CO2.2 - Final Project (RF-DETR for Object Detection)

CSS182-4_AM4

Group 23

Nathan Joseph L. Perez

Project Overview & Resources

Dataset Link:
https://universe.roboflow.com/visocomputacional/datasetships
Notebook Link:
https://colab.research.google.com/drive/10ji04MgUedX5oWHAezfydZwdn0MLJcw8?usp=sharing
Github Link:
https://github.com/Natj02/RF-DETR-for-Object-Detection.git

Table Of Contents

Introduction

Object detection plays a critical role in computer vision applications, enabling systems to localize and classify objects within an image. Recent advancements in transformer-based architectures, such as DETR (DEtection TRansformer), have significantly improved the accuracy and efficiency of object detection models by eliminating the need for handcrafted anchor boxes and non-maximum suppression. This project investigates RF-DETR, an advanced variant of DETR that incorporates Receptive Field Enhancement (RFE) modules to improve the model's ability to capture multi-scale spatial information. By expanding the receptive field, the model becomes more adept at recognizing objects of various sizes and contextual relationships within an image, particularly useful for tasks involving aerial or maritime imagery.

The model is trained using a balanced, ten-class ship detection dataset, formatted in COCO JSON and sourced through Roboflow. All training and evaluation processes are implemented in PyTorch, with specific hyperparameters chosen to ensure stable convergence over 50 epochs. The dataset includes vessel types such as tankers, container ships, and yachts, making it ideal for assessing the model's capacity to generalize across categories.

This project aims to demonstrate the effectiveness of RF-DETR in real-world object detection scenarios and provide insights into how receptive field enhancements contribute to improved recognition performance.

Objectives

1. **Model Implementation**
   Implement the **RF-DETRMedium** model with a CNN backbone, Transformer encoder-decoder, and RFE modules using PyTorch.
2. **Training on Custom Dataset**
   Train the model using a COCO-format dataset (from Roboflow) with appropriate hyperparameters and augmentation.
3. **Performance Evaluation**
   Evaluate model performance using:
   - **COCO metrics**: Average Precision (AP) and Average Recall (AR) at multiple IoU thresholds.
   - **Classification accuracy**: Correctness of predicted object categories compared to ground truth after bounding box matching.
4. **Metric Reporting**
   Present performance metrics in tabular and graphical format for interpretability.

Dataset Description and Preprocessing

The dataset used in this study is the "DatasetShips" sourced from Roboflow. It consists of 10 ship classes, each with exactly 500 labeled images, resulting in a total of 5,000 samples. The dataset is structured in COCO format, suitable for object detection tasks. The dataset used in this study consists of images categorized into ten distinct ship types:

Table 1: Ship Class Distribution

Class Name	Count
Bulk Carrier	500
Container Ship	500
General Cargo	500
Oil Products Tanker	500
Passengers Ship	500
Tanker	500
Trawler	500
Tug	500
Vehicles Carrier	500
Yacht	500

Each class is represented with exactly 500 labeled images, ensuring a balanced distribution across all categories. This uniformity aids in preventing class imbalance issues during training and evaluation.

The dataset was exported in COCO JSON format, structured with separate directories for training, validation, and test sets. All images were resized and preprocessed to meet the input requirements of the RFDETRMedium model, with a typical resolution of 512×512 pixels. Importantly, no data augmentations were applied during preprocessing or training. This decision was intentional to assess the model's raw performance on the original dataset without introducing variability from synthetic transformations. All input images were normalized to conform to the expected distribution of the model's convolutional backbone (e.g., ResNet). This involved either scaling pixel values to the [0, 1] range or applying mean-std normalization using standard ImageNet statistics.

RF-DETR Model Details (Screenshots of code)

Training Code

```
model = RFDETRMedium()

model.train(
   dataset_dir="/content/DatasetShips-1",
   epochs=50,
   batch_size=16,
   grad_accum_steps=4,
   lr=5e-5,
   output_dir="./rfdetr_output",
   tensorboard=True
)
```

