Street Navigation for Visual Impairment using CNN and Transformer Models

Hasan Ali[1], Faithful Chiagoziem Onwuegbuche[1,2]

[1]National College of Ireland, Ireland.
[2]SFI Centre for Research Training in Machine Learning (ML-Labs), University College Dublin, Ireland.

Abstract

This paper addresses the challenge of street navigation for individuals with visual impairments and explores the potential of Artificial Intelligence (AI) to enhance navigation safety and effectiveness. We evaluate the performance of state-of-the-art Computer Vision Object Detection models, focusing on accuracy and speed. The central question is whether Transformer-based Object Detection models outperform other models. We use the specialized dataset "Walking On The Road" adapted to include only relevant classes, to compare deep learning and Transformer models in pre-trained and fine-tuned states. Metrics used include Mean Average Precision (mAP) for accuracy and Average Inference Time in milliseconds for speed. Our results show that YOLO models surpass Transformer-based models in both accuracy and speed. In Phase 1, YOLOv8x achieved the highest mAP of 0.399 with an average inference time of 14ms, while Transformer-based DETR had a lower mAP of 0.344 and a significantly longer inference time of 818.2ms. In Phase 2, after fine-tuning, YOLOv8x again outperformed with an mAP of 0.471 compared to DETR's 0.323. These findings indicate that YOLO models are more effective for street navigation applications, providing superior accuracy and speed for visually impaired individuals.

Keywords

Artificial Intelligence, Vision Impairment, Blindness, Computer Vision, Object Detection,

1. Introduction

This paper aims to solve the problem of street navigation in the context of visual impairment. Street navigation is an important activity for the visually impaired people as it substantially contributes to increased public health and lower rates of chronic diseases [1] . We identify here a great opportunity to use Artificial Intelligence (AI) to increase the quality of life of visually impaired people. We define street navigation as walking in an outdoor street by foot, where the individual can navigate freely and identify if there are challenges on the street to avoid.

Visual impairment represents a significant global health issue with considerable prevalence rates. According to a study conducted by Flaxman et al. [2], as of 2015, approximately 36 million individuals were affected by blindness, while an additional 217 million people experienced moderate to severe visual impairment. The global prevalence of visual impairment, including blindness, was estimated at 0.49% of the total population, with moderate to severe visual impairment affecting 2.9% of the global population. Visual impairment restricts the ability to engage in activities such as walking on the street. Furthermore, individuals with blindness encounter significant challenges in securing employment, often resulting in lower income levels and heightened poverty rates, which can have adverse societal implications and impact education and social advancement, leading to reduced quality of life. The aforementioned challenges are what this paper aims to solve by using Artificial Intelligence.

To implement an AI-based solution, it is essential to identify the most appropriate model or architecture for this specific application. Previous research has explored suitable models for street navigation, with a focus on CNN-based models like YOLO and MobileNetv2. Other studies have examined Transformer-based models, though not within the context of street navigation for individuals with visual impairments. This paper seeks to address this gap by conducting a comprehensive comparison of Transformer-based and CNN-based models for street navigation applications designed to assist visually impaired individuals.

AICS'24: 32nd Irish Conference on Artificial Intelligence and Cognitive Science, December 09–10, 2024, Dublin, Ireland
x22142291@student.ncirl.ie (H. Ali); faithful.onwuegbuche@ncirl.ie (F. C. Onwuegbuche)

The key contributions of this paper are as follows:

1. Evaluate whether object detection models that utilize the Transformer architecture can be more effective (in accuracy and speed) than other state-of-the-art models in the context of street navigation for visually impaired people.
2. Introduce an enhanced version of the Walking On the Road Dataset (WOTRv2), specifically designed for object detection in street navigation applications for visual impairment.
3. Develop and implement a comprehensive methodology for the evaluation of object detection models, focusing on street navigation for visually impaired individuals.

It is crucial to have a fast model to ensure that the feedback provided is timely. If the model detects and identifies objects with a delay, the information becomes less useful or even irrelevant. Therefore, we evaluated the performance of the selected models (to be discussed in the related work) in two main areas: **accuracy** and **speed**. In terms of accuracy, the primary focus of this paper is to assess the model's ability to detect ground truth labels accurately using the Precision metric (Mean Average Precision), as it is vital to detect objects correctly rather than miss them, making precision more critical than Recall in this context. We also measured the speed of the model, its inference time in milliseconds (ms). This measurement allowed us to ascertain that the models can provide effective real-time feedback performance.

2. Related Work

Navigation aids through the use of object detection machine learning models have been studied in the past as part of computer vision. The applications of this area have increased with the advent of autonomous car driving. Moreover, this area has the potential to be used for visually impaired people to help them navigate while walking in the street. This literature review looks at recent applications of object detection for the purpose of aiding visually impaired people, as well as deep-diving into the recent advancements of the object detection models and their architectures.

