National College of Ireland

Street Navigation for Visual Impairment using CNN and Transformer Models

MSc Research Project
Masters of Science in Artificial Intelligence

Hasan Ali
Student ID: 22142291

School of Computing
National College of Ireland

Supervisor: Faithful Onwuegbuche

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Hasan Ali
Student ID:	22142291
Programme:	Masters of Science in Artificial Intelligence
Year:	2024
Module:	MSc Research Project
Supervisor:	Faithful Onwuegbuche
Submission Due Date:	12/08/2024
Project Title:	Street Navigation for Visual Impairment using CNN and Transformer Models
Word Count:	XXX
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Hasan Ali
Date:	16th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Street Navigation for Visual Impairment using CNN and Transformer Models

Hasan Ali
22142291

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.

1 Introduction

This paper aims to solve the problem of visual impairment in the context of street navigation. We believe it is an important activity for the visually impaired people to walk outside as it is healthy for the mind (better well-being), the body (increased cardiovascular activity), and socially (to interact with other people). Hence, we see a great opportunity to use machine learning (ML) models to increase the quality of life of visually impaired people. We define street navigation as walking on an outdoor street by foot publicly. The major contribution of this paper is to evaluate whether Transformer-based object detection models can be a better solution than other state-of-the-art models in street navigation for visually impaired individuals.

In the application of street navigation for visually impaired people, it is important to have both an accurate and a fast model to ensure that the visually impaired person has the information needed to make decisions on the street. The data that is needed is split into two parts: **detection** of objects of concern and **identification** of these objects. The model must first detect objects that are of concern, as these objects can influence decision-making or lead to significant consequences, such as falling, tripping, or being exposed to hazards or moving objects. Examples of objects of concern include

people, trees, poles, and tactile pavement, while non-examples include stores, the sky, and airplanes - essentially any object outside the set of objects of concern. Once the model detects an object of concern, it is equally important to identify these objects to assist in making the appropriate decision at that moment, such as determining whether the detected object is a person or a dog.

It is crucial to have a fast model to ensure that the feedback provided to visually impaired people is timely. If the model detects and identifies objects of concern but does so with a delay, the information becomes less useful or even irrelevant. Therefore, we will evaluate the performance of the selected models in two main areas: **accuracy** and **speed**. In terms of accuracy, the primary focus of this thesis is to assess the model's ability to detect ground truth labels accurately. We will emphasize the Precision metric, as it is vital to detect objects correctly rather than miss them, making precision more critical than Recall in this context. To evaluate accuracy, we will use the Mean Average Precision (mAP). In addition to accuracy, we will measure the speed of the model, particularly its inference time, which will be recorded in milliseconds (ms). This measurement is essential to ensure that the model can provide real-time feedback necessary for effective navigation assistance. We aim to evaluate the performance of Transformer-based object detection models in comparison with state-of-the-art models to be discussed in the related work. All of the Transformer-based models are pre-trained on the COCO 2017 dataset.

1.1 Research Question

Can object detection models that utilize the Transformer architecture be more effective (in accuracy and speed) than other state-of-the-art models in the context of street navigation for visually impaired people?

In order to answer the research question, we need to achieve the following objectives:

1. Understand the problem and design suitable experiment structure.
2. Identify relevant dataset and prepare dataset for the experiments.
3. Conduct Phase 1 and Phase 2 Experiments using pre-trained and fine-tuned weights, respectively.
4. Evaluate results based on the relevant metrics.

1.2 Limitations and Assumptions

Conducting research such as this involves a lot of nuance; hence it is imperative to identify and mention the limitations of the experiment and the assumptions taken.

1.2.1 Assumptions:

We assume that, through the literature review conducted, we will be able to identify the most effective state-of-the-art models suitable for our study. This assumption is crucial as it guides the selection of models that are expected to perform optimally in the given context.

Additionally, we assume that the dataset employed in this study is adequately representative of the relevant classes and situations pertinent to our use-case. This assumption

ensures that the findings and outcomes of the study will be applicable and meaningful in real-world scenarios that align with the intended application.

