National College of Ireland

# A Comparative Analysis of YOLOv8 and Vision Transformers for Road Damage Detection and Classification Using the RDD2022 Dataset

Practicum II
MSc Artificial Intelligence

Abdul Ahad Shaikh
Student ID: 23380756

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

**National College of Ireland**
**Project Submission Sheet**
**School of Computing**

National College of Ireland

| | |
|---|---|
| **Student Name:** | Abdul Ahad Shaikh |
| **Student ID:** | 23380756 |
| **Programme:** | MSc Artificial Intelligence |
| **Year:** | 2025 |
| **Module:** | Practicum II |
| **Supervisor:** | Vikas Sahni |
| **Submission Due Date:** | 11/12/2025 |
| **Project Title:** | A Comparative Analysis of YOLOv8 and Vision Transformers for Road Damage Detection and Classification Using the RDD2022 Dataset |
| **Word Count:** | 5680 |
| **Page Count:** | 19 |

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:** | Abdul Ahad Shaikh |
| **Date:** | 11th December 2025 |

## 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): | |

**Abstract**

The problem of road damage detection is a long-standing issue of road authorities, because manual inspection is time-consuming, labour-intensive, and has the tendency of inconsistency. This study meets the requirement of the automation, precision, and scalability of detection strategies that can detect different types of road defects in various settings. To address this requirement, this paper compares two recent deep-learning models, including YOLOv8 that is a object-level detector and Vision Transformers (ViT), a model that is an image-level classification network. What this work contributes is the adoption of an end-to-end pipeline, i.e. the conversion of the dataset into a trainable format, evaluation and the comparative analysis of the various models, and also illustrating how each model performs on the RDD2022 dataset. The experiments demonstrate that YOLOv8 can reach an mAP at 0.50 of 0.5627, i.e. more than 56 percent of predicted bounding boxes overlap ground truth at 50 or above, and ViT can gain higher accuracy 71.48 at validation, which means that the model is capable of high performance in single-label classification. The findings indicate that YOLOv8 is more compatible with the existing state-of-the-art localisation strategies, and ViT has more useful applications in larger categorisation problems. Nevertheless, there are still shortcomings when it comes to the way ViT recognizes numerous small cracks, and YOLO perceives repair types as visual uncertainties.

# 1 Introduction

## 1.1 Background and Motivation

The high rate of urbanization of road systems has led to the need of effective and scalable road maintenance systems. Manual methods of inspection are still slow, labour inductions and subjective to the human factor thus, causing delay in the identification of any critical flaws along the roads like cracks and potholes. Deep learning has created the opportunities of automated road assessment with the progress in computer vision. CNN-based models have shown satisfactory performance but there are existing challenges which are attributed to the changing lighting conditions, the difference of the quality of cameras, diverse texture on the road and the detailed structures of most types of cracks. These restrictions note the necessity to be more flexible and able to generalise detection strategies.

## 1.2 Methods of deep learning: YOLO and Vision Transformers

There are two opposing paradigms of deep learning that dominate the current computer vision. YOLOv8 is an object detector which can identify multiple defects in an image in real-time by the prediction of multiple bounding boxes and classes. Vision Transformers (ViT), conversely, are based on self-attention which classifies an image or an extracted part of an image by learning the global connections between patches. Although CNN-based detectors such as YOLO are still popular in road damage detection, transformer-based models are still new in that field, which is why it is both timely and significant to compare them.

## 1.3 Gap in Research and Study Purpose

Even though earlier experiments have demonstrated that CNN-based detectors do well in the identification of road-damage, there is a paucity of literature that compares directly the CNN-based detectors with transformer-based classifiers on the same dataset under controlled experimental settings. This insufficiency of comparison confines practitioners to an inability to choose the most appropriate model family to use in the automated road inspection pipelines. This paper aims to fill this gap by comparing YOLOv8 and ViT on RDD2022 using the data on their accuracy, localisation, and strength in heterogeneous road conditions.

## 1.4 Research Questions

The following research questions are used in this study:
**RQ1**: How the use of YOLOv8 and Vision Transformer in relation to the RDD2022 road-damage dataset vary in terms of accuracy, localisation, and robustness?
**RQ2**: What are the effects of the architectural features of convolution-based identifiers and transformer-based classifiers on image patterns representing small, irregular, and visually small road-damage?
All these questions are expected to discover not only the differences in the quantitative performances but also the underlying architectural behaviours.

## 1.5 Research Objectives

The paper plans to prepare RDD2022 data referring to both YOLOv8 and ViT, train YOLOv8 on multiple classes to identify road-damages and use that to detect multi-class data and use mAP to measure it, train ViT classifier on region patches and use it to evaluate it with accuracy and recall measures and finally make a comparative analysis highlighting the strength, weaknesses and practical suitability of each of the models to automated road-monitoring scenarios.

The rest of the report will be structured as follows: the literature review will present the previous research on the topic of road-damage detection through CNN and transformer models. The section of methodology describes the experiment design, data preparations, and evaluation plan. The system level and architecture information is provided in the design specification. The implementation section outlines the process of development, tools, and training set up. The evaluation part shows the quantitative results and the comparative results of YOLOv8 and ViT. Lastly, the conclusion provides a round-up of the insight and limit of the study and possible future searches.

# 2 Related Work

This section places the current research in the wider academic context of automated road damage detection. The existing literature can broadly be separated into two strands: (1) object detectors based on convolutional neural networks (CNNs), such as YOLO, and (2) more recent transformer-based architectures, such as Vision Transformers (ViT) and Swin Transformer. Each group is critically assessed in terms of methodology, contribution, strengths, and limitations. The section concludes by identifying gaps that motivate the comparative YOLO–ViT analysis carried out in this research.