2.1. Visual Impairment

Visual impairment, whether it manifests as blindness or moderate to severe impairment, substantially impacts individuals' quality of life [3]. The economic impact associated with visual impairment is multifaceted, encompassing both direct costs, such as medical expenses, and indirect costs, including loss of productivity [2]. Another study showcases that the prevalence of vision impairment is concentrated among older demographics, which is expected to increase due to global aging [4].

While these studies provide a comprehensive overview of the global impact of visual impairment and suggest various mitigation strategies, they lack a direct connection to the technological solutions that are the focus of this review. Specifically, there is no discussion on how different technological interventions, particularly modern machine learning models, compare in addressing the needs of visually impaired individuals. Moreover, the lack of focus on how these technologies are integrated into real-world applications creates a gap that this paper aims to fill, especially in comparing newer models like Transformers with state-of-the-art methods.

2.2. Object Detection as Aid for Visually Impaired People

Object detection systems have been employed to assist visually impaired people in navigating their environment, especially while walking, using various technologies such as sensors or smartphones that provide feedback through audio or tactile signals [5]. Other methods, including infrared, laser, GPS, or RFID-based technologies, have limitations in recognizing specific objects or providing detailed information [6]. Recent advancements in computer vision, particularly with the introduction of fuzzy logic and uncertainty-aware approaches, have improved these systems' ability to handle complex,

real-time scenarios [7]. This section explores previous research on deploying assistive technologies for visually impaired people using machine learning.

Masud et al. developed a smart assistive system using a Raspberry Pi 4B, camera, ultrasonic sensors, and an Arduino. The system employed the Viola-Jones algorithm for face detection and TensorFlow's Object Detection API trained on the COCO 2017 dataset. While it achieved 91% accuracy and enhanced user mobility and safety, it faced challenges like low-light conditions and out-of-frame objects. Although effective initially, the Viola-Jones algorithm is now outdated, and modern machine learning models could improve performance [8]. Islam et al. [9] found that MobileNetv2 combined with SSDLite was the most effective for real-life applications on embedded systems like Raspberry Pi, offering the best tradeoff between accuracy and computational power. In contrast, more accurate models like YOLOv4 and EfficientDet-D3 required more resources, making them less suitable for embedded devices. Acar et al. [10] proposed "SIGHT" a mobile application using YOLOv8 for object detection and MiDaS for depth estimation, enabling real-time navigation on smartphones. YOLOv8 was chosen for its processing speed and efficiency, achieving 0.547 mAP with a 228ms inference time on a mid-range smartphone, demonstrating successful implementation in a similar use-case to ours.

In another study, Atitallah et al. [11] utilized the YOLOv5 model with enhancements like CSPNet backbone and improved data augmentation, achieving 0.8102 mAP with an 89 FPS frame rate after compression. These studies predominantly focus on CNN-based models like YOLO and MobileNetv2, overlooking the rapidly emerging Transformer-based models. Our work addresses this gap by comparing these models with other Transformer-based approaches, particularly for street navigation for visually impaired individuals.

2.3. Transformer Architecture in Object Detection

The advancement of assistive technologies for visually impaired individuals has accelerated. A notable breakthrough is the implementation of Transformer architecture.

Introduced in "Attention is All You Need" by Vaswani et al. [12], the Transformer architecture represented a shift in sequence processing. It relies entirely on the attention mechanism, eliminating recurrence and convolutions to improve efficiency. The architecture includes key components such as self-attention, multi-head attention, positional encoding, and a feed-forward network, which together enhance the ability to process sequences effectively.

Building on this, Carion et al. [13] proposed the Detection Transformers (DETR) model, which introduces significant innovations by simplifying object detection into a direct set prediction task, eliminating traditional steps like non-max suppression and anchor generation. DETR uses a Transformer encoder-decoder architecture to capture global context and relationships within images, improving accuracy. DETR also incorporates features like bipartite matching loss and parallel processing, achieving competitive performance, particularly on large objects.

Deformable DETR (D-DETR), introduced by Zhu et al. [14], addresses challenges such as slow convergence and small object detection by using sparse sampling and a two-stage detector for improved precision. D-DETR outperforms both DETR and Faster R-CNN, particularly on the COCO 2017 dataset.

Real-Time DETR (RT-DETR) is an evolution of the DETR architecture designed for real-time object detection by Zhao et al. [15], crucial for applications like street navigation for visually impaired individuals. RT-DETR features a hybrid encoder, High-Quality Initial Queries, and flexible speed tuning, eliminating non-max suppression to reduce inference time. It achieves superior accuracy and speed, outperforming YOLOv5, v6, and v7.

These studies and papers, while thorough in detailing Transformer-based models, lack a direct comparison with state-of-the-art non-Transformer models like YOLO in aiding visually impaired individuals. This paper aims to fill that gap by comparing these models in real-world applications.

