

# Algorithmes d'intelligence artificielle multimodaux pour la detection d'activité dans le véhicule



*Stagiaire :*  
Jeremy Jaspar 3401421 (ISSI)

*Maitre de Stage :*  
Matthieu Donain

# Plan de la présentation

---

1. Presentation du projet
2. Environnement de travail
3. Detection & Reconnaissance Faciale :
  - A. Detection Faciale
  - B. Reconnaissance Faciale
4. Detection de voix & Reconnaissance de locuteur :
  - A. Detection de voix
  - B. Reconnaissance de Locuteur
5. Integration dans le véhicule
6. Conclusion

# 1. Présentation du projet

---

# 1. Présentation du projet

---

## Problématique générale :

Mettre en place des algorithmes de deep learning multimodaux pour la detection d'activité dans le véhicule

## Multimodaux ? :

- Utilisation de plusieurs sources d'information (audio, video, ...)
- Pour de la robustesse

## Cas d'utilisation :

- Identification du conducteur
- Surveillance de la concentration du conducteur sur la route
- Personnalisation des fonctions de conduite
- Couplage avec un chatbot

## Mes objectifs :

- Reconnaissance Faciale (video)
- Reconnaissance de Locuteur (audio)

# 1. Présentation du projet

---

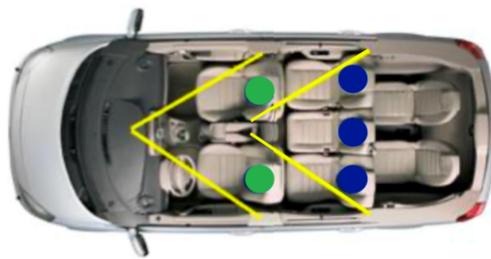
## Cahier des charges :

- Embarquable
- Robuste
- Video :
  - Luminosité (Jour/Nuit)
  - Position (conducteur et passagers)
- Audio :
  - Bruit environnant
  - Multi-locuteurs
  - Texte libre
- Rapide (temps-réel ou presque)

