AN AUTONOMOUS BALL-PICKING MECHANIC TO REDUCE MANUAL LABOR DURING BALL RETRIEVAL USING SENSORS AND MACHINE LEARNING

Cheng Chang [1], Jonathan Sahagun [2]

[1] BASIS Independent Silicon Valley, 1290 Parkmoor Ave, San Jose, CA 95126
[2] Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Accurate object detection is crucial in various computer vision applications. This paper explores the use of deep learning for the specific task of tennis ball detection in images, a task with applications in sports analytics and automated systems [1]. We propose a deep learning-based object detection model trained in a curated dataset of annotated tennis ball images. The model utilizes convolutional neural network architecture and is optimized using various loss functions. Challenges included data scarcity and variations in tennis ball appearance, which were addressed through careful data curation and potential data augmentation strategies [2]. Experiments were conducted to evaluate the model's performance in accurately locating and classifying tennis balls, with mAP as the primary metric. Results demonstrate promising performance, showcasing the potential of our approach. This focused approach, utilizing a tailored dataset and analysis of key performance indicators, provides a solid foundation for developing robust tennis ball detection systems, potentially impacting areas like automated officiating and player performance analysis.

KEYWORDS

Robotics, Sonar distancing, Efficiency, Collection of tennis ball

1. INTRODUCTION

In sports such as tennis, golf, and baseball, the task of collecting balls scattered across the field, court, or range is a repetitive and labor-intensive process that disrupts practice sessions and consumes valuable time [3]. Players, coaches, and support staff often find themselves devoting a significant portion of their efforts to this unnecessary task, which detracts from the primary focus on skill development, gameplay, and overall performance enhancement. This issue becomes even more critical in professional and competitive training environments, where every moment is crucial for maximizing productivity and preparing for competition.

Historically, ball collection has relied heavily on manual labor or simple tools, such as ball baskets, tubes, and rollers, to accelerate the process. While these methods have provided some relief, they still require human labor, making them inefficient and laborious. The advent of

David C. Wyld et al. (Eds): ADCO, CMCA, NeCoM, NLPI, CSITEC, BDBS, DMML, SP – 2025
pp. 269-278, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150521

technology in sports has led to innovations in training equipment, yet ball collection remains an area for automation.

Frequent manual ball collection can lead to fatigue, injuries, and distraction for athletes and coaches. In addition, recreational facilities, schools, and sports organizations often incur significant costs hiring personnel specifically for ball retrieval tasks.

This problem affects a wide range of individuals and organizations. Amateur players may experience frustration due to interrupted practice sessions, while professional athletes and coaches face inefficiencies that could impact their training schedules.

The repetitive and time-consuming nature of ball collection not only detracts from the quality of training and gameplay but also underlines the potential of integrating advanced technologies into sports. With the rise of automation and robotics, there is a significant opportunity to revolutionize this neglected aspect of sports by creating an autonomous solution that addresses these inefficiencies comprehensively.

YOLO: Aims for real-time object detection by framing it as a regression problem [4]. Its limitation lies in handling small or densely packed objects due to grid cell restrictions. This project aims to improve upon this by using more flexible architecture and focusing on a specific domain.

Faster R-CNN: Improves upon previous two-stage detectors by using a Region Proposal Network, increasing accuracy, especially for smaller objects [5]. However, its complexity remains a challenge. This project explores a simplified end-to-end approach for comparable accuracy with improved efficiency.

SSD: Combines elements of YOLO and Faster R-CNN, offering a balance between speed and accuracy [6]. Its reliance on default boxes for object shapes can be limiting. This project explores more adaptable methods for handling object shapes, potentially through anchor-free approaches or learned anchor shapes, within our specific domain.

To solve this problem, the proposed solution is an autonomous ball-picking robot designed to modernize the process of ball retrieval in sports. This robot combines advanced robotics, artificial intelligence, and sensor technologies to navigate sports fields and courts independently, detect balls, and collect them efficiently without human intervention.

The robot operates by employing a combination of sensors, such as cameras, ultrasonic detectors, and infrared systems, to identify and locate balls scattered across the area. Using an optimized algorithm, it calculates the most efficient path to collect the balls while avoiding obstacles like players, equipment, or uneven terrain. The collected balls are stored in an onboard container, which can be emptied as needed, ensuring uninterrupted operation.

