HANDS ON AI – Certificat IA de l'UMONS

Défi 1 : IA et analyse d'images : détection et localisation sur images

Protocole du défi 1

I. Introduction :

L'objectif de ce défi est de classifier des images, détecter et localiser des objets dans des images capturées à partir de caméras en utilisant des réseaux de neurones profonds. Le but de ce défi se résumé en trois points :

1. Maîtriser les outils de collecte, préparation et annotation de données nécessaires pour l'entraînement ;
2. Développer et exploiter des architectures DNN de classification d'images et de localisation d'objets ;
3. Analyser et évaluer et expliquer « XAI » les résultats des architectures neuronales profondes.

Durant ce défi, les étudiants travailleront sur des exemples concrets tels que la détection des feux de forêts (problème de classification d'images) et la localisation d'objets personnels (problème de localisation d'objets).

II. Environnement de travail :

Pour ce défi, nous vous proposons de travailler avec les outils de développement suivants :

a. **Python [1]:** langage de programmation interprété, multiparadigme et multiplateformes ;

b. **OpenCV [2]:** bibliothèque de traitement d'images et de vidéos ;

c. **Keras [3]:** bibliothèque permettant d'interagir avec les algorithmes de Machine Learning et réseaux de neurones profonds et de machine ;

d. **Google Colab [4]:** outil simple et gratuit utilisant un environnement de développement (Jupyter Notebook) qui ne nécessite aucune configuration. Google Colab permet de tester des applications en Deep Learning en utilisant des bibliothèques Python populaires telles que Keras, TensorFlow, PyTorch et OpenCV. Google Colab offre également la possibilité d'utiliser gratuitement un processeur graphique GPU à distance ;

[1] Python. https://www.python.org/
[2] OpenCV. https://opencv.org/
[3] Keras. https://keras.io/
[4] Google Colab. https://colab.research.google.com

III. Enoncé :

Ce défi consiste à réaliser un programme capable de classifier des images et de localiser les objets en utilisant les réseaux de neurones profonds (Deep Learning). Le défi comporte trois parties : classification d'images, localisation d'objets, traitement temps réel.

Figure 1: exemples d'images de feu, pas de feu et début de feu (fumée)

III.1. Classification d'images : la première partie de ce défi consiste à développer un classifieur utilisant les réseaux de neurones profonds pour la détection de feux de forêts à partir d'images de la caméra. Le classifieur devra identifier trois classes : feu, pas de feu, début de feu (fumée) (Figure 1). Pour réaliser ce classifieur, il faudra réaliser les trois étapes suivantes :

a. **Préparation et annotation de données :** avant de lancer les entrainements, il faudra préparer et annoter les données afin de fournir trois sous ensemble d'images : feu, pas de feu, début de feu. Afin de faciliter le démarrage de votre travail, nous vous proposons trois bases de données déjà annotées (DB1, DB2 et DB3). Bien évidemment vous pourrez augmenter ces bases de données avec vos propres images. Pour cela, vous pouvez utiliser le programme « ***extract_from_frames.py*** » partagé via Moodle et qui permet d'extraire des images à partir de séquences vidéo.

b. **Entraînement des modèles :** une fois que votre base de données est annotée, vous pouvez commencer le développement de votre classifieur qui peut se faire de deux manières différentes :

- Développer votre propre architecture de réseaux de neurones

- Utiliser une architecture de classification déjà existante (VGG, Inception, Xception, ResNet, etc.) avec ou sans transfert des poids (modèles pré-entrainés avec ImageNet par exemple). Cette option est fortement conseillée pour le développement de votre premier classifieur.

  Les entrainements de modèles peuvent être réalisés avec la plateforme **Google Colab.**

c. **Test et évaluation du modèle :** une fois que votre entrainement de données est finalisé, vous pouvez tester le modèle résultant avec les données de test, partagées via Moodle, afin d'évaluer votre modèle. Bien évidemment, l'objectif est d'avoir le meilleur taux de précision en détectant au mieux les trois types d'images.

- **NOTE IMPORTANTE :** un code de démarrage « ***input_classification.ipynb*** » est fourni pour le développement de ce classifieur, il faudra bien l'analyser et le comprendre avant de le compléter et le tester avec les différentes bases de données et architectures de classification (ou votre architecture). Il faudra veiller à l'analyse des hyper paramètres pour obtenir des résultats satisfaisants.

III.2. Localisation d'objets : la deuxième partie de ce défi consiste à développer un modèle pour la localisation d'objets sur images. Plus particulièrement, vous travaillerez sur le développement d'un réseau de neurones profond pour la localisation de clés à partir d'images de la caméra. Une application utile pour beaucoup de gens tels que les personnes atteintes de la maladie d'Alzheimer. Le modèle permettra de détecter la présence ou non d'une clé (ou un trousseau de clés) dans une image. Si l'image présente une ou plusieurs clés (ou trousseaux), il faudra les entourer en fonction de leurs positions. Pour réaliser ce classifieur, il faudra réaliser les trois étapes suivantes :

d. **Préparation et annotation de données :** là aussi, il faudra lancer les entraînements sur des données avec une différence importante lors de l'annotation des données en fournissant :
   - Les images présentant des clés/trousseaux de clés ainsi que d'autre objets ;
   - Le ou les labels (nom de classes) d'objets présents dans chaque image ;
   - Pour chaque objet, il faudra fournir sa position ainsi que sa taille.

