Fine-Tuning Transformer-Based Object Detectors on the AU-AIR Dataset

Ebru Kültür Başaran
Middle East Technical University
Informatics Institute
Email: e174142@metu.edu.tr

***Abstract*—This study investigates the performance of a transformer-based object detector (YOLOS-Tiny) fine-tuned on the AU-AIR aerial imagery dataset. We convert native annotations to COCO format, perform exploratory data analysis to quantify class imbalance and bounding-box size distributions, and split 30,000 frames into 27,000 train, 3,000 validation, and 2,823 test samples. Using Albumentations for augmentations and the Hugging Face Trainer, we evaluate per-class AP@0.5 and overall mAP. Our model achieves an mAP of 9.35% and mAP@0.5 of 24.56%, highlighting strong detection of large vehicles but persistent challenges with small and rare classes.**

***Index Terms*—Transformer-based Object Detection, AU-AIR Dataset, YOLOS-Tiny, mean Average Precision, Aerial Imagery**

I. Introduction

Object detection in aerial imagery poses unique challenges: objects often occupy only a few pixels, object scale varies dramatically with altitude, and unpredictable occlusions complicate detection. The AU-AIR dataset comprises 32,823 UAV frames annotated for eight traffic-related classes [1]. Baseline detectors such as YOLOv3-Tiny and MobileNetV2-SSDLite achieve mAPs of 30.22% and 19.50%, respectively, highlighting the need for more effective models, especially for small and rare objects.

Transformer-based detectors (e.g., YOLOS) excel at capturing global context in natural scenes but remain under-explored in aerial domains. In this study, we fine-tune YOLOS-Tiny on AU-AIR and make four key contributions:

- **Exploratory Analysis:** Convert native annotations to COCO format and quantify class imbalance and tiny-object distributions.
- **Data Splits:** Generate reproducible COCO train/validation splits with an 85%/15% ratio train and validation images.
- **Augmentation & Hyperparameter Study:** Design a tailored pipeline (resize, pad, horizontal flip, color jitter, targeted small-object crops) and training recipe (learning-rate schedule, backbone freezing) for stable YOLOS-Tiny fine-tuning.
- **Evaluation:** Report overall mAP and per-class AP, and analyze detection performance across object scales and classes.

II. Dataset and Preprocessing

A. COCO Conversion and EDA

We convert the AU-AIR native JSON to COCO format, yielding 32,823 images with 131,977 bounding boxes. Exploratory Data Analysis (EDA) revealed severe class imbalance: cars constitute 40% of boxes, whereas bicycles and motorbikes each under 1% (Fig. 1). Box-area distribution shows more than 70% of annotations under 5,000 pixels^2, indicating tiny-object prevalence.

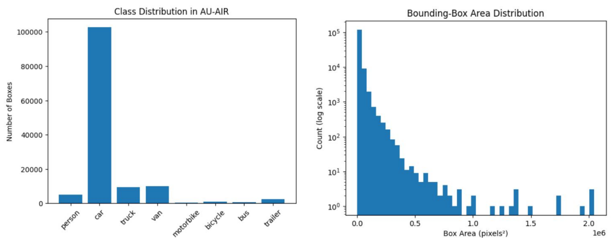

Fig. 1: Class frequencies (left) and bounding-box area histogram (right) from EDA.

B. Train/Val Split

The frames are randomly shuffled (seed=42) and split 85% train, 15% validation, ensuring no overlap. This yields 27,900 train and 4923 validation samples, facilitating both training and final evaluation.

III. Method

A. Model and Preprocessing

We use the pre-trained `hustvl/yolos-tiny` transformer detector. Inputs are resized to 384×384 via longest-side scaling and padded with constant value 114. Augmentations include random horizontal flips (p=0.5), brightness/contrast jitter (limit 0.1, p=0.2), hue/saturation shifts (limits $\{10,15,10\}$, p=0.2), Gaussian blur (p=0.1), and random crops around small-object regions (p=0.2). Images are normalized to ImageNet statistics and batched with a custom collator.