## 2.1 Road Damage Detection with CNNs

Early approaches relied on classical image processing methods such as edge detection, thresholding, and morphological operations. Although computationally inexpensive, these methods perform poorly in real-world conditions. A comprehensive review by Du et al., 2021 [1] showed that such traditional pipelines lack the robustness needed to handle variations in illumination, shadows, occlusions, and diverse pavement textures.

A major shift occurred with the release of annotated datasets. Maeda et al., 2018 [2] introduced the first multi-country Road Damage Dataset (RDD), enabling deep learning methods to significantly outperform classical vision algorithms. Their experiments demonstrated that CNN-based detectors achieve strong accuracy but still face persistent issues such as class imbalance and difficulty detecting very thin cracks.

YOLO quickly became a dominant architecture due to its real-time inference and strong localisation capabilities. Redmon & Farhadi, 2018 [3] introduced YOLOv3 with multi-scale predictions and a feature pyramid design, while Bochkovskiy et al., 2020 [4] proposed YOLOv4, integrating spatial attention, advanced augmentation, and PANet-based feature aggregation.

Later studies adapted YOLO specifically for road damage detection. Munir et al., 2023 [5] designed a practical road inspection pipeline using YOLO, demonstrating its feasibility for real-time deployment.Wan et al., 2022 [6] proposed YOLO-LRDD, a lightweight YOLOv5-based model that effectively detects small cracks while remaining efficient for embedded devices. More recently, Jiang , 2024 [7] optimised YOLOv8 for the RDD2022 dataset, reporting improved mAP and inference speed suitable for real-world deployment.

Despite these advances, CNN-based detectors still exhibit limitations. Multiple studies report reduced performance on extremely thin cracks, domain shifts across geographical regions, and challenges in modelling long-range structural continuity Du et al., 2021 [8]. Because convolutions primarily capture local spatial context, CNNs can struggle to encode global patterns such as long, connected crack networks or subtle structural variations across entire pavement surfaces.

## 2.2 Transformer-Based Models and Remaining Challenges

The introduction of the Vision Transformer (ViT) by Dosovitskiy et al., 2020 [9] marked a paradigm shift in computer vision. ViT demonstrated that pure self-attention architectures can outperform convolutional neural networks when trained on sufficiently large datasets such as ImageNet-21k. Unlike convolutions, transformers model long-range dependencies via global attention, making them theoretically well-suited to capturing extended cracks and structural pavement patterns.

Transformers have since been explored for pavement distress analysis. Chen et al., 2022 [10] proposed a fast ViT-based asphalt pavement classifier, achieving higher accuracy and better interpretability than several CNN baselines. Tang et al., 2022 [11] introduced PicT, a slim vision transformer that achieves high throughput and improved recognition of pavement distress. Tong et al., 2023 [12] proposed an evidential transformer for thin crack segmentation, achieving state-of-the-art accuracy. Guo et al., 2023 [13] introduced the Crack Transformer (CT), a Swin Transformer–based architecture capable of detecting long and complex cracks under noisy conditions.

However, transformer-based models also have notable drawbacks. Their high representational capacity typically requires large training datasets for stable optimisation. ViT underperforms on smaller, domain-specific datasets without extensive pretraining

Dosovitskiy et al., 2020 [9]. Road-damage datasets such as RDD are relatively small and imbalanced by transformer standards, increasing the risk of overfitting. Furthermore, most transformer-based pavement work focuses on image-level classification or segmentation rather than bounding-box detection, limiting direct applicability to end-to-end road inspection systems comparable to YOLO.

Recent comparative work has begun examining transformers versus CNNs in infrastructure inspection. Zhang et al., 2025 [14] compared ViT and CNN models for UAV-based pavement distress detection and found that ViT achieved the highest classification accuracy. However, this study also focuses only on image-level distress classification and does not compare transformer-based object detectors (e.g., DETR) with high-performing YOLO variants on a unified benchmark dataset such as RDD2022.

### 2.3 Summary of Gaps

In summary, CNN-based YOLO detectors provide strong real-time performance and robust localisation but struggle with very small defects and global structural reasoning. Transformer-based models excel at modelling global context and capturing extended crack patterns but require large training datasets and have limited use in bounding-box road damage detection. Critically, the existing literature lacks a systematic, controlled comparison of YOLO and a transformer-based model on the same multi-class road damage dataset. This research addresses that gap by comparing YOLOv8 and a ViT-based classifier on the RDD2022 dataset, analysing detection accuracy, computational efficiency, robustness, and practical deployability.

## 3 Research Methodology

It was based on a systematic experimental design to compare the capability of two fundamentally different deep learning schemes including YOLOv8 and Vision Transformer (ViT) into automated road harm identification and categorisation. The whole process was modeled to mimic the operation of a real world road inspection system: identifying defects on the roads in the full-resolution street images, isolating the defects, and lastly classifying the defects into categories.

The pipeline included eight steps that include dataset organisation, annotation processing, data cleaning, YOLOv8 training, masking the areas of predictable damage, ViT classification, quantitative assessment, and concluding comparative analysis. This workflow is presented in the Figure 1.

In the methodology, particular attention was paid to the nature of the RDD2022 data that comprises a wide variety of road images in a variety of countries and the processing limitations of the available environment (only CPU training). These limitations defined the approach to methodology and guaranteed that the results would be applicable to the conditions of the real low-resource deployment process.