1.2.2 Limitations:

Our dataset, while comprehensive, has some limitations that affect the generalizability and applicability of our findings. Firstly, **country specificity** is a notable concern. Although the dataset includes images from various countries, a significant portion of these images are taken in Asia. This presents a limitation because infrastructure elements like roadblocks and tactile pavements differ from country to country. As a result, a more accurate and usable experiment would involve focusing on images from one specific country at a time, ensuring that the model is tailored to the nuances of that region.

Secondly, there is an issue with **weather specificity**. The dataset predominantly features images taken during daylight and under clear skies. This is not fully representative of the conditions under which visually impaired individuals might need to navigate streets. In reality, they may require assistance in various weather conditions, particularly in adverse conditions such as rain or fog, where the need for object detection is even more critical to avoid potential hazards. Therefore, a more accurate and practical experiment would include a broader range of weather conditions, with a particular emphasis on ambiguous and challenging weather scenarios.

Finally, the **input format** used in this experiment is limited to still images. While this provides valuable insights, extending the experiment to include video formats would be highly beneficial. Video data is more representative of how an object detection application would be utilized in real-world scenarios, where continuous motion and real-time processing are essential for effective navigation assistance.

1.3 Experiment Structure

We follow the CRISP-DM method in this project. The main experiment structure is:

1. Understand the problem we're trying to solve
2. Prepare the data and conduct any pre-processing needed
3. Conduct the experiments: We structure the experiment in 2 main phases:
   (a) Phase 1 will evaluate the model's performance based on their default pre-trained weights. The classes we will evaluate the models on are exclusive to the COCO dataset. The evaluation method in Phase 1 pertains to **predicting** on classes based on their pre-trained weights.
   (b) Phase 2 will evaluate the model's performance based on their fine-tuned state. Firstly, we **train and fine-tune** the models on the custom dataset. Secondly, we **predict** using the fine-tuned weights. The classes used in Phase 2 are COCO classes in addition to custom, additional classes.
4. Evaluate results

When conducting the prediction methods, we will ensure that the model outputs YOLO-format labels, which contain the predictions, as well as image outputs that visualize the predictions using bounding boxes (when possible). Moreover, when training the model, the custom weights are exported for future use.

1.4 Structure of the Report

1. Related Work: This report begins with a comprehensive literature review, surveying relevant papers in the field. We will first examine research related to vision impairment, followed by studies focused on object detection models, and then delve into model-specific papers. Additionally, we will discuss any limitations identified in the literature and highlight gaps that warrant further investigation.

2. Methodology, Design, and Implementation: The report will then outline the methodology employed in the experiment to ensure transparency. This section includes a detailed description of the datasets and models used, the steps taken to pre-process and post-process the data, and the methods followed to assess the metrics. We will also address any limitations of our methodology as necessary.

3. Evaluation: Next, we will evaluate the performance of the models based on the experiment results, focusing on the metrics most relevant to our use-case. This section aims to provide answers to the research question presented earlier in the report.

4. Discussion and Conclusion: Finally, we will discuss potential future implementations and the scope of the research, including the opportunities and challenges encountered. The report will conclude with a summary of the findings, comparison with literature, and the next steps for further research.

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 represents a significant global health issue with considerable prevalence rates. According to a study conducted by Flaxman et al. (2017), 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, whether it manifests as blindness or moderate to severe impairment, substantially impacts individuals' quality of life (Yekta et al.; 2022). It restricts their ability to engage in activities traditionally designed for those with full eyesight. 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 economic burden associated with visual impairment is multifaceted,

encompassing both direct costs, such as medical expenses, and indirect costs, including loss of productivity (Flaxman et al.; 2017). Another study showcases that the prevalence of vision impairment is concentrated among older demographics, which is expected to increase due to global aging (Stevens et al.; 2013).