This solution addresses the problem by completely eliminating the need for manual ball collection. Athletes and coaches can focus entirely on training, while facility operators save time and resources. The robot is equipped with energy-efficient motors and rechargeable batteries, ensuring it is both cost-effective and environmentally friendly. Additionally, its compact design allows it to function across various settings, including tennis courts, golf ranges, and baseball fields.

The proposed solution is superior to existing methods due to its high degree of automation and versatility. Unlike manual or semi-automated tools, which still require significant human effort,

the robot operates independently, saving labor costs and time. Compared to traditional ball collection systems like stationary nets or manual rollers, the autonomous robot offers flexibility by adapting to different terrains and sports environments.

This method is not only practical but also aligns with the increasing trend of incorporating technology and automation into various aspects of life, including sports. As the global demand for smarter and more efficient solutions continues to rise, the adoption of autonomous ball-picking robots could become a standard in sports training and facility management. By addressing the inefficiencies of traditional methods and embracing innovation, this solution has the potential to transform how ball collection is approached, ultimately benefiting athletes, coaches, and facility operators worldwide.

This project investigated the application of deep learning for tennis ball detection in images [7]. The primary experiment focused on training an object detection model to accurately locate and classify tennis balls. The model was trained using a dataset of annotated images featuring tennis balls under various conditions. Performance was evaluated using mean Average Precision (mAP) and analyzed in conjunction with loss metrics (box, class, and object loss). The initial experiment revealed the model's learning progression, demonstrating a rapid improvement in performance after an initial learning phase. The analysis of loss functions provided insights into the model's ability to localize and classify objects. Subsequent experiments, focusing on individual loss components, confirmed the correlation between decreasing loss and increasing mAP, validating the importance of each aspect of the model's learning process. The fluctuations in loss during early epochs suggested potential challenges with data variability or learning rate optimization, highlighting areas for future refinement.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Accuracy

A major challenge in designing the ball-detection system is ensuring that the robot can accurately identify and differentiate balls from other objects or environmental elements [8]. Factors such as low visibility due to poor lighting conditions, reflective surfaces, overlapping objects, or the presence of debris on the ground could lead to errors in detection. These issues might cause the robot to misidentify balls or fail to detect them altogether, reducing the system’s overall efficiency. For example, in outdoor environments like golf driving ranges, changing weather conditions and uneven terrain might further complicate the detection process.

To address these challenges, I could use a combination of sensors, such as cameras, infrared systems, and depth sensors, to create a multi-modal detection system. The combination of these technologies would help the robot differentiate balls based on shape, size, and color while filtering out irrelevant objects. Additionally, I could incorporate machine learning algorithms that allow the robot to analyze and classify data more effectively over time. This system would enable it to adapt to dynamic environments by learning from previous detection errors. Regular testing in controlled and uncontrolled environments would ensure the system’s robustness and enhance its ability to detect balls accurately under diverse conditions.

2.2. Developing a Reliable Navigation System

Another significant challenge lies in developing a reliable navigation system for the robot to maneuver effectively across various terrains while avoiding obstacles [9]. Sports fields and courts can have numerous obstacles, including players, equipment, uneven surfaces, and moving objects. Without a well-designed navigation system, the robot could collide with obstacles or fail to collect balls efficiently, compromising its functionality. For example, in a tennis court, the robot might encounter net posts, boundary lines, or balls in hard-to-reach areas.

To resolve this challenge, I could implement a path-planning algorithm capable of real-time obstacle detection and avoidance. The robot could be equipped with ultrasonic sensors, LiDAR, and gyroscopes to map its surroundings and calculate the most efficient route for ball collection. Additionally, I could incorporate simultaneous localization and mapping technology to enable the robot to create and update a map of its environment dynamically. This would allow the robot to navigate accurately and adjust its movements in response to changes in the environment, such as players moving across its path.

Thorough testing across various sports fields, including both indoor and outdoor settings, would help identify weaknesses in the navigation system and ensure it performs reliably under diverse conditions.