Comme point de départ, nous vous fournissons une base de données déjà annotée. Bien évidemment vous pourrez l'augmenter avec plus de classes et images à annoter par vos soins. Pour annoter vos images, vous pouvez utiliser différents outils d'annotations tels que : Roboflow, labelme, labelimg, supervisely, etc.

e. **Entraînement des données :** une fois que votre base de données est annotée, vous pouvez commencer le développement de votre classifieur qui pourra s'appuyer sur des architectures de localisation d'objets pré-entrainées (Yolo V3, Yolo V4, Yolo V5, Faster R-CNN, SSD, etc.). La technique de Transfert Learning est fort conseillée en vue de réduire les temps de calcul et d'améliorer la précision des résultats.

Les entrainements de modèles peuvent être réalisés avec la plateforme **Google Colab.**

f. **Test et évaluation du modèle :** une fois que votre entrainement de données est finalisé, vous pouvez tester le modèle résultant avec les données de test, partagées via Moodle, afin d'évaluer votre modèle. Là aussi, l'objectif est d'avoir le meilleur taux de précision en localisant au mieux les clés ou trousseaux de clés.

- **NOTE IMPORTANTE :** un code de démarrage « ***input_localisation.ipynb*** » est fourni pour le développement de ce modèle, il faudra bien l'analyser et le comprendre avant de le compléter et le tester avec les différentes bases de données et architectures de localisation (ou votre architecture). Il faudra veiller à l'analyse des hyper paramètres pour obtenir des résultats satisfaisants.

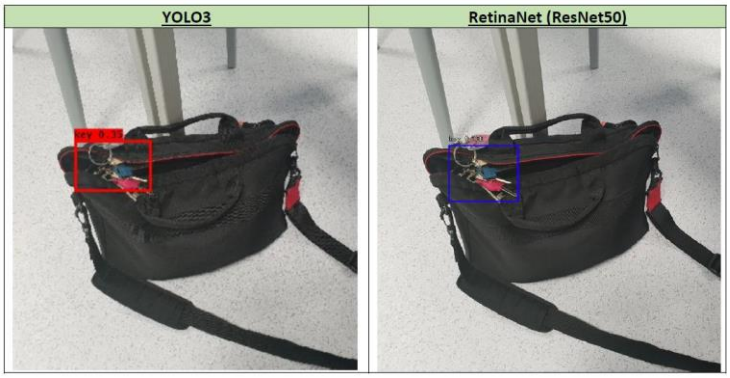

Figure 2: exemple de trousseaux de clés localisés sur image (architectures Yolo V3 et RetinaNet)

III.3. Traitement temps réel : la troisième partie de ce défi consiste à exploiter les deux modèles décrits ci-dessus pour le traitement d'une séquence vidéo en temps réel. Le traitement à appliquer pour chaque image de la vidéo serait :

- Pour une vidéo filmée en forêt : détecter la présence d'un feu ou un début de feu (fumée) ;
- Pour une vidéo filmée chez soi : détecter la présence d'une clé ou un trousseau de clés avec encadrement sur la bonne position.

III.4. Explicabilité et interprétabilité de la solution : cette partie de ce défi consiste à expliquer les modèles Deep Learning de classification d'image forestières. Vous pouvez utiliser les technique d'explicabilité « **XAI** » permettant d'identifier les pixels responsables de chaque de décision. Dans notre cas, pour chaque détection de feu par exemple, la méthode d'explicabilité permettra d'identifier, grâce à une carte de couleurs, les parties de l'images (pixels) ou il y eu un départ de feu (pixels responsables de la décision du modèle).

Pour développer cette partie « facultative », vous pouvez utiliser la libraire « tf-explain ». Un exemple de son utilisation sera donné durant la séance 3 de ce défi (27/10).

III.5. Les Vision Transformes (ViT) pour la classification d'images forestières : l'objectif de cette partir est de découvrir, explorer les réseaux de type « Vision Transformers » pour classifier les images forestières. Il faudra présenter la différence de fonctionnement (entre CNN et Vit) avant de comparer et interpréter les résultats. Pour réaliser cette partie, nous invitons à utiliser cette solution : https://paperswithcode.com/paper/an-image-is-worth-16x16-words-transformers-1#code

Vous pouvez également vous s'appuyer sur d'autres solutions en fonction de votre recherche.

IV. Mode d'évaluation :

Afin d'évaluer votre travail, chaque groupe devra remettre via Moodle les éléments suivants :

- Rapport : description de la solution proposée, justification des choix, interprétation des résultats ;
- Code source et données : liés aux trois parties : codes de tests, modèles entrainés et données ;
- Manuel d'utilisation : un petit manuel décrivant comment peut-on tester les codes fournis.

L'évaluation de vos travaux se fera via trois critères : qualité du rapport et manuel d'utilisation (20%), qualité de codes et de données (20%), taux de classification, (40%) et localisation (20%) obtenus avec les images de test.

- **Note :** les troisième et quatrième partie sont est facultatives mais permet d'avoir un bonus lors de l'évaluation (+ 20%).