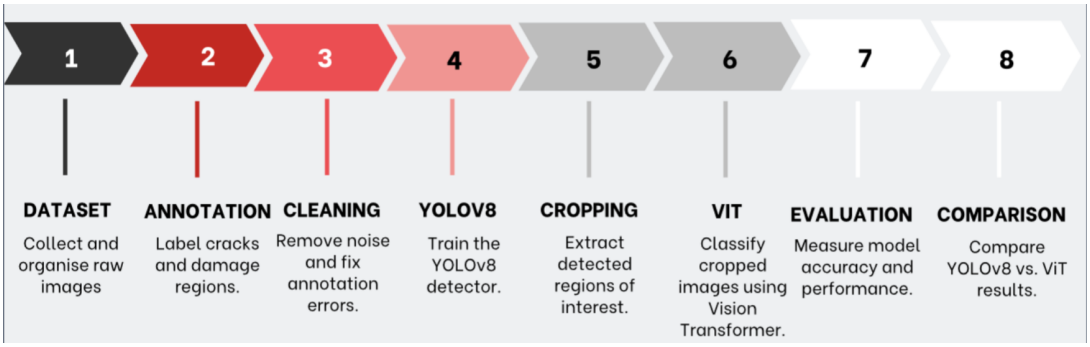



Figure 1: ***Workflow showing the eight-stage methodology used in this study, from dataset preparation to final model comparison.***

## 3.1 Dataset Preparation

The study employed the RDD2022 road damage dataset from Dataset Ninja: `https://datasetninja.com/road-damage-detector`, which comprised of 47,420 road pictures in Japan, India, Czech Republic, Norway, the United States, and China. There are one or more road defects annotated in each picture with five categories, alligator crack, longitudinal crack, transverse crack, pothole, and other damage.

All the images in this project were placed into one dataset structure, which was compatible with YOLOv8. Heterogeneity of the dataset (changes in the lighting, weather changes, road texture, and camera quality and country specific road environment) significantly contributed to the methodology. This variance made the experiment test the extent of generalisation of YOLOv8 and ViT when they are tested outside of controlled lab environments.

## 3.2 Annotation Processing

The RDD2022 gives annotation Pascal VOC XML, which is a format of bounding-box annotation. In the case of YOLOv8, they had to be transformed into YOLO text format:

```
classid, xcenter, ycenter, width, height (normalized)
```

This was necessary since YOLOv8 uses normalised coordinate system in prediction of bounding boxes. In this project, caution processing the annotations was completely automated, although, it was necessary to handcheck since there were discrepancies in the original dataset.

This conversion enabled the YOLOv8 model to be used on all road images and make accurate predictions of bounding-boxes during training and inference.

## 3.3 Data Cleaning

In this project, it was observed that the raw annotations had a number of critical problems that might impact the performance of the training in a negative way:

- The overlapping or redundant bounding boxes.
- Improper order of coordinates (e.g. xmin not the smallest)
- Area enclosure of very tiny pixels.
- Categories of defects that are not given or labeled.

The latter errors were particularly critical to the use of YOLOv8 which heavily depends on the correct supervisory signals in regressing bounding-boxes. To be studied within the context of this research, all the annotations that were problematic were located and either deleted or fixed.

Through this cleaning process, model stability was greatly enhanced, as well as lessening aggressiveness of losses and enhancing the accuracy of detection since there were no falsely bounding boxes that depicted non-real damage on the roads.

## 3.4 YOLOv8 Training

The main object detector in this article was YOLOv8. Its two-fold feature, i.e. to identify the location of a damage as well as the type of the damage, renders it to be applicable to road inspection in the real world.

- YOLOv8 was trained for 50 epochs.
- It had 78.5% images for training and 21.5% images for testing.
- The resolution of the input was adjusted to 640x640.
- The training was fully CPU based, and this was to resemble a low-resource deployment setting.
- Horizontal flips, colour jitter, scaling and mosaic transforms were augmented to make the datasets more varied.

A significant portion of the methodology was monitoring the performance improvements and declines of YOLOv8 with epochs (particularly the capacity of this model to identify thin cracks that often hide beneath asphalt patterns). The prediction of YOLOv8 resulted in the creation of the bounding boxes which constituted the input of the ViT phase.

## 3.5 Cropping of Determined Damage Areas

After the training and validation of YOLOv8, road defects in the dataset were detected using the algorithm. All the identified bounding boxes were cropped to create a new image data set with the damage area.

There were two functions of this stage:

- It modeled the isolation of damage points that was done by real-world systems in advance of secondary analysis.
- It provided a heading input dataset to the Vision Transformer and made sure that ViT was fed with damage-oriented images that were in standardised images as opposed to complicated images on the street.

These cropped photographs were of different shapes and clarity and they resembled actual problem in the world like motion blur, shadows interference and noise in the background.

## 3.6 Vision Transformer (ViT) Classification

The cropped patches were fed to the Vision Transformer model which was trained to classify damages into the five damage categories. In contrast to YOLOv8, the ViT is not a localisation model; the model can just predict the type of damage.

- The ViT was trained with 5 epochs because of the computational shortage.

- It had 80% images for training and 20% images for testing.
- It was not sent out in entire pictures, but in portions cut out.

This step gave the study the ability to test the handling of fine-grained, during the crack patterns, in the transformer-based model, which seems to be seen in fine, irregular lines that may be hard to identify when cut into patches. This was especially where the methodology was concerned since the architectural differences between convolutional and transformer models (YOLO) and transformer models (ViT) were emphasized.

### 3.7 Model Evaluation

The project is two-fold in the kind of evaluation metrics chosen:

**YOLOv8 was evaluated on:** Precision, Recall, mAP@50, mAP@50-95 (strict IoU thresholds) These metrics quantify quality in classification and also in localisation.

**ViT was evaluated on:** Overall accuracy, class accuracy, recall, Confusion matrix.

### 3.8 Comparative Analysis

This is the last phase of the methodology that compared YOLOv8 and ViT on performance measures, robustness and usability in real-life and actual scenarios of automated road monitoring system. Such comparison is the main asset of the study, showing the behavior of various architectural philosophies in the context of applying them to the same task in real life: convolution vs. self-attention to the task of road-inspection.

## 4 Design Specification

Here the internal architecture of the system that was used to run and compare YOLOv8 and Vision Transformer models in road-damage images are defined. The system is organized into five separate layers, the responsibility of each of which is to encapsulate particular part of the computational design. This specification provides the inner structure of the system and data representations and the module contracts that enable the system to run in a uniform model-agnostic environment unlike the methodology which concentrates on the implementation of experiments.