2.4. YOLO Evolution and Architecture

The YOLO (You Only Look Once) model, introduced by Redmon et al. [16], marked a significant advancement in object detection through the use of convolutional neural networks (CNNs) for feature extraction. Unlike earlier models like R-CNN, which relied on multi-stage processes, YOLO simplifies detection with a single network, treating object detection as a single regression problem that predicts bounding boxes in one evaluation, leading to faster performance. YOLO employs a grid-based prediction system where each grid cell predicts a bounding box and confidence score. However, YOLO has limitations, particularly in localizing smaller objects and handling multiple objects in close proximity.

YOLOv2, or YOLO9000, introduced in 2017, improved real-time detection across 9,000 categories with enhancements like a high-resolution classifier, anchor boxes, and multi-scale training. It increased detection speed through direct location prediction and batch normalization, achieving a 0.768 mAP, surpassing Faster R-CNN with ResNet, which achieved 0.764, and operating at 67 FPS compared to R-CNN's 5 FPS [17].

YOLO-World extended YOLO's capabilities by supporting open vocabulary detection, leveraging vision-language pre-training, particularly the RepVL-PAN technique, to enhance the interaction between visual and linguistic data. This model, which uses the pre-trained CLIP text encoder, enabled detection beyond predefined categories by integrating vision-language modeling. YOLO-World achieved a zero-shot average precision of 0.354 on the LVIS dataset with 52 FPS, outperforming models like DETCLIP-T [18].

Terven et al. [19] discuss YOLOv8, which introduced several enhancements, including a C2f Module and an Anchor-free Model, along with a new loss function (Complete IoU), collectively improving accuracy and speed. These features make YOLOv8 a strong candidate for applications like street navigation for the visually impaired, representing the current state-of-the-art in object detection.

Several applications have successfully implemented YOLO models in assistive systems for the visually impaired, including YOLOv7, PC-YOLO, and YOLOv8. Alsultan and Mohammad [20] highlight YOLOv7's practical benefits in enhancing environmental interaction and independence. Xia et al. [21] proposed PC-YOLO, designed specifically for visual impairment, achieving better average precision with a 0.6% improvement over YOLOv7.

Despite extensive documentation of YOLO's evolution, there is a lack of comparison with Transformer-based models, which are becoming important benchmarks in object detection. This gap limits the understanding of how YOLO compares to these models, particularly in contexts like real-time street navigation for the visually impaired. Our paper aims to address this by evaluating and comparing the effectiveness of Transformer-based models against other state-of-the-art models, focusing on their applicability in visual impairment scenarios.

3. Methodology

In this section, we outline the methodology followed and provide reasoning behind the choices made. This includes the models, dataset, and processes used to conduct the research.

3.1. Models

The models evaluated were YOLO-based and DETR-based. The goal of this paper is to evaluate the effectiveness and efficiency of the Transformer architecture in comparison with other state-of-the-art models.

Below are the DETR-based models implemented, both of which trained on COCO 2017 dataset and follow COCO labeling format when fine-tuned:

1. **DETR:** This model uses the Transformer and attention-based architecture and, when released, showed promising results; hence, we wanted to include it in this work and evaluate its performance. We utilized the ResNet-50 backbone.

2. **RT-DETR:** Real Time Detection Transformer (RT-DETR) is an enhanced implementation of the DETR model, built for increased speed and accuracy. We utilized the ResNet-50 backbone.

Below are the YOLO-based models implemented, both of which pre-trained on COCO 2017 dataset, follow the YOLO labeling format, and are maintained by Ultralytics[1], a library focused on computer vision models:

1. **YOLOv8:** This model was chosen because it is the state-of-the-art model in the YOLO family. Several sizes of the YOLOv8 model were implemented: nano, small, medium, large, and x-large.
2. **YOLOv8-Worldv2:** This variant was also tested, which employed vision-language modeling for Open-Vocabulary Detection tasks. The sizes implemented were small, medium, large, and x-large.

3.2. Dataset

We used two main datasets to conduct the experiments:

1. **COCO 2017:** Contains 80 common classes and follows the COCO labeling format. The models we utilized were already pre-trained on this dataset. Any further training does not aim to modify the existing weights with which the models were trained on.
2. **WOTR (Walking On The Road):** This dataset was created by Xia et al. [21] which also focused on vision impairment. Our paper heavily relies on Xia et al. [21]'s paper and especially the dataset. The WOTR dataset contains both COCO and non-COCO classes which are relevant to street navigation for visually impaired people. We split the classes in this dataset into 2 groups: **1) Phase 1:** classes which exist in COCO, and **2) Phase 2:** classes which don't exist in COCO. All classes here were labeled in ground truth using axis-aligned bounding boxes using the PASCAL-VOC labeling format.