Inference and Visualization of Predictions

```
# How many images to show
NUM_IMAGES = 12
TEST_DIR = "/content/DatasetShips-1/test"

# Pick random test images
image_paths = random.sample([
   os.path.join(TEST_DIR, f)
   for f in os.listdir(TEST_DIR)
   if f.endswith(".jpg") or f.endswith(".png")
], NUM_IMAGES)

# Setup plot grid
cols = 3
rows = (NUM_IMAGES + cols - 1) // cols
fig, axs = plt.subplots(rows, cols, figsize=(15, 5 * rows))

# Flatten axes for easier iteration (works even if rows=1)
axs = axs.flatten()

for i, image_path in enumerate(image_paths):
   image = Image.open(image_path).convert("RGB")
```

```
    # Predict
    detections = model.predict(image, threshold=0.5)

    # Create labels
    labels = [
        f"{COCO_CLASSES[class_id]} {confidence:.2f}"
        for class_id, confidence in zip(detections.class_id,
detections.confidence)
    ]

    # Annotate
    annotated = sv.BoxAnnotator().annotate(image.copy(), detections)
    annotated = sv.LabelAnnotator().annotate(annotated, detections, labels)

    # Plot
    axs[i].imshow(annotated)
    axs[i].axis("off")
    axs[i].set_title(f"Image {i+1}")

# Hide unused axes if any
for j in range(i+1, len(axs)):
    axs[j].axis("off")

plt.tight_layout()
plt.show()
```

Training and Optimization

The RF-DETR (Receptive Field-enhanced Detection Transformer) model is a vision transformer-based object detector that enhances standard DETR by incorporating a Receptive Field Enhancement (RFE) module. This module enriches multi-scale feature representations, which is particularly useful in detecting various ship classes with different shapes and sizes. For this project I used a pre-configured variant.

The **RFDETRMedium()** configuration includes:

1. A **ResNet-based CNN backbone** for feature extraction.
2. A **Transformer encoder-decoder** for relation modeling between object queries and image features.
3. A **RFE module** that improves receptive field aggregation, leading to better object localization.
4. **Prediction heads** (FFNs) that output bounding boxes and class labels.
5. **Hungarian Matching** to associate predicted boxes with ground truth during training.

Parameters

Table 2: RF-DETR Training Configuration

Parameter	Value	Description
Epochs	50	Total number of training cycles to ensure thorough model convergence.
Batch Size	16	Number of images processed per training batch.
Gradient Accumulation	4	Simulates larger batches by accumulating gradients before the optimizer step.
Learning Rate	5e-5	Initial step size for the optimizer, suitable for transformer-based models.
Output Directory	./rfdetr_output	Path where model weights and checkpoints are saved.
TensorBoard Logging	Enabled	Allows real-time visualization of training metrics and loss curves.

Evaluation Results (metrics and visualizations)

