**Tic-Tac-Toe Game Analysis Report**

**Introduction**

In this report, we present the methodology and findings of our Tic-Tac-Toe game analysis project. The objective was to classify Tic-Tac-Toe game images, determine the winner, and draw a line that crosses the row of three marks (X or O).

**Methodology**

**Data Preprocessing**

We initially faced a challenge of limited time for training a robust model. To mitigate this, we tried to divide a task into a smaller ones to reduse the amount of data needed to train a well generalizing model. Here this steps:

|  |  |
| --- | --- |
|  | **Grid Detection:**  We aimed to simplify the task by first detecting the grid of the Tic-Tac-Toe game and its angle. The idea was to reduce the number of different cases by ensuring that the grid was correctly aligned. However, our attempts to build a generalized grid rotation model did not yield satisfactory results. Consequently, we decided not to use this approach for the final solution, but we used it to rotate our images for future annotation. We acknowledge that this strategy has potential for future data generalization. |

|  |  |
| --- | --- |
|  | **Object Detection:**    In the absence of a generalized grid alignment, we opted for a second approach. We labeled our data to detect X, O, and empty places on the grid. This allowed us to create a game map. During this phase, we encountered a challenge where our model performed exceptionally well on images it was trained on but struggled to generalize to unseen images. We attributed this behavior to data-specific patterns. |

**Model Training**

To address the data-specific performance issue, we trained our model using a combination of two datasets: our labeled data and the provided dataset. Training on both datasets simultaneously improved the model's ability to generalize and perform on unseen images, and that gave us an ability to create emages with line that crossses the winner.

**Annotation and Line Drawing**

Once the winner was determined, we developed a strategy to draw the winning line differently for various cases:

|  |  |
| --- | --- |
|  | **Row Intersection:**  If the line crosses three marks in a row, we calculated the average y-coordinate of the leftmost and rightmost boxes for drawing the line horizontally. |
|  | **Column Intersection:**  For a line crossing three marks in a column, we calculated the average x-coordinate of the topmost and bottommost boxes for drawing the line vertically. |
|  | **Diagonal Intersection:**  When the line intersects diagonally, we handled the main diagonal by taking coordinates from the top-left box's top-left corner and the bottom-right box's bottom-right corner. Similarly, for the secondary diagonal, we considered the top-right corner of the top-right box and the bottom-left corner of the bottom-left box. |

**Results and Conclusion**

Our approach, which involved combining labeled data with provided data, helped improve the model's generalization. While the model's performance was highly data-dependent, it managed to achieve the desired results on both our data and the provided dataset, only when we trained on all the data including given one.

In conclusion, the success of our project was heavily reliant on data, and we acknowledge that acquiring data similar to the provided dataset would have been beneficial. Nevertheless, our approach demonstrates the importance of the training data.

The provided code (inference.py) accepts Tic-Tac-Toe images, determines the winner, and annotates the images with winning lines. It represents our solution for the given task.