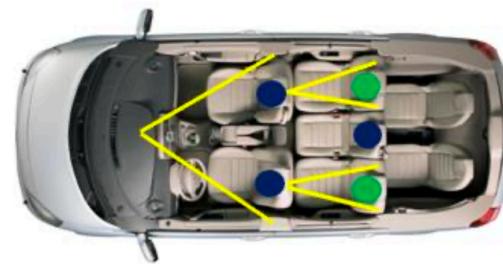
## 2. Environnement de travail

---

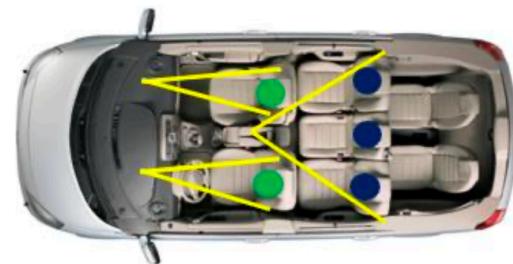
## Position des Cameras



Configuration 0

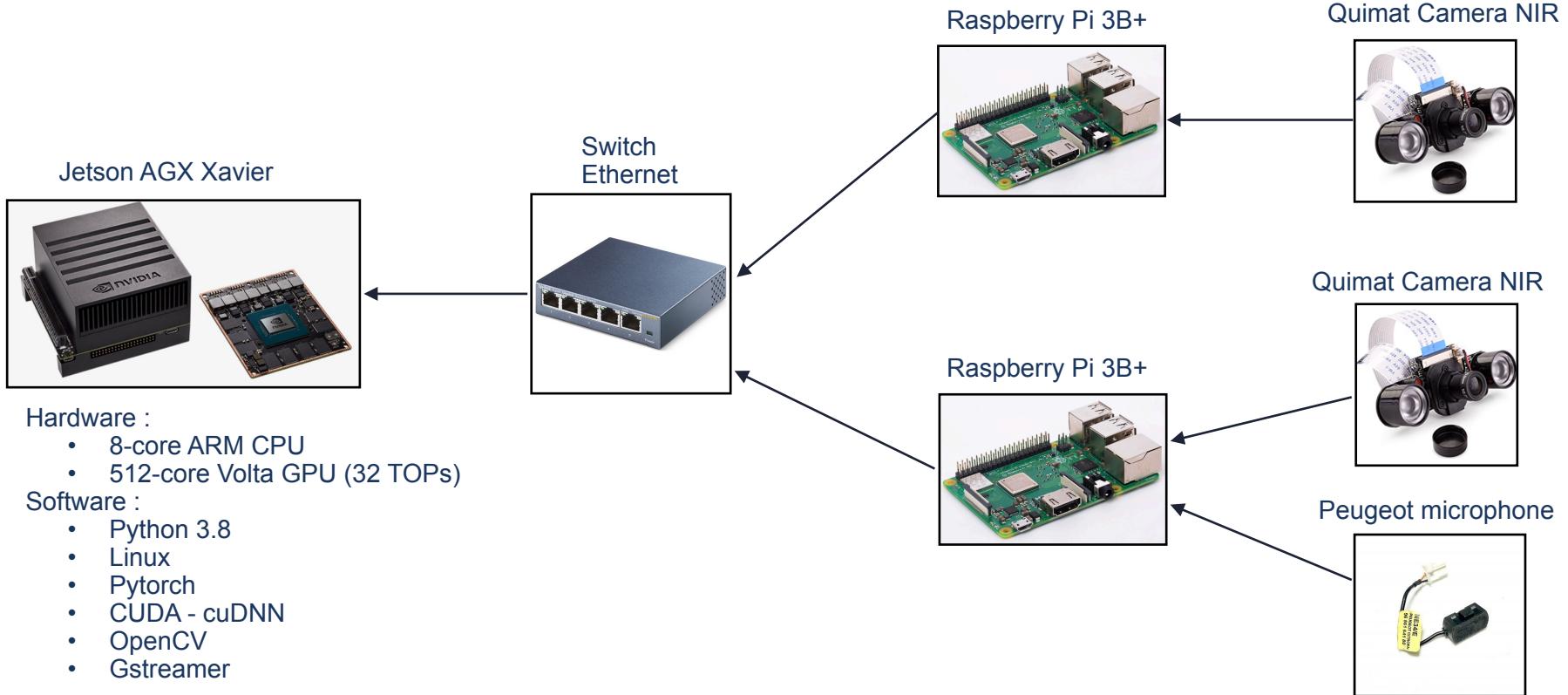


Configuration 1



Configuration 2

## 2. Environnement de travail



# 3. Détection et Reconnaissance Faciale

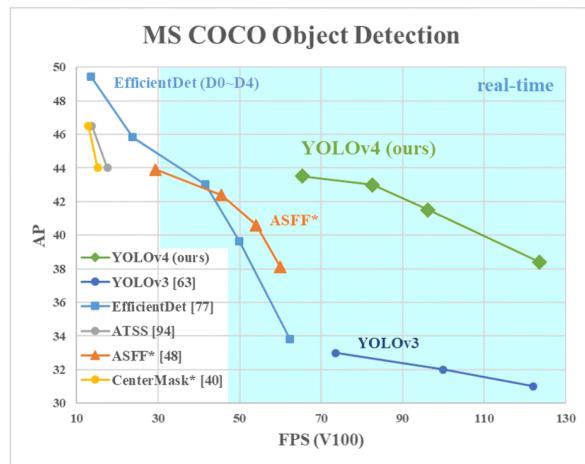
---

### 3. Détection et Reconnaissance Faciale

#### A. Détection de visage

## YOLOv4 : You Only Look Once

- Github : AlexeyAB (Alexey Bochkovskiy), Caffe
- Conversion Pytorch



Wider Face

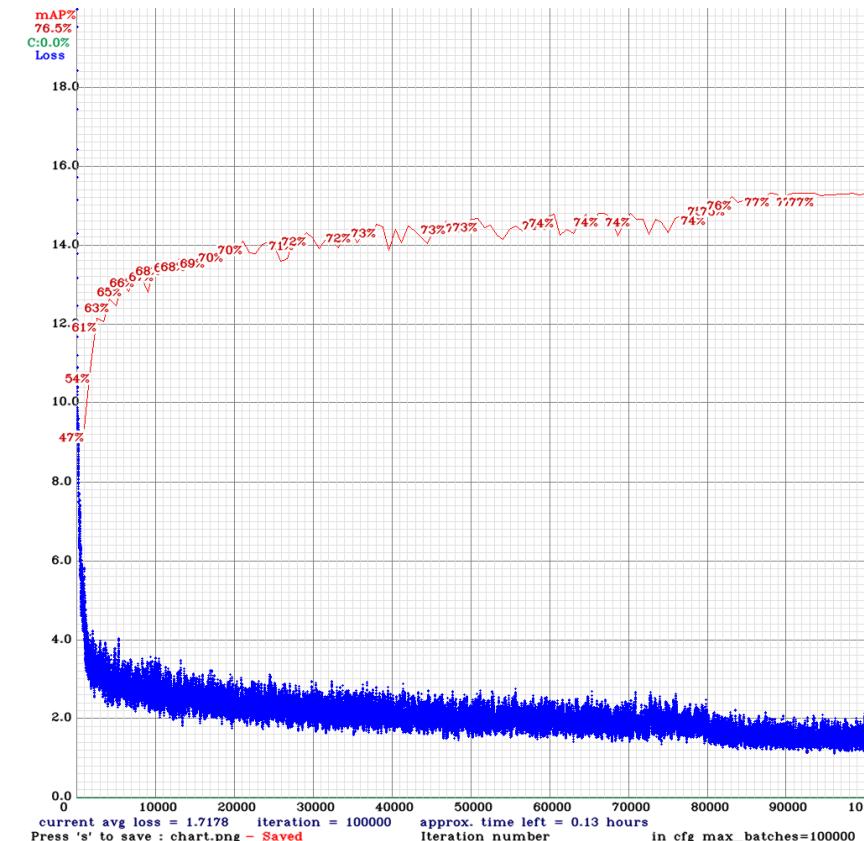
YOLOv4 performances sur COCO

### 3. Détection et Reconnaissance Faciale

## A. Détection de visage

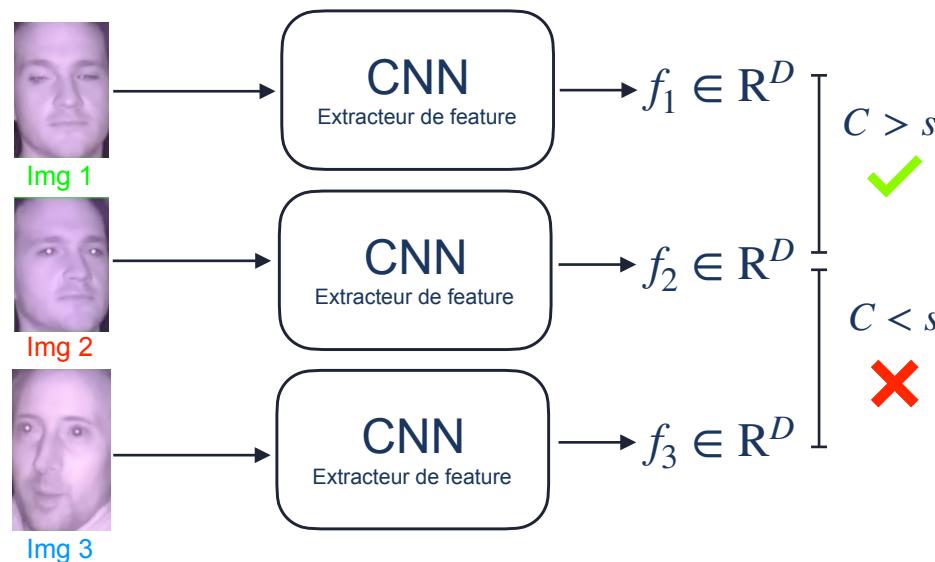
# YOLOv4 : You Only Look Once

- **Results :**



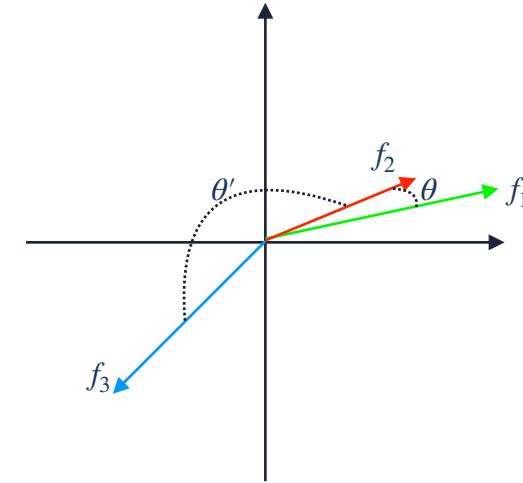
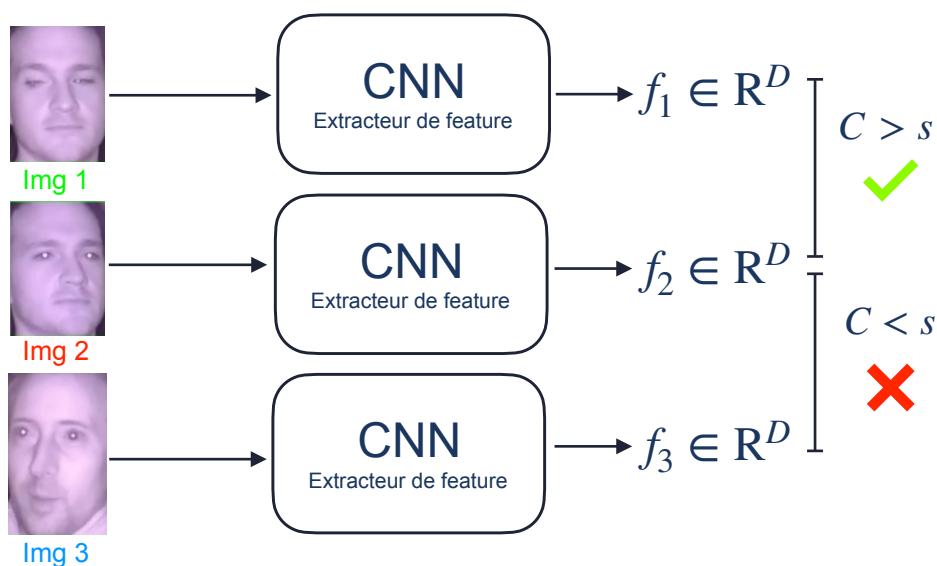
## Réseaux Siamois

- Pourquoi ?
  - Inclus la classe « inconnu »
  - Facilite l'ajout d'identité
- Fonctionnement :



## Réseaux Siamois

- Pourquoi ?
  - Inclus la classe « inconnu »
  - Facilite l'ajout d'identité
- Fonctionnement :

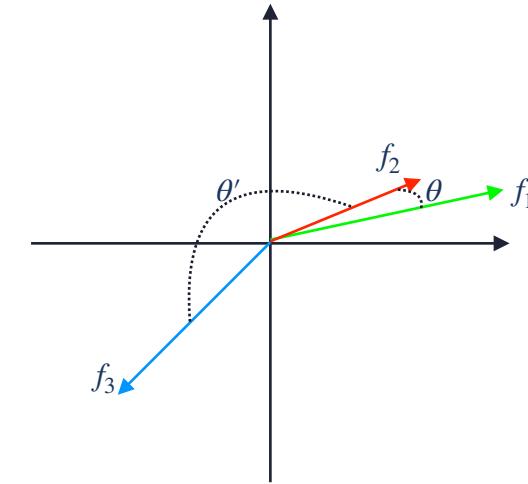
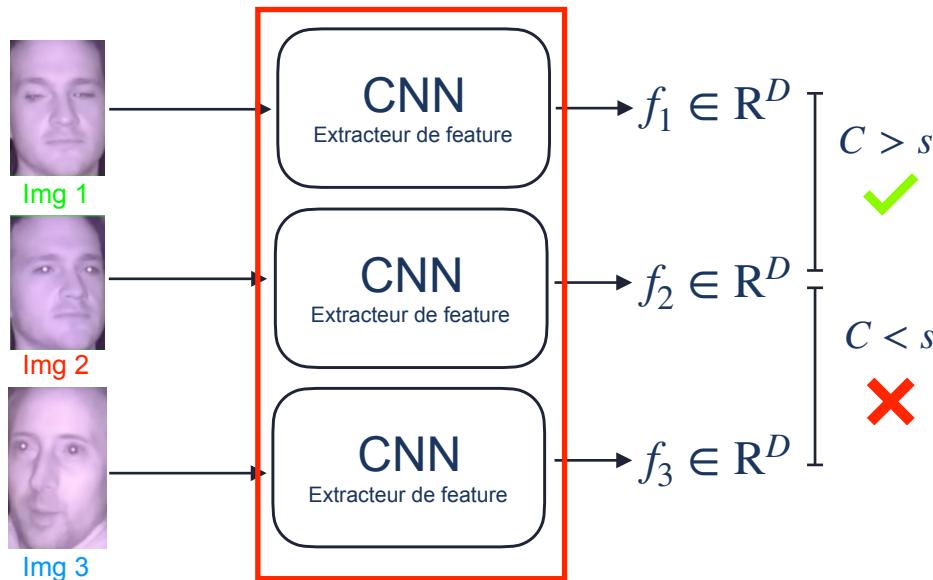


$$x_1 \cdot x_2 = \|x_1\| \cdot \|x_2\| \cdot \cos(\theta)$$

$$C(x_1, x_2) = \frac{x_1 \cdot x_2}{\max(\|x_1\| \cdot \|x_2\|, \epsilon)} = \cos(\theta) \in [-1, 1]$$

## Réseaux Siamois

- Pourquoi ?
  - Inclus la classe « inconnu »
  - Facilite l'ajout d'identité
- Fonctionnement :



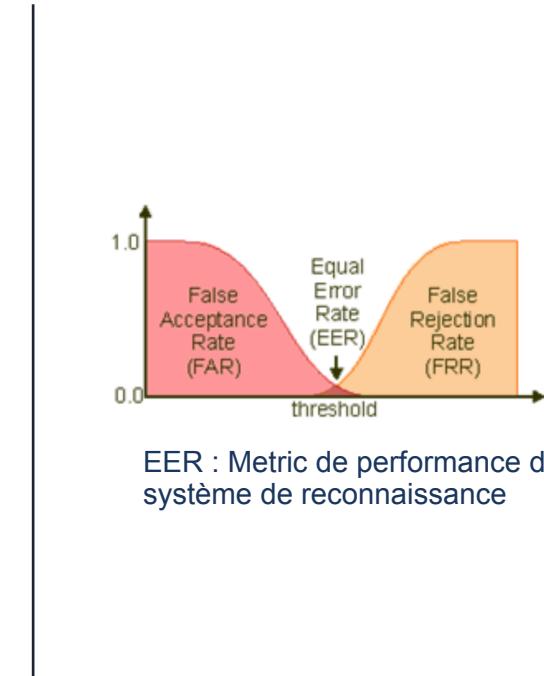
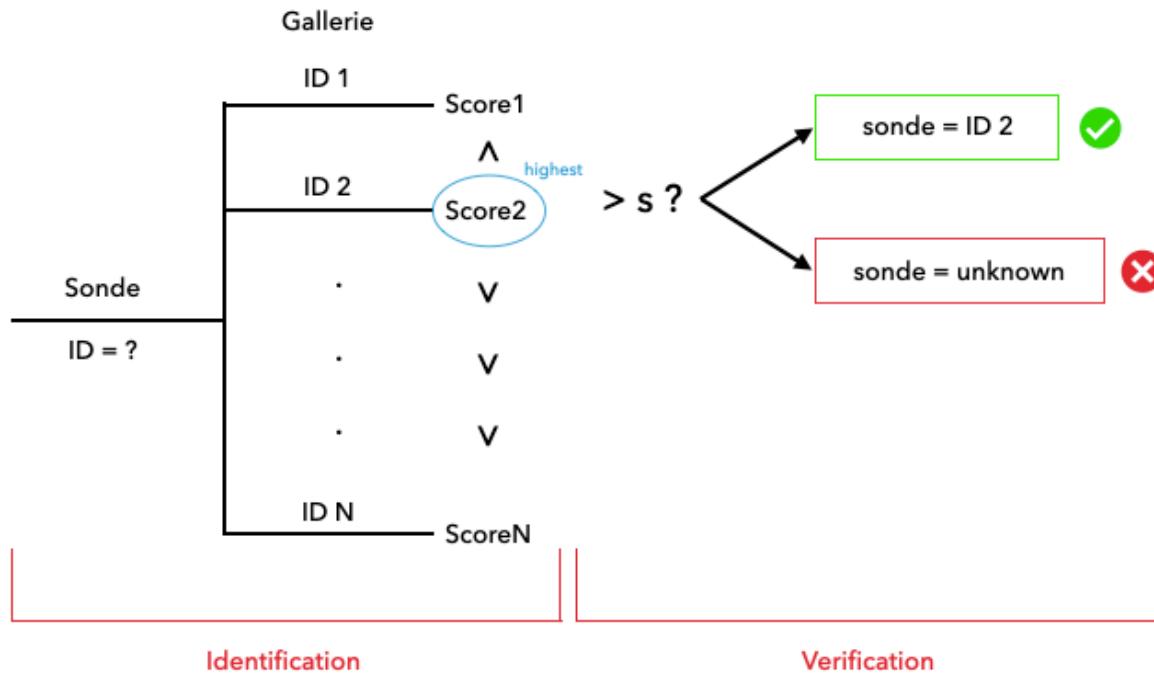
$$x_1 \cdot x_2 = \|x_1\| \cdot \|x_2\| \cdot \cos(\theta)$$