2.3. Optimizing the Robot's Ball-Collection Mechanism

A third challenge is optimizing the robot's ball-collection mechanism to handle different types, sizes, and weights of balls without causing damage or mechanical failures. Sports balls, such as tennis balls, golf balls, or baseballs, vary in material and dimensions. This variation could lead to jamming or inefficiencies in the collection process, particularly if the mechanism is not flexible enough to adapt to these differences. Additionally, the system must be designed to operate continuously without excessive wear and tear, which could result in costly maintenance or frequent breakdowns.

To address this challenge, I could design a versatile ball-collection mechanism that combines adjustable rollers, suction systems, or flexible conveyor belts. The mechanism would need to apply the right amount of force to pick up balls without damaging them. By incorporating modular components, the system could be easily adjusted or replaced based on specific requirements.

Stress testing the mechanism with different types of balls and under varying conditions, such as wet or uneven surfaces, would help ensure its durability and reliability. Additionally, implementing self-cleaning or anti-jamming features could reduce downtime and enhance the system's performance. This approach would make the robot adaptable to a wide range of sports scenarios, improving its overall efficiency and usability.

3. SOLUTION

The ball-picking robot program consists of three major components: Detection, Navigation, and Collection. These components are integrated to allow the robot to detect tennis balls, move toward them, and collect them efficiently. The program starts with the detection component using a camera and AI algorithms to identify the tennis balls' location. Once detected, the navigation component calculates the closest path to the ball while avoiding obstacles, ensuring smooth movement. Finally, the collection mechanism engages to pick up the ball and store it securely.

The program flow begins with the robot taking a picture of its surroundings and using an AI-powered detection system to identify the nearest tennis ball. If no ball is detected, the robot rotates in search of one. After determining the closest ball, the robot adjusts its position to ensure the ball is centered in its field of view. It then moves forward while continuously checking for obstacles. Upon encountering an obstacle, the system determines whether the obstacle is a ball or an unrelated object. If it is a ball, the robot collects it; otherwise, it navigates around the obstacle.
This program was developed using AI algorithms for ball detection, sonar sensors for obstacle avoidance, and a motor-controlled chassis for movement [10]. The components communicate seamlessly, ensuring a smooth transition between tasks for efficient ball collection.

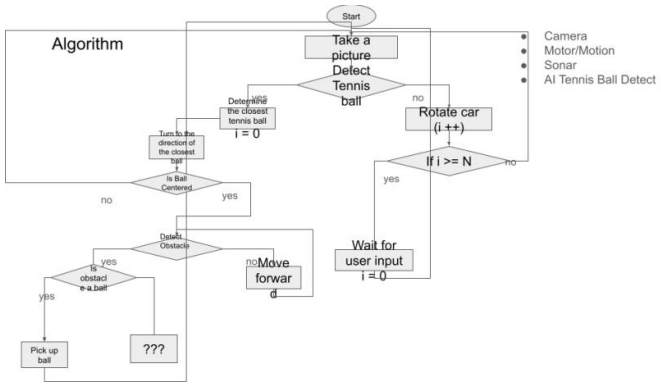


Figure 1. Overview of the solution

Purpose: To identify and locate tennis balls within the robot's environment.
Implementation:
Primary Sensor: Camera
Processing: Pre-trained Convolutional Neural Network (CNN)
Special Concept:
Convolutional Neural Networks (CNNs): Specialized neural networks highly effective for image recognition.
Function:
Input: Processes images captured by the camera.
Analysis: Identifies tennis balls within the images using the pre-trained CNN.
Output: Provides the coordinates of detected tennis balls relative to the robot.

