**AIML2402 TEACHERS ASSESSMENT COMPUTER VISION**

**LAB- 09**

**Case Study Report**

**On**

**2D Object Recognition**

**Submitted By**

**Group No.: 13**

|  |  |  |
| --- | --- | --- |
| **Roll No.** | **Name** | **Signature** |
| **36** | **Arnab Chakraborty** |  |
| **44** | **Jayushna Mahadule** |  |
| **17** | **Rutuja Balbudhe** |  |
| **31** | **Aayush Zade** |  |

**Image Classification and Object Detection**

# **End-to-End Object Detection with Transformers**

**(DETR – DEtection TRansformer)**

**[1] Introduction:**

Object recognition is a technology within computer vision used to detect and identify objects within an image or video. While humans can easily recognize a wide variety of objects in images, despite differences in viewpoints, size, scale, or even partial obstruction, this task remains challenging for computer vision systems. Over the years, various methods have been developed to tackle this problem.

**References:**

<https://en.wikipedia.org/wiki/Outline_of_object_recognition>

**[2] Problem Definition:**

Traditional object detection systems often involve complex, manually designed components such as anchor generation, non-maximum suppression, and other task-specific techniques. These elements are used to encode prior knowledge and improve detection performance but also add unnecessary complexity, limit the system's ability to generalize, and make optimization more difficult. Additionally, many existing detection methods struggle to handle tasks like panoptic segmentation in a unified framework. A novel approach is introduced that treats object detection as a direct set prediction problem, simplifying the detection process. This method eliminates the need for many traditionally hand-designed components, such as non-maximum suppression or anchor generation, which typically rely on prior task-specific knowledge. The framework, called DEtection TRansformer (DETR), utilizes a set-based global loss that enforces unique predictions through bipartite matching, alongside a transformer-based encoder-decoder architecture. DETR works with a small, fixed set of learned object queries, enabling it to reason about the relationships between objects and the overall image context to output the final set of predictions in parallel. The model is conceptually simple and does not require specialized libraries, unlike many other modern detection systems. DETR achieves accuracy and run-time performance comparable to the highly optimized Faster RCNN baseline on the challenging COCO object detection dataset. Furthermore, DETR can be easily adapted to generate panoptic segmentation in a unified way, significantly outperforming other competitive baselines.

**[3] Methodology:**

**Traditional Methods of Object Recognition:**

***Edge Matching***

Edge matching involves using edge detection methods, such as the Canny edge detector, to identify the edges within an image. Changes in lighting or color typically have little impact on the edges of an image.

**Strategy:**

* Detect edges in both the template and the image.
* Compare the edge images to locate the template.
* Consider all possible positions where the template could be placed.

***Divide-and-Conquer Search***

**Strategy:**

* Treat all positions as a set, representing a cell within the position space.
* Calculate a lower bound on the score for the best position within that cell.
* If the bound exceeds a certain threshold, prune the cell and discard it from further consideration.
* If the bound is within acceptable limits, divide the cell into smaller subcells and apply the search recursively to each subcell.
* The process terminates when the cell becomes "small enough" to stop further division.

***Greyscale Matching***

Edges are generally resistant to changes in illumination, but they tend to discard much of the image's detail. To improve accuracy, pixel distance should be computed considering both the pixel's position and its intensity value. This approach can also be extended to color images by incorporating color information into the distance calculation.

***Gradient Matching***

To maintain robustness to illumination changes while retaining more image details, gradient matching compares the gradients of the image rather than just the raw pixel values. The matching process is similar to greyscale image matching. A simpler alternative is to use normalized correlation for comparison.

***Histograms of Receptive Field Responses***

This approach eliminates the need for explicit point correspondences by encoding the relationships between different image points within the receptive field responses. These responses capture the spatial patterns and features of the image, making the method more flexible. Notable works in this area include Swain and Ballard (1991), Schiele and Crowley (2000), and Linde and Lindeberg (2004, 2012).

