Rapport Technique: DeepTracel - Ziyad TAIA-ALAOUI

Données natives:

L'implémentation de MOTRv2 réalisée au cours de ce projet respecte les règles d'un MOTChallenge en 2D avec bounding boxes (4 coordonnées (x,y)) autour des cellules trackées au cours du temps. Le dataset fourni par l'équipe de recherche contient une séquence de 168 images annotées dans un format natif contenant les informations de segmentation par le biais de 64 coordonnées 2D formant le contour de chaque cellule.

Format natif:

- Fichiers images.mat et masks.mat contenant un ensemble de 168 images annotées avec l'encodage suivant: {Id_cell, x1_cell, y1_cell, …., xN_cell, yN_cell}, avec autant de lignes que de cellules dans les fichiers d'annotations.

Données pré-traitées par Ziyad

- Format MOTChallenge:

```
MyDataset/
  train/
    seq1/
      img1/
        000001.jpg
        000002.jpg
        ...
      gt/
        gt.txt
    seq2/
      img1/
        000001.jpg
        000002.jpg
        ...
      gt/
        gt.txt
```

- Il y a trois images set de 100, 34, et 34 images respectivement pour les sets de: train, val, et test.

- Les fichiers d'encodage sont enregistrés dans le sous-dossier gt au nom de gt.txt. L'encodage tient compte de plusieurs informations: `<frame>, <id>, <bb_left>, <bb_top>, <bb_width>, <bb_height>, <conf>, <x>, <y>, <z>` (https://motchallenge.net/instructions/)
    - `<frame>` → numéro d'image dans la séquence
    - `<id>` → Id de cellule das la frame actuelle
    - `<bb_left>, <bb_top>`: coordonnées X et Y du coin supérieur gauche de la boîte englobante (bounding box).
    - `<bb_width>, <bb_height>:` largeur et hauteur de la bounding box.
    - `<conf>`: indicateur de confiance (1 = vérité terrain valide, 0 = ignorer, X = valeur calculée par détecteur, ex. YOLOX ou personnalisé comme ce qui a été réalisé dans ce travail à partir du fichier gt).
    - `<x>, <y>, <z>`: coordonnées cartésiennes des cellules, généralement ignorées dans les MOTChallenge 2d et mis à leur valeur par défaut: -1, -1, -1.

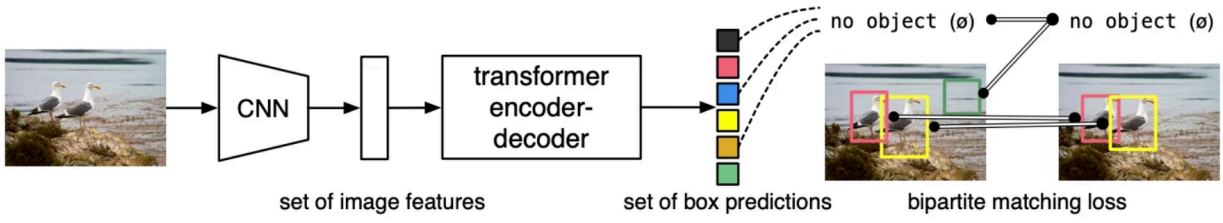
Description du modèle retenu:

Macro-Archi:

“ The first layer processes the input image using a CNN backbone. It produces features for the input image. These feature maps are then fed into the transformer layer to generate **bounding box, class, and class confidence scores** for a **fixed number of predictions** ”

Number of detectable objects:

“This number, N, is chosen to be much larger than the typical number of objects in an image” Labels = Ids + “No-object” Id

