Objective

In this assignment, you will fine-tune a DETR (Detection Transformer) model for balloon detection using a small dataset. You will learn about:

1. Working with limited data in object detection tasks
2. Implementing data augmentation techniques
3. Fine-tuning transformer-based object detection models
4. Evaluating model performance against a baseline

Dataset

The dataset consists of 74 images containing balloons, along with their corresponding bounding box annotations. The small dataset size presents an interesting challenge that you'll need to address through various techniques.

V2 Balloon Detection Dataset | Kaggle

Part 1: Data Preparation (20 points)

Load the dataset from Kaggle and do all necessary preparations, mainly:

1- Data Loader: it is recommended to use `torchvision.datasets.CocoDetection`

2- Data Split: consider using 70% of the data for training, 15% for validation, and 15% for testing.

```
# Set random seeds for reproducibility
torch.manual_seed(42)
np.random.seed(42)
random.seed(42)
torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
```

Part 2: Data Augmentation (15 points)

Try to utilize data augmentation techniques to cover the small size of the training data. For that you might find `Albumentations` library to be useful.

Part 3: Model Fine-tuning (30 points)

1. Load the pre-trained DETR model:

```
from transformers import DetrForObjectDetection, DetrImageProcessor

processor = DetrImageProcessor.from_pretrained("facebook/detr-resnet-50")
model = DetrForObjectDetection.from_pretrained("facebook/detr-resnet-50")
```

2. Implement the training loop with the following requirements:
    - Use AdamW optimizer
    - Implement learning rate scheduling
    - Save best model based on validation performance
    - Track and plot training metrics

Part 4: Evaluation and Analysis (20 points + 15 points)

1. Evaluate your fine-tuned model using:
    - Average Precision (AP): You should achieve an AP@0.5 score of at least 0.6.
    - Qualitative analysis of predictions
2. Analyze the impact of data augmentation:
    - Compare performance with and without augmentation
    - Discuss which augmentation techniques were most effective

Deliverables

- A .zip file containing a directory named after your unique name (i.e. your_name_assignment_01.zip) with the structure:
  1. Code implementation (Jupyter notebook) - with all necessary instructions on how to run.
  2. Model checkpoint
  3. Training logs and visualizations (consider using `Tensorboard`)

- A .pdf report (max 3 pages) that includes:
  1. Methodology description
  2. Optional: Implementation details (if you think that could help us understand your solution)
  3. Results analysis
  4. Discussion of challenges and solutions