Table 1
Classes of the WOTR dataset

Phase 1 Classes	Phase 2 Classes
Person, Bicycle, Bus, Truck, Car, Motorcycle, Fire Hydrant, Dog	Tree, Reflective Cone, Crosswalk, Blind Road, Pole, Warning Column, Roadblock, Litter Bin, Signs

3.3. Pre-Processing

Due to the various datasets and formats utilized, data pre-processing was an extensive task. To use the WOTR dataset, we had to first pre-process it so that it contains accurate and relevant data for our purpose. The dataset originally came in PASCAL-VOC format and therefore, we did most of the pre-processing on the PASCAL-VOC format directly, and then exported it to the desired format based on the model architecture (different models natively support different formats).

3.3.1. Formatting

1. **Images**: All formats were in jpg. The image sizes were not uniform and vary.
2. **Labels**: The dataset came in PASCAL-VOC, and after pre-processing, we exported it to YOLO format. We opted to use the YOLO labeling format as the default labeling format as it is well supported and simple to use. Moreover, it is the default format for YOLO models. When necessary, we also used other labeling formats such as COCO format.

[1] See [22] for more details.

We used the YOLO label format for several key operations: exporting predictions for all YOLO models, exporting predictions during Phase 1 for DETR/RT-DETR models, and evaluating model metrics across all models by comparing ground-truth labels with predictions. Additionally, for Phase 2 training of the DETR/RT-DETR models, we used the COCO dataset label format to import ground-truth labels so that the model can be trained on the custom weights.

3.3.2. Data Preparation

To prepare our dataset, we followed several key steps to ensure its quality and suitability for our research. We began with an audit, manually assessing the accuracy of the ground truth labels to meet our high standards.

We next standardized and refined the baseline dataset, specifically by removing the "sign" class, which we identified as unreliable and irrelevant for our use case. The original WOTR dataset contained both stop signs and directional signs within this class, which presented issues. This grouping did not align with the COCO dataset's class definitions, as COCO has a separate "stop sign" class that should not include directional signs. Additionally, the "sign" class offered minimal utility for our target users, further justifying its removal. As a result, we reviewed our PASCAL VOC labels, removed the "sign" class, and discarded any empty labels along with their corresponding images.

We then consolidated certain classes within the WOTR dataset, such as merging "red light" and "green light" into the "traffic light" class, and "tricycle" into "bicycle," to enhance detection and create more meaningful training data. Additionally, we renamed several classes to ensure consistency with the COCO dataset, aligning names like "fire_hydrant" with the standard "fire hydrant". We do these operations to ensure consistency with the COCO dataset format and ensure each class has a meaningful number of instances to train the models.

For phase-based datasets, we tailored each phase to include only relevant classes. Starting with the Phase 2 dataset (We start with the Phase 2 dataset because it is the large set between the two), we identified and retained the necessary classes, then derived the Phase 1 dataset by removing out-of-scope classes. Phase 1 uses "WOTR_v2_Phase1" dataset while Phase 2 uses "WOTR_v2_Phase2" dataset.

The datasets were then split into Training, Validation, and Testing collections using an 80/10/10 ratio, with sci-kit learn[2] ensuring randomness and bias reduction. Finally, we converted the labels from PASCAL-VOC to YOLO format, chosen for its native compatibility with the YOLO models and ease of conversion to other formats. The DETR architecture uses COCO format natively, which will be handled dynamically at a later stage described in next sections.

These steps ensured our datasets were ready for predictions, fine-tuning, and performance evaluation.

3.3.3. Model Prediction and Training

When evaluating the models, we ensured consistency and level-setting by utilizing the same hyper-parameters across all experiments. For prediction, the confidence level was set at **0.5**, and the IOU (Intersection Over Union) threshold for Non-Maximum Suppression was set at **0.8**. During training, the models were trained for **20** epochs with a batch size of **10**. The learning rate was fixed at **0.0001**, and we used the AdamW optimizer with a weight decay of **0.0001** to prevent over-fitting. For metrics calculation, the IOU threshold was consistently maintained at **0.8** to ensure uniformity in the evaluation process.

3.3.4. Post-Processing

After performing DETR-based predictions in Phase 1, it was necessary to perform post-processing to align the class numbers with the original COCO mapping. This involved converting the class names, which were outputted as names rather than numbers, into the corresponding class numbers in the YOLO label files (e.g., converting 'person' to '0'). Similarly, in Phase 2, post-processing was required not

[2]See [23] for more details.

only to ensure that the class numbers match the original COCO mapping but also to align the filenames of each label with the original filenames. This involved two steps: first, correcting the class numbers to match the original class numbers in the map file to ensure that the evaluation script correctly interprets the data; and second, correcting the filenames from the raw output to the original filenames, which is crucial for our evaluation script to work (because it compared 2 labels with same name)

3.4. Evaluation Methodology and Metrics

After the model completed its prediction processes, exported the results in YOLO label formats, and we performed the post-processing for DETR labels, a script was run to evaluate performance and calculate relevant metrics. Below is the evaluation methodology:

1. **Data Parsing and Preparation:** This included parsing YOLO .txt files to extract ground truth and predicted bounding boxes and converting YOLO box format to (x1, y1, x2, y2) corners.
2. **IoU Calculation:** Compute IoU for pairs of ground truth and predicted boxes.
3. **Metric Computation:** This included computing Precision, Recall, and F1 Score based on IoU values, calculating Average Precision (AP) for each class across various confidence thresholds, and computing Mean Average Precision (mAP) across multiple IoU thresholds.
4. **Class-wise Metrics and Confusion Matrix:** This included computing True Positives (TP), False Positives (FP), and False Negatives (FN) for each class, constructing Confusion Matrix, and computing False Negative Rate (FNR).

The below metrics are calculated:

1. **Evaluation Metrics**
   a) **Accuracy**
      i. Mean Average Precision (mAP) at various IOU thresholds:

$$mAP = \frac{1}{n} \sum_{i=1}^{n} AP_i \tag{1}$$

Where:

- AP_i is the Average Precision (AP) at the i-th IoU threshold.
- n is the total number of IoU thresholds considered.

      ii. Precision and Recall:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

      iii. False Negative Rate (FNR):

$$FNR = \frac{FN}{FN + TP} \tag{4}$$

      iv. Counts of TP, TN, FP, and FN.
   b) **Speed**
      i. Average Inference Time:

$$\text{Avg Inference Time} = \frac{1}{n} \sum_{i=1}^{n} t_i \tag{5}$$

Where:

- t_i is the inference time for the i-th sample.
- n is the total number of inference samples considered.

Table 2
Summary of mAP and Inference Time for Phase 1 and Phase 2

Phase	Architecture	Model	Params (M)	Average - mAP	Average - Inference Time (ms)
Phase 1	DETR	DETR-Original	41.6	0.3449	818.2
Phase 1	DETR	RTDETR	41.6	0.3801	560.2
Phase 1	YOLO	yolov8l	43.7	0.3954	13.9
Phase 1	YOLO	yolov8m	25.9	0.3771	11.7
Phase 1	YOLO	yolov8n	3.2	0.2951	9.4
Phase 1	YOLO	yolov8s	11.2	0.3412	9.6
Phase 1	YOLO	yolov8x	68.2	0.3997	14.0
Phase 1	YOLO-WORLD	yolov8l-worldv2	43.7	0.3675	18.3
Phase 1	YOLO-WORLD	yolov8m-worldv2	25.9	0.3469	16.7
Phase 1	YOLO-WORLD	yolov8s-worldv2	11.2	0.3033	15.0
Phase 1	YOLO-WORLD	yolov8x-worldv2	68.2	0.3728	19.3
Phase 2	DETR	DETR-Original	41.6	0.3236	
Phase 2	DETR	RTDETR	41.6	0.4387	
Phase 2	YOLO	yolov8x	68.2	0.4710	
Phase 2	YOLO-WORLD	yolov8x-worldv2	68.2	0.4681	

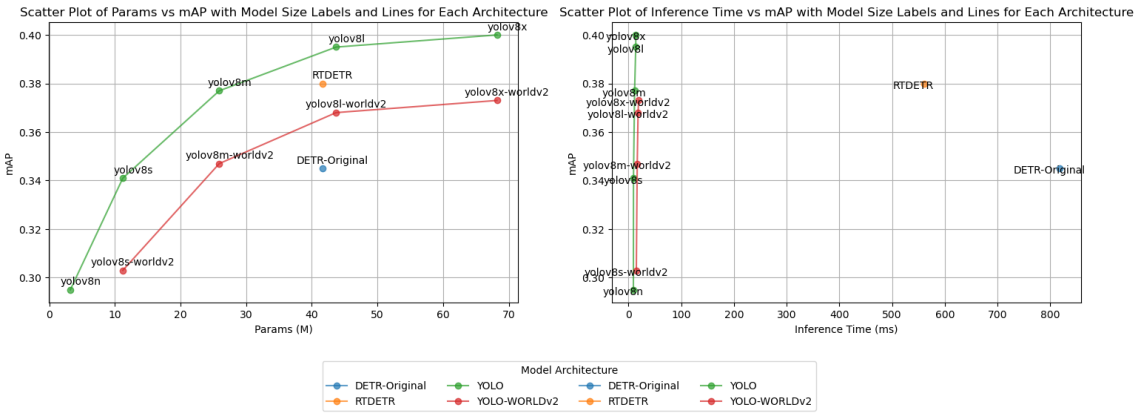

Figure 1: Phase 1 Results which include mAP and speed performance

2. **Secondary Metrics**
   a) Intersection over Union (IOU): Measures the overlap between two bounding boxes:

$$IOU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{6}$$

Please note, if we perform Phase 1 predictions, they must be benchmarked against Phase 1 Ground Truth labels, and respectively the same for Phase 2.