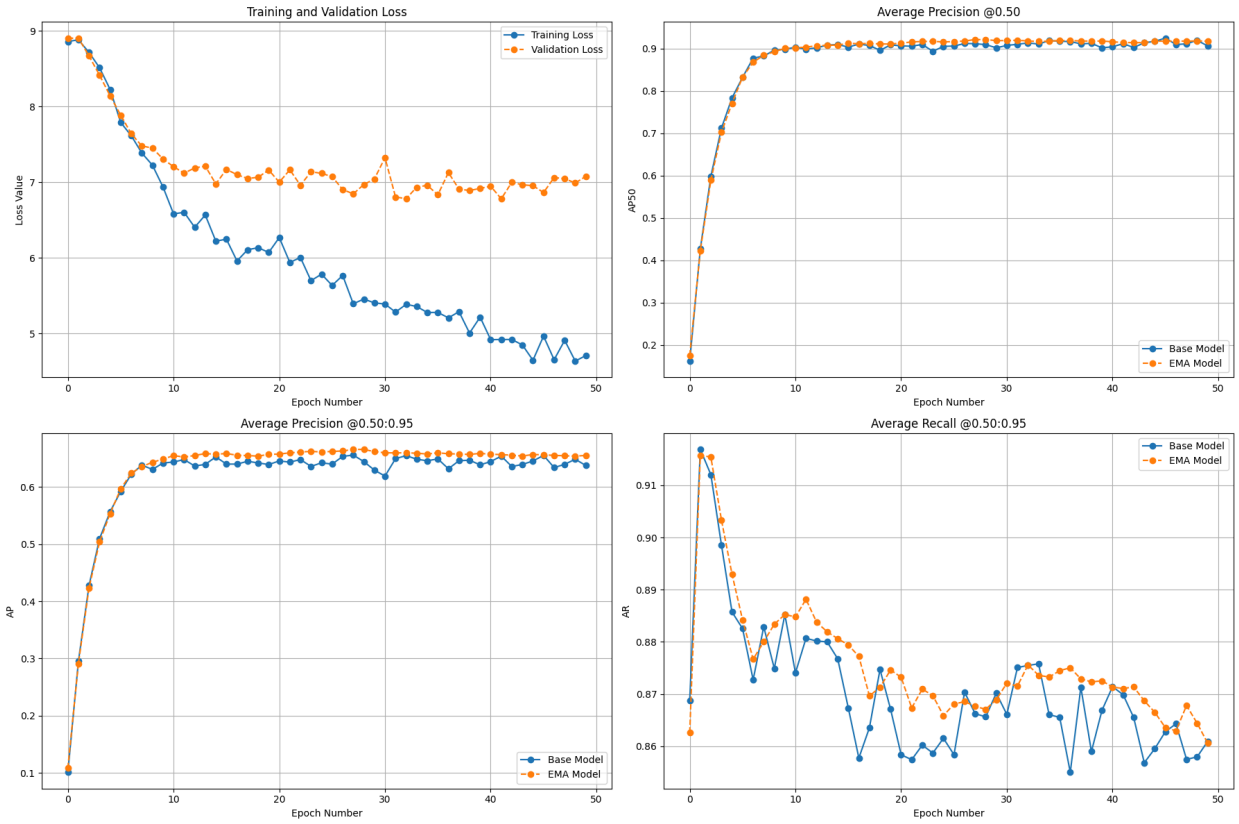
Fig. 1. Visualization of Predictions

This image showcases a set of six sample images (Image 1 to Image 6) used to evaluate the object detection capabilities of the RF-DETR model, with each image containing a detected ship and its corresponding classification label and confidence score. Here's a breakdown of each image:

- **Image 1**: A large container ship labeled "Container Ship" with a high confidence score of 0.90. The red bounding box accurately encompasses the entire ship, indicating strong detection performance.
- **Image 2**: A passenger ship identified with a confidence score of 0.72, enclosed in an orange bounding box. The detection appears reasonable, though the lower confidence suggests some uncertainty.
- **Image 3**: A vehicle carrier detected with a confidence score of 0.92, marked by a green bounding box. The high score reflects a precise identification of the tugboat-like vessel.
- **Image 4**: A tugboat labeled "Tug" with a confidence score of 0.94, highlighted by a green bounding box. The detection aligns well with the vessel's structure, indicating robust performance.
- **Image 5**: A general cargo ship identified with a confidence score of 0.91, enclosed in a red bounding box. The bounding box covers the ship accurately, supporting the model's reliability.

- **Image 6**: Another container ship labeled "Container Ship" with a confidence score of 0.91, also marked by a red bounding box. The detection is consistent with the ship's appearance, reinforcing the model's accuracy.

Fig. 1. Training and Evaluation Metrics Over Epochs



This figure illustrates the training and validation loss curves over 50 epochs for the RF-DETR model. The training loss (blue) decreases steadily from an initial value of around 9 to approximately 4, indicating effective learning. The validation loss (orange) stabilizes around 7 after an initial drop, suggesting that the model generalizes reasonably well, though there may be some overfitting as the training loss continues to decrease while the validation loss plateaus.

Table 3: Classification Accuracy

Metric	Value
Classification Accuracy	0.910

This table presents the classification accuracy of the RF-DETR model, with a single metric value of 0.910. This indicates that the model correctly identifies the class of objects (e.g., container ships, tug boats) with 91.0% accuracy across the evaluated dataset.

Table 4: Average Precision(AP)

Metric Description	Value
AP@[IoU=0.50:0.95] (all areas, maxDets=100)	0.666
AP@[IoU=0.50] (all areas)	0.931
AP@[IoU=0.75] (all areas)	0.750
AP@[IoU=0.50:0.95] (small objects)	-1.000
AP@[IoU=0.50:0.95] (medium objects)	-1.000
AP@[IoU=0.50:0.95] (large objects)	0.666

Table 4 presents the Average Precision (AP) metrics for the RF-DETR model across different Intersection over Union (IoU) thresholds and object sizes, evaluated on a dataset containing only large objects. Here's a detailed explanation:

