Tic-Tac-Toe: Hand Gesture Recognition Using YOLO

Group 2

September 16, 2024

Abstract

This project presents a real-time Tic-Tac-Toe game where players use hand gestures as input, powered by a YOLO object detection model. The gestures (crossed fingers) and (OK sign) represent the "X" and "O" moves, respectively. The game is implemented using a webcam, which detects player hand gestures, processes them through YOLO, and updates the game grid accordingly. This paper documents the approach, methodology, results, and challenges faced during the development of the system.

1 Introduction

Tic-Tac-Toe is a simple two-player game traditionally played on a 3x3 grid, where players take turns marking empty spaces with an "X" or "O." Our goal was to modernize this classic game by incorporating computer vision techniques, allowing players to interact with the game using hand gestures. We built a model to detect two specific hand gestures: the crossed fingers for Player X and the OK sign for Player O.

The core of the project involved data collection, training a YOLO model for hand gesture recognition, and implementing the game logic. This project required integrating real-time object detection, grid management, and user interaction via gestures.

2 Methodology

2.1 Dataset Collection

We captured a total of **500 images** for our dataset, focusing on hand gestures. The data collection process involved the following:

- **Capturing a variety of hand positions and conditions**: This included images with multiple hands, partial hands, and both blurry and clear images.

- **Diversity**: We ensured the dataset represented real-world conditions, such as varying lighting, hand angles, and backgrounds. We intentionally included blurred images to simulate potential real-time scenarios.

2.2 Data Annotation

We used **Roboflow** to annotate the images with high precision. Each image was carefully labeled as either an (X) gesture or an (O) gesture.

2.3 Data Augmentation

To increase the robustness of the dataset and to ensure the model generalizes well, we performed data augmentation:

- **Techniques Used**: Image flipping, rotation, and scaling.
- **Final Dataset Size**: After augmentation, the dataset grew to **370 images**, providing sufficient variety for training.

2.4 Preprocessing

A Python script was written to resize all images to a uniform size for consistency in training.

2.5 YOLO Model Training

We trained a **YOLO (You Only Look Once)** model to detect the two specific hand gestures. The following steps were taken:

- **Model Configuration**: YOLOv5 was selected for its speed and accuracy in object detection tasks. We fine-tuned the model for the two classes: (X) and (O).
- **Training Parameters**:
  - Training: 390 image
  - Testing: 90 image
- **Training Environment**: The model was trained using GPU for efficient processing.

Initially, the model's performance was weak due to insufficient data, but we resolved this by adding more images to the dataset and retraining the model.

2.6 Game Implementation

Once the YOLO model was trained, we proceeded with the game logic implementation.

2.6.1 Tic-Tac-Toe Grid

- **Grid Layout**: The screen was divided into a 3x3 grid using OpenCV, and each move by a player was registered in the corresponding grid square.
- **Gesture Detection**: The YOLO model processed real-time video from the laptop's camera, detecting gestures and placing the appropriate symbol ("X" or "O") in the respective grid area.

2.6.2 Game Logic

- **Turn-based System**: Players alternated turns, and gestures were recognized for the respective player.
- **Win Conditions**: Standard Tic-Tac-Toe rules were implemented to detect a winner when a player successfully completes a row, column, or diagonal.
- **Error Handling**: The game handled situations where a gesture was not properly detected or placed outside the grid.

3 Results

The Tic-Tac-Toe game was successfully implemented and played by two players using real-time hand gestures. The final trained YOLO model exhibited the following performance:

- **Model Accuracy**: After data augmentation and retraining, the model achieved an accuracy of **92%** on our test dataset.
- **Real-Time Detection**: The model was able to detect hand gestures in real time with minimal latency, providing a seamless gaming experience.

4 Challenges Faced

4.1 Initial Model Performance

Initially, the YOLO model did not perform well, primarily due to the limited size and diversity of our dataset. The accuracy was low, and the model struggled to generalize to new images. This issue was addressed by:

- **Expanding the dataset**: We captured additional images under various conditions, bringing the dataset size to 500.

4.2 Grid Management

Ensuring that the detected gestures were accurately placed in the correct grid positions was a challenge. This required careful calibration of the grid's dimensions and alignment with the real-time camera feed. The issue was resolved through thorough testing and debugging.

4.3 Gesture Misclassification

We occasionally encountered misclassification of hand gestures, particularly in poor lighting conditions. We mitigated this issue by fine-tuning the model's detection thresholds and improving the dataset diversity.

5 Conclusion

This project successfully combined hand gesture recognition with a classic game, providing a fun and interactive experience for players. The YOLO model demonstrated strong performance in detecting hand gestures, and the game logic operated seamlessly once the challenges were resolved.