***Large Modelbases***

One approach to efficiently searching a database for a specific image is to use eigenvectors of templates, often referred to as "eigenfaces," which represent the most significant features of the images. Modelbases, on the other hand, consist of a collection of geometric models that represent the objects to be recognized. These models help in matching and identifying objects in the search process.

**DETR Approach**



In contrast to traditional computer vision methods, DETR frames object detection as a direct set prediction problem. It utilizes a set-based global loss that ensures unique predictions through bipartite matching, combined with a Transformer encoder-decoder architecture. Using a small, fixed set of learned object queries, DETR evaluates the relationships between objects and the overall image context to generate the final set of predictions in parallel. This parallel processing enables DETR to be both fast and efficient.

The DETR architecture is notably straightforward due to the Transformer’s powerful representational capabilities. It comprises two primary components:

* **Convolutional Backbone**: In this case, ResNet-50 is utilized as the backbone.
* **Transformer**: The default PyTorch nn.Transformer module is employed for this purpose.

**Transformer Encoder**

The encoder begins by applying a 1x1 convolution to the high-level activation map, reducing its channel dimensions from the original number of channels to a smaller number. This creates a new feature map with the smaller channel dimension while retaining the height and width of the original map. Since the encoder operates on sequences, the spatial dimensions of this feature map are flattened into a single dimension, resulting in a sequence of features.

Each encoder layer uses a standard structure that includes a multi-head self-attention mechanism and a feed-forward network. Because the Transformer architecture is inherently permutation-invariant, fixed positional encodings are added to the input of each attention layer to retain spatial information. Detailed specifications of the architecture, which follow the design described in earlier works, are provided in supplementary materials.

**Transformer Decoder**

The decoder employs the standard Transformer architecture, processing **N** embeddings of a fixed size using multi-headed self-attention and encoder-decoder attention mechanisms. Unlike the original Transformer design, which uses an autoregressive approach to predict one element of the output sequence at a time, this model decodes all **N** objects in parallel at each layer. For readers unfamiliar with these concepts, further details are available in supplementary materials.

Since the decoder is permutation-invariant, the **N** input embeddings must be unique to ensure distinct outputs. These embeddings, referred to as object queries, are learned positional encodings added to the input of each attention layer. The decoder transforms these object queries into output embeddings, which are then decoded independently into bounding box coordinates and class labels by a feed-forward network (described further in the next subsection). This process results in **N** final predictions.

Through self-attention and encoder-decoder attention, the model reasons globally about all objects simultaneously, leveraging pairwise relationships between them while also incorporating the entire image as context.

**Prediction Feed-Forward Networks (FFNs)**

The final predictions are generated using a feed-forward network consisting of three layers, with ReLU activation functions and a hidden dimension equal to the embedding size. This network includes a linear projection layer that outputs the results. The FFN predicts the normalized center coordinates, height, and width of bounding boxes relative to the input image. Simultaneously, the linear layer predicts the class label using a softmax function.

Since the model produces a fixed set of **N** bounding boxes—where **N** is typically larger than the actual number of objects in an image—an additional special class label, denoted as "∅," is introduced. This label indicates that no object is detected for a given slot. It serves a similar purpose to the "background" class in traditional object detection methods.

**References:**

<https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123460205.pdf>

<https://arxiv.org/abs/2005.12872v3>

<https://en.wikipedia.org/wiki/Outline_of_object_recognition>

<https://doi.org/10.1023/A:1008120406972>

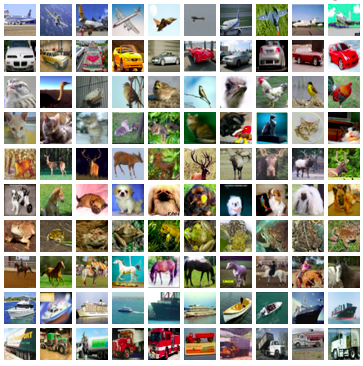
<https://doi.org/10.1007/BF00130487>

<ftp://ftp1.nada.kth.se/pub/documents/CVAP/reports/LinLin04-ICPR.pdf>

<http://www.csc.kth.se/~tony/abstracts/LinLin11-CompComplCueHist.html>

**[4] Dataset Used:**

**MS COCO (Microsoft Common Objects in Context) Dataset**

The MS COCO dataset is a comprehensive resource designed for tasks such as object detection, segmentation, key-point detection, and image captioning. It includes a total of 328,000 images and provides detailed annotations for a variety of use cases.

