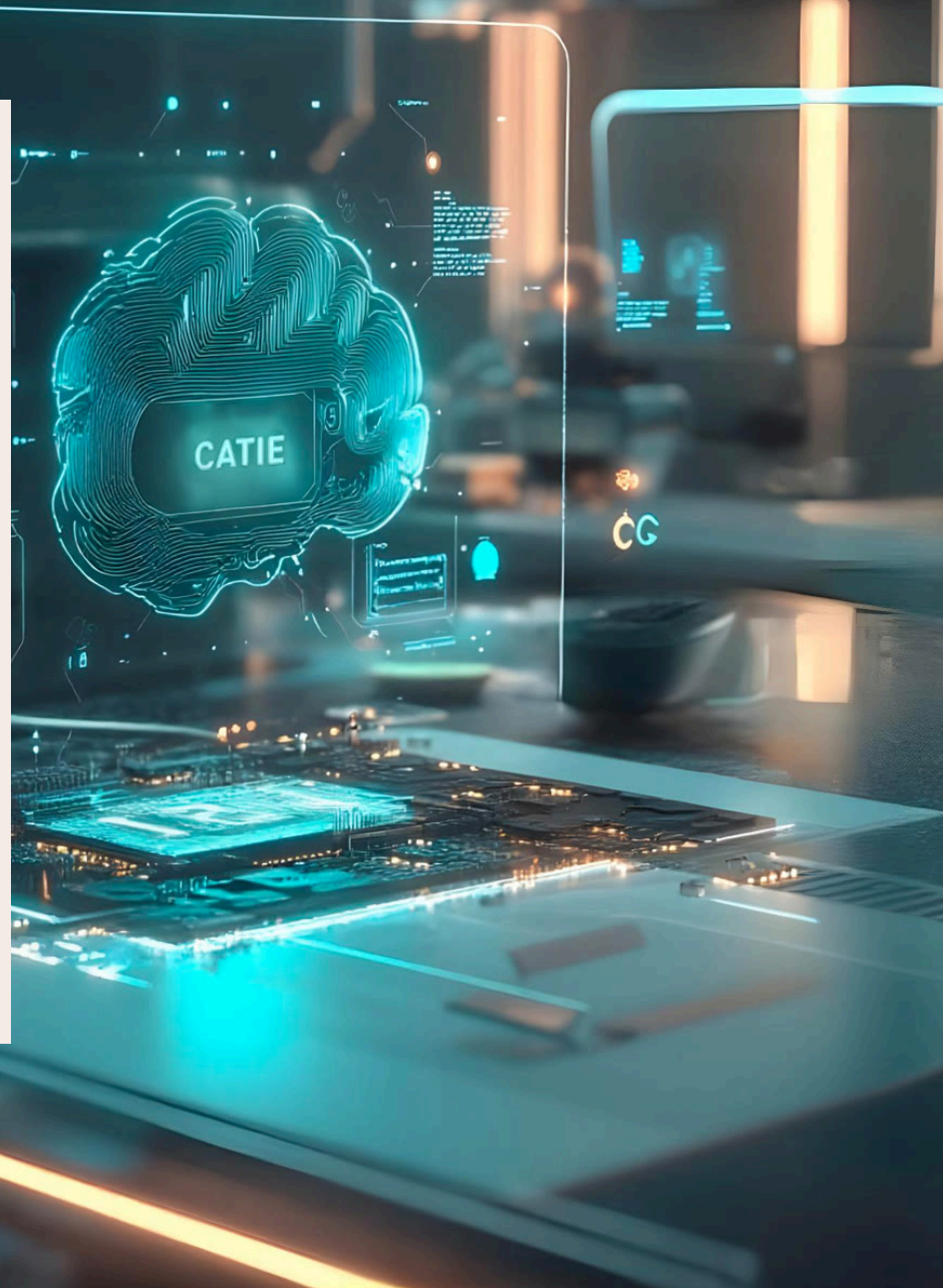




CATIE

4. Sémantique par IA

1/2. Les réseaux de neurones convolutifs



Point sur le programme

- 08/10 : 0. Introduction (SDE)
- 10/10 : 1. Acquisition (CPI)
- 14/10 : 2. Traitement (CPI)
- 15/10 : 2. 3. Traitement & Analyse (CPI)
- **24/10 : 4. Sémantique par IA 1/2 (SDE)**
- 28/10 - 03/11 : Vos Vacances
- 04/11 - 11/11 : Vacances CPI & SDE
- 12/11 : 2. Traitement (CPI)
- 18/11 : Extérieur - Proditec
- 19/11 : 2. 3. Traitement & Analyse (CPI)
- 25/11 : Extérieur - Aérospline
- 26/11 : 4. Sémantique par IA 2/2 (SDE)
- 28/11 : 3. Analyse (CPI)
- 02/12 : Extérieur - Aérospline
- 03/12 - 11/12 : Projet (CPI + SDE)

Sémantique par IA

Objectifs des deux séances

- Comprendre les limites des méthodes classiques d'analyse d'images.
- Comprendre les principes des réseaux de neurones convolutifs.
- Utiliser (intelligemment) des réseaux de neurones pré-entraînés.
- Comprendre les limites des réseaux de neurones.
- ⚠ **Ne pas devenir un expert en IA en 2 séances -> comprendre comment en être un bon utilisateur et avoir un regard critique sur les résultats.** ⚠

La mise en oeuvre

- Avec CPI : outils et méthodes d'acquisition, de traitement et d'analyse d'images -> constitution d'une boîte à outils.
- Avec SDE : ajout de l'extraction de sémantique par IA dans cette boîte à outils.
- Votre job pendant les 2 séances : mettre en place un pipeline d'analyse d'images complet sur un exemple concret utilisant les outils et méthodes vus en cours.