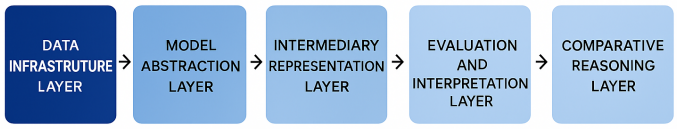



Figure 2: ***Layered System Architecture for Independent Model Processing and Comparative Analysis.***

## 4.1 Data Infrastructure Layer

DAIS is tasked with the responsibility of ensuring there is a single organisational structure to be applied to images, annotations and metadata. As the RDD2022 data is a mixed source, the system applies an internal annotation schema to abstract external representation (e.g. Pascal VOC XML) to a model-independent neutral format. This internal schema provides a steady interpretation of the areas of damage by the broader system.

## 4.2 Model Abstraction Layer

This layer is a cover of the YOLOv8 detection module and the ViT classification module. They are all viewed as autonomous subsystems that have clear input-output contracts that foster architectural neutrality. The system neither assumes nor uses any knowledge of any internal learning mechanisms- modules simply must respond to their interface specifications.

### 4.2.1 ViT Classification Output Contract.

The ViT subsystem processes each cropped on par with the logit-length fixed-length vector of classes, it can be expressed as follows:

```
[ c1, c2, c3, c4, c5 ]
```

This is forwarded to the evaluation layer which makes inter consistent predictions across architectures. Since the system is not interested in the computations of the model, being just the shape and semantics of the output, the classifier may be replaced by any other equivalent model.

## 4.3 Intermediate Representing Layer

This layer serves the central storage of organized outputs of both models. Bounding boxes, confidence scores, and spatial tags connecting the identified areas to the original images are detection artefacts. The artifacts of classification are logits, final category

decisions and metadata related to the logits. Such design will make sure that all the data streams into evaluation and comparison modules under a single and architecture-independent form. Both pixel-level geometry and categorical outputs help to compare the high-resolution yielding made by both YOLOv8 and ViT.

### 4.4 Evaluation Layer

Evaluation Layer Standardises the interpretation of a system of outputs which the system makes the artefacts of detection and classification consistent in a computational structure. Instead of having to do raw metric calculation per se, this layer packages together organized data into clear evaluation objects which can be acquired by metric engines or analysis tools. Such segregation allows metric logic to be kept apart with model execution logic, keeping architectural clarity intact.

### 4.5 Comparative Reasoning Layer.

It is a layer that converts the results of the evaluation into a clear comparative work of the two architectures. It analyses variations in robustness and error patterns, spatial sensitivity, damage-type behaviour and class imbalance responses. This layer is fundamental to it being conceptual and not computational: it gives a structural process of demonstrating how independent behaviours of a model can be translated into one comparative structure. The design can eliminate potential biases of sequential processing or interdependence of models because it isolates the comparison logic.

## 5 Implementation

This section covers the last phase of the implementation of the research, which identifies the tools, schemes, and results that were produced in the creation of the models. The aim of the phase was to convert the prepared data into working inputs to object detection and classification systems, to train and localize both the YOLOv8 and Vision Transformer (ViT) systems and to generate the reproducible artifact of the experiment.

### 5.1 Tools and Programming Environment Used

Python 3.12 was used to implement the entire project on an Apple MacBook Air (M3, 2024, 16 GB RAM) running macOS Sonoma. The project was run entirely on CPU in a simulated resource-constrained environment that matched the deployment conditions for low-infrastructure regions. AI Resources [15].

A mixture of deep learning and computer vision libraries was used, which included: PyTorch (v2.1) [16],Ultralytics YOLOv8 (v8.3.228) [17], TorchVision [18], OpenCV [19], Albumentations [20],NumPy and Pandas [21], Matplotlib and Seaborn [22][23], scikit-learn [24].

The software stack was chosen to maximize reproducibility and compatibility between object detection and transformer-based classification pipelines.

## 5.2 Outputs of Data Transformation and Preparation

The initial implementation phase consisted of converting the RDD2022 dataset Arya et al., 2024 [25] into a normalized form suitable for training both models. The original annotations were in Pascal VOC XML format and were reformatted into YOLO text format, where bounding box coordinates are normalized to values between 0 and 1. This ensures scale invariance and consistency across different image resolutions Redmon et al., 2016 [26].

Incorrect or noisy labels were identified and corrected. Bounding boxes that were duplicated, inverted, or too small were removed to avoid training instability. Data augmentation was applied to increase diversity and prevent overfitting using operations such as horizontal flipping, color jittering, scaling, translation, random erasing, and mosaic blending.

The final dataset was organized into training and validation folders containing images and corresponding label files.

## 5.3 Outputs of YOLOv8 Implementation

After dataset preparation, YOLOv8 was trained to detect multiple types of road damage. The training generated several implementation artifacts demonstrating its learned localization and classification capability.

- **Model Weights:** Two weight files were produced: `best.pt` (best validation performance) and `last.pt` (final checkpoint).
- **Detection Outputs:** For each test image, YOLOv8 generated bounding box coordinates, class labels, and confidence scores. It also produced color-annotated images for visual inspection.
- **Performance Logs:** Each epoch recorded precision, recall, and mean Average Precision (mAP), stored automatically by the Ultralytics training pipeline.

All YOLO outputs were stored under the default Ultralytics directory `runs/detect/`, ensuring compatibility with future evaluation and fine-tuning tasks Jocher et al., 2023 [27].

## 5.4 Cropped Dataset Generation

A cropped road-damage dataset was created using YOLO-generated bounding boxes. Each detected damage region was cropped, resized to 224×224 pixels, and saved with its respective class label.

