TELECOM SudParis

INSTITUT POLYTECHNIQUE DE PARIS

## FINAL TERM PRESENTATION

# DEEP LEARNING BASED METHOD FOR AUTOMATIC HUMAN POSE UNDERSTANGING

KAÏS DJEDDOU, ALEXANDRE MAUREL, MATISSE JEAN-MARIE ET LOUIS KEMPF

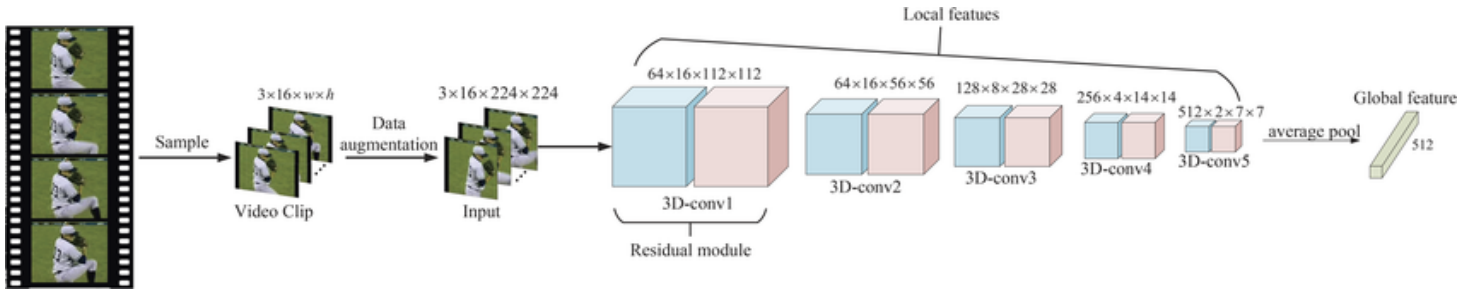
Institut Mines-Télécom

# OBJECTIVES

## AUTOMATIC ACTION RECOGNITION

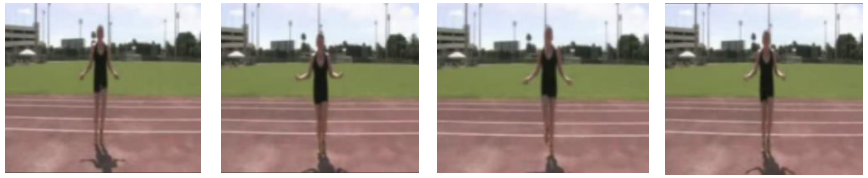
1. To build basic action recognition with the use of Deep Learning based computer vision method.
2. To evaluate the capability of the developed method on the common datasets
3. To apply the method on the real life setting
4. Use image or keypoints for action recognition

(b) stretching leg



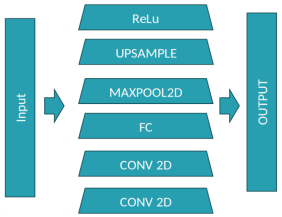
# PIPELINE

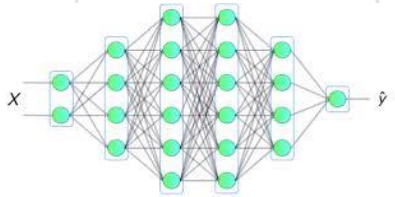
## Data loader

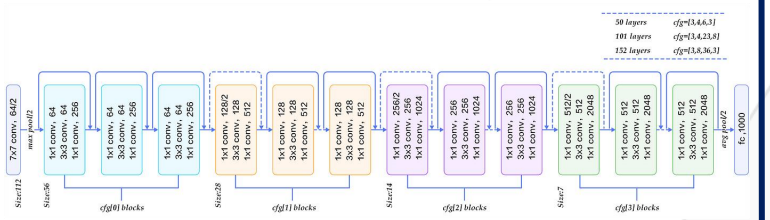


(b) stretching leg

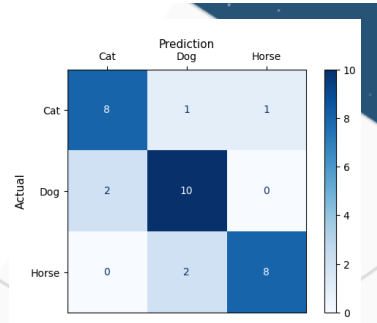
## Model

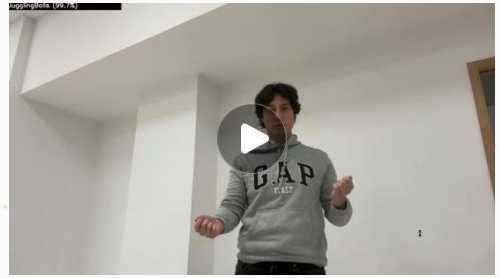


## Output



# Data Loader – Activity Net and UCF 101

## UCF 101

- Data pre-processing :
  - Statistic : 13 320 videos & 101 labels
  - Input : frames (16/video)
  - Output :  class labels + frames

```
================================================
Début de l'extraction des frames pour l'action : JumpRope
Dossier de destination : /home/amine_tsp/DL2026/Datasets/UCF101/mes_frames_video
Terminé ! Toutes les images sont générées.
================================================
```