- Relève automatique des dépôts GitHub le 02/12 à 23h59 (fin du projet fil rouge, début du projet complet).
- ⚠ **Pas de soutenance pour ce projet fil rouge.** ⚠

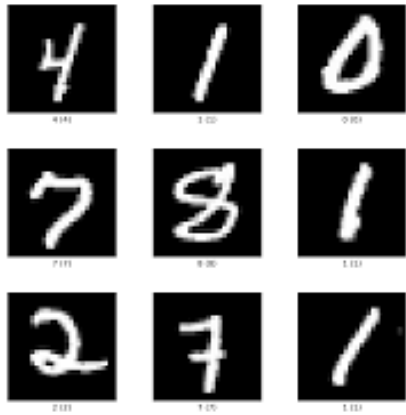
Informations obligatoires

- **README.md** :
 - explication de votre pipeline, des choix effectués, des résultats obtenus. (texte ou schéma, faites pas un roman non plus)
 - instructions pour lancer votre pipeline
 - critiquer les résultats obtenus (ce qui marche, ce qui ne marche pas, pourquoi)
- 1 module Python pour tout votre pipeline.
 - Module nommé **ts341**, script principal **pipeline.py**
 - **pip install .** doit fonctionner et suffire
 - **python3 -m ts341.pipeline** doit lancer votre pipeline (google est votre ami, contactez-moi si vous avez des soucis)

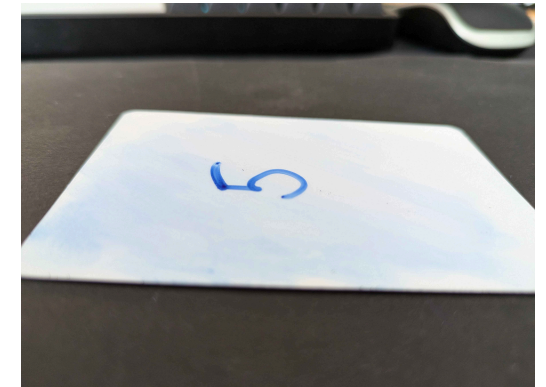
Critères d'évaluation

- Principaux :
 - Compréhension du besoin et traduction technique
 - Fonctionnalité du pipeline (absolument obligatoire)
 - Capacité à le distribuer à un tiers (CPI et moi)
 - Critique des résultats obtenus
- Secondaires :
 - Qualité du code (PEP8, PEP257, PEP484, ... -> **black**, **pylance** etc)
 - Intelligence du pipeline (à quel point vos choix sont pertinents et efficaces)
- Bonus :
 - Si vous essayez de containeriser, d'intégrer des tests unitaires, de la CI/CD et que c'est bien fait et pertinent.
 - ⚠ **Pas de bonus si le pipeline est médiocre.** ⚠