This dataset allowed the Vision Transformer to specialize in identifying appearance and texture characteristics of different road damage types, rather than their spatial location Dosovitskiy et al., 2021 [9].

## 5.5 ViT Implementation Results and Visualization

The Vision Transformer was trained on the cropped dataset, generating the following artifacts:

- **Trained Model:** Saved as `vitroaddamage.pth`.
- **Training Metrics:** A CSV file containing epoch-level training and validation accuracy.

- **Evaluation Artifacts:** Confusion matrix plots and per-class accuracy visualizations.
- **Label Mapping:** A JSON file linking numeric labels to road damage categories.

These outputs constitute the full set of ViT artifacts, capable of reproducible inference and direct comparison with YOLO's performance.

### 5.6 Output Organization and Final Deliverables

At the completion of the implementation phase, the project directory contained a complete set of reproducible artifacts used for object detection and classification analysis. The main deliverables included:

- Cleaned and augmented RDD2022 dataset in YOLO format
- YOLOv8 weights (`best.pt` and `last.pt`)
- YOLO detection visualizations, logs, and predictions
- Cropped datasets generated from YOLO detections
- Vision Transformer weights (`vitroaddamage.pth`)
- Evaluation products: accuracy tables, confusion matrices, performance charts

A strict hierarchical structure ensured clear organization of results and checkpoints.

### 5.7 Summary

The implementation stage involved transforming theoretical system designs into operational machine learning pipelines. All preprocessing, training, and evaluation tasks were executed using reproducible Python workflows. The resulting datasets, trained models, cropped image sets, and performance summaries establish a clear and verifiable foundation for future analysis and replication by other researchers,, Dosovitskiy et al., 2021 [9] Jocher et al., 2023 [27].

## 6 Evaluation

The aim of this part is to display, discuss and interpret the findings as observed when comparing two models, namely YOLOv8 and Vision Transformer (ViT), for identification of road damage based on experiments conducted using RDD2022 dataset. Rather than just showing numbers, the section defines what each result represents and explains why it happened and how it answers the main research question:

> "Which architecture is most suitable for road-damage classification on real-world, small form factor dataset: YOLO-v8 or Vision Transformer?"

The assessment is organized into a series of tasks, each with the goal to evaluate a particular dimension of model behavior. For the sake of transparency and the scientific merit of any contingency, both positive and negative cases were analyzed, as the latter demonstrate architectural fragilities and are necessary for a fair comparison.

## 6.1 Experiment-1 – YOLOv8 Road-Damage Detection Performance

### 6.1.1 Experimental Setup

YOLOv8 was trained for 50 epochs on CPU-only hardware (Apple M3, 16GB of RAM). The purpose of this experiment is to quantify the detector's ability in finding and classifying damage on full-resolution images.

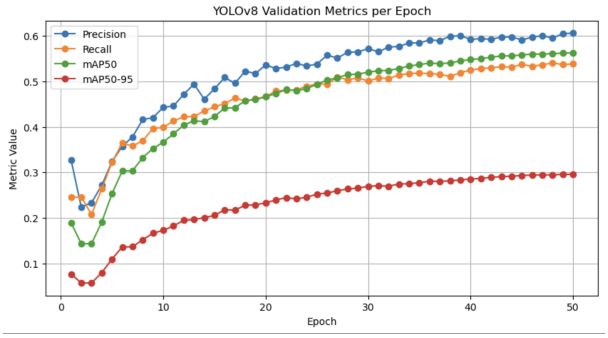



Figure 3: ***YOLOv8 validation metrics per epoch*** *The graph illustrates the continuous progress of YOLOv8 validation results throughout multiple training epochs because all four metrics precision, recall, mAP50, and mAP50-95 show consistent growth before reaching stable values during training.*

### 6.1.2 Quantitative Results

The trained YOLOv8 model achieved the following:

| Metric | Value |
|---|---|
| mAP@50 | 0.5627 |
| mAP@50–95 | 0.2961 |
| Precision | 0.606 |
| Recall | 0.538 |

Table 1: Performance metrics of the model, including mAP@50, mAP@50–95, precision, and recall.

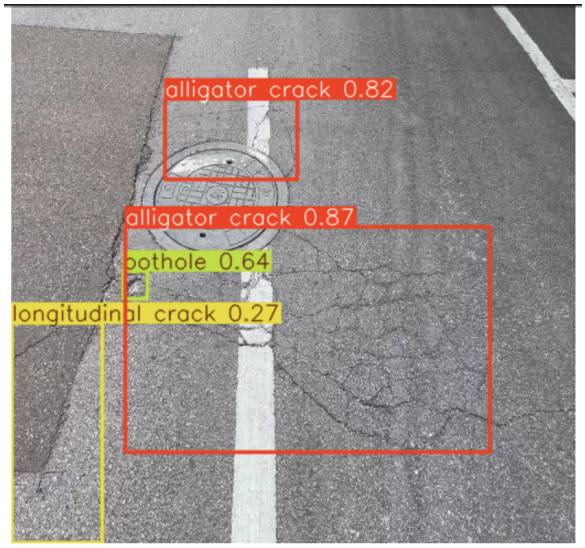
### 6.1.3 Per-Class Results and Interpretation

| Class | mAP@50 |
|---|---|
| Alligator Crack | 0.670 |
| Other Damage | 0.710 |
| Longitudinal Crack | 0.481 |
| Transverse Crack | 0.476 |
| Pothole | 0.476 |

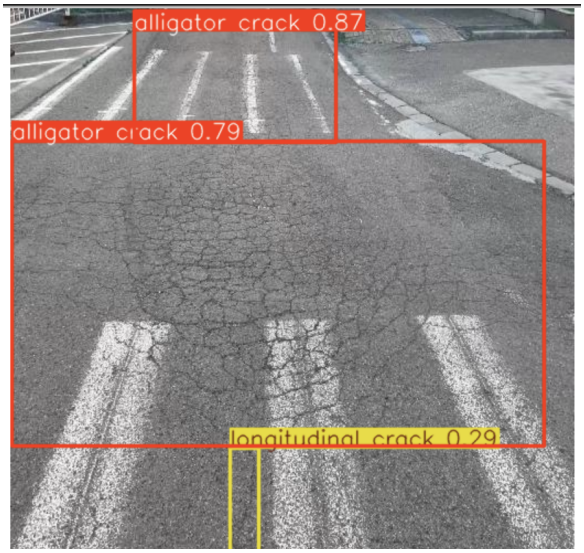
Table 2: Class-wise mAP@50 performance metrics for the model across different road damage categories.