$$C(x_1, x_2) = \frac{x_1 \cdot x_2}{\max(\|x_1\| \cdot \|x_2\|, \epsilon)} = \cos(\theta) \in [-1, 1]$$

### 3. Détection et Reconnaissance Faciale

#### B. Reconnaissance Faciale

## Identification - Vérification



### 3. Détection et Reconnaissance Faciale

#### B. Reconnaissance Faciale

---

## À la recherche du CNN :

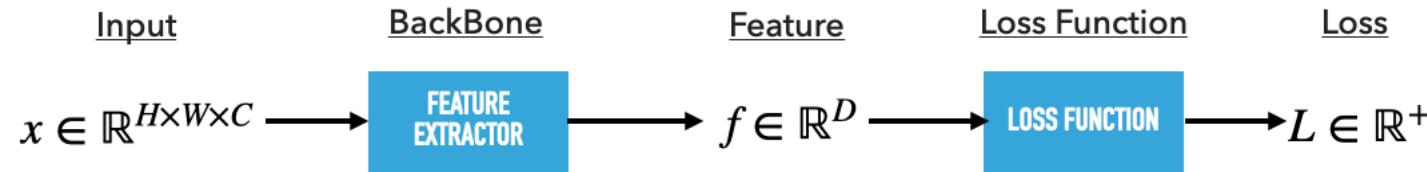
- Critères :
  - Capacités discriminatoire
  - Vitesse d'inférence
  - Besoins mémoire faible

### 3. Détection et Reconnaissance Faciale

#### B. Reconnaissance Faciale

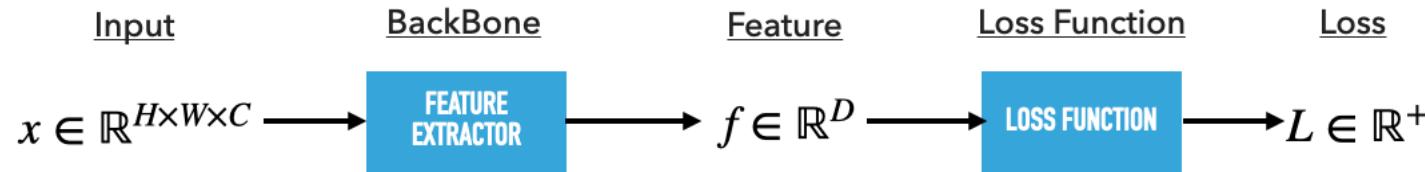
## À la recherche du CNN :

- Critères :
  - Capacités discriminatoire
  - Vitesse d'inférence
  - Besoins mémoire faible
- Entrainement du CNN:



## À la recherche du CNN :

- Critères :
  - Capacités discriminatoire
  - Vitesse d'inférence
  - Besoins mémoire faible
- Entrainement du CNN:



- Architectures choisies :
  - Backbone : ResNet 50, ResNet 100, Inception-ResNet
  - Loss Function : ArcFace

### 3. Détection et Reconnaissance Faciale

#### B. Reconnaissance Faciale

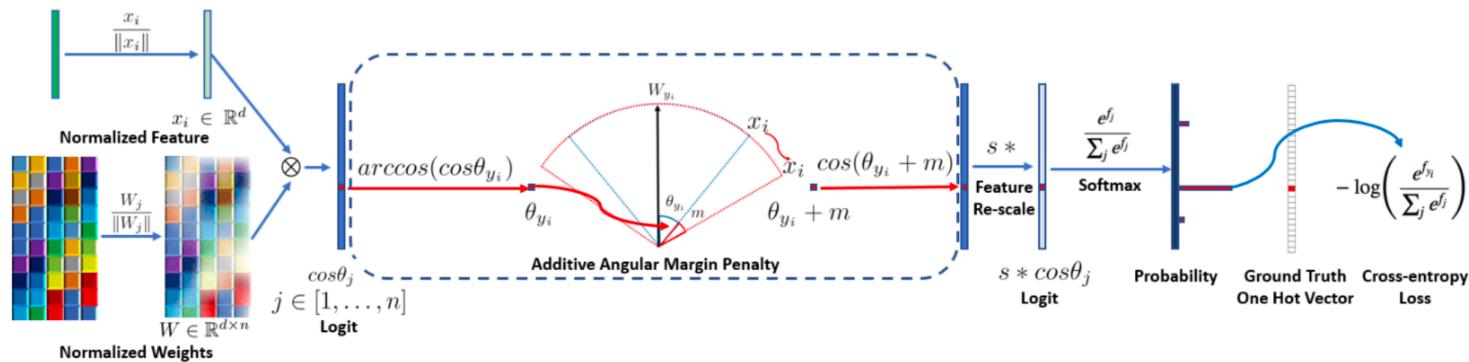
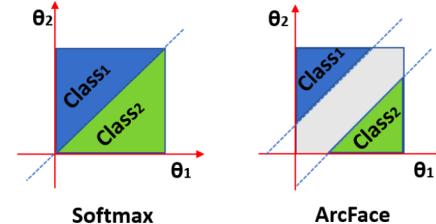
ArcFace :

$$L_1 = -1 \frac{1}{N} \sum_{i=1}^N \log \frac{e^{W_{y_i}^T x_i + b_i}}{\sum_{j=1}^n e^{W_j^T x_i + b_j}}$$

$$x_1 \cdot x_2 = \|x_1\| \cdot \|x_2\| \cdot \cos(\theta)$$

$$L_2 = -1 \frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cos \theta_{y_i}}}{e^{s \cos \theta_{y_i}} + \sum_{j=1, j \neq y_i}^n e^{s \cos \theta_j}}$$

$$L_3 = -1 \frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y_i}^n e^{s \cos \theta_j}}$$

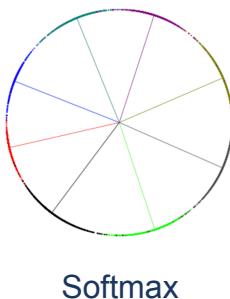


### 3. Détection et Reconnaissance Faciale

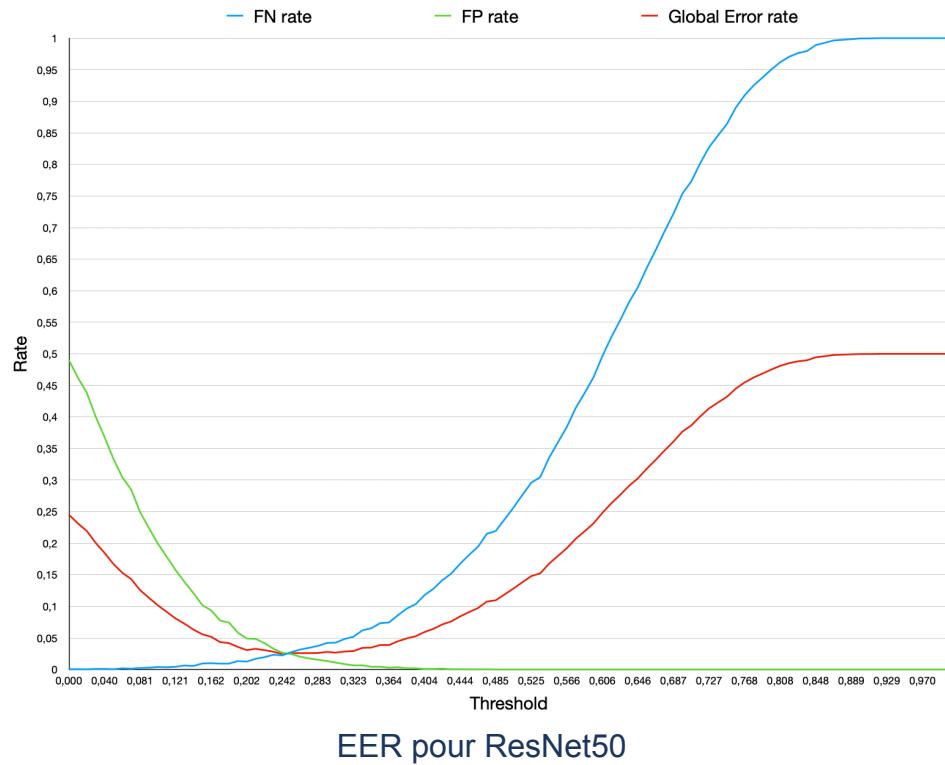
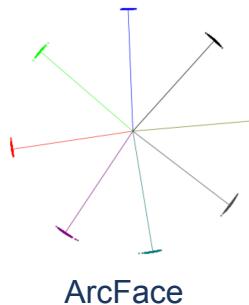
#### B. Reconnaissance Faciale