One approach to mitigating visual impairment is through enhanced access to eye care services, particularly in low- and middle-income countries. Such improvements encompass medical interventions, including vision correction and surgical procedures. Furthermore, raising public health awareness by promoting education and increasing access to preventative measures, such as nutritional supplements and healthy lifestyle practices, is essential (Steinmetz et al.; 2021). In addition to these measures, the utilization of technology to support existing cases presents a significant opportunity. Notably, the application of machine learning and artificial intelligence offers the potential for a substantial positive impact, given the large number of individuals affected and the severity of the issue addressed.

While these studies provides a comprehensive overview of the global burden of visual impairment and suggests various mitigation strategies, it lacks 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 (Islam et al.; 2019). Other methods, including infrared, laser, GPS, or RFID-based technologies, have limitations in recognizing specific objects or providing detailed information (Khan et al.; 2021). 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 (Dimas et al.; 2020). 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 (Masud et al.; 2022).

Islam Islam et al. (2023) 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. (2024) 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. (2024) 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 study 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, particularly in computer vision and object detection. A notable breakthrough is the implementation of transformer architecture, which enhances performance, especially in accuracy.

Introduced in "Attention is All You Need" by Vaswani et al. (2017), the transformer architecture represents 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. (2020) 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. (2020), 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. (2024), 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. (2016), 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 (Redmon and Farhadi; 2017).

YOLO-World extends 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, enables 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 (Cheng et al.; 2024).

Terven et al. (2023) discuss YOLOv8, which introduces 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 (2023) highlight YOLOv7's practical benefits in enhancing environmental interaction and independence. Xia et al. (2023) propose 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 are YOLO architecture-based models and DETR-based models. 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. DETR uses the Transformer architecture and is a promising model for our use-case. DETR employs a CNN backbone.

Below are the DETR-based models implemented:

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 thesis and evaluate its performance. DETR is trained on the COCO 2017 dataset and follows the COCO labeling format. We utilize 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. It is also pre-trained on the COCO dataset and utilizes the ResNet-50 backbone.

Below are the YOLO-based models implemented:

1. **YOLOv8:** This model was chosen because it is the state-of-the-art model in the YOLO family. This model is maintained by Ultralytics, a library focused on the YOLO family of computer vision models. This model was pre-trained on the COCO 2017 dataset. There are several variations of the YOLOv8 model based on different sizes (nano, small, medium, large, and x-large). This model follows the YOLO labeling format.

2. **YOLOv8-Worldv2:** We also test this variant of YOLO, which employs vision-language modeling for Open-Vocabulary Detection tasks. We make use of the Ultralytics library for this model and implement the small, medium, large, and x-large sizes.

3.2 Dataset

We will use two main datasets to conduct the experiments:

1. **COCO 2017:** The COCO dataset contains 80 common classes and follows the COCO labeling format. The models we are utilizing are already pre-trained on this dataset. Any further training will not aim to modify the existing weights with which the models were trained on.

2. **WOTR (Walking On The Road):** The dataset used is called Walking On The Road. It was created by Xia et al. (2023) who wrote the paper which also focuses on vision impairment street navigation. Our paper heavily relies on Xia et al. (2023)'s paper and especially the dataset. The WOTR dataset contains both COCO and non-COCO (new) classes which are relevant to visually impaired people street navigation. The classes in this dataset are split into 2 main groups: **1) Classes in COCO:** person, bicycle, bus, truck, car, motorcycle, fire hydrant, dog and **2) Classes not in COCO:** tree, reflective cone, crosswalk, blind road, pole, warning column, roadblock, litter bin, signs. All classes here were labeled in ground truth using axis-aligned bounding boxes using the PASCAL-VOC labeling format (however, we may use other formats at later stages).

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

Table 1: Classes

3.3 Pre-Processing

Due to the different formats we use and the extensive datasets we utilize, pre-processing of the data is an extensive task in this paper. To use the dataset, we had to first pre-process it so that it contains accurate and relevant data for our purpose. The dataset originally comes in PASCAL-VOC format and therefore, we will do most of the pre-processing on the PASCAL-VOC format labels directly, and then export 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 comes in PASCAL-VOC, and after pre-processing, we export 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 use other labeling formats such as the COCO labeling format.

