T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

CAR COMPONENTS CLASSIFICATION

ERTUGRUL KASIKCI

SUPERVISOR
DOC. DR. HABIL KALKAN

GEBZE
2024

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CAR COMPONENTS CLASSIFICATION

ERTUGRUL KASIKCI

SUPERVISOR
DOC. DR. HABIL KALKAN

2024
GEBZE

GEBZE TECHNICAL UNIVERSITY	GRADUATION PROJECT JURY APPROVAL FORM

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 13/06/2024 by the following jury.

JURY

Member
(Supervisor) : Doc. Dr. Habil KALKAN

Member : Dr. Burcu Yilmaz

ABSTRACT

This project aims to accurately classify car components, specifically distinguishing between left and right sides as well as front and back sides. The YOLOv8 (Ultralytics) computer vision model was used for this purpose. The dataset was created and labeled manually.

Car components were filmed in various environments and lighting conditions to ensure a diverse dataset. Labeling was conducted using the CVAT.AI platform. Several data augmentation techniques were applied to enhance the model's robustness. Multiple YOLO models were experimented with to identify the optimal configuration. The final model was tested using both pre-recorded videos and real-time scenarios.

Keywords: YOLO, labeling, classify, augmentation.

ÖZET

Bu proje, sol ve sag araba parçalarının önleri ve arkaları dahil olmak üzere aralarında ayrım yaparak doğru bir şekilde sınıflandırmayı amaçlamaktadır. Bu amaçla YOLOv8 (Ultralytics) bilgisayarla görme modeli kullanılmıştır. Dataset manuel olarak oluşturuldu ve etiketlendi.

Araç parçaları, genis bir dataset sağlamak için çeşitli ortamlarda ve aydınlatma koşullarında videoya alınmıştır. Etiketleme CVAT.AI platformu kullanılarak gerçekleştirilmiştir. Modelin doğruluğunu artırmak için çeşitli veri çoğaltma teknikleri uygulanmıştır. Optimum konfigürasyonu belirlemek için birden fazla YOLO modeli denenmiştir. Nihai model hem önceden kaydedilmiş videolar hem de gerçek zamanlı senaryolar kullanılarak test edilmiştir.

Anahtar Kelimeler: YOLO, etiketleme, sınıflandırma, veri çoğaltma.

CONTENTS

Abstract **iv**

Özet **v**

Contents **vii**

List of Figures **viii**

1 Problem Description **1**
1.1 Problem Overview 1
1.2 The Approach 1

2 Model **2**
2.1 YOLOv8 2
2.2 Why YOLOv8 is a Good Candidate 2
2.3 Which YOLOv8 Model to Select 2

3 Data **4**
3.1 Image Data 4
3.2 Data Preparation 4
3.3 Text Data 5
3.3.1 YOLO Format 5
3.4 Creating the Dataset 6

4 Training **8**

5 Results **9**
5.1 Training Metrics 9
5.2 Confusion Matrix 9
5.3 Label Distribution and Bounding Box Analysis 10
5.4 Real-Time Classification Result 11

6 Conclusions **13**

Bibliography **14**

CV **15**

LIST OF FIGURES

2.1 Comparison of YOLO models 3

3.1 Left Front 4
3.2 Left Back 4
3.3 Right Front 4
3.4 Right Back 4
3.5 Labeling Process using CVAT.AI 5

5.1 Training metrics including box loss, classification loss, DFL loss, precision, recall, and mAP (mean Average Precision) over epochs. 9
5.2 Confusion Matrix for the car component classification model. 10
5.3 Label distribution and bounding box analysis for the car component dataset. 11
5.4 Screenshot of real-time classification result using the YOLOv8 model. 12

1. PROBLEM DESCRIPTION

1.1. Problem Overview

In a factory setting, workers often need to separate and sort various components and pass them onto different platforms. This task can be challenging, especially when dealing with components that are very similar in appearance. Such similarities can lead to errors, resulting in inefficiencies and potential quality control issues. For instance, distinguishing between left and right car components, which may look almost identical, can be a source of mistakes.

