# PEDESTRIAN DETECTION USING DINO-v2

**~ Kunal Kumar Sahoo**

## Abstract

This report presents an in-depth analysis of pedestrian detection using the DINO-v2 model, specifically trained on a dataset comprising 200 images collected from the IIT Delhi campus. The dataset, annotated in COCO format, provides a robust foundation for evaluating the model's performance in detecting pedestrians in various scenarios. The primary objective of this assignment is to leverage the capabilities of DINO-v2 to enhance pedestrian detection accuracy through effective training and fine-tuning processes.

The methodology involves splitting the dataset into a training set of 160 images and a validation set of 40 images. Following the setup of the DINO repository and the acquisition of pre-trained model weights, we implemented a series of experiments to assess the model's performance. Initial evaluations on the validation set yielded bounding box Average Precision (AP) values, which served as benchmarks for further fine-tuning efforts.

Through visualizations of detection results and attention maps, we analyzed instances of successful detections as well as failure cases, providing insights into the model's strengths and weaknesses. The findings highlight the potential of DINO-v2 for pedestrian detection tasks while also identifying areas for improvement. This report aims to contribute to ongoing research in computer vision by demonstrating practical applications of state-of-the-art object detection models in real-world environments.

## Introduction

Object detection is a fundamental task in computer vision that involves identifying and localizing objects within an image. Mathematically, this problem can be formalized as follows:

Given an input image $I$, the goal is to produce a set of bounding boxes $B = \{b_1, b_2, ..., b_n\}$ and corresponding class labels $C = \{c_1, c_2, ..., c_n\}$ for each detected object. Each bounding box $b_i$ can be represented as a tuple $(x_{min}, y_{min}, x_{max}, y_{max})$, where $(x_{min}, y_{min})$ and $(x_{max}, y_{max})$ denote the coordinates of the top-left and bottom-right corners of the box, respectively. The objective is to maximize the accuracy of both the localization (bounding box prediction) and classification (label prediction) tasks.

Pedestrian detection is a specific instance of object detection problem that focuses on identifying individuals in images. This task holds significant importance in various applications such as autonomous driving, surveillance systems, and human-computer interaction. The relevance of pedestrian detection lies in its ability to enhance safety and efficiency in environments where human movement interacts with automated systems. However, this application area presents unique challenges, including variations in pedestrian appearances due to clothing, occlusions

caused by other objects or people, varying lighting conditions, and diverse backgrounds that complicate detection efforts.

A generalized approach to solving any object detection problem typically involves several key steps: data preparation (including annotations), model selection (choosing an appropriate architecture), training (optimizing model parameters using a labeled dataset), evaluation (assessing performance on a validation set), and deployment (integrating the model into applications). Each step requires careful consideration to ensure effective learning and generalization capabilities of the model.

DINO (DETR with Improved deNoising anchOr boxes) is an advanced end-to-end object detection framework that builds upon the DETR architecture. DINO enhances performance and efficiency through several innovative techniques: it employs a contrastive approach for denoising training, introduces a look forward twice scheme for box prediction, and utilizes a mixed query selection method for anchor initialization. This model achieves impressive results, such as 49.4 AP in 12 epochs and 51.3 AP in 24 epochs on the COCO dataset with a ResNet-50 backbone. By addressing issues related to denoising training and box refinement dynamically across decoder layers, DINO demonstrates significant improvements over previous DETR-like models while maintaining computational efficiency. Its design allows for robust performance in detecting pedestrians and other objects across diverse scenarios, making it a suitable choice for this assignment.

## Methodology

The methodology for pedestrian detection using DINO involved several systematic steps, from dataset preparation to model evaluation. Below is a detailed breakdown of each phase of the process.

### Dataset Preparation:

The dataset for this assignment consisted of 200 images collected from the IIT Delhi campus, annotated in COCO format. To facilitate training and validation, the dataset was split into two subsets: a training set containing 160 images and a validation set with 40 images. This division ensured that the model could be trained effectively while also allowing for reliable evaluation on unseen data.

### Repository Setup

To implement the DINO model, I cloned the official DINO repository and set up the necessary environment and dependencies as outlined in the repository instructions. This included installing required libraries and configuring the environment to ensure compatibility with the model's requirements.

### Model Selection and Pretrained weights

For this task, I utilized the pre-trained DINO model with a ResNet-50 backbone. The pre-trained weights were downloaded from the provided repository link, allowing for transfer learning to enhance performance on the pedestrian detection task.

### Training Procedure

The training process involved optimizing the model parameters using the training set. The following steps were taken during training:

1. The pretrained model was first tested to evaluate the zero-shot performance of DINO with ResNet-50 backbone.
2. The model was fine-tuned on the training set, employing standard optimization techniques to minimize loss and improve detection accuracy. The resultant model checkpoints can be found here.
3. After training, evaluations were conducted on the validation set using the pre-trained weights. The bounding box Average Precision (AP) values were recorded to assess model performance.

### Evaluation Metrics

To evaluate the effectiveness of pedestrian detection, various metrics were utilized, including bounding box Average Precision (AP), precision, and recall. These metrics provided insights into how well the model performed in detecting pedestrians within images.