We will use 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 will use 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.

Next, we standardized and improved the baseline dataset, focusing on the removal of the "sign" class, which we found unreliable and irrelevant for our use-case. The class included both stop signs and directional signs, causing issues during evaluation and offering little utility to our target users, visually impaired people. As a result, we scanned our PASCAL VOC labels, removed the "sign" class, and deleted any empty labels and their corresponding images to reduce noise.

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."

For phase-based datasets, we tailored each phase to include only relevant classes. Starting with the Phase 2 dataset (We start with 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.

The datasets were then split into Training, Validation, and Testing collections using an 80/10/10 ratio, with SKLearn 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. 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 Equipment and Applications

Equipment Used: Local resources, including CPU and GPU, were employed for tasks that were not resource-intensive. However, predictions and training were primarily conducted using Google Colab. Training of the DETR architecture specifically required the GPU instances available in Google Colab.

Labeling Software: We utilized several tools for image labeling, auditing labels, and managing class structures. LabelImg and Label Studio were employed for viewing and labeling images, as well as exporting datasets into the necessary formats. Additionally, Roboflow was used both as a labeling tool and for managing the format of labels to ensure consistency across different stages of the process. Roboflow is important for use with DETR architecture as it allows for COCO format exporting of labels used for training.

Dataset Management: To manage the datasets used for training, testing, and validation, a combination of Google Drive and Roboflow was utilized. Google Drive was prioritized for organizing files into appropriate folders and directory structures when possible. Roboflow also played a significant role in dataset management, offering storage solutions and Python API access, which facilitated consistent and efficient handling of the datasets throughout the experiments, especially DETR models.

3.3.4 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.

3.3.5 Post-Processing

After performing DETR-based predictions in Phase 1, it is necessary to perform post-processing to align the class numbers with the original COCO mapping. This involves converting the class names, which are 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 is required not 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 involves 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 completes its prediction processes, exports the results in YOLO label formats, and we perform the post-processing for DETR labels, a script is run to evaluate performance and calculate relevant metrics. Below is the architecture of our evaluation methodology and metrics:

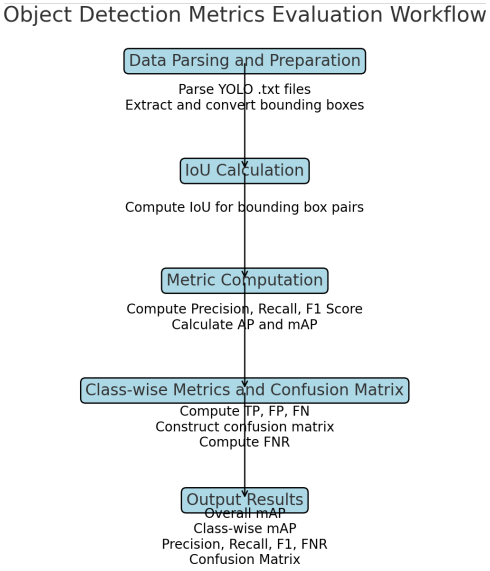

Figure 1: Metrics Calculation Script

1. **Data Parsing and Preparation:** This includes 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 includes 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 includes 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:

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

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:

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

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 Design Specification

This project leverages deep learning techniques and advanced neural network architectures to address the challenge of street navigation for the visually impaired. The two primary architectures to be used are DETR (Detection Transformers) and YOLO (You Only Look Once). These architectures were selected for their strengths in accuracy and speed.

4.1 Architectures

1. **DETR:** DETR models employ a transformer-based architecture that eliminates the need for traditional processes like anchor generation and non-maximum suppression. The architecture is particularly well-suited for capturing global context. The variants used are the original DETR model and an enhanced version: Real-Time DETR. RT-DETR was included due to its claimed improvements in inference speed.