Sujet du projet fil rouge

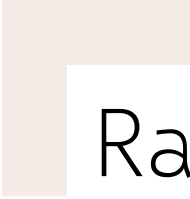


- Réaliser un pipeline de détection des chiffres manuscrits
- Basé sur le jeu de données MNIST
- Non limité à MNIST, ajout de données supplémentaires (écrit à la main, par un enfant etc)



Des questions ?

Liste des outils déjà vus en cours



Rappel sur les Réseaux de Neurones

Rappels des Réseaux de Neurones

- **Système de neurones** : Plusieurs couches connectées.
- **Problème de classification** : Calcul de scores pour chaque classe.
- **Transformation des données** : Combinaison linéaire + fonction d'activation.
- **Empilement de couches** : Sortie d'une couche = entrée de la suivante.
- **Fonction différentiable** : Sortie finale du réseau.

Réseaux de Neurones Convolutifs

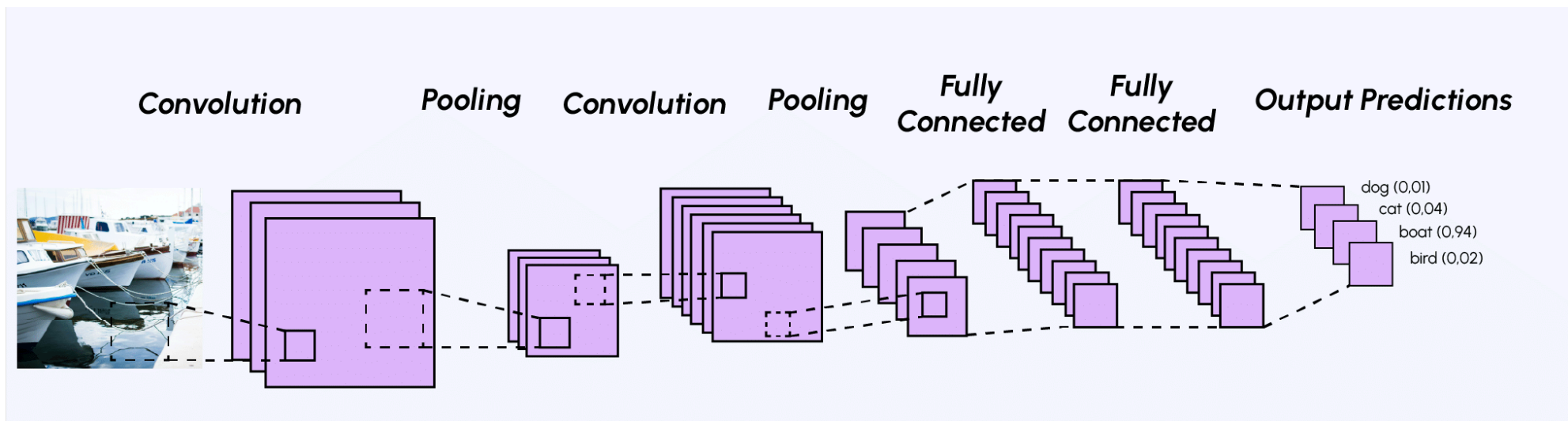
Différences avec les Réseaux de Neurones

- **Sous-catégorie des réseaux de neurones**
- **Traitement d'images en entrée**
- **Architecture spécifique : deux blocs principaux**

Toujours 2 étapes : extraction de features puis classification.