##### ArcFace :

- Résultats :
  - ResNet 50 : 1.7% EER
  - ResNet 100 : 2.7% EER
  - Inception-ResNet : 1.4% EER



Difference entre Softmax et ArcFace sur 8 classes



### 3. Détection et Reconnaissance Faciale

#### B. Reconnaissance Faciale

##### ArcFace :

- Vitesse d'inférence :

	Mac (ms)	Xavier (ms)
<b>ArcFace 50</b>	<b>80 - 110</b>	<b>27 - 30</b>
<b>ArcFace 100</b>	<b>130</b>	<b>48 - 60</b>
<b>Inception-ResNet</b>	<b>680 - 750</b>	<b>134 - 157</b>

# 4. Détection de voix et Reconnaissance de Locuteur

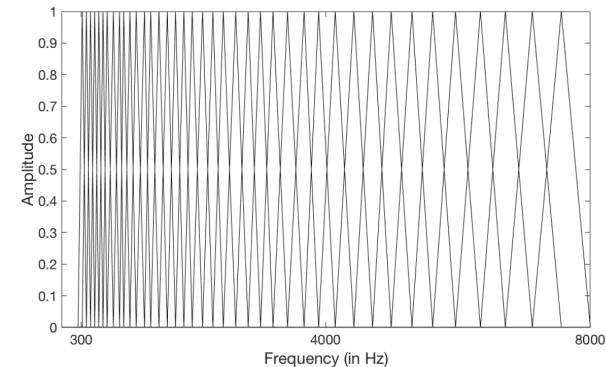
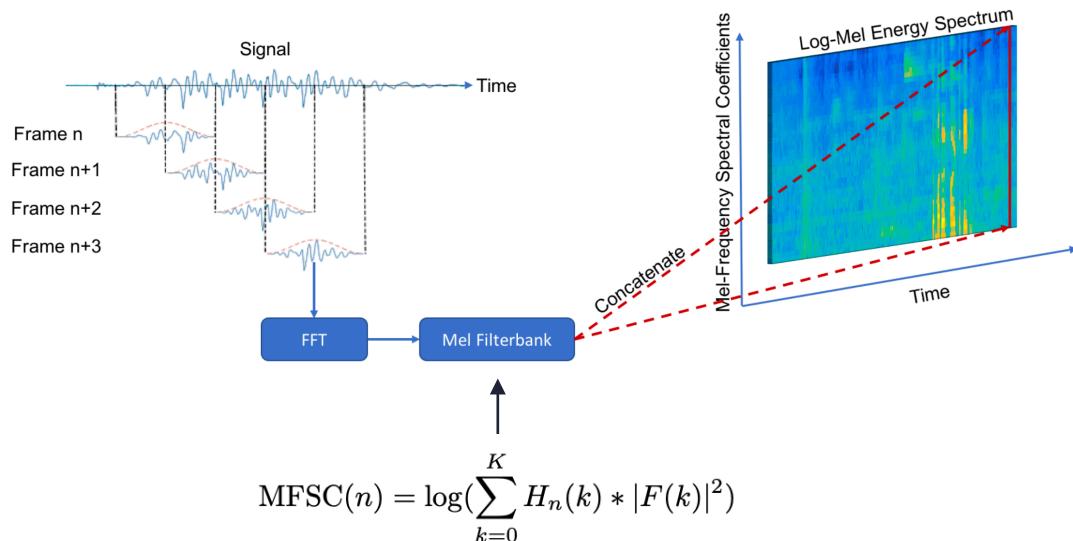
---

## 4. Détection de voix & Reconnaissance de Locuteur

### A. Détection de voix

#### Détection de voix :

- Input : Log-Mel Spectrogramme
  - Mel : L'échelle de fréquences qui désigne une échelle perceptuelle de fréquences qui sont subjectivement jugées égales en distance les unes des autres en termes de sensation auditive humaine.



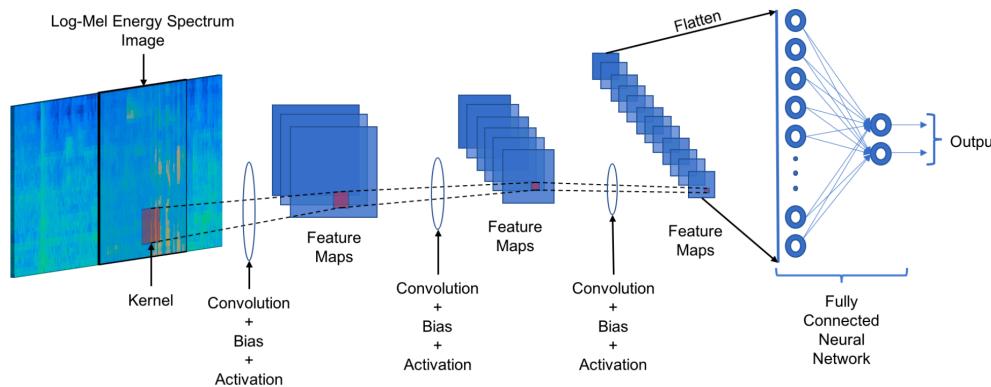
Filtres de Mel

## 4. Détection de voix & Reconnaissance de Locuteur

### A. Détection de voix

#### Détection de voix :

- Architecture :

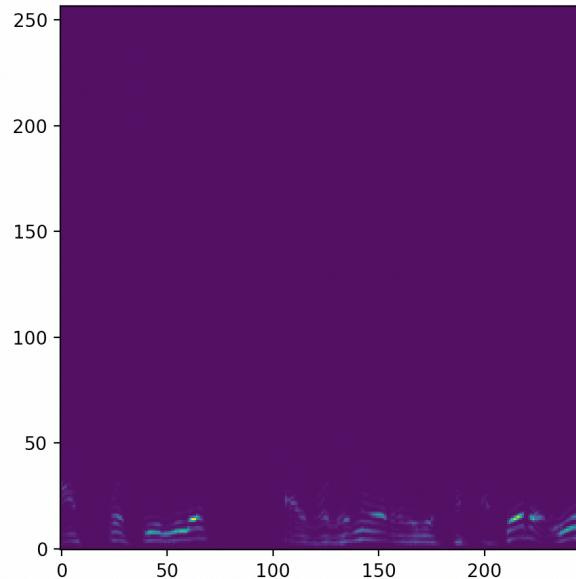


Layer	Number of Kernels/Nodes	Kernel Width
Convolution x 3	40-20-10	5 x 5
Fully Connected	100	-
Softmax	2	-

- Résultats :
  - Dataset AVA : films (video+audio annotés)
  - Précision sur validation : 88%

## Reconnaissance de Locuteur :

- Réseau Siamois : VGGVox
- Input : Spectrogramme



## 4. Détection de voix & Reconnaissance de Locuteur

### B. Reconnaissance de Locuteur

#### Reconnaissance de Locuteur :

- Extracteur de feature : VGGVox
  - Tronc (Trunk) : Thin-ResNet

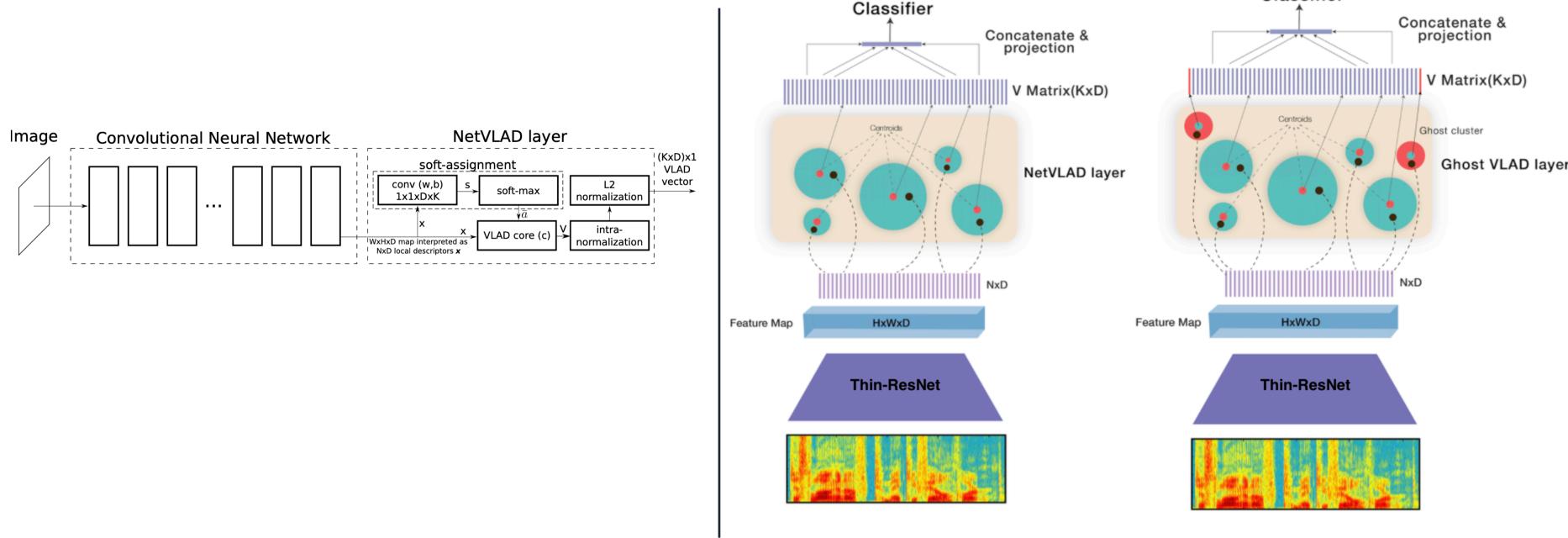
Layer name	Thin-ResNet
conv1	$7 \times 7, 64$ , stride 1
pool1	$3 \times 3$ , max pool stride 2
conv2_x	$\begin{bmatrix} 1 \times 1, 48 \\ 3 \times 3, 48 \\ 1 \times 1, 96 \end{bmatrix} \times 2$
conv3_x	$\begin{bmatrix} 1 \times 1, 96 \\ 3 \times 3, 96 \\ 1 \times 1, 128 \end{bmatrix} \times 3$
conv4_x	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv5_x	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 3$
fc1	$7 \times 1, 512$ , stride 1
pool_time	$3 \times N$ , max pool stride 2

