TP4

Activité 1

Dans cette activité, nous allons réaliser une détection des objets dans une vidéo à l'aide de **Faster RCNN** en utilisant **ResNet** comme Backbone,

Prérequis

```
pip install torch torchvision matplotlib
```

Code

```
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import os
import torch
from torchvision.models.detection import fasterrcnn_resnet50_fpn
from torchvision.transforms import functional as F
from PIL import Image
# Charger le modèle Faster R-CNN avec ResNet-50 comme backbone (pré-entraîné sur COCO)
model = fasterrcnn_resnet50_fpn(pretrained=True,weights='DEFAULT') #COCO_V1
model.eval()  # Mode évaluation
os.environ["KMP_DUPLICATE_LIB_OK"] = "TRUE"
# Charger une image d'exemple
image_path = '../../images/image2.jpeg'  # Remplacez par le chemin de votre image
image = Image.open(image_path).convert("RGB")
# Transformer l'image pour correspondre au format d'entrée attendu par le modèle
image_tensor = F.to_tensor(image)  # Convertit l'image en tenseur
images = [image_tensor]  # Le modèle attend une liste d'images
# Faire une prédiction
with torch.no_grad():
    predictions = model(images)
```

```
# Afficher les prédictions
boxes = predictions[0]['boxes']
labels = predictions[0]['labels']
scores = predictions[0]['scores']
# Afficher l'image et les boîtes
fig, ax = plt.subplots(1, figsize=(12, 9))
ax.imshow(image)
# Ajouter les boîtes englobantes
for idx, box in enumerate(boxes):
   if scores[idx] > 0.8:  # Afficher uniquement les prédictions de haute confiance (score > 0.5)
      x1, y1, x2, y2 = box
      width = x2 - x1
      height = y2 - y1
      # Créer un rectangle pour chaque boîte englobante
      rect = patches.Rectangle((x1, y1), width, height, linewidth=2, edgecolor='red',
facecolor='none')
      ax.add_patch(rect)
      # Ajouter l'étiquette et le score
      ax.text(x1, y1 - 10, f'{labels[idx].item()} ({scores[idx]:.2f})', color='red', fontsize=12,
backgroundcolor='white')
plt.axis('off')
plt.show()
```

1. Modifier le Backbone en utilisant VGG19, EfficientNet, Inception et Vision Transformers (ViT).

Activité 2

Utiliser ResNet pour détecter des objets dans une vidéo.

```
import torch
import cv2
from torchvision.models.detection import fasterrcnn_resnet50_fpn
from torchvision.transforms import functional as F
# Charger le modèle Faster R-CNN avec ResNet-50 comme backbone
```

```
model = fasterrcnn_resnet50_fpn(pretrained=True,weights='DEFAULT')
model.eval()  # Mettre le modèle en mode évaluation
def draw_boxes(image, boxes, labels, scores, threshold=0.5):
    for i in range(len(boxes)):
        if scores[i] > threshold:
            box = boxes[i]
            x1, y1, x2, y2 = map(int, box)
            cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
            label = f"{labels[i]}: {scores[i]:.2f}"
            cv2.putText(image, label, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0),
2)
    return image
# Charger la vidéo d'entrée
input_video_path = 'test.mp4'
output_video_path = 'output_video_with_detections.mp4'
cap = cv2.VideoCapture(input_video_path)
# Obtenir les propriétés de la vidéo
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
fps = int(cap.get(cv2.CAP_PROP_FPS))
# Configurer la vidéo de sortie
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(output_video_path, fourcc, fps, (frame_width, frame_height))
# Boucle pour lire chaque frame de la vidéo
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
    # Convertir l'image en tensor pour le modèle
    image_tensor = F.to_tensor(frame).unsqueeze(0)
    # Faire des prédictions
```

```
    with torch.no_grad():
        predictions = model(image_tensor)
    # Extraire les boîtes, les scores et les labels
    boxes = predictions[0]['boxes'].cpu().numpy()
    scores = predictions[0]['scores'].cpu().numpy()
    labels = predictions[0]['labels'].cpu().numpy()
    # Dessiner les boîtes englobantes sur la frame
    frame = draw_boxes(frame, boxes, labels, scores, threshold=0.5)
    # Afficher ou sauvegarder la frame avec les détections
    out.write(frame)
    cv2.imshow('Video Detection', frame)
    # Quitter la boucle si 'q' est pressé
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# Libérer les ressources
cap.release()
out.release()
cv2.destroyAllWindows()
```

1. Tester d’autres Backbone