2. **YOLO:** YOLO models are known for their single-stage object detection process, which makes them highly efficient. These models predict bounding boxes and class probabilities directly from full images in a single evaluation, making them great for real-time applications. The variants used are YOLOv8 and YOLOv8-Worldv2, which integrates vision-language modeling for open vocabulary detection.

4.2 Framework and Tools

Our implementation made use of several frameworks and tools which facilitated development, training, and evaluation:

- **Ultralytics:** We utilized Ultralytics for implementing YOLO models as this library provides a robust and efficient environment for training and deploying the models on custom datasets.
- **Transformers and PyTorch:** These frameworks were essential for implementing DETR models. PyTorch, in particular, facilitated fine-tuning and evaluation.
- **Google Colab:** The prediction and training of models needed intensive computing resources, especially for DETR training. It was conducted using Google Colab (A100 GPU).

5 Implementation

In the final stage of the project, the focus was on the practical application and integration of the models. This section outlines the steps taken, the tools, languages, and outputs produced during implementation.

5.1 Model Training and Fine-Tuning

The implementation involved fine-tuning pre-trained models on the custom dataset.

- **YOLO:** The X-Large size of YOLOv8 and YOLOv8-Worldv2 were selected for fine-tuning based on their performance in the Phase 1 analysis. These models were trained using the Ultralytics library. The models were trained using Google Colab, and hyper-parameters such as learning rate, batch size, and the number of epochs were optimized to ensure the best performance. The training was monitored to prevent over-fitting.

5.2 Output Generation

When running the predictions, the models produced key outputs:

- **Prediction Labels:** Each model generated prediction labels in the YOLO format, which were then used to calculate the mAP. These labels included the bounding boxes and class names of the detected objects.
- **Visualized Predictions:** For qualitative analysis and when possible, the bounding boxes predicted by the models were overlaid on the original images, providing a visual representation of the model's performance.

- **Inference Time Reports:** Detailed logs of inference times were generated, providing insight into the real-time applicability of each model.

5.3 Tools and Programming Languages

The implementation phase made extensive use of the following tools and programming languages:

- **Python:** The primary programming language used for data processing, model training, evaluation, and output generation. Libraries such as PyTorch, Ultralytics, and OpenCV played a crucial role in these processes.
- **Roboflow:** Used for dataset management, including pre-processing and format conversion, ensuring consistency across the different phases of the implementation.

5.4 Final Outputs of the Experiment

1. **Trained Models:** Fine-tuned versions of YOLOv8x, YOLOv8-Worldv2, DETR-Original, and RT-DETR, ready for deployment in real-world applications.
2. **Evaluation Reports:** Comprehensive reports detailing the performance of each model, including mAP scores, inference times, and qualitative assessments based on visualized predictions.
3. **Processed Datasets:** Enhanced and Improved WOTR dataset in YOLO format, along with the associated ground truth labels, archived for future reference and further experimentation.

6 Evaluation

Below is the summary of the findings from the experiments conducted.

Phase	Architecture	Model	Params (M)	Average - mAP	Average - Inference Time (ms)
Phase 1	DETR	DETR-Original	41.6	0.34494	818.2
Phase 1	DETR	RTDETR	41.6	0.38005	560.2
Phase 1	YOLO	yolov8l	43.7	0.3953731	13.9
Phase 1	YOLO	yolov8m	25.9	0.3770995	11.7
Phase 1	YOLO	yolov8n	3.2	0.2950705	9.4
Phase 1	YOLO	yolov8s	11.2	0.3412478	9.6
Phase 1	YOLO	yolov8x	68.2	0.399727	14
Phase 1	YOLO-WORLD	yolov8l-worldv2	43.7	0.3675223	18.3
Phase 1	YOLO-WORLD	yolov8m-worldv2	25.9	0.3468728	16.7
Phase 1	YOLO-WORLD	yolov8s-worldv2	11.2	0.3032878	15
Phase 1	YOLO-WORLD	yolov8x-worldv2	68.2	0.3728289	19.3
Phase 2	DETR	DETR-Original	41.6	0.3236444	
Phase 2	DETR	RTDETR	41.6	0.4386833	
Phase 2	YOLO	yolov8x	68.2	0.471	
Phase 2	YOLO-WORLD	yolov8x-worldv2	68.2	0.4681389	