1.2. The Approach

To address this issue, I developed a computer vision model to classify car components in real time. This model aims to minimize mistakes and potentially automate the process.

2. MODEL

2.1. YOLOv8

YOLOv8 (You Only Look Once, Version 8) is a state-of-the-art real-time object detection model developed by Ultralytics. It is particularly suitable for complex classification tasks, such as distinguishing between similar-looking car components.

Key Advantages of YOLOv8:

- **Speed and Efficiency**: Capable of real-time processing, essential for maintaining production efficiency in a factory setting.
- **Accuracy**: Advanced architecture and optimized algorithms provide high accuracy, crucial for distinguishing subtle differences between components.
- **Robustness**: Performs well under various lighting conditions and environments, ensuring reliable classification in diverse factory conditions.
- **Scalability**: Adaptable to different computational resources, making it versatile for various factory setups.
- **Ease of Implementation**: Well-documented and supported, facilitating integration into existing workflows.

2.2. Why YOLOv8 is a Good Candidate

YOLOv8's speed, accuracy, and robustness make it ideal for real-time classification of car components, minimizing errors and enhancing sorting efficiency. Its adaptability and ease of implementation further support its use in dynamic factory environments.

2.3. Which YOLOv8 Model to Select

[1] YOLOv8 offers models of different sizes, each suited for specific needs. The comparison of these models and previous versions is illustrated below:

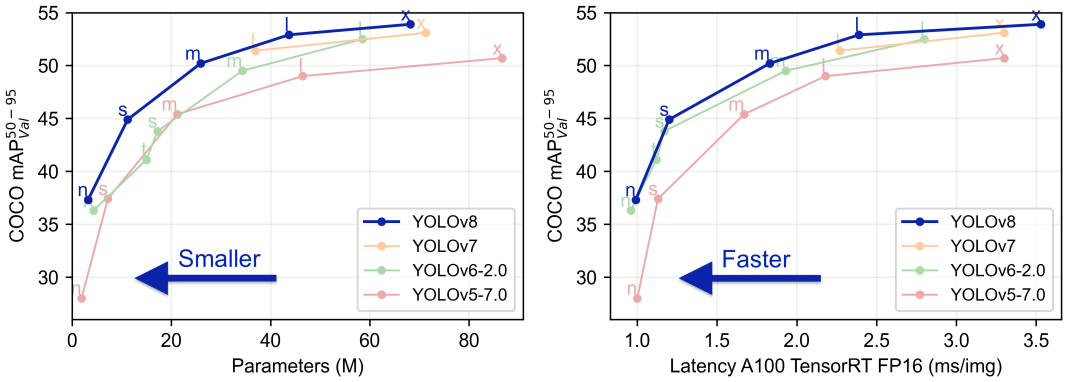

Figure 2.1: Comparison of YOLO models

After thorough testing and comparison of results, I have chosen YOLOv8n for this project. This decision was based on its balance of performance and computational efficiency.

3. DATA

3.1. Image Data

The project involves classifying two types of car components, which are mirror reflections of each other. This similarity poses a challenge for accurate classification. Therefore, both the front and back sides are classified, resulting in four distinct classes: Left Front, Left Back, Right Front, and Right Back.