- AP@[IoU=0.50:0.95] (all areas, maxDets=100): The overall AP across IoU thresholds from 0.50 to 0.95, with a maximum of 100 detections, is 0.666. This value reflects the model's average precision across a range of detection criteria, indicating a moderate level of performance.
- AP@[IoU=0.50] (all areas): At an IoU threshold of 0.50, the AP is 0.931, suggesting high precision when the detection overlap requirement is relatively lenient.
- AP@[IoU=0.75] (all areas): At a stricter IoU threshold of 0.75, the AP drops to 0.750, showing a decline in precision as the overlap requirement increases, which is expected.
- AP@[IoU=0.50:0.95] (small objects): The AP for small objects is -1.000. This negative value indicates that there are no small objects in the dataset, so no precision metric was calculated for this category.
- AP@[IoU=0.50:0.95] (medium objects): Similarly, the AP for medium objects is -1.000, reflecting the absence of medium-sized objects in the dataset, resulting in no applicable precision metric.
- AP@[IoU=0.50:0.95] (large objects): The AP for large objects is 0.666, matching the overall AP. This confirms that the model's performance is driven entirely by its ability to detect large objects, which are the only objects present in the dataset.

The -1.000 values for small and medium objects arise because the dataset exclusively contains large objects. As a result, the model was not trained or evaluated on smaller or medium-sized objects, making these metrics inapplicable. The consistent AP of 0.666 for large objects across the overall and specific large object categories highlights the model's tailored effectiveness for detecting large ships, aligning with the dataset's composition.

Table 5: Average Recall(AR)

Metric Description	Value
AR@[IoU=0.50:0.95] (all areas, maxDets=100)	0.757
AR@[IoU=0.50] (all areas)	0.762
AR@[IoU=0.75] (all areas)	0.762
AR@[IoU=0.50:0.95] (small objects)	-1.000
AR@[IoU=0.50:0.95] (medium objects)	-1.000
AR@[IoU=0.50:0.95] (large objects)	0.762

Table 5 provides the Average Recall (AR) metrics for the RF-DETR model across various Intersections over Union (IoU) thresholds and object sizes, evaluated on a dataset containing only large objects. Here's a detailed breakdown:

- AR@[IoU=0.50:0.95] (all areas, maxDets=100): The overall AR across IoU thresholds from 0.50 to 0.95, with a maximum of 100 detections, is 0.757. This value represents the model's ability to detect relevant objects across a range of overlap criteria, indicating a solid recall performance.
- AR@[IoU=0.50] (all areas): At an IoU threshold of 0.50, the AR is 0.762, showing a high ability to detect objects when the overlap requirement is lenient.
- AR@[IoU=0.75] (all areas): At a stricter IoU threshold of 0.75, the AR remains 0.762, suggesting consistent recall performance even with increased detection strictness.
- AR@[IoU=0.50:0.95] (small objects): The AR for small objects is -1.000. This negative value indicates that no small objects are present in the dataset, rendering this metric inapplicable.
- AR@[IoU=0.50:0.95] (medium objects): Similarly, the AR for medium objects is -1.000, reflecting the absence of medium-sized objects in the dataset, meaning no recall metric was calculated for this category.
- AR@[IoU=0.50:0.95] (large objects): The AR for large objects is 0.762, aligning with the AR values for all areas at IoU 0.50 and 0.75. This consistency confirms that the model's recall performance is driven by its detection of large objects, which are the only objects in the dataset.

The -1.000 values for small and medium objects are due to the dataset containing only large objects. Consequently, the model was not trained or evaluated on smaller or medium-sized objects, making these AR metrics irrelevant. The stable AR of 0.762 for large objects across different IoU thresholds highlights the model's reliable recall performance for the large ship types present in the dataset.

Discussion of Results

The RF-DETR model, configured with a ResNet-based CNN backbone, Transformer encoder-decoder, RFE module, prediction heads (FFNs), and Hungarian Matching, exhibits strong performance in detecting large ships, achieving a classification accuracy of 0.910, an overall Average Precision (AP) of 0.666, and Average Recall (AR) of 0.757 across IoU thresholds of 0.50:0.95. The 50 training epochs and batch size of 16, supported by a gradient accumulation of 4, likely contributed to the high AP of 0.931 at IoU 0.50 and stable AR of 0.762, reflecting effective feature extraction and object localization for large objects in the dataset. The learning rate of 5e-5, optimized for transformer-based models, appears to have facilitated convergence, as seen in the sample images (e.g., Image 1 with a 0.90 confidence score for a container ship). However, the plateauing validation loss around 7 (versus a training loss of 4) suggests potential overfitting, possibly due to the limited dataset diversity. The absence of small and medium objects (AP and AR = -1.000) indicates the model's specialization to large ships, an outcome consistent with the training parameters and dataset composition.