*Dataset Splits:*

- 2014 Release: Included 164,000 images, divided into training (83,000), validation (41,000), and test (41,000) sets.

- 2015 Update: Added 40,000 new test images, expanding the test set to 81,000 images in total.

-2017 Update: Adjusted the training/validation split to 118,000/5,000 while maintaining the same images and annotations. The test set in 2017 became a subset of the 2015 test set, containing 41,000 images. Additionally, an unannotated dataset with 123,000 images was introduced.

*Annotations:*

The dataset includes the following types of annotations:

- Object Detection: Bounding boxes and instance segmentation masks for 80 object categories.

- Captioning: Natural language descriptions of the images.

- Keypoint Detection: Over 200,000 images with 250,000 person instances labeled with 17 keypoints, such as the left eye, nose, and right ankle.

- Stuff Segmentation: Per-pixel segmentation masks for 91 categories like grass, wall, and sky.

- Panoptic Segmentation: Full-scene segmentation combining 80 "thing" categories (e.g., person, bicycle) and a subset of 91 "stuff" categories (e.g., sky, road).

- DensePose: More than 39,000 images with 56,000 person instances annotated with DensePose data. This includes instance IDs and a mapping between image pixels and a 3D model template of a person. DensePose annotations are available only for training and validation images.

This extensive dataset has become a cornerstone in computer vision research, supporting a wide range of applications and evaluation benchmarks.

COCO-O(Out-of-Distribution) consists of six domains—sketch, cartoon, painting, weather, handmade, and tattoo—featuring COCO objects that are challenging for most current detectors to identify. The dataset includes 6,782 images and 26,624 labeled bounding boxes.