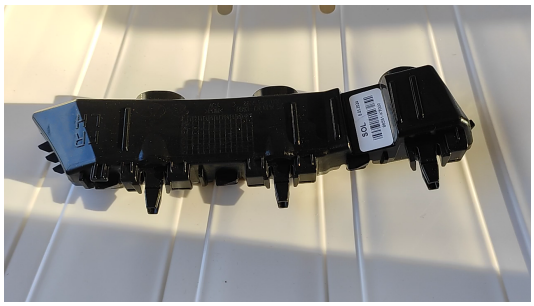

Figure 3.1: Left Front

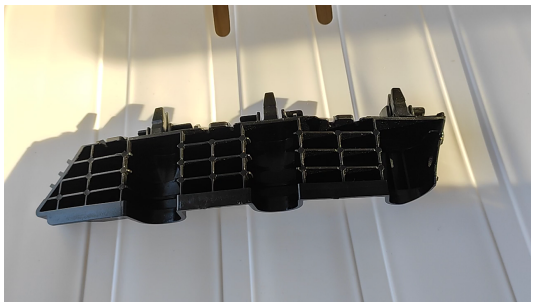

Figure 3.2: Left Back

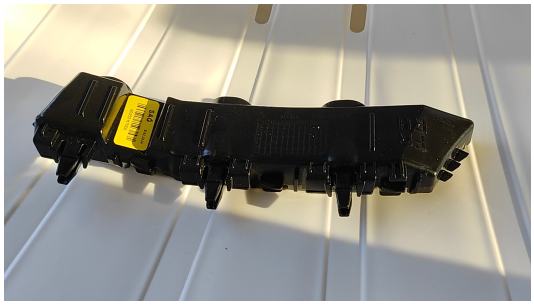

Figure 3.3: Right Front

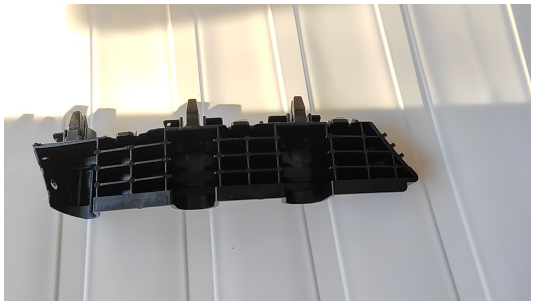

Figure 3.4: Right Back

3.2. Data Preparation

To create a comprehensive dataset, several videos were recorded in different environments and lighting conditions. Frames were extracted from these videos using the following ffmpeg command:

```
ffmpeg -i D:\Grad_Project_Files\Videos\R_B_1.mp4 -r 30.01 -q
0 -start_number 0 D:\Grad_Project_Files\Images\R_B_1\R_B_1_frame_%06d.png
```

This command specifies the frame rate, source and destination addresses, and the naming convention for the resulting frames. Consistent naming is crucial for matching image data with corresponding text data.

3.3. Text Data

Each frame must be labeled to prepare it for the computer vision model. The CVAT.AI platform was used for labeling, as illustrated below:

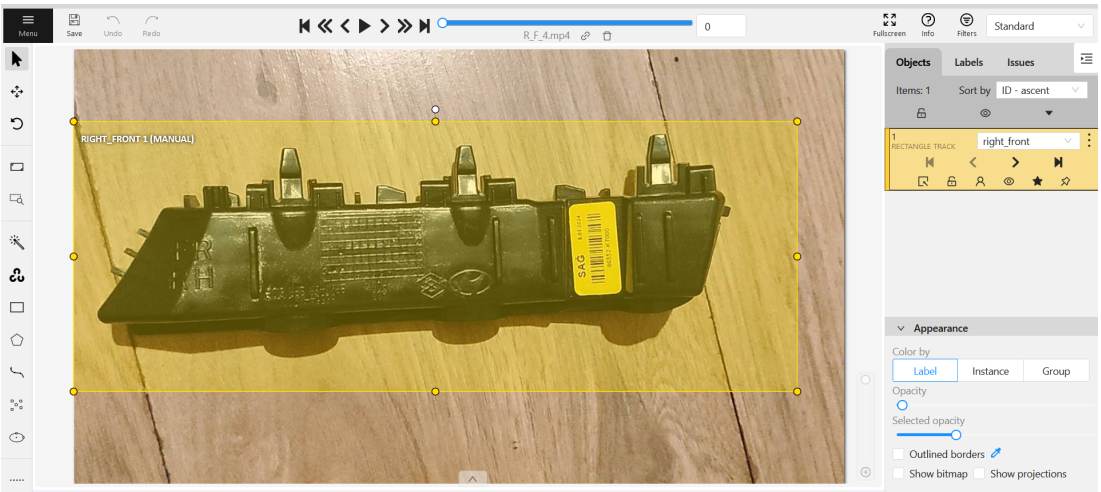

Figure 3.5: Labeling Process using CVAT.AI

Boundaries were defined, and each frame was labeled accordingly. This process generates text files for each frame, containing information about the classes, their locations, and sizes. The image data and corresponding text data must have matching names. For example, the image `L_B_1_frame_000000.png` must have a corresponding text file `L_B_1_frame_000000.txt`.

