones **fc** prend en argument le nombre d’unités, la fonction d’activation et lambda (régularisation L2).

1. **Ajuster les hyper paramètres d’apprentissage** **du CNN**, en modifiant la variable **« hyp… »** du fichier de configuration « **my\_parameters.py** » :

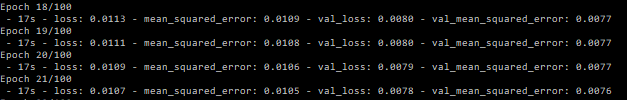


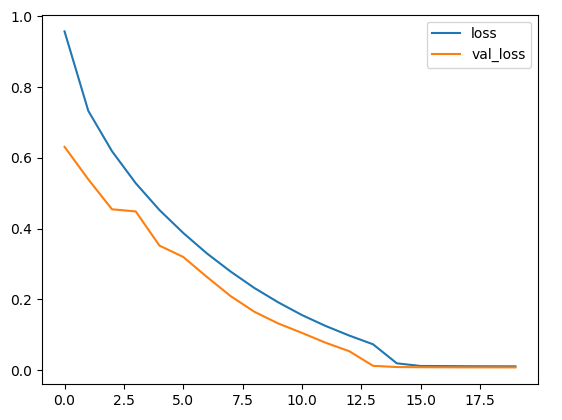
1. Lancer **« 02\_Train.py »** pour lancer l’apprentissage.

L’architecture modèle est affichée au début de l’apprentissage :

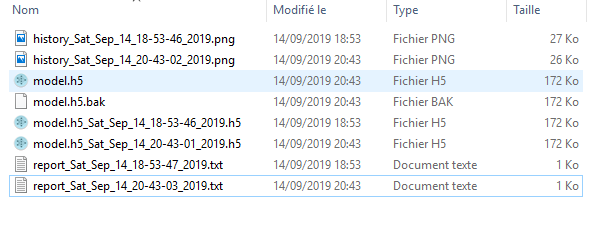


Les métriques sont affichées au fil de l’apprentissage (sortie standard et graphique) :





1. A la fin de l’apprentissage, le modèle généré est stocké localement dans le dossier « **model** » dans un fichier « .h5 » daté et dans un fichier générique « model.h5 ». Note : le précédent modèle est automatiquement sauvegardé (.bak).



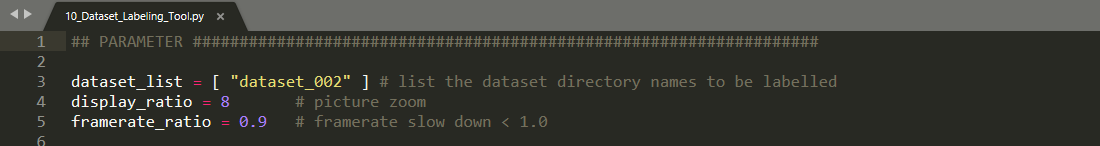
1. Le graphique des métriques est stocké localement dans le dossier « model » dans un fichier daté « .png ».
2. Un rapport d’apprentissage est stocké localement dans le dossier « model » dans un fichier daté « .txt ». Ce rapport rappelle succinctement l’architecture du CNN et les métriques finales pour le dataset de validation (Loss et MSE).

# Outils

## Outil de lebalisation d’un dataset SOURCE

Le script Python de labélisation est « 10\_Dataset\_Labeling\_Tool.py ». Il génère un fichier « label.txt » dans le dossier du dataset sélectionné.

1. Sélectionner le dataset à labéliser, en modifiant la variable **« dataset\_list »** du script.

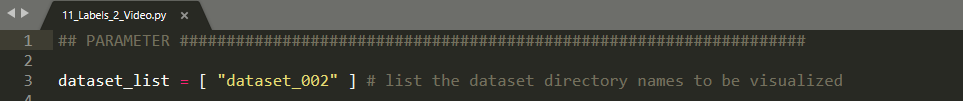


1. Lancer le script et labéliser à la souris.

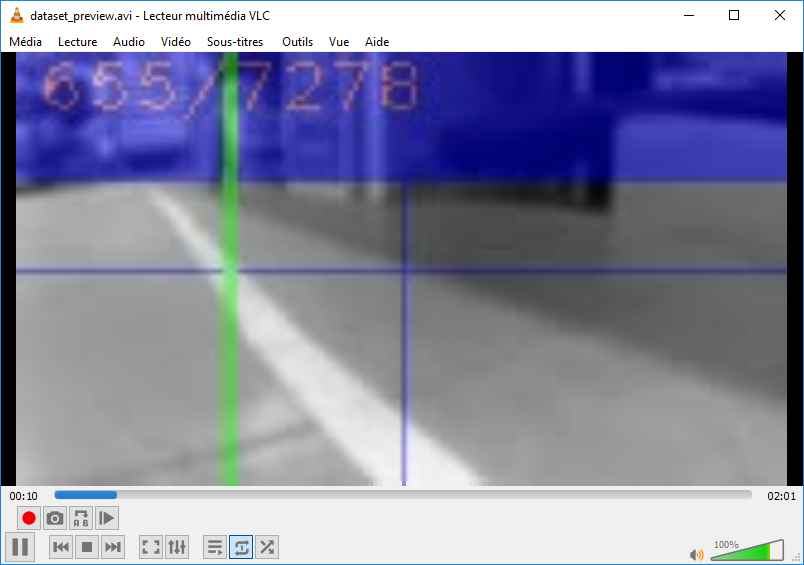
## Outil de visualisation d’un dataset SOURCE labélisés

Le script Python de visualisation est « 11\_Labels\_2\_Video.py ». Il génère un fichier video « dataset\_preview.avi » dans le dossier « video » du dossier local contenant les datasets.

1. Sélectionner le dataset à visualiser, en modifiant la variable **« dataset\_list »** du script.



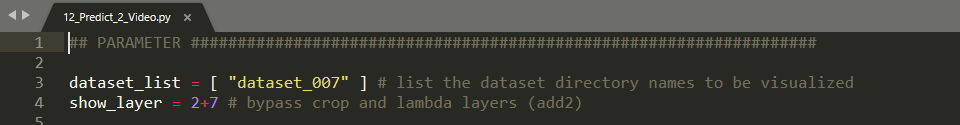
1. Lancer le script.
2. Ouvrir la vidéo générée.



## Outil de visualisation d’un dataset SOURCE labélisés avec prédiction par CNN

Le script Python de visualisation est « 12\_Predict\_2\_Video.py ». Il génère un fichier video « dataset\_prediction.avi » dans le dossier « video » du dossier local contenant les datasets.

1. Sélectionner le dataset à visualiser et la couche du CNN à visualiser, en modifiant les variables du script.



1. Lancer le script. La visualisation de la couche interne complète la visualisation du dataset.



1. Ouvrir la vidéo générée.