Interpretation: Overall, YOLOv8 is the most accurate on alligator cracks and "other damage," which are larger spatial regions. Thin cracks are the least detected categories of damage because of penalizing errors in the detection of narrow areas. The problem of inconsistent lighting also affects potholes' detection recall.

### 6.1.4 YOLOv8 Detection Examples




(a) ***YOLOv8 detection showing multiple crack types with overlapping classifications in a complex road surface.***




(b) ***YOLOv8 identifying extensive alligator cracking with additional longitudinal crack detections across lane markings.***

Figure 4: ***YOLOv8 detection examples showing different crack patterns.***

## 6.2 Experiment 2 - Vision Transformer Patch Classification

### 6.2.1 Experimental Setup

The cropped patches produced from YOLO detections were classified by ViT. ViT was trained using patch size 16×16 for 5 epochs on CPU-only setup.

### 6.2.2 Quantitative Results

| Metric | Value |
|---|---|
| Validation Accuracy | 71.48% |

Table 3: Validation accuracy achieved by the model during evaluation.

### 6.2.3 Confusion Matrix Interpretation

The confusion matrix gives a comprehensive overview of the classification model of Vision Transformer (ViT) on the cropped patches obtained from YOLO detections. The matrix consists of the sum of samples belonging to the true class (rows) that were predicted

as the particular class (columns) for each cell. The values along the diagonal indicate accurate predictions, while off-diagonal cells show the number of misclassifications.

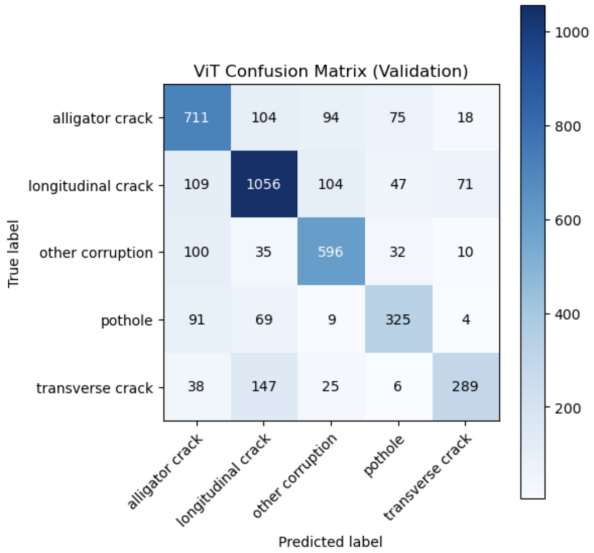



Figure 5: ***This confusion matrix illustrates how accurately the ViT model classifies each road-distress category, highlighting strong diagonal correctness but noticeable misclassifications between similar crack types.***

**1. Alligator Crack - High Accuracy but Some Confusion with Longitudinal & Transverse Cracks** ViT performed correctly in identifying 711 alligator crack samples. The cases of misclassification went mainly as follows: longitudinal crack (104 samples) other corruption (94 samples)

Reason: Alligator Cracks have very clear distinctive patterns and if only a small area of the web is cropped then the pattern might look like a tiny crack or general surface deterioration. The ViT gets only a crop and does not know the global context so it sometimes interprets a pattern as a single crack instead of a network.

**2. Longitudinal Crack - Strong Performance but Overlap with Transverse Crack** Correctly classified: 1,056 Major overlapping: transverse crack (71 samples) alligator crack (109 samples) other corruption (104 samples)

Reason: Longitudinal and transverse cracks are very thin linear features, orientation being the only difference. The ViT applies patch embeddings that sometimes do not allow for orientation hints when cracks are divided into several sections. Patch fragmentation leads the model to misclassify vertical cracks (longitudinal) as horizontal ones (transverse). The scenario of mislabeling as alligator cracks arises when the crop has intersecting fragments of cracks that look like the initial stages of alligator cracking.

**3. Other Corruption - Moderate Clarity, But Overlaps With Potholes and Cracks** Correct predictions: 596 Misclassified as: Alligator crack (100) Longitudinal crack (35) Pothole (32)

Reason: The term "Other corruption" refers to a wide range of issues including the likes of stains, patches, surface erosion, and non-crack deformations. These textures do not have a shared visual structure which results in ViT: misinterpreting roughness as

cracking (this leads to misclassification as alligator/longitudinal) or misinterpreting dark depressions or patches as potholes. This situation shows a taxonomic confusion that is present in the dataset.