## 4. Détection de voix & Reconnaissance de Locuteur

### B. Reconnaissance de Locuteur

#### Reconnaissance de Locuteur :

- Extracteur de feature : VGGVox
  - Aggregation : NetVLAD - GhostVLAD

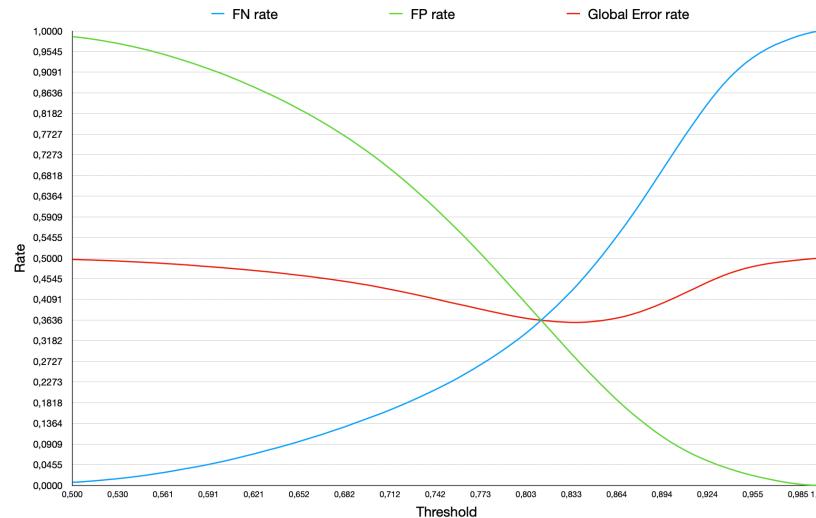


## 4. Détection de voix & Reconnaissance de Locuteur

### B. Reconnaissance de Locuteur

#### Reconnaissance de Locuteur :

- Résultats :
  - Entraîné sur VoxCeleb 2 (1 millions, 6000 classes)
  - Testé sur VoxCeleb 1 (100,000 échantillons, 1251 classes)



Test : 36% d'EER

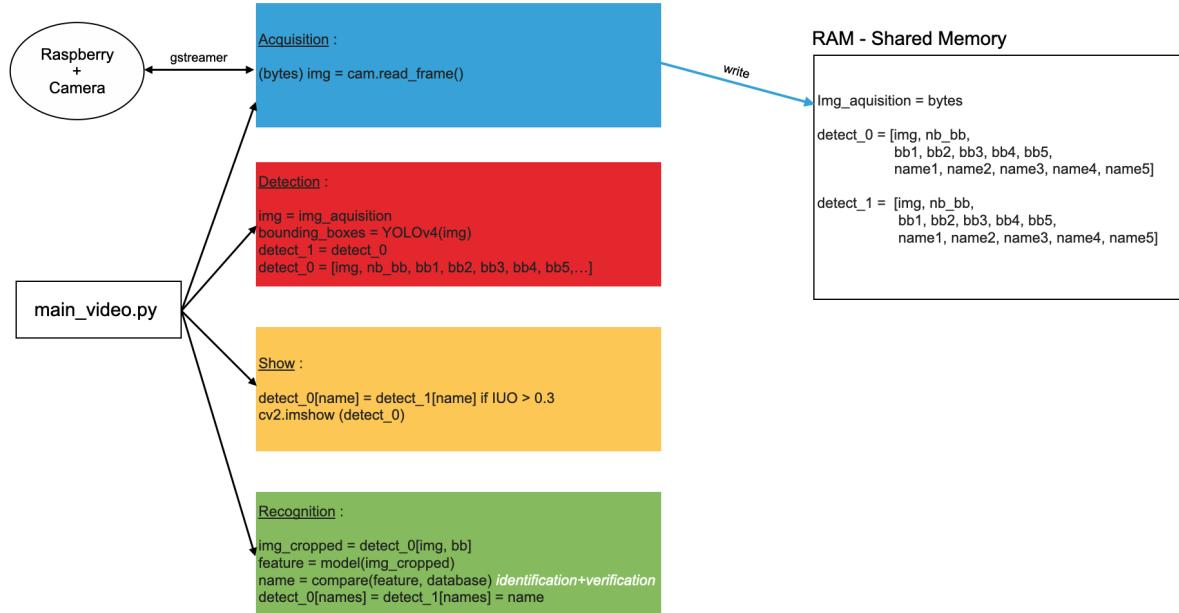
# 5. Intégration dans le véhicule

---

## 5. Intégration dans le véhicule

### Reconnaissance Faciale

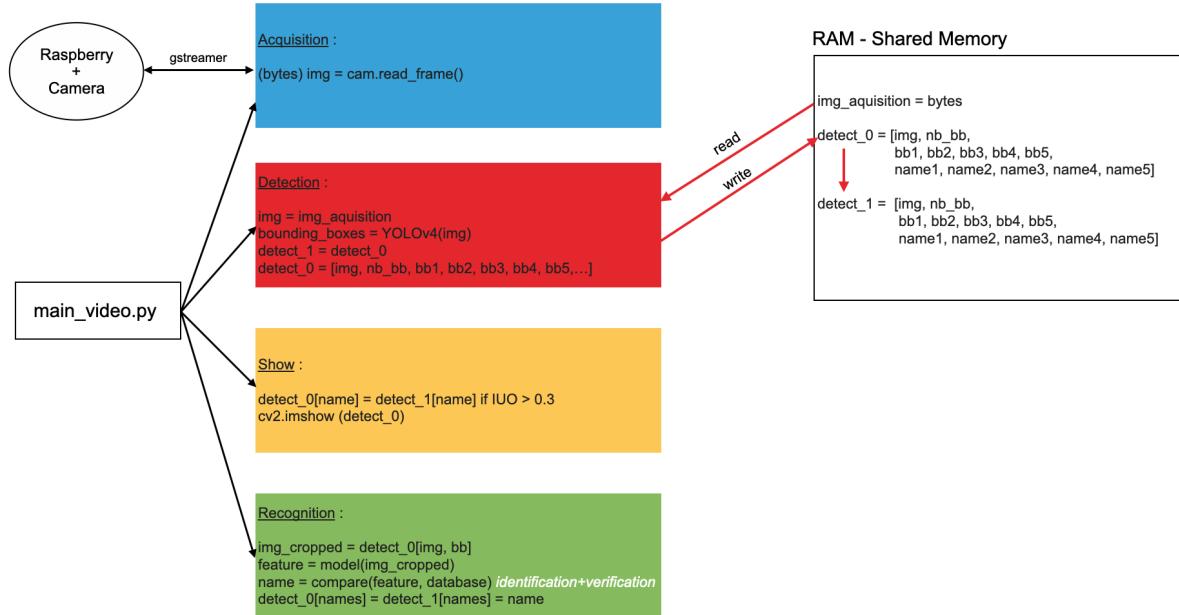
- Fonctionnement pour 1 caméra : Acquisition



## 5. Intégration dans le véhicule

### Reconnaissance Faciale

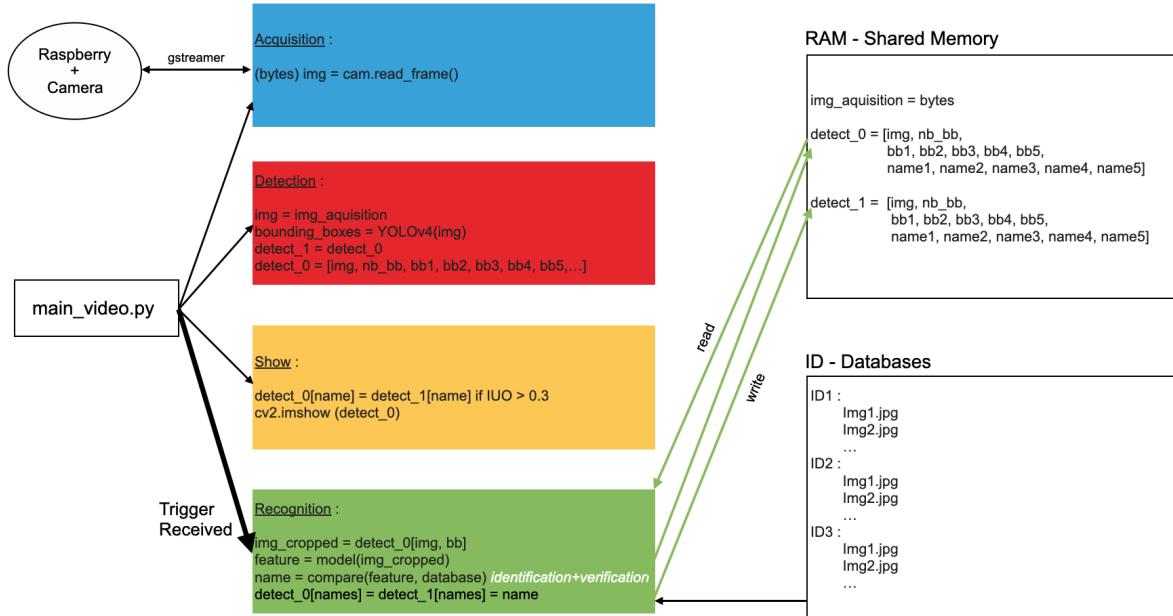
- Fonctionnement pour 1 caméra : Détection