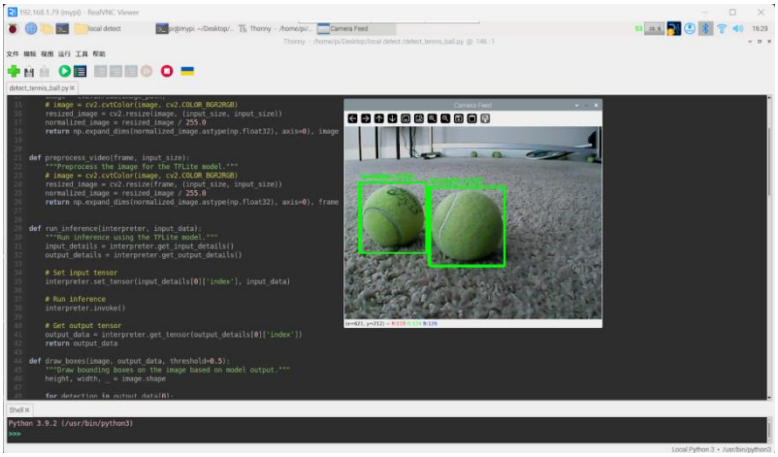
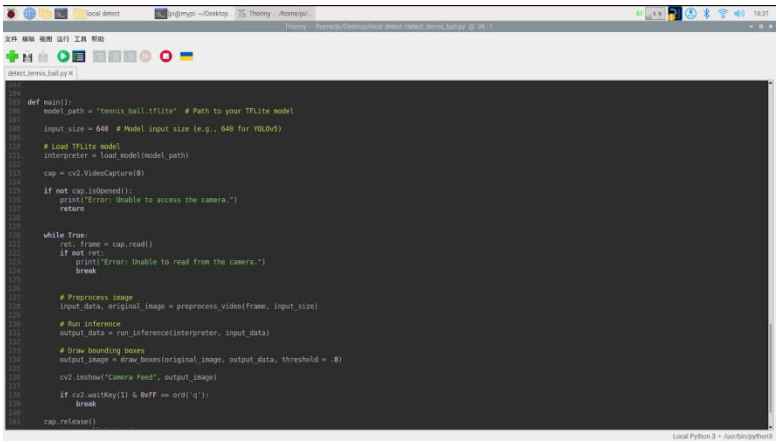


Figure 2. Screenshot of the tennis ball



```
def main():
    model_path = "tennis_ball.tflite"  # Path to your TFLite model

    input_size = 640  # Model input size (e.g., 640 for YOLOv5)

    # Load TFLite model
    interpreter = load_model(model_path)

    cap = cv2.VideoCapture(0)

    if not cap.isOpened():
        print("Error: Unable to access the camera.")
        return


    while True:
        ret, frame = cap.read()
        if not ret:
            print("Error: Unable to read from the camera.")
            break


        # Preprocess image
        input_data, original_image = preprocess_video(frame, input_size)

        # Run inference
        output_data = run_inference(interpreter, input_data)

        # Draw bounding boxes
        output_image = draw_boxes(original_image, output_data, threshold = .8)

        cv2.imshow("Camera Feed", output_image)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
```

Figure 3. Screenshot of code

This code represents the detection phase of the tennis ball-picking robot's operation. It initializes the TFLite (TensorFlow Lite) model to recognize tennis balls and connects to the camera for live video capture. The main() method begins by defining the model's file path and its input size for processing images.

The load_model method loads the specified TFLite model into memory, while cv2.VideoCapture(0) initializes the camera feed.Within a continuous loop, the program captures frames from the camera and preprocesses the images using preprocess_video(). This step prepares the image data for the model by resizing it to match the input size.

The run_inference() method processes the image using the loaded model and outputs the detection results. These results are used by the draw_boxes() function to highlight detected tennis balls with bounding boxes. The output image is displayed using OpenCV's imshow(). When the user presses 'q', the loop terminates, and the camera resource is released. This code is critical for accurately detecting tennis balls in real time, providing input for the navigation component.
Objective: Real-time tennis ball detection on a Raspberry Pi using TensorFlow Lite.
Model: Loads a pre-trained roboflow model.

Camera: Initializes and accesses the default camera (index 0). Includes error handling for camera access.

Processing Loop:
Continuously captures frames from the camera.
Includes error handling for frame reading.
Preprocesses each frame (resizing, normalization, etc.) for the model.
Performs inference using the loaded TensorFlow Lite model.
Draws bounding boxes around detected tennis balls on the original frame.
Displays the annotated frame in a window.
Cleanup: Releases the camera resource after exiting the loop.
Key Technologies: OpenCV (cv2) for camera handling and image processing, TensorFlow Lite for model inference.
Platform: Raspberry Pi
Code Structure: High-level structure is shown, but details of model loading, preprocessing, and bounding box drawing are abstracted into separate functions.