Table 2: Summary of mAP and Inference Time

6.1 Phase 1 Analysis

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

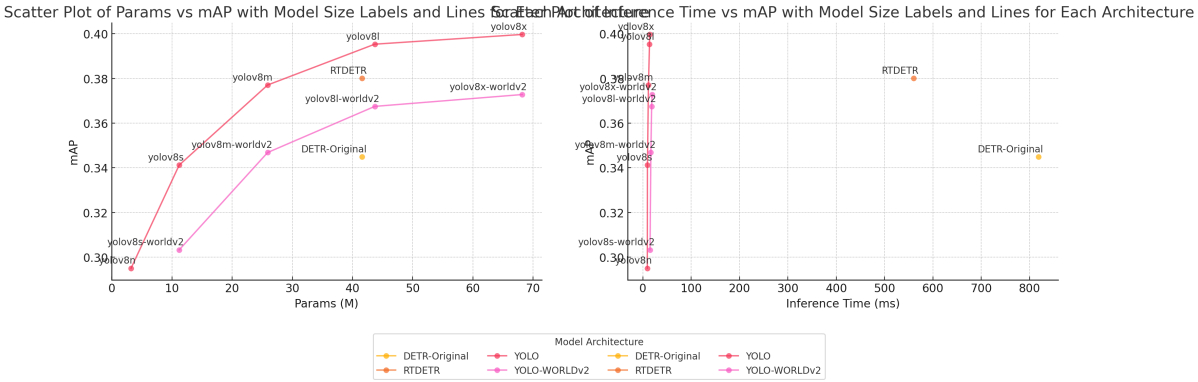

Figure 2: Phase 1 Results

Based on the data analyzed and evaluated after running the models, YOLOv8x emerged as the top performer in terms of accuracy, achieving a Mean Average Precision (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 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 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.

6.2 Phase 2 Analysis

As a next step, we take the best performing models in Phase 1 and conduct Phase 2 experiment on them; this includes training and fine-tuning them, and performing predictions once again. We measure only accuracy.