## 5. Intégration dans le véhicule

### Reconnaissance Faciale

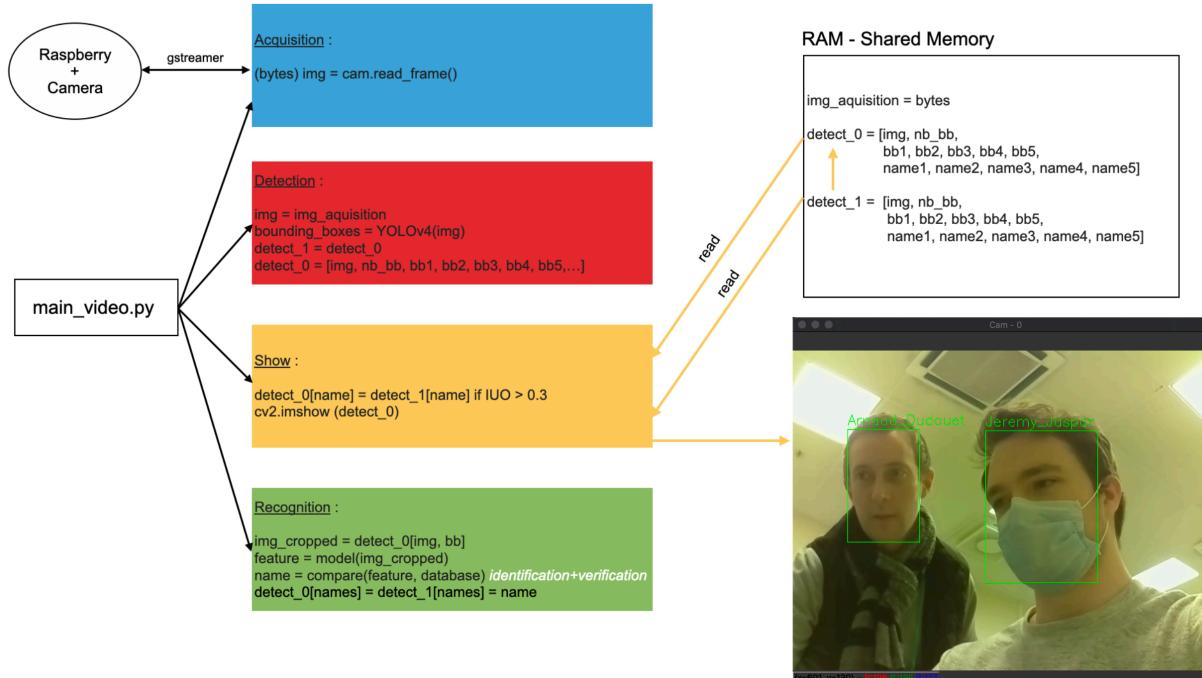
- Fonctionnement pour 1 caméra : Recognition