Les couches à connaître

- **Convolution** : Application de filtres sur l'image.
- **Pooling** : Réduction de la taille de l'image.
- **ReLU** : Fonction d'activation.
- **Fully Connected** : Couche classique de réseaux de neurones.



Détails de fonctionnement des CNN

- Voir avec Frédéric Alexandre si vous voulez plus de détails.
- Objectif du cours : comprendre comment les utiliser intelligemment, pas comment les construire.

Un peu de concret

Utilisation de CNN pré-entraînés

- **Réseaux de neurones déjà entraînés sur des jeux de données**
- **Utilisation de ces réseaux pour des tâches spécifiques**
- **Fine-tuning : ré-entraînement partiel du réseau**
- **Transfer Learning : réutilisation de couches pré-entraînées**
- **Exemple : VGG16, ResNet, Inception, MobileNet, YOLO, ...**

YOLO

- Pour ce cours : utilisation de YOLO
- **You Only Look Once**
- **Réseau de neurones pour la détection d'objets**
- **Très rapide et précis**
- Site officiel

Utilisation de YOLO

Dans les dernières versions de YOLO (v11) il est capable de :

- Classifier
- Détecter
- Segmenter
- Suivre
- Déterminer la position d'un corps

Utilisation “simple” de YOLO

- Charger un modèle pré-entraîné
- Charger une image
- Faire une prédiction

Utilisation “complète” de YOLO

- Charger un modèle pré-entraîné
- Charger un jeu de données
- Faire un fine-tuning (ie ré-entraînement partiel sur un jeu de données spécifique)
- Faire une prédiction

Comment faire un fine-tuning ?

- Travail de labélisation (c'est long, c'est chiant, c'est nécessaire) : **labelstudio**
- Entrainement des dernières couches du réseau sur un jeu de données spécifique
- Evaluation des performances

Exercice 1 : Utilisation simple de YOLO

- Installer YOLO sur votre machine (package `ultralytics`)
- Chargez `yolo11n.pt` et faites un train avec `mnist`
- Inférez sur quelques images de `mnist`

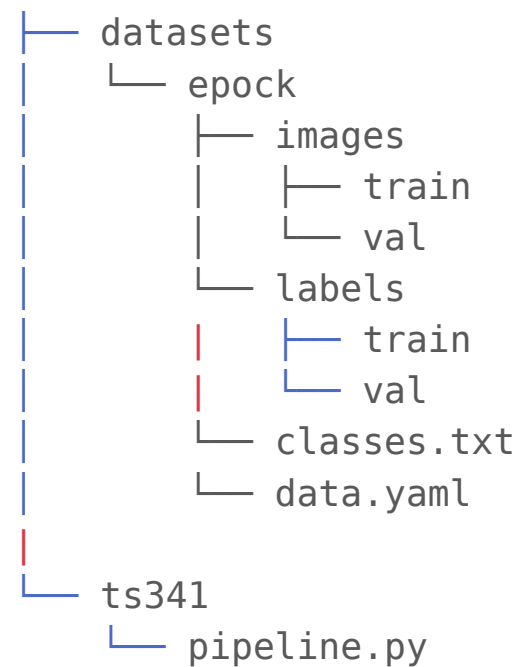
Exercice 2.1 : Labeliser des données (pas trop non plus)

- Téléchargez la vidéo : https://youtu.be/qH_8ZUGkEEA?list=PLeTQsjMI20jGFJOJUNOz3le4LbhH_F-hT
- Utilisez **labelstudio** pour labeliser Epock (le robot)

Téléchargez le dataset

- Format : yolo

ts341



Exemple de fichier data.yaml

```
path: /path/to/ts341/datasets/epock  
train: 'images/train'  
val: 'images/val'
```

```
# class names
```

```
names:
```

```
  0: Epock
```

```
  1: Object
```

Exercice 2.2 : Fine-tuning de YOLO

- Chargez **yolo11s.pt** et faites un fine-tuning sur vos données labelisées pour détecter Epock et un objet quelconque.
- Inférez sur quelques images sur vos images