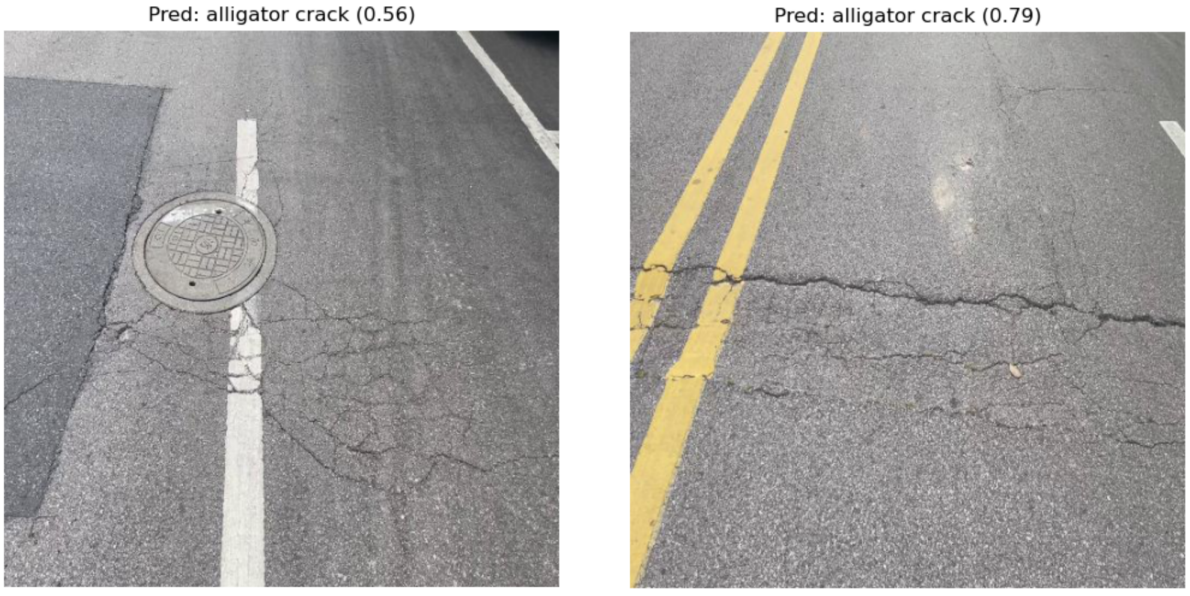
**4. Pothole - Poor Performance Compared to Other Classes** 325 predictions were correct. Misclassification by: 91 - Alligator crack 69 - Longitudinal crack 9 - Other corruption

Reason: Detection of the pothole requires ViT to have access to the shape, depth cues, and shading, which are rendered pointless by the: Flat 2D patch-based representation (no 3D geometric context). Potholes differ a lot in terms of texture, color and size. Potholes are often located in shadowed regions which causes the patch to look like a dark area or crack cluster.

**5. Transverse Crack - Ambiguous and Often Misclassified** 289 were the correct predictions. Main confusion: 147 - Longitudinal crack 25 - Other corruption

Reason: When it comes to longitudinal cracks, orientation clarity is very important but patch-based processing divides long cracks into short fragments making orientation estimation hard. So ViT sometimes guesses the crack to be longitudinal when the patch has only a part of the full crack in it.

### 6.2.4 ViT Detection Examples




**(A) alligator crack (0.56)** **(B) alligator crack (0.79)**

Figure 6: ***Vision Transformer (ViT) predictions showing alligator crack detections with confidence scores.***

## 6.3 Summary Comparison Table

| Property | YOLOv8 | ViT |
|---|---|---|
| Task | Detection (Localisation + Classification) | Classification |
| mAP@50 | 0.5627 | N/A |
| Accuracy | N/A | 71.48% |
| Handles multiple objects | Yes | No |
| Sensitive to patch size | No | Yes |
| Performs localisation | Yes | No |

The table shows a direct comparison between YOLOv8 and ViT regarding their fundamental abilities and operational results. YOLOv8 performs both object localisation and classification, achieving an mAP50 of 0.5627 and handling multiple objects without patch-size sensitivity. In contrast, ViT is a pure classifier with 0.7148 accuracy, but it cannot localise objects and is sensitive to patch size. YOLOv8 functions as a detection system but ViT operates exclusively as a single-object classification model.

## 6.4 Functional Comparison Visualisation

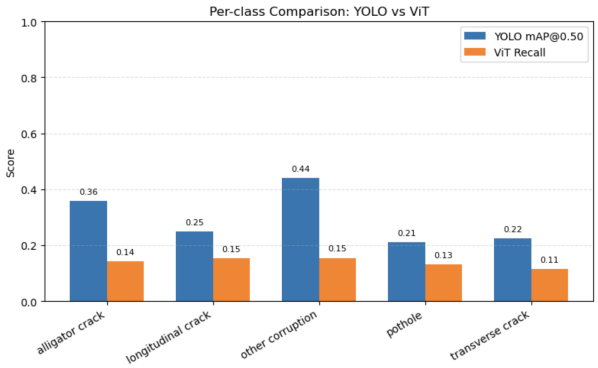



Figure 7: ***Per-class Comparison: YOLO vs ViT***

The per-class comparison shows a consistent performance gap between YOLOv8 and the Vision Transformer across all five RDD2022 damage categories. The results show that YOLOv8 performs better than all other classes at mAP@50 with the highest scores for other corruption (0.44) and alligator crack (0.36). The performance of ViT differs because it produces recall values which stay between 0.11 and 0.15 when it comes to recognizing cracks in isolated patches. The fundamental architectural difference between YOLO and ViT emerges from their design principles because YOLO uses convolutional inductive biases to maintain local structures like edges and crack textures but ViT cannot handle small fragmented crack patterns through its patch-based representation which lacks spatial locality. The chart demonstrates that YOLO outperforms other methods in road-damage detection because it identifies small structural details together with their spatial positions.

# 7 Conclusion and Future Work

This paper evaluated two complementary research questions to gain insight on how various deep-learning paradigms perform in the automated road-damage analysis. RQ1 address the accuracy and localisation capabilities and robustness of YOLOv8 and Vision Transformer (ViT) on the RDD2022 dataset, whereas RQ2 investigated the impact of the architectural design of convolutional detectors and transformer-based classifiers on their behaviour with subtle, irregular patterns of road-damage.