3.3.1. YOLO Format

The YOLO format for text data follows a specific structure for each labeled object within an image. Each line in the text file represents one object and contains the following information:

```
<Class ID> <x_center> <y_center> <width> <height>
```

- **Class ID**: An integer representing the class of the object.
- **Normalized bounding box coordinates**: These are the coordinates of the object's bounding box, normalized to values between 0 and 1. The coordinates

consist of four values:

- **x_center**: The x-coordinate of the center of the bounding box.
- **y_center**: The y-coordinate of the center of the bounding box.
- **width**: The width of the bounding box relative to the width of the entire image.
- **height**: The height of the bounding box relative to the height of the entire image.

For example, the line:

```
3 0.467559 0.521479 0.879023 0.521458
```

represents an object of class ID 3 with a bounding box whose center is located at approximately (0.467559, 0.521479) in normalized coordinates. The width and height of the bounding box are approximately 87.9% and 52.1% of the width and height of the entire image, respectively.

3.4. Creating the Dataset

The dataset for YOLOv8 is organized in the following folder structure format:

```
data
    images
        train
        val
    labels
        train
        val
```

The structure is represented in the data.yaml file:

```
data.yaml

# dataset root directory
path: D:\Car_Component_Classification\data_without_augmentation

train: images/train # training images
```

```
val: images/val # validation images
nc: 4 # number of classes

# class names
names: ['right_front', 'right_back', 'left_front', 'left_back']
```

This configuration ensures that the dataset is properly structured for training and validation purposes in the YOLOv8 model.

4. TRAINING

The training process proved to be more challenging than anticipated, primarily due to the similarity between car components. Various data augmentation techniques were tested; some yielded positive results, while others adversely affected the training.

The primary issue was that the components are mirror reflections of each other. Therefore, augmentation techniques like vertical and horizontal flips degraded the training quality. These techniques were automatically implemented by YOLOv8, and it took considerable time to identify this problem. After careful testing and comparisons, the following training parameters were chosen for optimal results: [2]

```
train_config = {
    'epochs': 200,
    'patience': 5,
    'batch': 16,
    'imgsz': 640,
    'dropout': 0.2,
    'lr0': 0.01,
    'lrf': 0.01,
    'momentum': 0.937,
    'weight_decay': 0.0005,
    'kobj': 1.0,
    'hsv_h': 0.015,
    'hsv_s': 0.7,
    'hsv_v': 0.4,
    'degrees': -180 - +180,
    'translate': 0.1,
    'scale': 0.5,
    'flipud': 0.0,
    'fliplr': 0.0,
    'mosaic': 0.1,
}
```

These parameters were the some of the parameters selected to balance performance and accuracy while addressing the specific challenges posed by the dataset. Disabling vertical and horizontal flips was crucial to prevent the model from confusing mirror-reflected components, thus improving the overall training effectiveness.

5. RESULTS

This chapter presents the results obtained from the training and evaluation of the YOLOv8 model. The results include loss metrics during training, precision and recall, a confusion matrix, and label distributions.

5.1. Training Metrics

The training metrics for the YOLOv8 model are depicted in Figure 5.3. These graphs show the progression of various loss metrics and performance indicators over the training epochs.