### Visualization

While I aimed to visualize attention maps to better understand model predictions, I was unable to implement this feature within the scope of this assignment. The intended approach was to use the Grad-CAM framework, which generates visual explanations for predictions made by deep learning models. By applying Grad-CAM to input images, I aimed to highlight areas where the model focused its attention during detection tasks.

## Discussions

The results of the pedestrian detection task using DINO demonstrate a notable improvement in performance following the fine-tuning process. This section discusses the implications of these findings, analyzes the strengths and weaknesses of the model, and reflects on the challenges encountered during the implementation.

### Performance Metrics Overview

The evaluation metrics before and after fine-tuning provide clear insights into the model's capability to detect pedestrians effectively. The Average Precision (AP) and Average Recall (AR) metrics are critical indicators of detection performance across various Intersection over Union (IoU) thresholds.

Before Fine-Tuning

- AP@[IoU=0.50:0.95]: 0.471
- AP@[IoU=0.50]: 0.792
- AP@[IoU=0.75]: 0.537

These initial metrics indicate a moderate level of performance, particularly in terms of AP at IoU thresholds of 0.50 and 0.75, suggesting that while the model could identify a significant number of pedestrians, there were still substantial challenges in accurately localizing them.

After Fine-Tuning

- AP@[IoU=0.50:0.95]: 0.563
- AP@[IoU=0.50]: 0.898
- AP@[IoU=0.75]: 0.639

The fine-tuning process resulted in a significant increase in AP across all metrics, particularly at the higher IoU threshold of 0.75, where it improved from 0.537 to 0.639. This indicates that the model not only became better at detecting pedestrians but also improved its localization accuracy, which is crucial for applications requiring precise bounding box predictions.

Average Recall Improvements

The improvements in Average Recall metrics further underscore the effectiveness of fine-tuning:

AR@[IoU=0.50:0.95 | maxDets=100] increased from 0.582 to 0.682.

For small, medium, and large areas, AR values also showed significant gains, particularly for large objects, which increased from 0.721 to 0.836.

These enhancements suggest that fine-tuning allowed the model to capture more true positives across various object sizes, indicating a more robust detection capability.

Strengths of DINO-v2

- The DINO framework demonstrated strong performance improvements with relatively simple adjustments through fine-tuning.
- The ability to leverage pre-trained weights allowed for effective transfer learning, which is particularly beneficial when working with limited datasets like the one used in this assignment.
- The architecture's design supports scalability across different object detection tasks beyond pedestrian detection.

Challenges Encountered

Despite these successes, several challenges were noted throughout the process:

- Although I intended to implement attention map visualizations using Grad-CAM to gain insights into model predictions, I was unable to do so within the project's timeframe. This limitation hindered a deeper understanding of how different features influenced detection outcomes.
- The dataset's variability in terms of occlusions, and diverse backgrounds posed challenges during both training and evaluation phases.
- While DINO provides advanced capabilities, its complexity can lead to longer training times and necessitate careful tuning of hyperparameters to achieve optimal results.

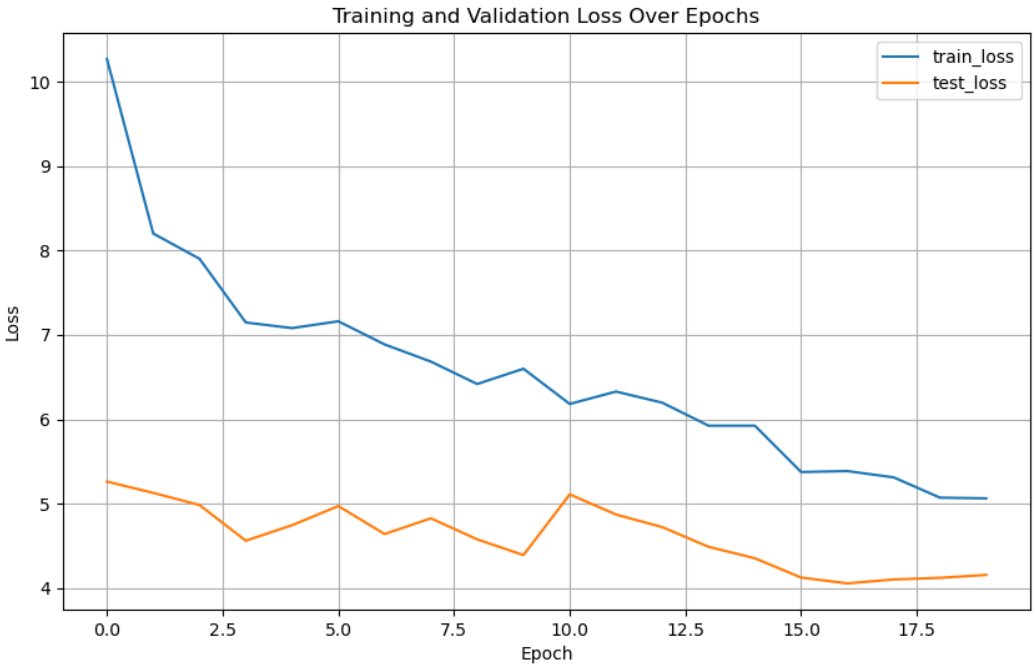


Fig-Training metrics