## ActivityNet

Json file (video url, timecode, label)

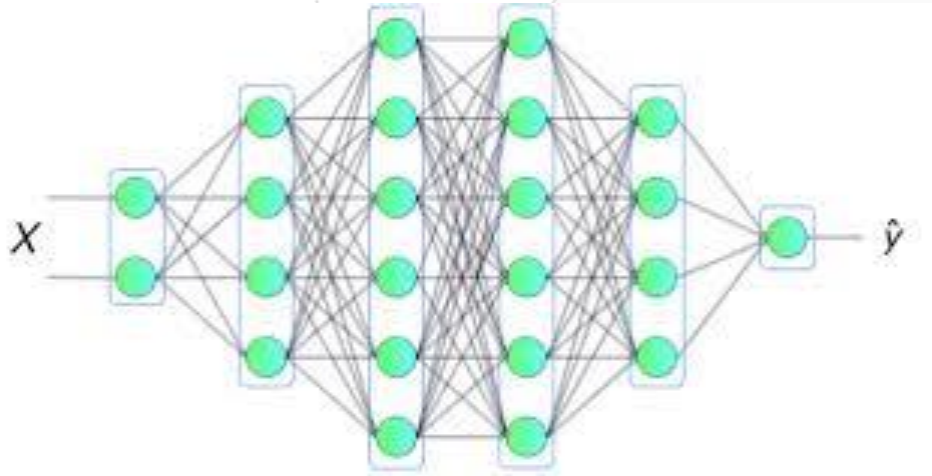
- Download videos (95% corrupted videos).
- -> ~1000 number of videos, 200 classes
- Croping videos
- Data augmentation
- Data splitting (training, testing, validation)
- Framing clips

(b) stretching leg

# Model : Fully Connected

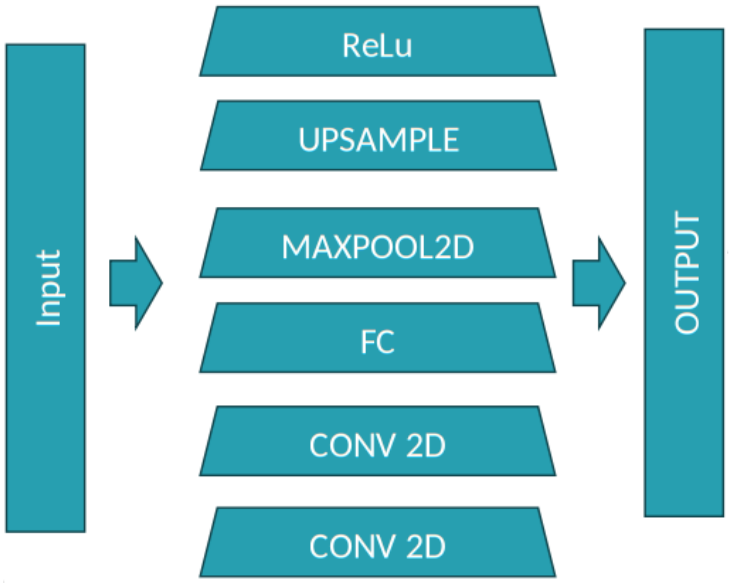
Image 224x224
Frames : 16
Color : 'RGB'
Shape : [3, 16, 224, 224]
InputNode = 3*16*224*224
= 2 408 448
OutputNode : 101

```python
class FC(nn.Module):
    def __init__(self,inputNode=561,hiddenNode = 256, outputNode=1):
        super(FC, self).__init__()
        #Define Hyperparameters
        self.inputLayerSize = inputNode
        self.outputLayerSize = outputNode
        self.hiddenLayerSize = hiddenNode

        self.relu = nn.ReLU()

        # weights
        self.Linear1 = nn.Linear(self.inputLayerSize, self.hiddenLayerSize)
        self.Linear2 = nn.Linear(self.hiddenLayerSize, self.outputLayerSize)
```