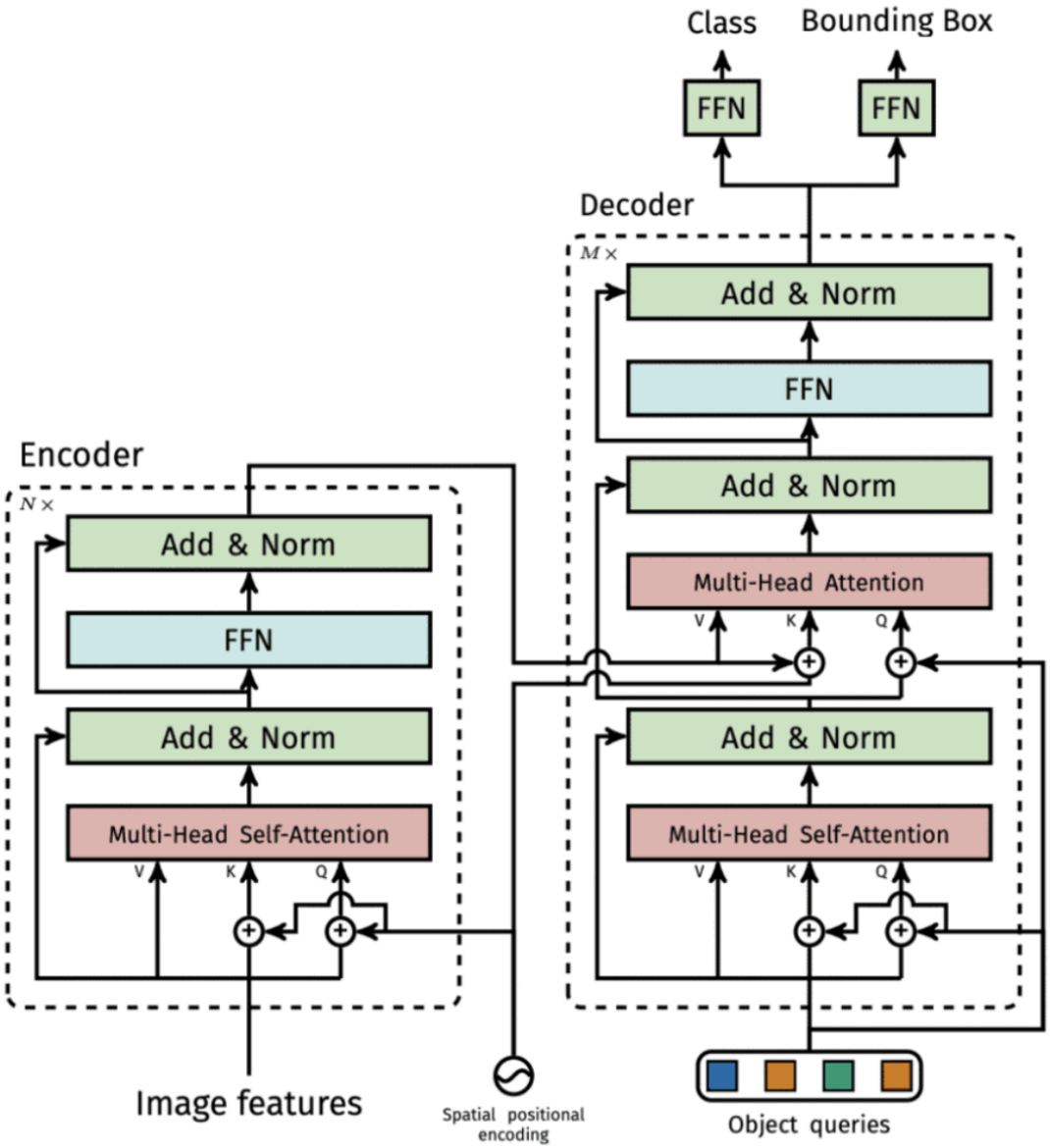


DETR architecture

CNN Layer

“ DETR paper uses ResNet-50 and ResNet-101.”

Transformer



Transformer Layer in DETR

- **Decoder Input**: The embeddings input into the decoder are learnable parameters referred to as object queries ⇒ Queries = Decoder Input.
- **Decoder Output**: The output of the transformer decoder is then independently decoded into box coordinates and class labels.