**References:**

<https://arxiv.org/pdf/1405.0312v3>

<https://cocodataset.org/>

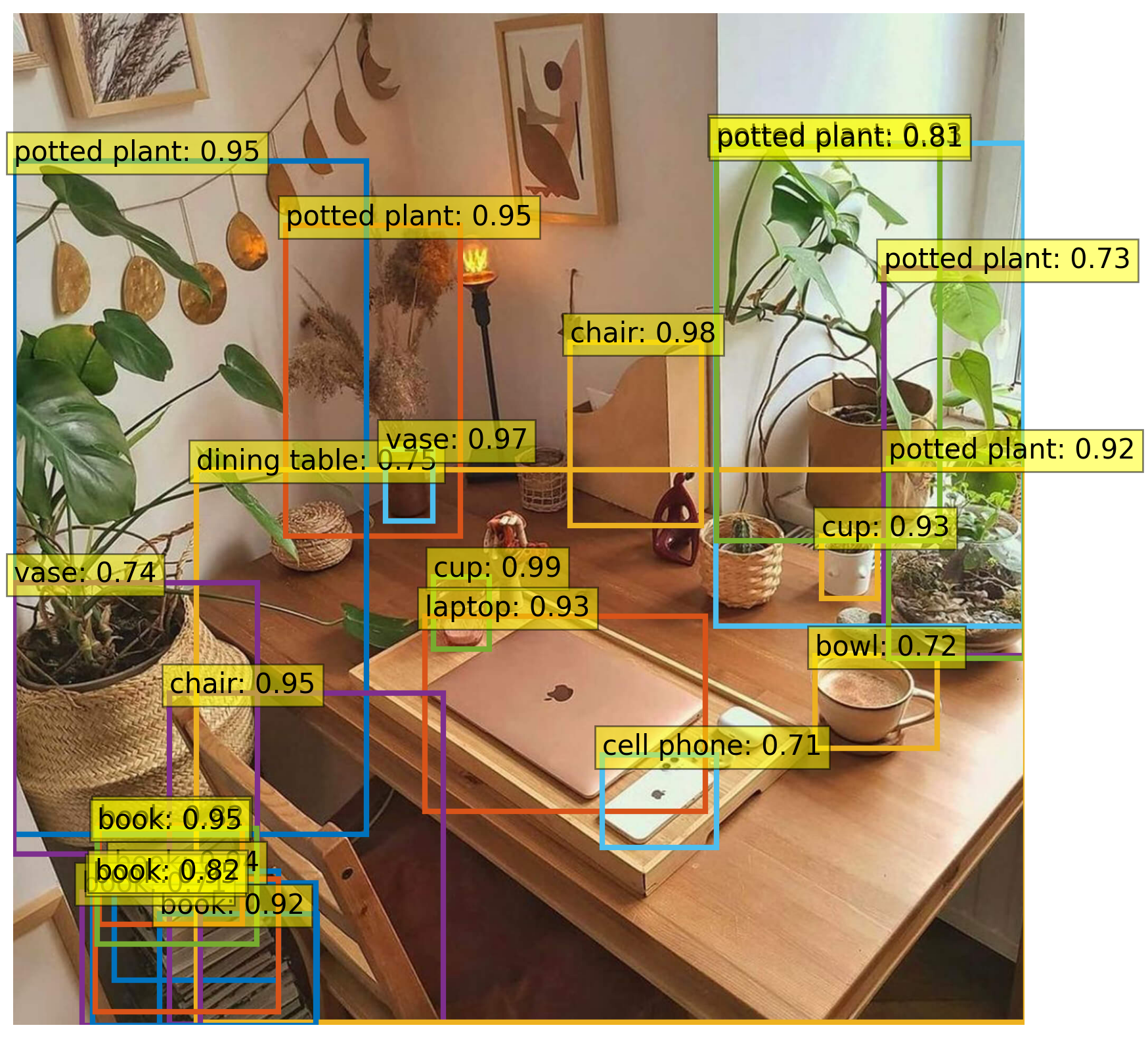
<https://github.com/alibaba/easyrobust/tree/main/benchmarks/coco_o>

**[5] Implementation Results:**

On taking a couple of sample images, the DETR model based on an encoder-decoder transformer architecture was able to accurately identify and label objects within an image making at great efficiency.

**Object Recognition and Classification**

**within a 2D Image-**

****

**[6] Conclusion:**

The DETR (DEtection TRansformer) framework marks a notable advancement in object detection by reimagining the task as a direct set prediction problem, removing reliance on traditional, manually-designed components. This innovative approach streamlines the detection process using a Transformer-based design. In essence, DETR introduces a streamlined and versatile method for object detection, demonstrating the potential for Transformer-based architectures to replace traditional techniques in computer vision tasks.

**[7] What do you learn from collecting the relevant data, code, etc.?**

* **Simplified Detection Process:** DETR streamlines object detection by removing complex steps like anchor generation and non-maximum suppression. Traditional methods rely on predefined bounding boxes and additional post-processing to refine predictions, which can be computationally intensive and require extensive tuning. Instead, DETR directly predicts a fixed number of object locations and labels in a single step, making the process more efficient, flexible, and easier to implement.
* **MS COCO:** The MS COCO dataset helps develop models that are more applicable to real-world scenarios, as it features diverse, everyday images with complex environments. Its detailed annotations for tasks like object detection, segmentation, and keypoint detection allow for comprehensive training. With 80 object categories and over 300K images, COCO exposes models to a broad range of objects and situations, promoting better generalization. Additionally, COCO is a trusted benchmark for evaluating model performance, making it a valuable resource for building and testing computer vision systems.
* **Diversity in image recognition techniques:** Diversity in image classification and object recognition methods is essential for creating robust models capable of handling various real-world challenges. By employing techniques like multi-scale processing, ensemble learning, data augmentation, and transfer learning, models can better address object variability, occlusion, and environmental changes. Additionally, diverse architectures and strategies, such as multi-task and few-shot learning, improve generalization and adaptability. This combination of approaches results in models that are not only more accurate but also more reliable and flexible, leading to improved performance in complex, real-world scenarios.