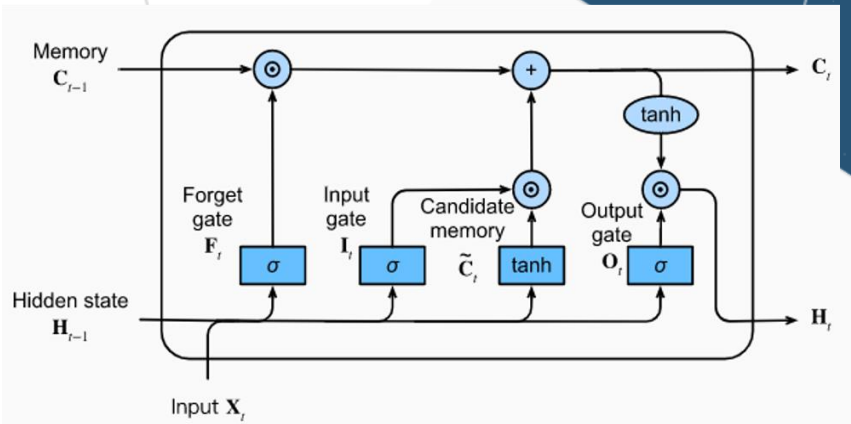


# Model : CNN+LSTM

Image 224x224
Frames : 16
Color : 'RGB'
Shape : [3, 16, 224, 224]
InputNode = 3*16*224*224
= 2 408 448
OutputNode : 101



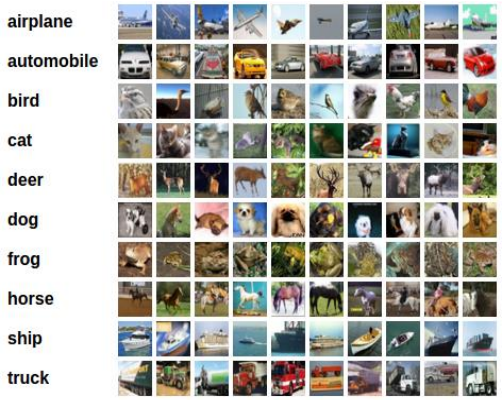

CNN

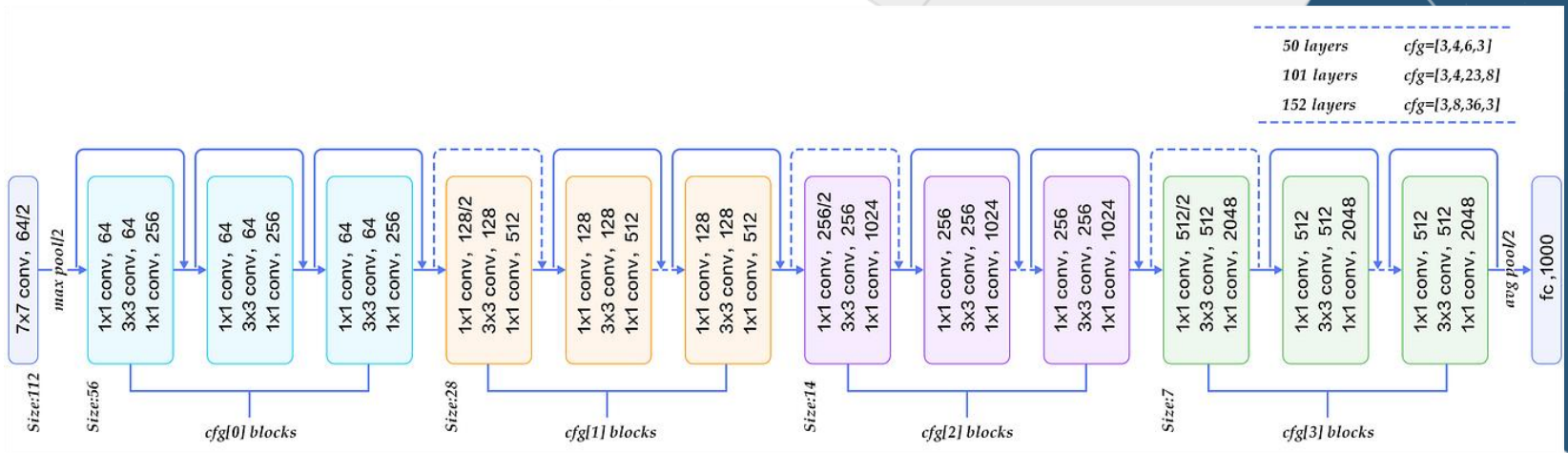



LSTM (RNN)