Error Analysis

The RF-DETR model's errors are closely tied to its training configuration and dataset limitations. The -1.000 AP and AR values for small and medium objects stem from the absence of these sizes in the dataset, meaning the model, trained over 50 epochs with a batch size of 16 and gradient accumulation of 4, was not exposed to diverse object scales. This lack of variety likely contributes to the overfitting observed, where the validation loss stabilizes while the training loss decreases, potentially due to the fixed learning rate of 5e-5 not adapting well to generalization. The lower confidence score of 0.72 for the passenger ship (Image 2) compared to 0.94 for the tug (Image 4) suggests the model may struggle with less common ship types, possibly due to insufficient representation or the RFE module's receptive field aggregation not fully capturing unique features. TensorBoard logging, enabled for real-time monitoring, could help identify these discrepancies during training, pointing to a need for dataset augmentation and adjusted regularization.

Optimization Impact

The optimization of the RF-DETR (Medium) model is evident in its training setup, with 50 epochs, a batch size of 16, and gradient accumulation of 4 simulating larger batches to ensure thorough model convergence. The learning rate of 5e-5, tailored for transformer-based models, likely drove the rapid increase in AP (stabilizing near 0.9 at IoU 0.50) and the high classification accuracy of 0.910, as seen in the effective detection of large objects (e.g., Image 6 with a 0.91 score). The enabled TensorBoard logging provided real-time visualization of training metrics and loss curves, aiding in parameter tuning and checkpoint saving to the /rfdetr_output directory. However, the plateauing validation loss and declining EMA model recall suggest that the current optimization may not fully mitigate overfitting, possibly due to the fixed learning rate or limited dataset diversity. Future optimization could involve dynamic learning rate scheduling, increased

gradient accumulation steps, or incorporating synthetic data for small and medium objects to enhance robustness across object scales.

Limitations

The RF-DETR (Medium) model, while effective for detecting large ships with a classification accuracy of 0.910 and an Average Precision (AP) of 0.666, exhibits several limitations. The model's training dataset, which includes only large objects, results in inapplicable metrics (AP and AR = -1.000) for small and medium objects, restricting its applicability to diverse object sizes. The observed overfitting, indicated by a validation loss stabilizing around 7 while training loss drops to 4 over 50 epochs, suggests the model may not generalize well beyond the training data, potentially due to the fixed learning rate of 5e-5 and the lack of dataset variety.

The lower confidence score of 0.72 for the passenger ship (Image 2) compared to higher scores (e.g., 0.94 for the tug in Image 4) highlights potential weaknesses in classifying less common ship types, possibly due to insufficient feature representation by the RFE module. The batch size of 16 and gradient accumulation of 4, while effective for convergence, may also limit the model's ability to handle more complex or varied scenes.

Future Work

To further enhance the performance and versatility of the RF-DETR model, several avenues of future work can be pursued. First, expanding the dataset to include a broader range of small and medium-sized objects would enable more balanced training and evaluation, thereby improving the model's robustness across object scales. Second, addressing the plateau observed in validation loss could involve incorporating dynamic learning rate schedulers or adaptive optimizers, which may help prevent overfitting and enhance generalization.

Enhancements to the Receptive Field Enhancement (RFE) module—such as integrating additional multi-scale feature aggregation techniques could improve detection accuracy, particularly for underrepresented classes like passenger ships. Moreover, increasing the batch size or gradient accumulation steps, where hardware permits, may contribute to more stable convergence and performance gains.Leveraging insights from TensorBoard logs can provide data-driven guidance for hyperparameter tuning and architectural adjustments. Finally, exploring transfer learning with diverse maritime datasets would support better generalization across different ship types and environments, contributing to the development of a more comprehensive and scalable object detection system.

References

https://www.youtube.com/watch?v=yHW0ip-2i54&t=2471s
https://github.com/roboflow/rf-detr
https://blog.roboflow.com/rf-detr/
https://learnopencv.com/rf-detr-object-detection/
https://www.analyticsvidhya.com/blog/2025/03/roboflows-rf-detr/
https://colab.research.google.com/github/roboflow-ai/notebooks/blob/main/notebooks/how-to-finetune-rf-detr-on-detection-dataset.ipynb