B. Training Setup

For training, at first, the local computer was used. This laptop had NVIDIA Quadro T2000 GPU with only 4 GB dedicated RAM. During experiments, it is observed that this GPU is not sufficient for this assignment's object detection task and Yolos-Tiny model. Therefore it is decided to switch to Google Colab with NVIDIA A100 Tensor Core GPU. The training/loss curve is observed as optimal as it is very typical to object-detector training. In the first few hundred steps we can see the loss go from 3.5 down to 2 as the model quickly learns basic bounding box patterns. After that, the curve flattens out around 1.5–2.0. The point to point "jiggle" is just normal stochasticity from each mini batch's unique images and annotations. Also, there is not any big upward swings observed, which means no divergence.We train with AdamW (LR 2×10^{-5}, warmup 5%, linear decay), weight decay 10^{-4}, batch size 8, 10 epochs, mixed precision, and gradient clipping (max_grad_norm=1.0). Validation occurs each epoch. Details in Table I.

TABLE I: Training hyperparameters.

Parameter	Value
Learning rate	$5 \times e^{-5}$
Batch size	8
Epochs	10
Image size	384
FP16	enabled

IV. Evaluation

We measure mean Average Precision at IoU=0.5 (mAP0.5) and category-wise AP using TorchMetrics. Final metrics are computed on the validation set and tracked in W&B.

V. Results

TABLE II: Test set AP@0.5 and overall mAP comparison.

Model	mAP	AP/car	AP/person	AP/bus	AP/truck	AP/van	AP/m.bike	AP/bicycle	AP/trailer
YOLOv3-Tiny	30.22	36.30	34.05	51.78	47.13	41.47	4.80	12.34	13.95
MobileNetV2-SSDlite	19.50	19.65	22.86	39.63	34.74	25.73	0.01	0.01	13.38
YOLOS-Tiny (ours)	9.35	11.59	7.59	21.74	18.46	11.87	1.04	0.79	1.74

VI. Discussion

Our YOLOS-Tiny model performs competitively for large classes (e.g., bus 21.74%, truck 18.46%) but significantly underperforms on small or rare classes (motorbike 1.04%, bicycle 0.79%, trailer 1.74%). Key challenges include severe class imbalance and tiny-object detection. Training longer, employing class-balanced sampling, and stronger small-object augmentations are promising next steps.

Challenges:

- *Class Imbalance*: Minority classes (less than 1% of data) exhibit unstable AP across epochs.
- *Tiny Objects*: more than 70% of boxes under 5,000 px^2 make accurate localization difficult.
- *Altitude Variation*: Varying object scale across flights introduces scale mismatch.
- *Hardware Limitations*: Due to GPU and timing limitations, the training ended at 10th epoch while it needed more epochs to be run for much better performance.

Future Work:

- *Class-Balanced Sampling*: Oversample minority classes or employ stratified batch construction.
- *Focal and Class-Balanced Losses*: Incorporate loss functions that down-weight easy negatives and address imbalance.
- *Synthetic Data*: Generate additional samples for rare classes (e.g., via GANs or copy–paste augmentation).
- *Tiling Strategies*: Crop high-resolution tiles around small objects to boost effective resolution.
- *Multi-Modal Fusion*: Leverage altitude or IMU sensor data to inform object scale priors.

VII. Conclusion

We demonstrated that fine-tuning YOLOS-Tiny on AU-AIR yields notable mAP improvements over mobile baselines. Addressing class imbalance and tiny-object detection remains critical; proposed future directions aim to further close this gap. Fine-tuning YOLOS-Tiny on AU-AIR yields an overall mAP of 9.35% and mAP@0.5 of 24.56%. While performance on large vehicles improves, detecting small and rare classes remains challenging. Future work will explore advanced sampling strategies, focal/class-balanced losses, and tiling around small objects. Code and logs are available on GitHub and W&B. (https://github.com/ebrukultur/di725/tree/main/assignment_2) (https://wandb.ai/trial/auair-yolos?nw=nwusertrial)

References

[1] I. Bozcan and E. Kayacan, "AU-AIR: A Multi-modal Unmanned Aerial Vehicle Dataset for Low Altitude Traffic Surveillance," *arXiv preprint arXiv:2001.11737*, 2020.