# Model : ResNet50+LSTM

## Pre-trained ResNet50 on CIFAR-10



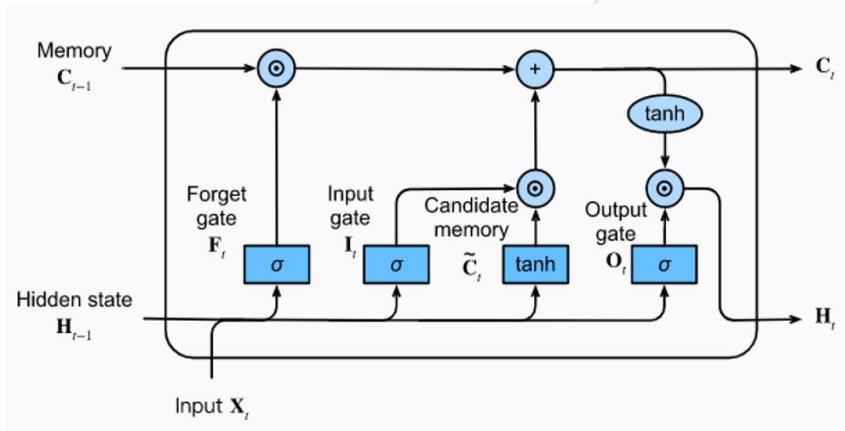




**Image 224x224**
**Frames : 16**
**Color : 'RGB'**
**Shape : [3, 16, 224, 224]**
**InputNode = 3*16*224*224**
**= 2 408 448**
**OutputNode : 101**

# Model : ResNet3D-18

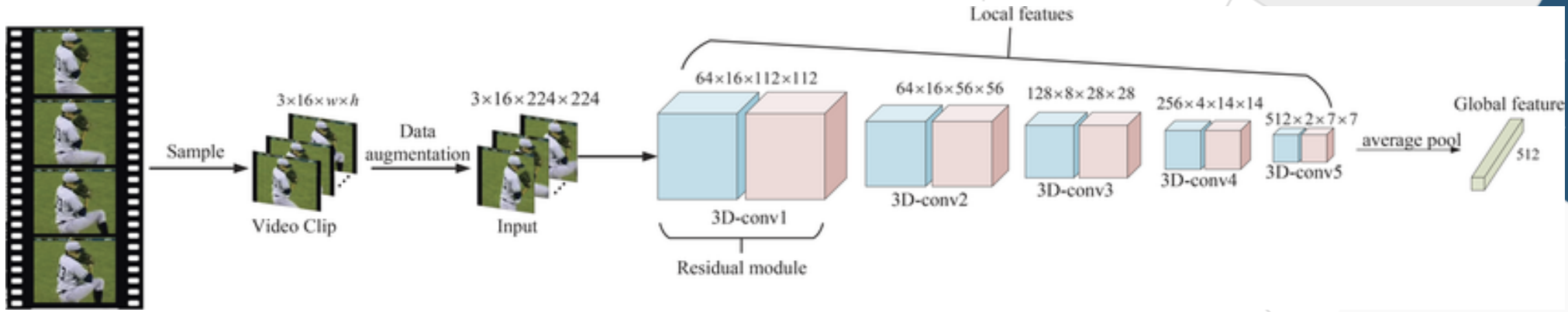
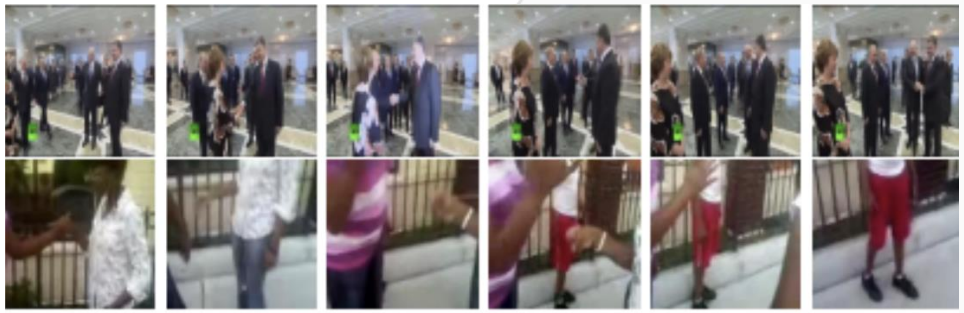



**Image:** 224 x 224
**Frames:** 16
**Colors:** 'RGB'
**Shape:** [3, 16, 224, 224]
**InputNode =** 3*16*224*224
**=** 2 408 448
**OutputNode :** 101