4. Evaluation

4.1. Phase 1 Analysis

In Phase 1, we evaluate the models in their pre-trained state. We measure both accuracy and speed.

Based on the experiment results, YOLOv8x emerged as the top performer in terms of mAP, achieving an mAP of 0.399. Several factors contributed to this result. Firstly, YOLOv8's architecture is designed for maximum accuracy and speed, utilizing a single network to predict objects and bounding boxes, which minimizes computational requirements. Additionally, YOLOv8 employs advanced techniques such as anchor-free detection, streamlining the prediction process. Furthermore, YOLOv8x, being the largest model in the YOLOv8 family with 68.2 million parameters, benefits from extensive weight training, leading to superior performance in predictions.

In comparison, YOLOv8-Worldv2 ranked second in performance, though it exhibited reduced overall accuracy. Notably, the anticipated benefits of its Open-Vocabulary detection architecture did not

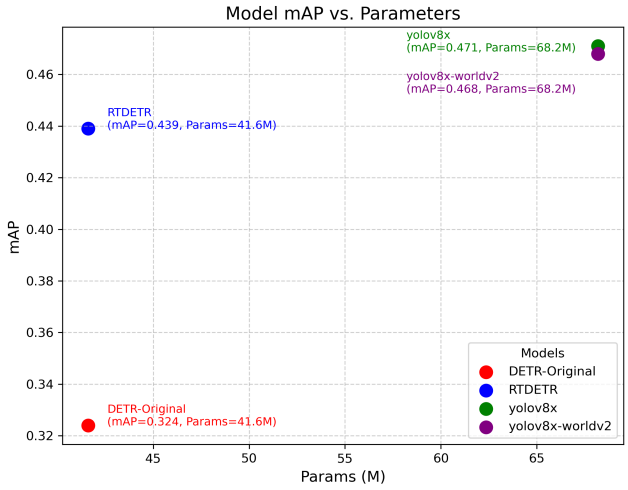

Figure 2: Phase 2 Results comparing mAP performance to model size (params)

materialize in the results, indicating that this approach did not significantly enhance performance in this context.

On the other hand, DETR-Original performed the worst in terms of both accuracy and speed, with an mAP of 0.34494 and an average inference time of 818.2 ms. Several factors contributed to this outcome. The architectural differences, particularly the use of the Transformer-based architecture, which is computationally intensive, may not be well-suited for real-time detection use cases like street navigation. Additionally, DETR utilized the ResNet-50 backbone, which, while effective, is smaller in size compared to other models evaluated, potentially limiting its performance.

However, Real-Time DETR (RT-DETR) outperformed the original DETR in both accuracy and inference time, achieving an mAP of 0.38005 and a reduced inference time of 560.2 ms. This improvement can be attributed to several architectural enhancements. RT-DETR incorporates parallel decoding, which significantly improves inference time, and its optimized DETR architecture reduces computational overhead. Moreover, RT-DETR includes an enhanced attention mechanism and benefits from hardware acceleration, allowing for better utilization of modern GPUs and thus superior performance.

The potential reasons that DETR performed worse than YOLO models could be attributed to their architecture which relies on a complex two-stage process. The process could be beneficial in capturing global context and relationships but it requires computational power. Moreover, DETR's set prediction mechanism might struggle with precise localization, especially with dense object images.

4.2. Phase 2 Analysis

As a next step, we took the best performing models in Phase 1 and conducted Phase 2 experiments using them; this included training and fine-tuning them, and performing predictions once again. We measured only mAP. In Phase 2 of the experiment, where the models were fine-tuned and trained on a custom dataset containing street navigation-specific classes not present in the COCO dataset—such as roadblocks and tactile pavement—YOLOv8x emerged as the top performer in terms of accuracy, achieving a Mean Average Precision (mAP) of 0.471.

YOLOv8-Worldv2 came in second place, with an mAP of 0.468. Real-Time DETR ranked third, with an mAP of 0.439, showing an improvement of 5.86% from Phase 1, indicating better performance after fine-tuning on the custom dataset. In contrast, DETR-Original performed the worst in terms of accuracy, with its performance even degrading from Phase 1. This degradation could be attributed to catastrophic interference or catastrophic forgetting, which can occur in models when adapting to new tasks or datasets during training. This can happen when a model is first trained on a large dataset, and then

fine-tuned with a smaller one, which happened in our case. This can lead the model to overwrite the weights and cause this "catastrophic forgetting" phenomena. According to Li and Hoiem [24], this could potentially be fixed by freezing all or some of the early layers in the backbone (in this case, ResNet-50).

Another reason could be due to the learning rate hyper-parameter which can either overshoot the optimal minima of the loss function if too high, or predict sub-optimally if too low, due to the gradient descent process converging slowly, requiring many iterations to reach an optimal or near-optimal solution. This can potentially be fixed by tweaking the learning rate hyper-parameter.