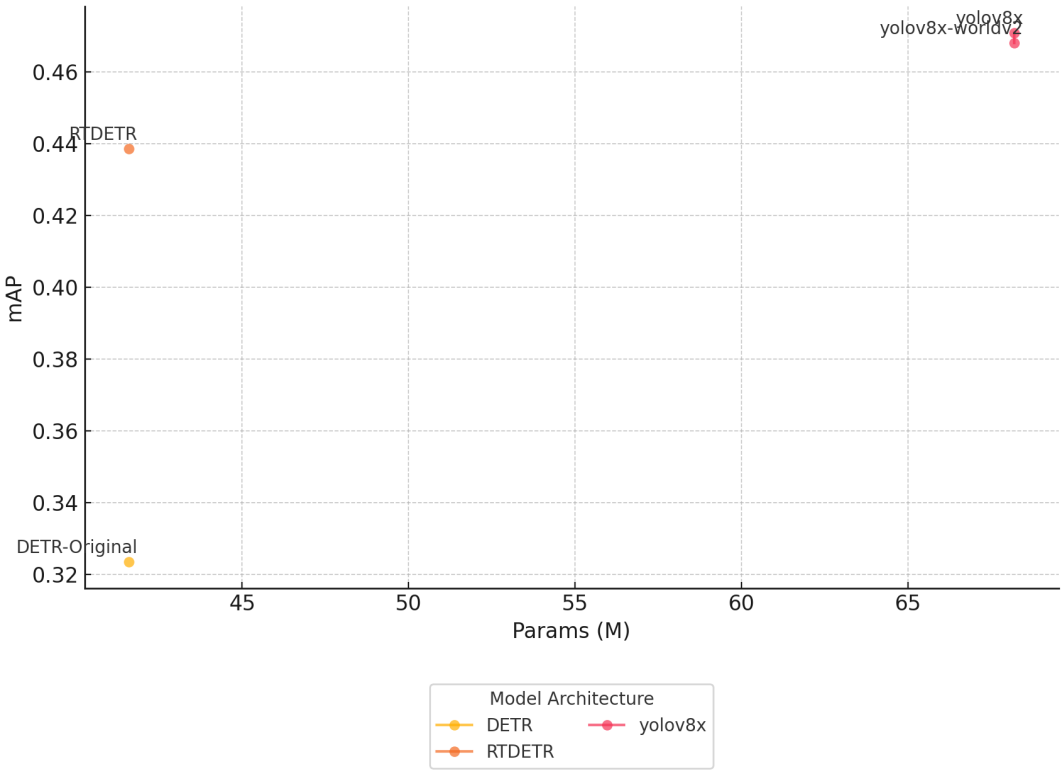

Figure 3: Phase 2 Results

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 0.438, 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. 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.

6.3 Making Sense of Results

The evaluation of the results provides 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 indicate 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 indicates that YOLO models, particularly YOLOv8x, outperform Transformer-based models. Thus, YOLOv8x emerges 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.

6.4 Discussion

In our study, the findings indicate 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. (2024) 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 suggests the potential of Transformer-based models, particularly in handling complex object detection scenarios, as noted by Carion et al. (2020) with the introduction of DETR. Despite this, our results show 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 Transformers and their practical application in time-sensitive environments, suggesting that further optimization is needed for Transformer models to become competitive in this domain.

While this research is a first step towards helping visually impaired people navigate the streets, it is important to note certain limitations such as:

1. **Models:** While the most prominent state-of-the-art models were included in this research, there is an opportunity to validate and conduct this experiment on more models for a more comprehensive landscape analysis.

2. **Computational Limitations:** Training computer vision models is resource-intensive. While this research had access to standard off-the-shelf computational resources, additional resources in the future can unlock a more robust evaluation methodology and fine-tuning, such as training for higher epochs and for real-time or video-based formats.

3. **Dataset:** The dataset used had limitations in terms of variety, including country, weather, and classes. A more varied dataset can help us assess the models more comprehensively.

4. **Evaluation specificity:** While having a varied dataset helps us validate on more situations and use-cases, having a specific validation methodology (such as walking while raining, walking in a specific country, etc) can help us evaluate models more accurately.

7 Conclusion and Future Work

This research aimed to validate if models utilizing Transformer architecture can outperform other state-of-the-art models in the application of street navigation for visual impairment. The primary metrics we used are Mean Average Precision (mAP) and Inference Time. The research successfully achieved the goal by building and utilizing a robust evaluation methodology and an enhanced and improved dataset. The research's key findings are that models with Transformer architecture were outperformed by other state-of-the-art models such as YOLOv8x and YOLOv8x-Worldv2 in both accuracy and speed. This confirms that DETR and RT-DETR are not ideal for street navigation for visual impairment. DETR-based models performed with less accuracy both in their pre-trained and fine-tuned state. Moreover, they were significantly slower in performing inference than YOLO. The research highlights the importance of selecting models that not only achieve high accuracy but also deliver timely feedback in real-world applications where speed is important. Future research can explore further optimizations and the integration of additional environmental factors to enhance model performance in diverse real-world scenarios.

7.0.1 Future Work

In terms of future work, several opportunities present themselves for further development. Firstly, addressing the limitations identified in the current study (mentioned in an earlier section) will help us deep-dive further with more specificity and accuracy. Secondly, 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. Thirdly, integrating hardware is necessary to make the software-based solutions more practical and accessible. This could involve using smartphones, specialized camera systems, or smart glasses to assist visually impaired users. Additionally, measuring the computational requirements of the models is important to ensure the models can be effectively integrated into hardware solutions. Lastly, building a more robust, specific, and varied dataset is essential to solve for specific activities to be performed by visually impaired people, ensuring that the models are trained on data that meets the specific needs of the application.