## 5. Intégration dans le véhicule

### Reconnaissance Faciale

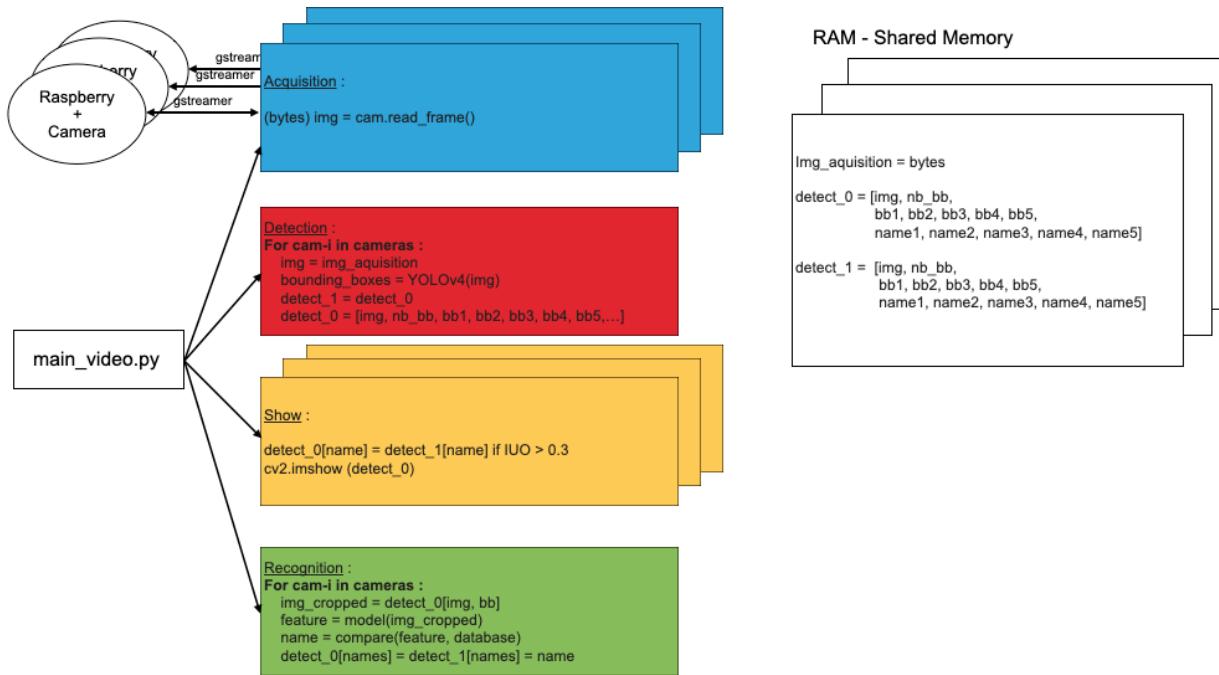
- Fonctionnement pour 1 caméra : Show



## 5. Intégration dans le véhicule

### Reconnaissance Faciale

- Fonctionnement multi-camera

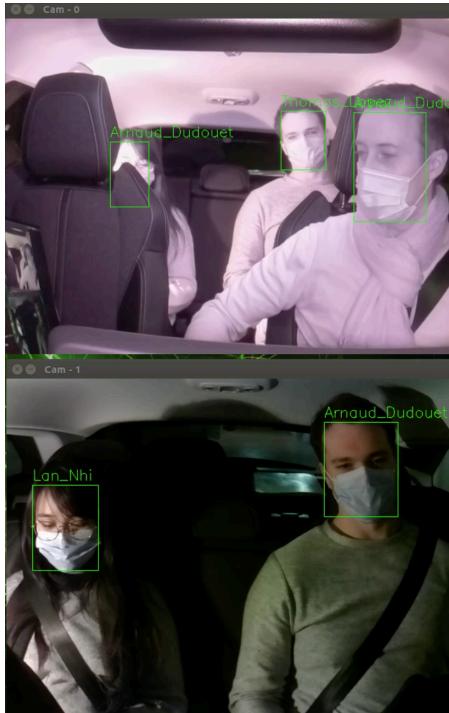


## 5. Intégration dans le véhicule

### Reconnaissance Faciale

- Résultats

Avec masque

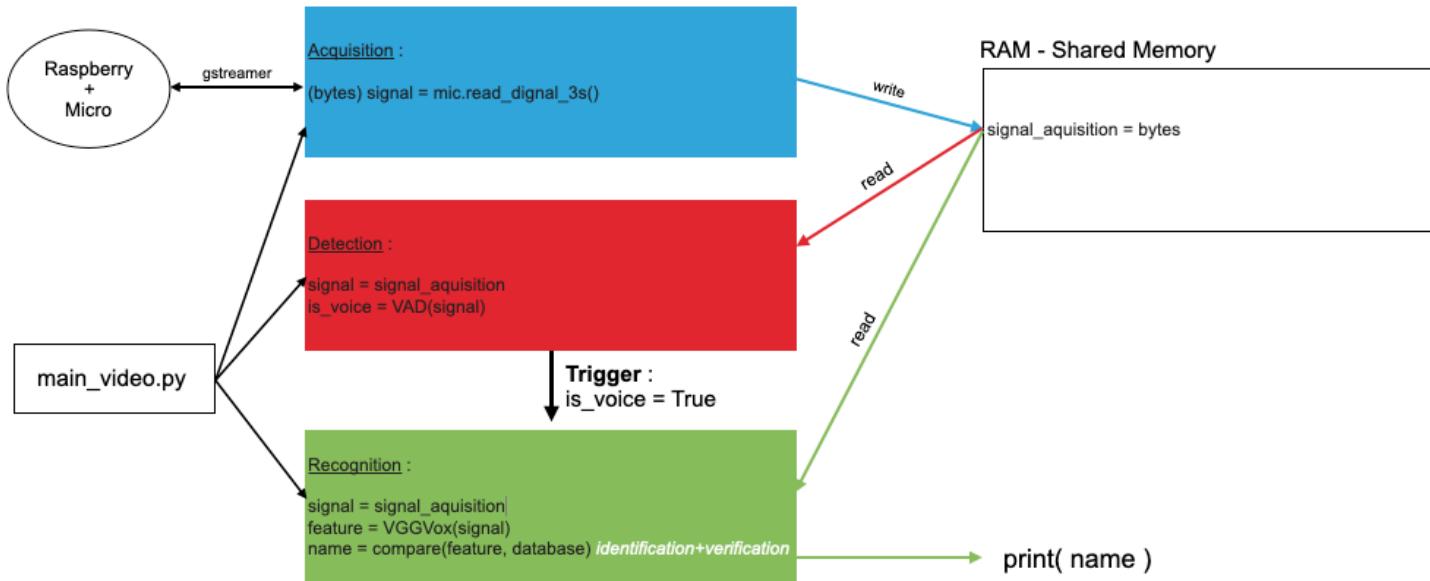


Sans masque



## 5. Intégration dans le véhicule

### Fonctionnement de l'audio :



## 6. Conclusion

---

## 6. Conclusion

---

### Apprentissage :

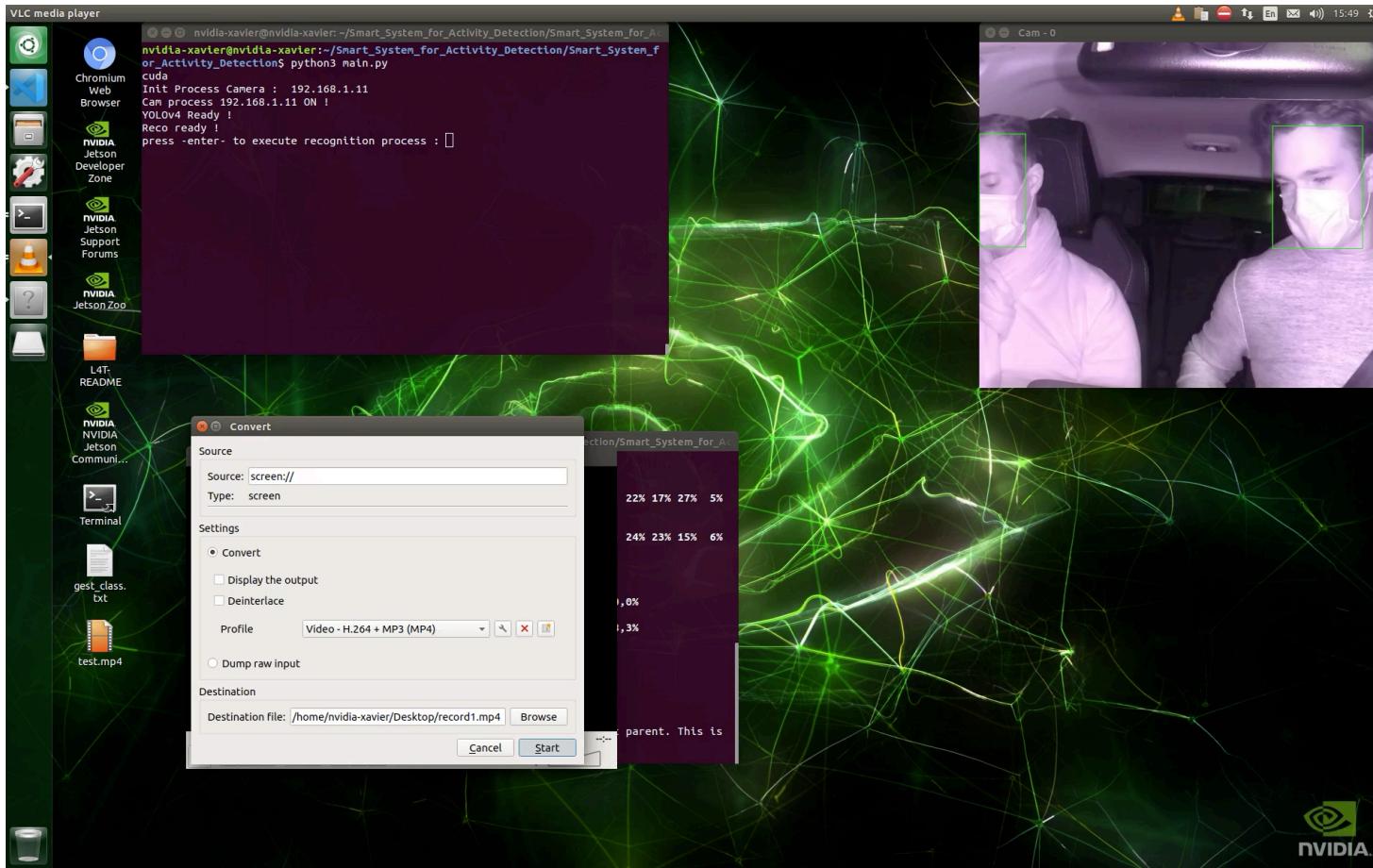
- Programmation : Pytorch, Python3.8
- Théorie Deep Learning
- Server NVIDIA DGX
- Fonctionnement d'une Entreprise

### Difficultés :

- Compatibilité Softwares - Hardware
- Implémentation à partir d'article

### Possible amélioration :

- Fusion de données
- Passage en C++ (ou FPGA)
- Séparation de Locuteurs

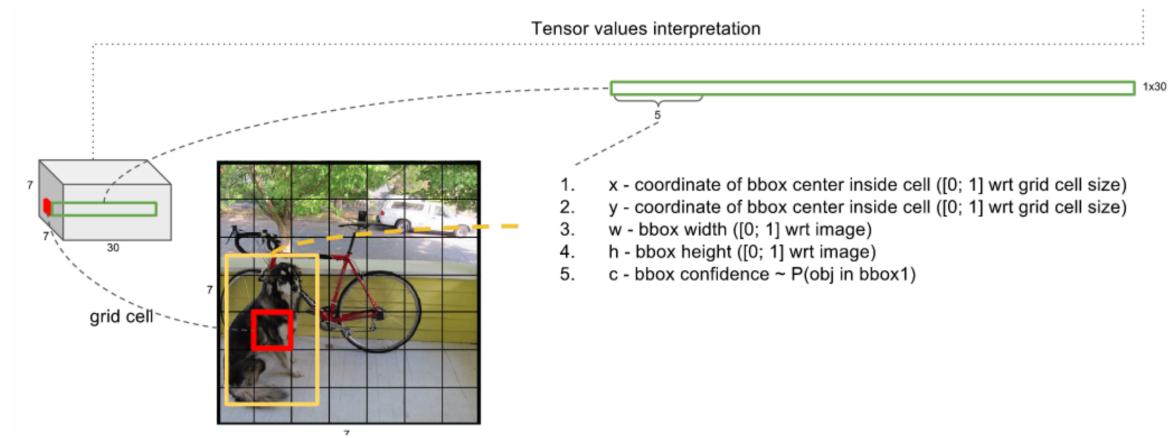


Merci pour votre attention.

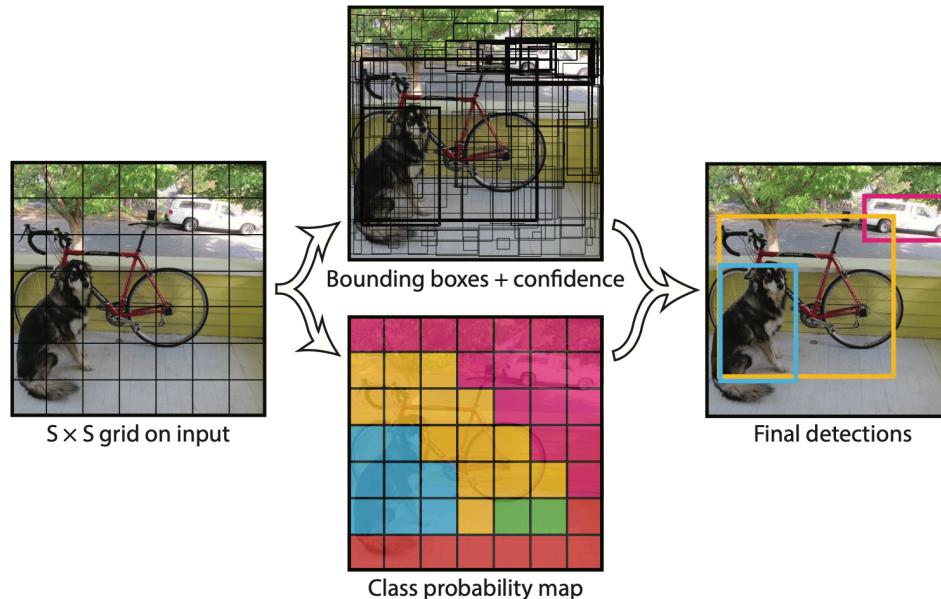
# YOLOv4

- Divise l'image en une grille SxS
- Si centre objet tombe dans une cellule, celle-ci est responsable de la detection de cet objet
- Chaque cellule prédit les boîtes contenant un objet et les scores de confiances
- Le score de confiance :  $P(\text{Objet}) * IoU_{pred}^{truth}$ , 0 si il n'y a pas d'objet
- Chaque boîte se compose de 5 predictions : x, y (centre), w, h , confiance

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

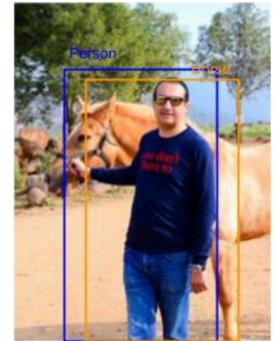
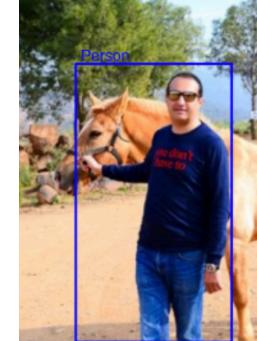
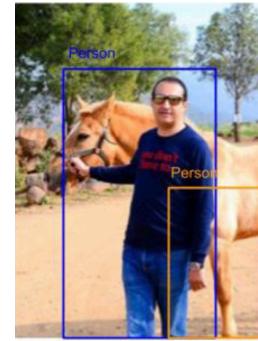
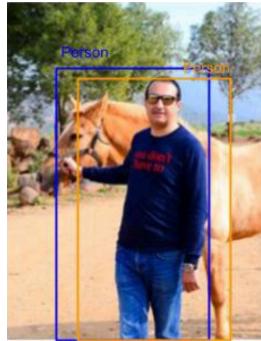


- Chaque cellule prédit également les roba :  $P(\text{Class}_i \mid \text{Objet})$  (un set de class par cellule)