**Pre-trained on Kinetics-400**



(c) shaking hands

# Quantitative evaluation

## Accuracy

Accuracy measures the absolute correct prediction of the model

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{All predictions}}$$

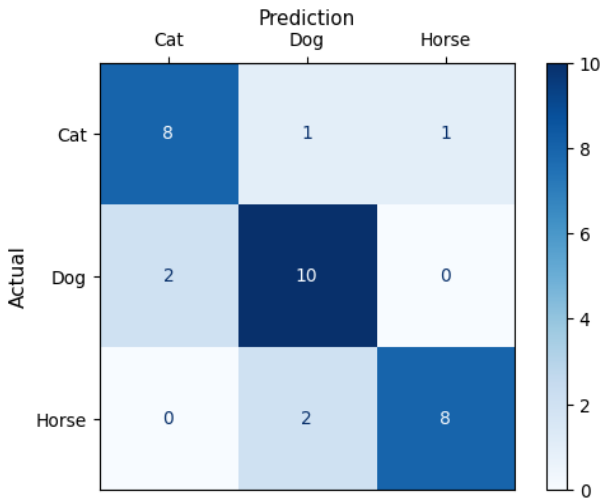
## F1score

F1 score takes into account the label distribution

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

## Confusion matrix

Compare the prediction with their true label



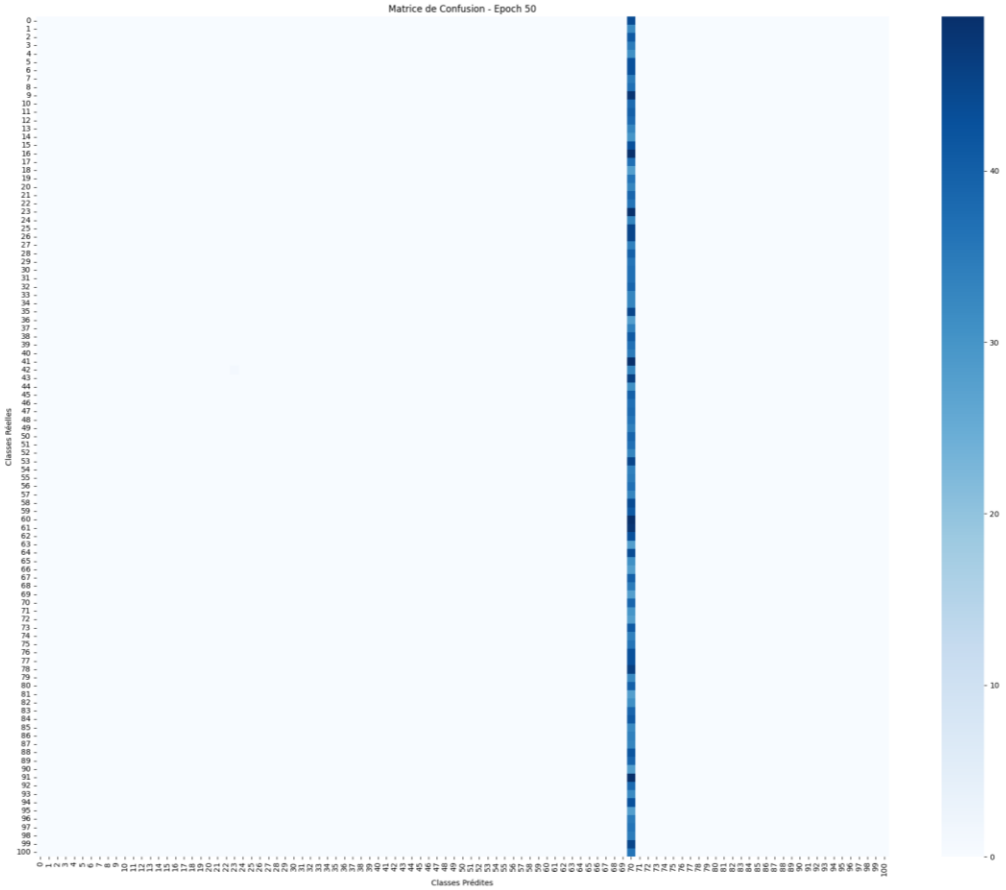
# Quantitative Comparisons

- The best methods are the ResNet3D, with accuracy score of 79,67% and F1 score of 0,7895
- FC doesnt good well bc of complexity of the model and data
- CNN+LSTM worked better than FC bc it take into account the image information (text, color etc) and the time information
- ResNet50 and ResNet3D worked even better given that they have been pre-trained on other datasets