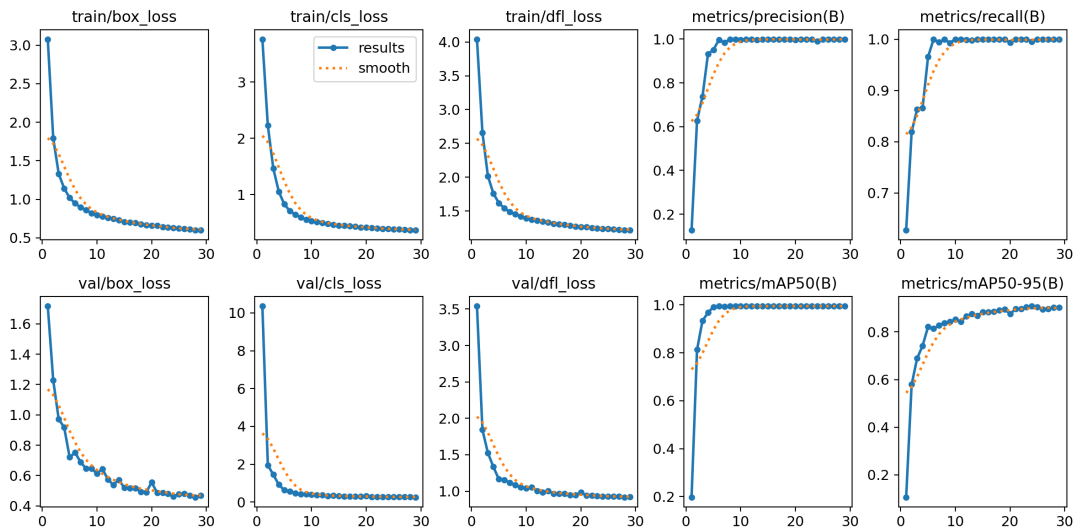

Figure 5.1: Training metrics including box loss, classification loss, DFL loss, precision, recall, and mAP (mean Average Precision) over epochs.

5.2. Confusion Matrix

The confusion matrix in Figure 5.3 illustrates the performance of the model in classifying the different classes of car components.

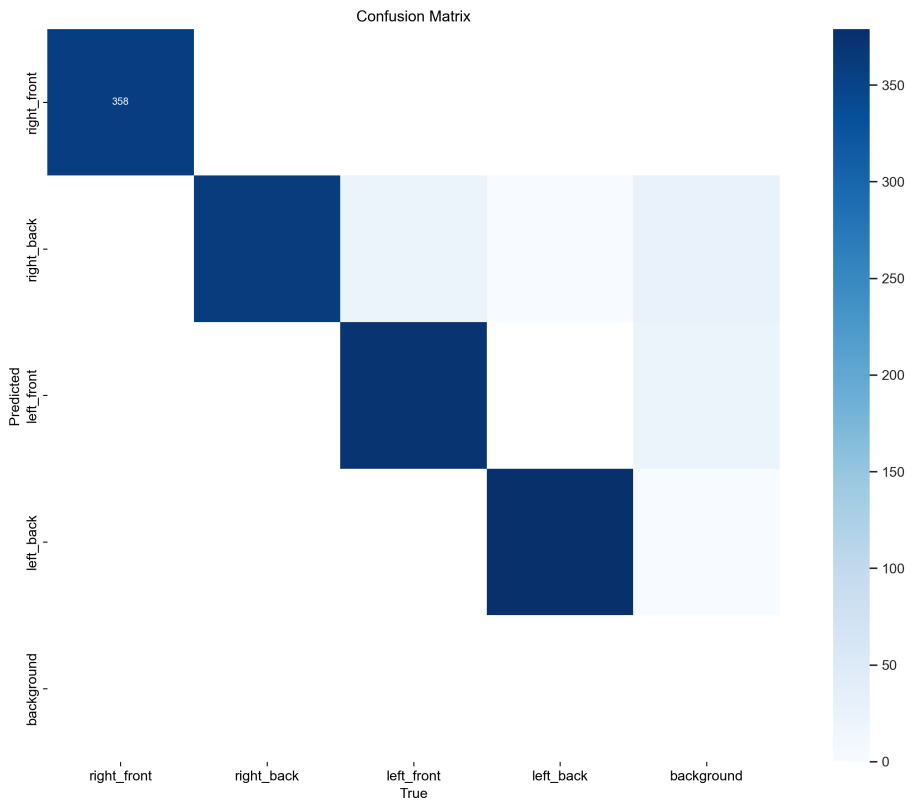

Figure 5.2: Confusion Matrix for the car component classification model.