## YOLOv4 : mAP

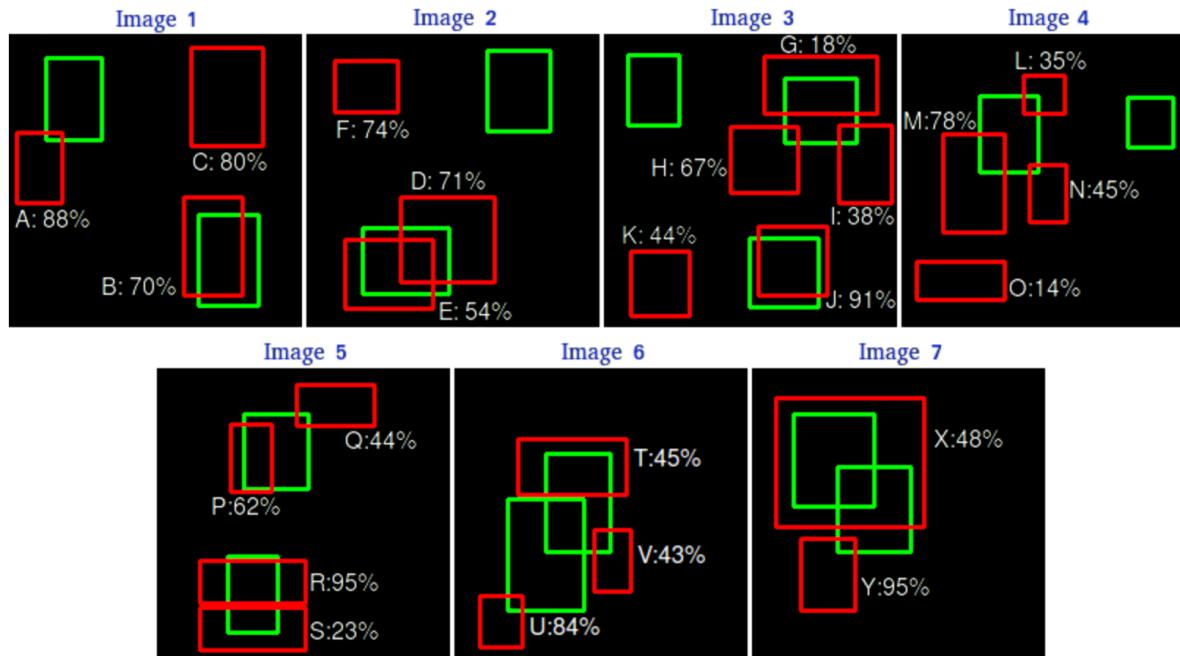
- True Positive : IoU > 0.5
- False Positive : IoU < 0.5 or Duplicated
- False Negative : no box or wrong class



$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detections}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}}$$

## YOLOv4 : mAP



7 images with 15 ground truth objects represented by the green bounding boxes and 24 detected objects represented by the red bounding boxes. Each detected object has a confidence level and is identified by a letter (A,B,...,Y).

## YOLOv4 : mAP

---

- TP is considered if IOU > 30%, otherwise it is a FP
- If multiple detection, first is TP others FP

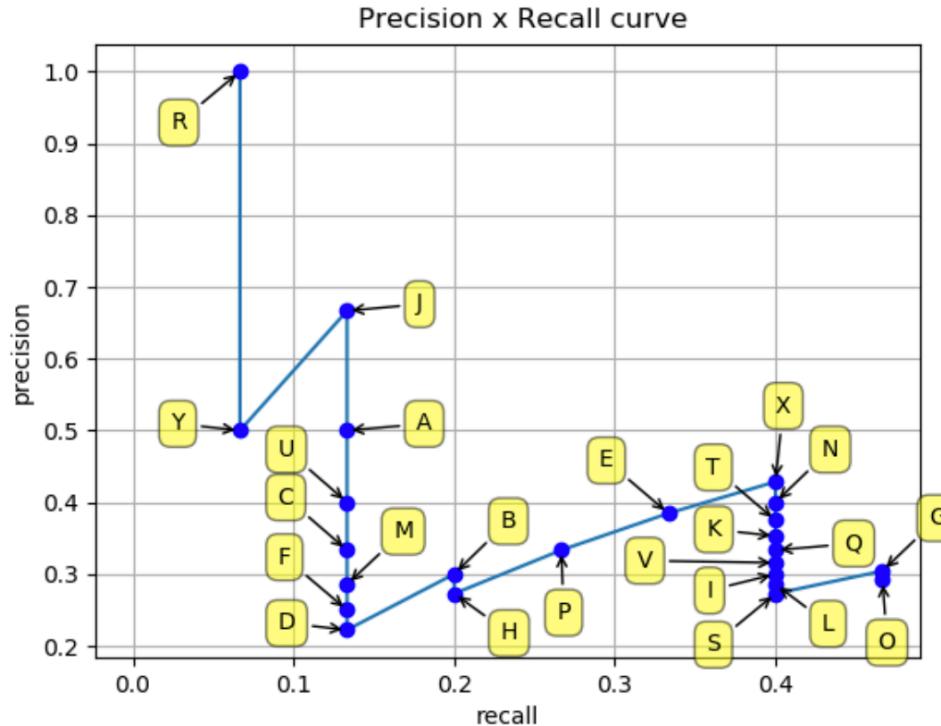
Images	Detections	Confidences	TP or FP
Image 1	A	88%	FP
Image 1	B	70%	TP
Image 1	C	80%	FP
Image 2	D	71%	FP
Image 2	E	54%	TP
Image 2	F	74%	FP
Image 3	G	18%	TP
Image 3	H	67%	FP
Image 3	I	38%	FP
Image 3	J	91%	TP
Image 3	K	44%	FP
Image 4	L	35%	FP
Image 4	M	78%	FP
Image 4	N	45%	FP
Image 4	O	14%	FP
Image 5	P	62%	TP
Image 5	Q	44%	FP
Image 5	R	95%	TP
Image 5	S	23%	FP
Image 6	T	45%	FP
Image 6	U	84%	FP
Image 6	V	43%	FP
Image 7	X	48%	TP
Image 7	Y	95%	FP

## YOLOv4 : mAP

---

- Order the detections by their confidences
- Then we calculate the precision and recall for each accumulated detection

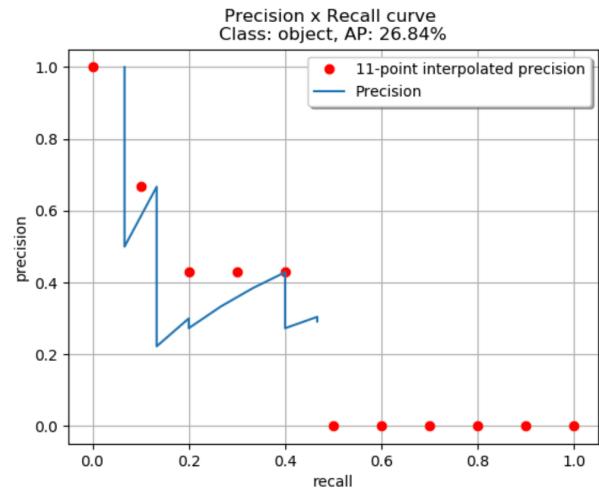
Images	Detections	Confidences	TP	FP	Acc TP	Acc FP	Precision	Recall
Image 5	R	95%	1	0	1	0	1	0.0666
Image 7	Y	95%	0	1	1	1	0.5	0.0666
Image 3	J	91%	1	0	2	1	0.6666	0.1333
Image 1	A	88%	0	1	2	2	0.5	0.1333
Image 6	U	84%	0	1	2	3	0.4	0.1333
Image 1	C	80%	0	1	2	4	0.3333	0.1333
Image 4	M	78%	0	1	2	5	0.2857	0.1333
Image 2	F	74%	0	1	2	6	0.25	0.1333
Image 2	D	71%	0	1	2	7	0.2222	0.1333
Image 1	B	70%	1	0	3	7	0.3	0.2
Image 3	H	67%	0	1	3	8	0.2727	0.2
Image 5	P	62%	1	0	4	8	0.3333	0.2666
Image 2	E	54%	1	0	5	8	0.3846	0.3333
Image 7	X	48%	1	0	6	8	0.4285	0.4
Image 4	N	45%	0	1	6	9	0.4	0.4
Image 6	T	45%	0	1	6	10	0.375	0.4
Image 3	K	44%	0	1	6	11	0.3529	0.4
Image 5	Q	44%	0	1	6	12	0.3333	0.4
Image 6	V	43%	0	1	6	13	0.3157	0.4
Image 3	I	38%	0	1	6	14	0.3	0.4
Image 4	L	35%	0	1	6	15	0.2857	0.4
Image 5	S	23%	0	1	6	16	0.2727	0.4
Image 3	G	18%	1	0	7	16	0.3043	0.4666
Image 4	O	14%	0	1	7	17	0.2916	0.4666



# YOLOv4 : mAP

## Calculating the 11-point interpolation

The idea of the 11-point interpolated average precision is to average the precisions at a set of 11 recall levels  $(0, 0.1, \dots, 1)$ . The interpolated precision values are obtained by taking the maximum precision whose recall value is greater than its current recall value as follows:



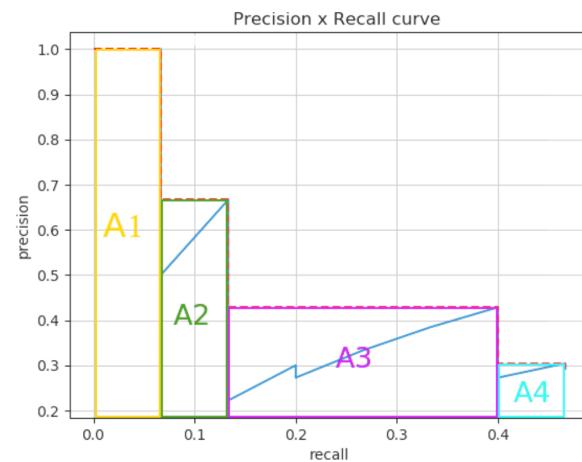
By applying the 11-point interpolation, we have:

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} \rho_{\text{interp}(r)}$$

$$AP = \frac{1}{11} (1 + 0.6666 + 0.4285 + 0.4285 + 0.4285 + 0 + 0 + 0 + 0 + 0 + 0)$$

$$AP = 26.84\%$$

## Calculating the interpolation performed in all points



Calculating the total area, we have the AP:

$$AP = A1 + A2 + A3 + A4$$

with:

$$A1 = (0.0666 - 0) \times 1 = \mathbf{0.0666}$$

$$A2 = (0.1333 - 0.0666) \times 0.6666 = \mathbf{0.04446222}$$

$$A3 = (0.4 - 0.1333) \times 0.4285 = \mathbf{0.11428095}$$

$$A4 = (0.4666 - 0.4) \times 0.3043 = \mathbf{0.02026638}$$

$$AP = 0.0666 + 0.04446222 + 0.11428095 + 0.02026638$$

$$AP = 0.24560955$$

$$AP = \mathbf{24.56\%}$$