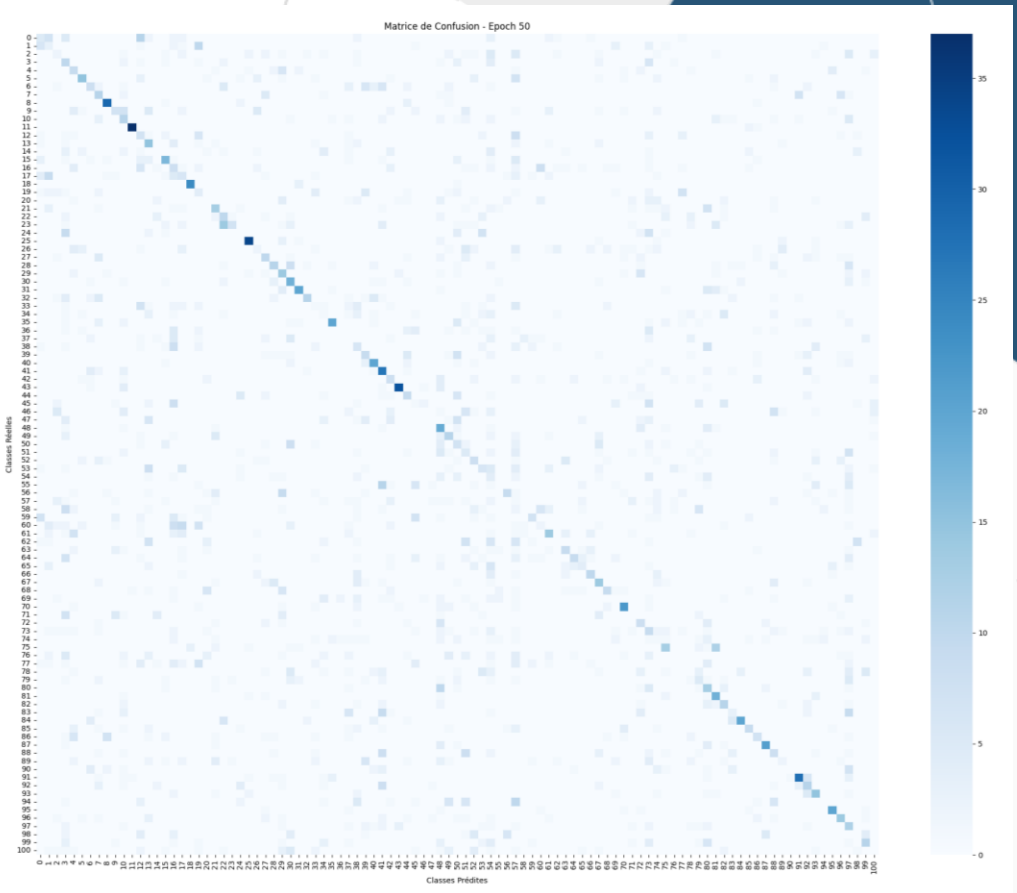
| **Methods** | **UCF 101** | |
|---|---|---|
| | Accuracy | F1 score |
| Fully connected | 1.14% | 0.0002 |
| CNN + LSTM | 25.67% | 0.2379 |
| ResNet50 + LSTM | 72.38% | 0.7072 |
| ResNet3D | 79.67 % | 0.7895 |