The main method best represents this component. It is called in the program when the user initiates the tennis ball detection process. When executed, it captures live video feed, processes it using a TensorFlow Lite (TFLite) model, and displays real-time detections [14]. There are two important variables: model_path specifies the path to the TFLite model file, which is used for detecting objects, and input_size defines the dimensions of the model's input data for inference.

When the method runs, it initializes the model using the load_model function. Then, it establishes a connection to the default camera using cv2.VideoCapture(0). The program reads frames from the camera and preprocesses them with the preprocess_video function to convert the video frames into a format suitable for the model. The run_inference function runs the preprocessed frames through the model to detect objects, and draw_boxes overlays bounding boxes on detected tennis balls.

There is no server-side component; all computation occurs locally on the device. The processed video, with detected objects highlighted, is displayed to the user in real-time. The user can exit the detection process by pressing 'q'. This method ties together all the program's critical components for real-time object detection and feedback.

4. EXPERIMENT

4.1. Experiment 1

A potential blind spot in the program lies in the detection accuracy of tennis balls in challenging environments, such as low lighting or when balls are partially obstructed. Ensuring high detection accuracy is critical because errors in identifying the ball could lead to failed pickups, wasting time and compromising the robot's functionality.

To test detection accuracy, the experiment involves running the robot in controlled environments with varying lighting conditions and ball placements. Scenarios include low-light setups, cluttered backgrounds, and partially hidden balls. These conditions simulate real-world challenges. Control data will include images and videos of tennis balls in diverse settings to ensure comprehensive testing.

Performance will be measured by calculating the detection rate as a percentage of successfully identified tennis balls compared to the total number of balls present. Repeated trials under different scenarios will provide reliable results, ensuring the robustness of the detection system in real-world applications.

Figure 4. Figure of experiment 1

The mean appears to be around 0.85, while the median around 0.90. The lowest value fluctuates around 0.40-0.50 during early epochs, reaching a high of just below 1.0 (likely 0.98-0.99).

The initial instability in "mAP" during the first 50 epochs is surprising. I expected a smoother, more consistent climb, indicating the model was struggling to learn initially. This might be due to insufficient initial data or a poorly tuned learning rate. The rapid jump in "mAP" after epoch 50 suggests a crucial learning phase.

The biggest impact on results is likely the training data itself.

Object detection models are heavily reliant on large, diverse datasets. The quality and quantity of annotations directly influence the model's ability to generalize. The choice of diverse pictures also play a role.

4.2. Experiment 2

From the "Box Loss" graph, the mean appears to be around 0.50-0.60 after the initial fluctuations. The median is likely slightly lower, around 0.40-0.50. The highest value is around 1.0, while the lowest value fluctuates around 0.20-0.30 in later epochs.

The "Box Loss" represents the error in predicting the bounding box coordinates. The decrease indicates the model's improvement in accurately predicting object locations. The fluctuations suggest similar challenges as with other losses, potentially linked to data or learning rate. The stabilized "Box Loss" in later epochs suggests reasonable accuracy in localization, though it seems to have more variability than "Class Loss" or "Object Loss".

Comparing "Box Loss" with "mAP", we again observe an inverse relationship. As "Box Loss" decreases, "mAP" increases, indicating that accurate bounding box predictions are essential for good object detection performance.

5. RELATED WORK

YOLO (Redmon et al., 2016) tackles object detection by reframing it as a regression problem, directly predicting bounding boxes and class probabilities from a single grid applied to the entire image [11].

This approach is fast, enabling real-time object detection. However, YOLO struggles with objects that are small or close together due to the grid cell limitations. It also has limitations in accurately predicting irregularly shaped bounding boxes. Our project improves upon YOLO by potentially using a more flexible architecture that doesn't rely on a fixed grid, allowing for better handling of varying object sizes and densities. We also focus on a specific domain (tennis balls) with tailored data, potentially leading to higher accuracy within that domain.

Faster R-CNN (Ren et al., 2015) addresses the region proposal bottleneck in previous two-stage detectors by introducing a Region Proposal Network (RPN) that shares convolutional features with the detection network [12].

This allows for near-costless region proposals, improving speed compared to methods like Selective Search. Faster R-CNN is more accurate than YOLO, especially for smaller objects. However, it is still a complex, multi-stage pipeline. Our project potentially simplifies the process

by using an end-to-end approach, aiming for comparable accuracy with improved efficiency and potentially better generalization within our specific domain due to focused training data.