4.3. Making Sense of Results

The evaluation of the results provided valuable insights into the most suitable models for future use in street navigation for visually impaired individuals, particularly in determining whether Transformer-based models like DETR or RT-DETR are the optimal choice. The findings indicated that while Transformer-based models offer certain advantages, they fall short in both accuracy and speed when compared to YOLO models, especially YOLOv8x. The superior performance of YOLOv8x, with its high accuracy and significantly faster inference times, suggests that it is better suited for real-time applications where timely and precise object detection is crucial, such as in the context of street navigation for the visually impaired.

Therefore, in addressing the research question - whether object detection models utilizing Transformer architecture can be more effective than other state-of-the-art models in terms of accuracy and speed for street navigation - the answer is no; Transformer-based models are not better suited. The result indicated that YOLO models, particularly YOLOv8x, outperform Transformer-based models. Thus, YOLOv8x emerged as the most effective model for this application, based on the models we evaluated, offering the best balance of accuracy and speed needed to assist visually impaired individuals in navigating streets safely and efficiently.

4.4. Discussion

In our study, the findings indicated that YOLOv8 models consistently outperform Transformer-based models, particularly DETR, in both accuracy and speed within the context of street navigation for visually impaired individuals. These results align with the literature, which highlights the efficiency and accuracy of YOLO models in object detection tasks. For instance, Acar et al. [10] emphasized the improved performance of YOLOv8 in real-time applications, which is confirmed by our findings where YOLOv8 achieved a higher mAP and significantly faster inference times compared to DETR.

However, the literature also suggested the potential of Transformer-based models, particularly in handling complex object detection scenarios, as noted by Carion et al. [13] with the introduction of DETR. Despite this, our results showed that DETR models, while innovative, do not yet surpass the well-optimized YOLO architecture in scenarios requiring rapid and accurate detection, such as street navigation for the visually impaired. This highlights a gap between the theoretical advantages of Transformer and their practical application in time-sensitive environments, suggesting that further optimization is needed for Transformer models to become competitive in this domain.

Some of the limitations we faced in this paper are computational power and dataset variation. In terms of computational power, while this research had access to standard off-the-shelf computational resources (Google Colab Pro+), having dedicated resources can help with further fine-tuning of hyper-parameters, such as training for higher epochs and real-time video formats. In terms of dataset variation, it is important to have variation in the dataset in terms of **geographical location, weather, and miscellaneous objects**. Since infrastructure elements like roadblocks and tactile pavements vary by region, it is crucial to customize datasets for specific locations. Moreover, models must be trained to handle a range of weather conditions, particularly challenging scenarios like rain or fog. Additionally, including objects such as potholes or temporary construction is vital. Crowd-sourcing this data could further enhance the dataset's comprehensiveness.

5. Conclusion and Future Work

In conclusion, this paper has made several important contributions to the field of street navigation for visually impaired individuals. **First**, we conducted a detailed evaluation of object detection models utilizing Transformer architectures, assessing their performance in terms of accuracy and speed against state-of-the-art models like YOLOv8. Our key findings indicate that Transformer-based models, such as DETR and RT-DETR, were outperformed by YOLOv8x and YOLOv8x-Worldv2 in both accuracy and speed, confirming that Transformer architectures are not yet ideal for street navigation applications in this context. Despite the aforementioned limitation, this paper manages to present a reliable and thorough evaluation of the models and architectures. YOLO models outperformed Transformer-based models like DETR primarily due to architectural differences. YOLO's single-stage detection which operates on grid-based bounding boxes allows for faster and more accurate object detection, whereas DETR's two-stage process, which captures global context, requires more computational power and struggled with precise localization. DETR also experienced issues like catastrophic forgetting during training and fine-tuning, leading to degraded performance. **Second**, we introduced an enhanced version of the Walking On the Road Dataset (WOTRv2), which is specifically tailored for object detection in street navigation for the visually impaired, offering a valuable resource for future research and development in this area. **Third**, we developed and implemented a comprehensive methodology and a structured approach for evaluating object detection models for visual impairment. These contributions collectively advance the development of more accurate and efficient solutions that can better assist visually impaired individuals in outdoor navigation.

In terms of future work, firstly, incorporating a feedback mechanism is crucial; while the detection and identification of objects by the models are useful, this information must be utilized effectively. Feedback could be auditory, visual, textual, or haptic, providing essential guidance for visually impaired individuals during navigation. Secondly, integrating hardware is necessary to make the software-based solutions more practical and accessible.

References

[1] I.-M. Lee, D. M. Buchner, The importance of walking to public health, Medicine & Science in Sports & Exercise 40 (2008) S512–S518.

[2] S. R. Flaxman, R. R. Bourne, S. Resnikoff, P. Ackland, T. Braithwaite, M. V. Cicinelli, T. Vos, Global causes of blindness and distance vision impairment 1990–2020: a systematic review and meta-analysis, The Lancet Global Health 5 (2017) e1221–e1234. doi:10.1016/S2214-109X(17)30393-5.