5.3. Label Distribution and Bounding Box Analysis

Figure 5.3 presents the distribution of instances per class, along with visualizations of the bounding boxes and their parameters (position and size). These analyses help to understand the dataset and the model's behavior in terms of detecting and classifying objects within the frames.

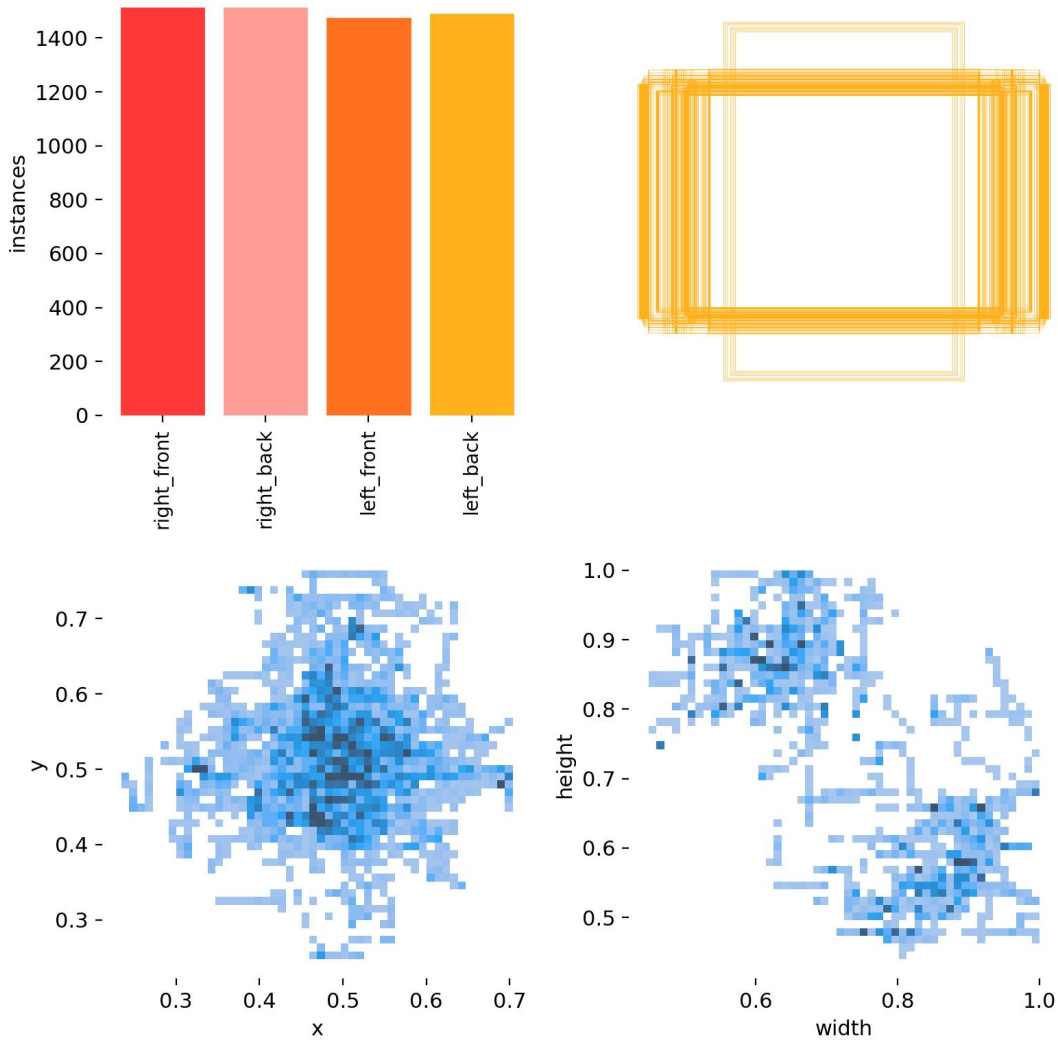

Figure 5.3: Label distribution and bounding box analysis for the car component dataset.