SSD (Liu et al., 2016) combines aspects of YOLO and Faster R-CNN, using a single convolutional network to predict bounding boxes and class probabilities at multiple scales [13]. This allows for faster detection than Faster R-CNN while maintaining relatively high accuracy. SSD uses default boxes of various aspect ratios to handle objects of different shapes. However, choosing appropriate default box sizes can be challenging. Our project potentially improves on SSD by using a more adaptable method for handling object shapes, potentially through an anchor-free approach or by learning the optimal anchor shapes from our specific dataset. We also aim to demonstrate the effectiveness of focused data curation for a specific object detection task.

6. CONCLUSIONS

A key limitation of this project lies in the limited diversity of the tennis ball dataset. While the model shows promising results on the current dataset, it might struggle with variations in lighting, background, ball wear, and camera angles not represented in the training data. Gathering a more comprehensive dataset encompassing these variations would significantly improve robustness. Another limitation is the potential overfitting to the specific characteristics of tennis balls. Testing on related objects (e.g., baseballs, softballs) would reveal the model's generalization ability and highlight areas for improvement.

If I had more time, I would implement data augmentation techniques (e.g., random rotations, flips, color adjustments) to artificially expand the dataset and improve the model's ability to generalize. I would also explore incorporating transfer learning, leveraging pre-trained models on larger, more diverse datasets to initialize the model and potentially reduce the amount of tennis-ball specific data needed. Analyzing failure cases (images where the model performs poorly) would also provide valuable insights for targeted data collection and model adjustments [15].

This tennis-ball picking project demonstrates the potential of deep learning for accurate tennis ball detection using a focused dataset and analysis of key performance metrics. While limitations exist, the results provide a solid foundation for future work towards a robust and generalizable tennis ball detection system.

REFERENCES

[1] Mathew, Amitha, P. Amudha, and S. Sivakumari. "Deep learning techniques: an overview." Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020 (2021): 599-608.

[2] Shorten, Connor, Taghi M. Khoshgoftaar, and Borko Furht. "Text data augmentation for deep learning." Journal of big Data 8.1 (2021): 101.

[3] Krzywdzinski, Martin, Maren Evers, and Christine Gerber. "Labor-Intensive Work Processes."

[4] Jiang, Peiyuan, et al. "A Review of Yolo algorithm developments." Procedia computer science 199 (2022): 1066-1073.

[5] Ren, Shaoqing, et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." IEEE transactions on pattern analysis and machine intelligence 39.6 (2016): 1137-1149.

[6] Liu, Wei, et al. "Ssd: Single shot multibox detector." Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11 – 14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016.

[7] Wu, Di, and Aiping Xiao. "Deep learning-based algorithm for recognizing tennis balls." Applied Sciences 12.23 (2022): 12116.

[8] Teimouri, Meisam, Mohammad Hossein Delavaran, and Mahdi Rezaei. "A real-time ball detection approach using convolutional neural networks." Robot World Cup. Cham: Springer International Publishing, 2019. 323-336.

[9] Hasan, Ahmed M., et al. "A review of navigation systems (integration and algorithms)." Australian journal of basic and applied sciences 3.2 (2009): 943-959.

[10] Baus, Jörg, Antonio Krüger, and Wolfgang Wahlster. "A resource-adaptive mobile navigation system." Proceedings of the 7th international conference on Intelligent user interfaces. 2002.

[11] Simon, Martin, et al. "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2019.

[12] Ren, Shaoqing, et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." IEEE transactions on pattern analysis and machine intelligence 39.6 (2016): 1137-1149.

[13] Liu, Wei, et al. "Ssd: Single shot multibox detector." Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11 – 14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016.

[14] Giedra, Henrikas, and DaliusMatuzevičius. "Assessing and Forecasting TensorFlow Lite Model Execution Time in Different Platforms." 2024 IEEE 11th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE). IEEE, 2024.

[15] Kim, Yeon-Hak, Sun-Woong Park, and Yeong-Wha Sawng. "Improving new product development (NPD) process by analyzing failure cases." Asia Pacific Journal of Innovation and Entrepreneurship 10.1 (2016): 134-150.