# Result 1 : confusion matrix for each model (1/2)

## Fully Connected

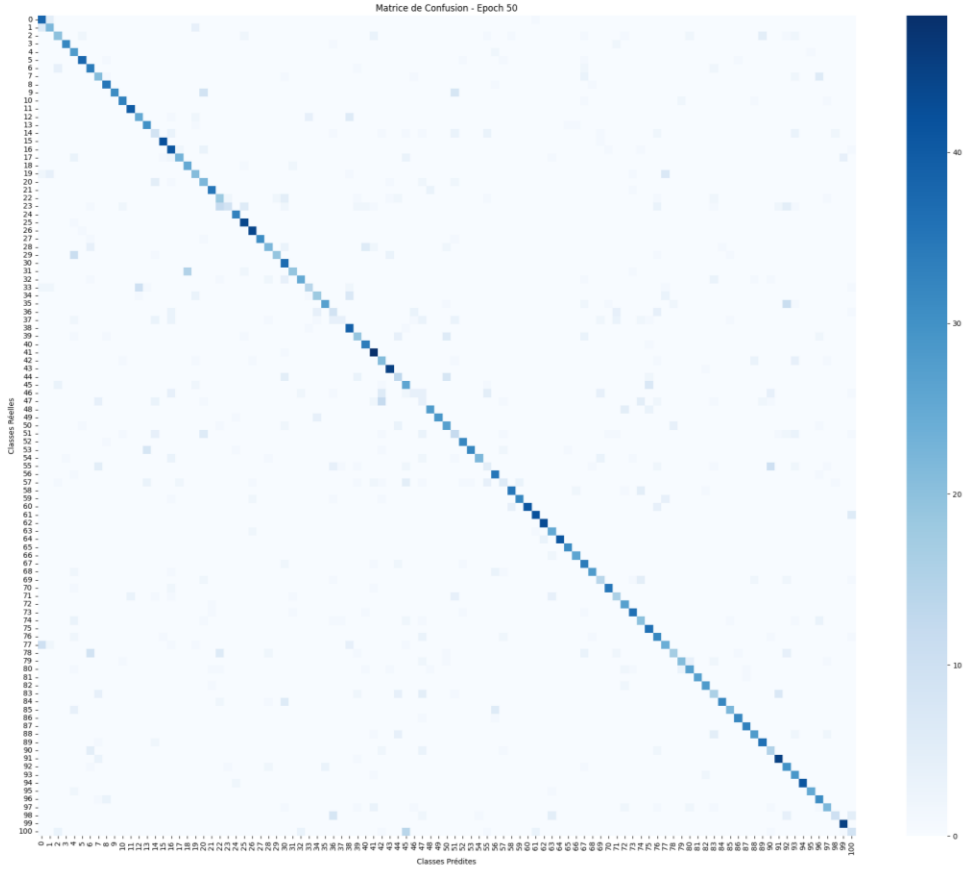



## CNN + LSTM

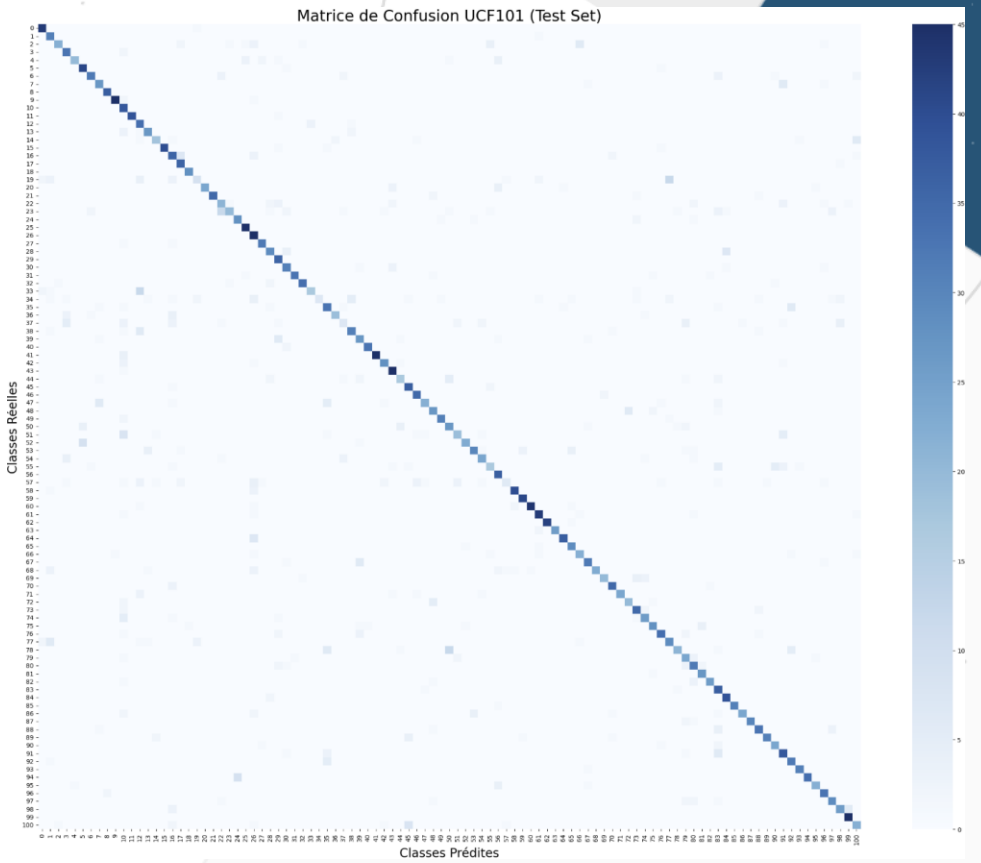


# Result 1 : confusion matrix for each model (2/2)

## ResNet50 + LSTM




## ResNet3D-18



# Result – Predict single video

**Label : Basketball**

Vidéo analysée : v_Basketball_g01_c01

| RANG | CLASSE | CONFIANCE |
|---|---|---|
| 1 | Punch | 1.23% |
| 2 | PlayingCello | 1.21% |
| 3 | ShavingBeard | 1.19% |
| 4 | PlayingGuitar | 1.19% |
| 5 | CricketShot | 1.18% |

**FC**

| RANG | CLASSE | CONFIANCE |
|---|---|---|
| 1 | Fencing | 36.68% |
| 2 | FloorGymnastics | 28.83% |
| 3 | Kayaking | 27.59% |
| 4 | Rafting | 4.41% |
| 5 | WalkingWithDog | 1.26% |

**CNN + LSTM**

Vidéo analysée : v_Basketball_g01_c01

| RANG | CLASSE | CONFIANCE |
|---|---|---|
| 1 | TennisSwing | 98.68% |
| 2 | Basketball | 1.04% |
| 3 | BasketballDunk | 0.15% |
| 4 | FieldHockeyPenalty | 0.05% |
| 5 | VolleyballSpiking | 0.02% |

**ResNet3D-18**

| RANG | CLASSE | CONFIANCE |
|---|---|---|
| 1 | Basketball | 99.59% |
| 2 | SoccerJuggling | 0.23% |
| 3 | VolleyballSpiking | 0.06% |
| 4 | TennisSwing | 0.04% |
| 5 | BodyWeightSquats | 0.01% |

**ResNet50 + LSTM**

# Live result with ResNet50 + LSTM


# Summary

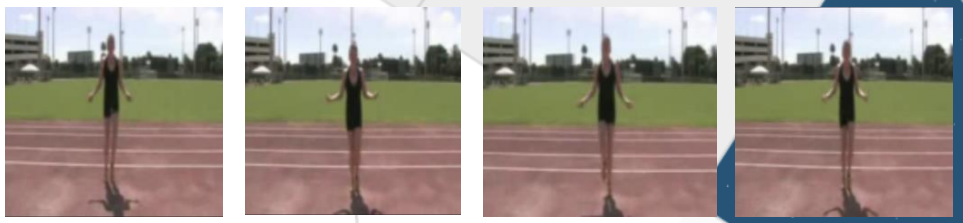
Whole project succesfully implemented the basic deep learning models for action recognitions.