5.4. Real-Time Classification Result

Here, a screenshot of the real-time classification result obtained using the trained YOLOv8 model is presented. This screenshot provides a visual representation of the model's performance in detecting and classifying car components in real-world scenarios.

Figure 5.4: Screenshot of real-time classification result using the YOLOv8 model.

These results demonstrate the effectiveness of the YOLOv8 model in accurately classifying the car components, taking into account various training metrics, precision and recall values, and the overall distribution and detection performance as visualized in the graphs and matrices. Additionally, the real-time screenshot further validate the model's ability to accurately identify and classify car components in dynamic environments.

6. CONCLUSIONS

In this study, the aim was to address the challenge of accurately classifying left and right car components, which are similar in appearance and can lead to errors in factory separation processes. Leveraging computer vision techniques and the YOLOv8 model, the goal was to automate this classification task and improve efficiency in factory settings.

Through the implementation of YOLOv8, significant advancements were achieved in classification accuracy.

The training process posed challenges, particularly in handling the similarity between car components, which required careful consideration of data augmentation techniques. After thorough testing and experimentation, the most effective training configurations were identified to optimize model performance.

Additionally, the comprehensive evaluation of different YOLO model variants allowed for the selection of the most suitable architecture for the specific classification task. By considering factors such as model size, computational resources, and classification accuracy, informed decisions were made to enhance the classification system's performance.

Overall, this project demonstrates the potential of computer vision technologies in improving efficiency and accuracy in industrial processes.

BIBLIOGRAPHY

[1] "Ultralytics github page." (2024), [Online]. Available: https://github.com/ultralytics/ultralytics.

[2] "Ultralytics documents." (2021), [Online]. Available: https://docs.ultralytics.com/tr.

CV

ERTUGRUL KASIKCI

COMPUTER SCIENCE STUDENT

ekasikci2000@gmail.com

+90 501 367 80 41

https://github.com/ekasikci

www.linkedin.com/in/ertugrulkasikci

EDUCATION

Gebze Technical Universtiy
Computer Science
2020- 2024

University of Primorska
Computer Science
2022 - 2023

SKILLS

- C, C++, C#, Java, Python, Assembly, Unity, TensorFlow
- Machine Learning Deep Learning Natural Language Processing
- Redis, SQL
- Communication skills, Teamwork skills

LANGUAGE

- Advanced English
- Native Turkish

ABOUT ME

I am an enthusiastic computer science student with a passion for learning and a history of meeting deadlines with outstanding results. My diverse cultural experiences have developed my strong communication and adaptability skills. I am dedicated to staying up-to-date in technology and continuously improving. I am eager to embrace new opportunities and contribute to the industry with my skills.

EXPERIENCE

PENCE ROCKET TEAM

02/2022 - 09/2022

- I was a member of the Software department at Gebze Technical University Pence Rocket Team. In our first year at Teknofest, our team succeeded in becoming a finalist in 2022. This experience, combined with my personal endeavors in developing small games, honed my expertise in Unity and C#

ERASMUS+

09/2022 - 02/2023

- During the fall semester of 2022-2023, I participated in an Erasmus program at the University of Primorska in Slovenia.

OBSS TECHNOLOGY

07/2023 - 08/2023

- My internship was based on learning Java Spring Boot and using my skills to develop back-end applications.

TURKCELL

08/2023 - 09/2023

- During my internship as a middleware engineer, I worked on technologies such as Redis, WebLogic, Apache Kafka and developed projects.

PREVLA

03/2023 - 11/2023

- I played an active role in a European Union project focused on validating patients' movement accuracy during physical therapy. Our aim was to combine computer vision and radar technology, eliminating the need for cameras due to privacy concerns. Through collaboration, we generated and analyzed data to enhance our pattern recognition model's accuracy.