References

Acar, T., Solmaz, A., Bozkir, A. S. and Cengiz, I. (2024). From pixels to paths: Sight - a vision-based navigation aid for the visually impaired, *2024 IEEE Conference on*

Computer Vision and Pattern Recognition (CVPR), IEEE.

Alsultan, O. K. T. and Mohammad, M. T. (2023). A deep learning-based assistive system for the visually impaired using yolo-v7, *Revue d'Intelligence Artificielle* **37**(4): 901–906.
URL: *http://dx.doi.org/10.18280/ria.370409*

Atitallah, A. B., Said, Y., Atitallah, M. A. B., Albekairi, M., Kaaniche, K. and Boubaker, S. (2024). An effective obstacle detection system using deep learning advantages to aid blind and visually impaired navigation, *Ain Shams Engineering Journal* **15**: 102387. Available online 16 July 2023.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. and Zagoruyko, S. (2020). End-to-end object detection with transformers, *European conference on computer vision*, Springer, pp. 213–229.

Cheng, T., Song, L., Ge, Y., Liu, W., Wang, X. and Shan, Y. (2024). Yolo-world: Real-time open-vocabulary object detection.
URL: *https://arxiv.org/abs/2401.17270*

Dimas, G., Diamantis, D. E., Kalozoumis, P. and Iakovidis, D. K. (2020). Uncertainty-aware visual perception system for outdoor navigation of the visually challenged, *Sensors* **20**(8): 2385.
URL: *https://www.mdpi.com/1424-8220/20/8/2385*

Flaxman, S. R., Bourne, R. R., Resnikoff, S., Ackland, P., Braithwaite, T., Cicinelli, M. V. and Vos, T. (2017). Global causes of blindness and distance vision impairment 1990–2020: a systematic review and meta-analysis, *The Lancet Global Health* **5**(12): e1221–e1234.

Islam, M. M., Sadi, M. S., Zamli, K. Z. and Ahmed, M. M. (2019). Developing walking assistants for visually impaired people: A review, *IEEE Sensors Journal* **19**(8): 2814–2827.

Islam, R. B., Akhter, S., Iqbal, F., Rahman, M. S. U. and Khan, R. (2023). Deep learning based object detection and surrounding environment description for visually impaired people, *Heliyon* **9**(6): e16924.
URL: *https://www.sciencedirect.com/science/article/pii/S2405844023004127*

Khan, S., Nazir, S. and Khan, H. U. (2021). Analysis of navigation assistants for blind and visually impaired people: A systematic review, *IEEE Access* **9**: 26712–26729.

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

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

Redmon, J. and Farhadi, A. (2017). Yolo9000: better, faster, stronger, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271.

Steinmetz, J. D., Bourne, R. R., Briant, P. S., Flaxman, S. R., Taylor, H. R., Jonas, J. B., Aboyans, V., Abrès, M., Abu-Gharbieh, E., Afshin, A. et al. (2021). Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to vision 2020: the right to sight: an analysis for the global burden of disease study, *The Lancet Global Health* **9**(2): e144–e160.

Stevens, G. A., White, R. A., Flaxman, S. R., Price, H., Jonas, J. B., Keeffe, J., Leasher, J., Naidoo, K., Pesudovs, K., Resnikoff, S. et al. (2013). Global prevalence of vision impairment and blindness: magnitude and temporal trends, 1990–2010, *Ophthalmology* **120**(12): 2377–2384.

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

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

Xia, H., Yao, C., Tan, Y. and Song, S. (2023). A dataset for the visually impaired walk on the road, *Displays* **79**: 102486.
URL: *https://www.sciencedirect.com/science/article/pii/S0141938223001191*

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

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

Zhu, X., Su, W., Li, L., Wang, X. and Dai, J. (2020). Deformable detr: Deformable transformers for end-to-end object detection, *International Conference on Learning Representations*.