Assignment 2
COL828: Advanced Computer Vision
Semester I, 2024-2025.
Due Date: Nov. 25, 2024

October 14

# Implementing the DETR Model for Object-Detection from Scratch

## 1 Objective

The goal of this assignment is to deepen your understanding of modern object detection techniques by implementing the DETR (**DE**tection **TR**ansformer) model from scratch. You will use a custom dataset to detect and classify different bone-fractures in x-ray images provided here.

## 2 Dataset

The dataset contains images categorized into different classes, each representing a specific type of bone fracture. These classes include *Elbow Positive*, *Fingers Positive*, *Forearm Fracture*, *Humerus Fracture*, *Shoulder Fracture*, and *Wrist Positive*.
The dataset is divided into train/val/test splits each containing 3361/348/169 images respectively. The corresponding annotations for each split is provided in COCO-format.
The annotation file typically follows a hierarchical structure to store bounding boxes, with the following key components:

- Images: This section contains information about each image, including its unique identifier, file name, height, width, and any other relevant metadata.
- Annotations: This section provides details about the annotations for each object instance present in the image. Each annotation entry includes the following fields:

- "id": A unique identifier for the annotation.
- "image_id": The identifier of the image to which the annotation belongs, linking it to the corresponding image entry in the "images" section.
- "category_id": The category label of the annotated object.
- "bbox": The bounding box coords of the annotated object, represented as [x, y, width, height]. Here, (x,y) denotes the top-left corner of the bounding-box, and width and height represent its dimensions.

# 3 Experiments

- [**E1**]: implement the DETR model architecture with the following key-components:
  - **CNN-Encoder**: Any pre-trained resnet of your choice.
  - **Transformer Encoder-Decoder**: Implement the transformer architecture as specified in the [1]. You can modify the number of encoder/decoder blocks to make training efficient. We recommend at-least 3 transformer block per encoder/decoder. This involves implementing classes for all the necessary components in a transformer block i.e **Self-Attention**, **Cross-Attention**, **FFNs**, etc.
  - **Position Encoding**: : Implement positional encoding to ensure that spatial information is preserved during the transformer operations.
  - **Object Queries**: Use learnable object queries in the transformer decoder to predict the objects in the images. **Hint:** Analyze the data and figure out the optimal number of object-queries.
  - **Loss Function**: This is a crucial component in DETR training. Read about Hungarian Matching and implement the set-matching loss as described in [1]

  We recommend following practices similar to [1] such as augmentations and other training details. Transformer based models require large number of training iterations, we recommend training this model for $\geq 100$ epochs.

- [**E2**]: Use a pre-trained DETR model https://huggingface.co/facebook/detr-resnet-50 and fine-tune on the given dataset.

Compute the mAP score on both the validation and test set. Also report Precision and Recall for your model.

# 4 Submission Requirements

1. Code
    - Your code should be well-structured, modular, and properly documented. You should provide clear explanations for each module and function in your implementation.
    - Include your training script as well as other utility files in the code.
2. Report: The following results should be presented in a concise report:
    - Any design choices or modifications made to DETR .
    - Training strategy and hyperparameters used.
    - Results: include both quantitative metrics (mAP, Precision, Recall) and qualitative results (visualizations).

# References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020.