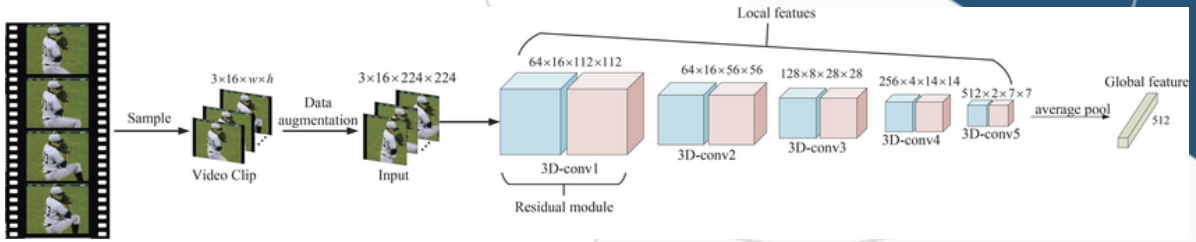
Four methods are implemted (FC, CNN + LSTM, Resnet + LSTM, and Resnet3D-18)

The best methods are Resnet3D, mainly that it is pretrained in relevant dataset (Kinetics-400)

Live inference shows the ability of the methods to do prediction.

To improve:
1. To use bigger methods, such as transformer
2. To use keypoints as the inputs (openpose)
3. To enhance the inference (Two Stream Fusion with 2 buffers pixels+keypoints)






| Methods | UCF 101 | |
|---|---|---|
| | Accuracy | F1 score |
| Fully connected | 1.14% | 0.0002 |
| CNN + LSTM | 25.67% | 0.2379 |
| ResNet50 + LSTM | 72.38% | 0.7072 |
| ResNet3D | 79.67 % | 0.7895 |

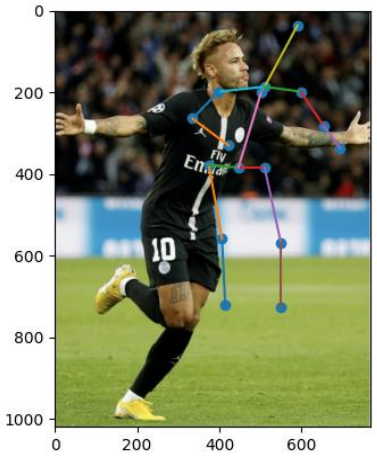
# What did we learn and challenges

**Topics:**
Keypoint prediction
Action recognition

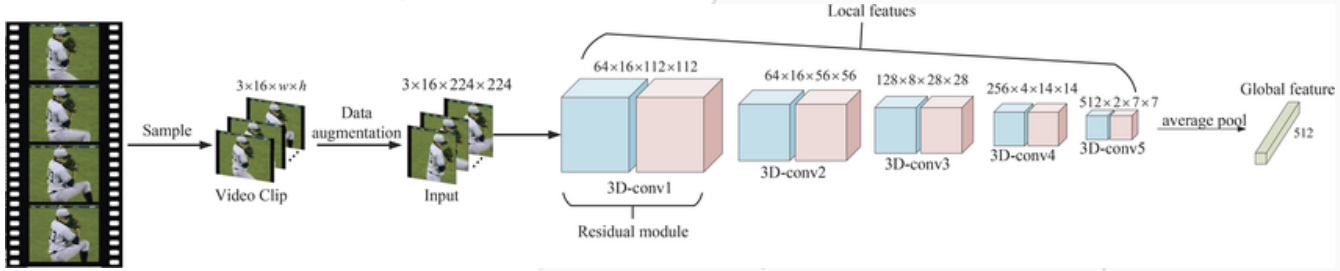
**Methods:**
FC, CNN+LSTM, RESNET50+LSTM, RESNET3D, Openpose (keypoints)

**Code management:** Access SSH server, install libraries, environements, GPU management

**Challenges :**
- Code management (beginning)
- Understand last year code (beginning)
- Modelling
- Technical limitations







# ANY QUESTIONS ?

# THANK YOU

- DJEDDOU Kaïs
- MAUREL Alexandre
- JEAN-MARIE Matisse
- KEMPF Louis