[3] A. Yekta, E. Hooshmand, M. Saatchi, H. Ostadimoghaddam, A. Asharlous, A. Taheri, M. Khabazkhoob, Global prevalence and causes of visual impairment and blindness in children: A systematic review and meta-analysis, Journal of Current Ophthalmology 34 (2022) 1–15. URL: https://www.jcurrophthalmol.org. doi:10.4103/joco.joco_135_21.

[4] G. A. Stevens, R. A. White, S. R. Flaxman, H. Price, J. B. Jonas, J. Keeffe, J. Leasher, K. Naidoo, K. Pesudovs, S. Resnikoff, et al., Global prevalence of vision impairment and blindness: magnitude and temporal trends, 1990–2010, Ophthalmology 120 (2013) 2377–2384. doi:10.1016/j.ophtha.2013.05.025.

[5] M. M. Islam, M. S. Sadi, K. Z. Zamli, M. M. Ahmed, Developing walking assistants for visually impaired people: A review, IEEE Sensors Journal 19 (2019) 2814–2827. doi:10.1109/JSEN.2018.2890423.

[6] S. Khan, S. Nazir, H. U. Khan, Analysis of navigation assistants for blind and visually impaired people: A systematic review, IEEE Access 9 (2021) 26712–26729. doi:10.1109/ACCESS.2021.3052415.

[7] G. Dimas, D. E. Diamantis, P. Kalozoumis, D. K. Iakovidis, Uncertainty-aware visual perception

system for outdoor navigation of the visually challenged, Sensors 20 (2020) 2385. URL: https://www.mdpi.com/1424-8220/20/8/2385. doi:10.3390/s20082385.

[8] M. O. Masud, M. F. Rahman, M. R. Islam, M. S. Hossain, M. K. Rahman, M. M. Hasan, A smart assistive system for the visually impaired using raspberry pi and machine learning, IEEE Access 10 (2022) 11650–11659. doi:10.1109/ACCESS.2022.3146320.

[9] R. B. Islam, S. Akhter, F. Iqbal, M. S. U. Rahman, R. Khan, Deep learning based object detection and surrounding environment description for visually impaired people, Heliyon 9 (2023) e16924. URL: https://www.sciencedirect.com/science/article/pii/S2405844023004127. doi:10.1016/j.heliyon.2023.e16924.

[10] T. Acar, A. Solmaz, A. S. Bozkir, I. Cengiz, From pixels to paths: Sight - a vision-based navigation aid for the visually impaired, in: 2024 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2024. doi:10.1109/HORA61326.2024.10550694.

[11] A. B. Atitallah, Y. Said, M. A. B. Atitallah, M. Albekairi, K. Kaaniche, S. Boubaker, An effective obstacle detection system using deep learning advantages to aid blind and visually impaired navigation, Ain Shams Engineering Journal 15 (2024) 102387. doi:10.1016/j.asej.2023.102387, available online 16 July 2023.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-end object detection with transformers, in: European conference on computer vision, Springer, 2020, pp. 213–229.

[14] X. Zhu, W. Su, L. Li, X. Wang, J. Dai, Deformable detr: Deformable transformers for end-to-end object detection, in: International Conference on Learning Representations, 2020.

[15] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, J. Chen, Detrs beat yolos on real-time object detection, 2024. URL: https://arxiv.org/abs/2304.08069. arXiv:2304.08069.

[16] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[17] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[18] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, Y. Shan, Yolo-world: Real-time open-vocabulary object detection, 2024. URL: https://arxiv.org/abs/2401.17270. arXiv:2401.17270.

[19] J. Terven, D.-M. Córdova-Esparza, J.-A. Romero-González, A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas, Machine Learning and Knowledge Extraction 5 (2023) 1680–1716. URL: https://www.mdpi.com/2504-4990/5/4/83. doi:10.3390/make5040083.

[20] O. K. T. Alsultan, M. T. Mohammad, A deep learning-based assistive system for the visually impaired using yolo-v7, Revue d'Intelligence Artificielle 37 (2023) 901–906. URL: http://dx.doi.org/10.18280/ria.370409. doi:10.18280/ria.370409.

[21] H. Xia, C. Yao, Y. Tan, S. Song, A dataset for the visually impaired walk on the road, Displays 79 (2023) 102486. URL: https://www.sciencedirect.com/science/article/pii/S0141938223001191. doi:10.1016/j.displa.2023.102486.

[22] G. Jocher, A. Chaurasia, J. Qiu, Ultralytics yolov8, 2023. URL: https://github.com/ultralytics/ultralytics.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[24] Z. Li, D. Hoiem, Learning without forgetting, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (2018) 2935–2947. doi:10.1109/TPAMI.2017.2773081.