The analysis also indicates that YOLOv8 beats ViT in terms of accuracy, localisation, and overall strength particularly in detecting big cracks and various flaws in a unity image. The convolutional inductive biases employed in YOLOv8, allowed the model to perform well on localisation and in only CPU-only settings. ViT was not yet able to capture fine-grained fine cracks but conflicts with dataset size, patch patches, and lacked the locality as a factor. These results directly answer RQ1 and prove that YOLOv8 is the more useful model when it comes to working on real-world road-damage detection. The findings also provide the answer to RQ2 by verifying that CNNs do not lose obvious advantages when it comes to small and texture-based features but transformers demand larger datasets and more structured visual patterns to demonstrate their competitiveness.

In general, the research has achieved its goals: it has prepared the data, trained both models in similar conditions and generated similar outputs as well as conducted a systematic comparative analysis. System reveals the difference in the interpretations of the same road environment by two contrasting architectures and offers some useful results to municipalities and researchers aiming to apply the efficient automated road-inspection solutions.

## 7.1 Future Work

This can be developed further in future research in more significant ways. One of these directions is hybrid CNN-Transformer designs that could incorporate the strength of localisation of the YOLO with the ability to reason on the global scale of ViT. The other direction is to increase the datasets which include a variety of road conditions or even generate synthetic data to enhance transformer performance. The addition of time-related data such as video streams might also be useful to augment robustness, with road damage being a more apparent object across frames. Lastly, as the subsequent work, the application of lightweight models to embedded systems or edge components may be possible, and they will allow real-time monitoring of the road at the city level and pave the way to commercialisation.

# References


[1] Du et al., "Application of image technology on pavement distress detection: A review," *Measurement*, vol. 186, 2021, **URL:**`https://www.sciencedirect.com/science/article/abs/pii/S0263224121008393`.

[2] H. Maeda et al., "Road damage detection and classification using deep neural networks with smartphone images," *arXiv preprint arXiv:1801.09454*, 2018, **URL:**`https://arxiv.org/abs/1801.09454`.

[3] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018,
**URL:**https://arxiv.org/abs/1804.02767.

[4] A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020,
**URL:**https://arxiv.org/abs/2004.10934.

[5] M. Munir et al., "Automated road inspection using deep learning-based detection," *Remote Sensing*, vol. 15, no. 3, 2023,
**URL:**https://www.mdpi.com/2072-4292/15/3/.

[6] X. Wan et al., "Yolo-lrdd: A lightweight road damage detection model for embedded systems," *Remote Sensing*, vol. 14, no. 8, 2022,
**URL:**https://www.mdpi.com/2072-4292/14/8/.

[7] Jiang, "Road damage detection and classification using deep neural networks," ,, 2024,
**URL:**https://link.springer.com/article/10.1007/s42452-024-06129-0?utm_source=chatgpt.com.

[8] Du et al., "Application of image technology on pavement distress detection: A review," *Measurement*, 2021,
**URL:**https://www.researchgate.net/publication/353394701_Application_of_Image_Technology_on_Pavement_Distress_Detection_A_Review.

[9] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020,
**URL:**https://arxiv.org/abs/2010.11929.

[10] Y. Chen, X. Gu, Z. Liu and J. Liang, "A fast inference vision transformer for automatic pavement image classification and its visual interpretation method," *Remote Sensing*, vol. 14, no. 8, p. 1877, 2022,
**URL:**https://www.mdpi.com/2072-4292/14/8/1877.

[11] Tang et al., "Pict: A slim weakly supervised vision transformer for pavement distress classification," ,, 2022,
**URL:**https://arxiv.org/abs/2209.10074?utm_source=chatgpt.com.

[12] Tong, "Evidential transformer for pavement crack segmentation," *Signal Processing*, vol. 200, 2023,
**URL:**https://onlinelibrary.wiley.com/doi/10.1111/mice.13018.

[13] Guo, "Pavement crack detection based on transformer network," *Signal Processing*, vol. 200, 2023,
**URL:**https://doi.org/10.1016/j.autcon.2022.104646.

[14] Zhang et al., "Evidential transformer for pavement distress segmentation," *Machine Intelligence & Civil Engineering*, 2025,
**URL:**https://onlinelibrary.wiley.com/doi/10.1111/mice.13018.

[15] J. Howard and S. Gugger, *Deep Learning for Coders with fastai & PyTorch*. O'Reilly Media, 2020,
**URL:**https://course.fast.ai/Resources/book.html.

[16] A. Paszke et al., *Pytorch: An imperative style, high-performance deep learning library*,
**URL:**`https://pytorch.org/`, 2019.

[17] Ultralytics, *Yolov8 documentation*,
**URL:**`https://www.ultralytics.com`, 2025.

[18] TorchVision Developers, *Torchvision documentation*,
**URL:**`https://pytorch.org/vision`, 2025.

[19] OpenCV Team, *Opencv library*,
**URL:**`https://opencv.org`, 2025.

[20] A. Buslaev et al., "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020,
**URL:**`https://www.mdpi.com/2078-2489/11/2/125`.

[21] Pandas Developers, *Pandas documentation*,
**URL:**`https://pandas.pydata.org`, 2025.

[22] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007,
**URL:**`https://matplotlib.org/`.

[23] M. L. Waskom, "Seaborn: Statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021,
**URL:**`https://seaborn.pydata.org/`.

[24] F. Pedregosa et al., "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011,
**URL:**`https://scikit-learn.org/`.

[25] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal and Y. Sekimoto, "Rdd2022: A multi-national image dataset for automatic road damage detection," *Geoscience Data Journal*, vol. 11, no. 4, pp. 846–862, 2024,
**URL:**`https://rmets.onlinelibrary.wiley.com/doi/10.1002/gdj3.260`.

[26] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*,
**URL:**`https://pjreddie.com/yolo/`, 2016.

[27] G. Jocher et al., *Yolo by ultralytics*,
**URL:**`https://github.com/ultralytics/ultralytics`, 2023.