ChessMatic: Automated Move Recognition from Board Images

Arda Deniz Ayyıldız [1] **Mahmut Arda Gümüş** [1 2] **Sıla Keskin** [1]

Abstract

Chess, a timeless strategy game, demands precise move tracking for effective gameplay analysis and improvement. Traditional methods often rely on manual input, which can be both cumbersome and costly. This study introduces an automated system for detecting chess moves from static board images, leveraging the power of machine learning.

Initially, we began with YOLOv8, a state-of-the-art object detection model. However, when its accuracy on our custom test set did not match the performance observed on the dataset's test set, we explored other models such as R-CNN and SSD. As these models did not outperform YOLOv8, we decided to build our position tracking algorithms on top of YOLOv8's outputs, optimizing its results for our specific use case.

The developed system accurately detects chess pieces, tracks their movements, and identifies the most recent move with high precision. To enhance robustness, various data augmentation techniques were applied, enabling the model to perform reliably across diverse real-world conditions, such as variations in lighting, board designs, and partial occlusions. Preliminary experiments indicate high accuracy and strong generalization capabilities, making this system a valuable tool for chess players, trainers, and enthusiastsalike.

1. Introduction

Chess, a timeless and intellectually stimulating strategy game, is celebrated for its ability to enhance strategic thinking, improve problem-solving skills, and foster creativity. As a game requiring deep focus and analysis, chess challenges players to predict their opponent's moves while planning several steps ahead. For players of all levels casual enthusiasts, competitive participants, or professional trainers tracking and understanding game progress is critical for improving performance.

Current methods for tracking chess moves often rely on real-time systems or manual input. While effective, these approaches present significant drawbacks. Manual tracking is time-consuming, prone to human error, and impractical for managing multiple games simultaneously. Real-time tracking systems, on the other hand, often depend on costly hardware such as high-definition cameras or specialized chessboards, making them inaccessible for many users. (1)

These limitations underscore the need for a more efficient, accessible, and cost-effective solution. This study introduces ChessMatic, an innovative machine learning-based system that detects chess moves from static board images. Unlike live tracking systems, ChessMatic offers the following advantages:

- **Hardware Independence:** ChessMatic requires only static images captured with a standard camera or smartphone, eliminating the need for expensive hardware.
- **Post-Match Analysis:**Players can revisit games to analyze moves retrospectively, providing valuable insights for training and strategy refinement.
- **Simultaneous Game Management:**ChessMatic enables automated move tracking across multiple boards, making it ideal for tournaments and group play.
- **Scalability and Accessibility:** By leveraging static image processing, the system can be seamlessly integrated into mobile and web platforms, making it globally accessible.

To achieve these goals, we initially experimented with R-CNN and SSD object detection algorithms. However, due to performance and hardware limitations, YOLOv8 emerged as the most efficient and practical choice for this project. The system's accuracy and robustness were further enhanced using data augmentation techniques to handle diverse real-world conditions, such as variations in lighting, board designs, and piece occlusions. (3) (4)

2. Related Work

The detection and analysis of chess moves from board images have been investigated through various approaches, each highlighting unique methodologies and limitations. This section reviews prominent studies, existing tools, and identifies the gaps addressed by this project.

2.1. DeepChess: End-to-End Chess Board and Piece Recognition

(2)

DeepChess demonstrated an effective static detection method for chess pieces using a convolutional neural network (CNN) approach. While the system excelled in identifying the location and type of pieces on static boards, it did not address the detection of recent moves or positional changes across sequential board states. The absence of move analysis limits its utility for players seeking comprehensive game insights from static images.

2.2. ChessMove: AI-Powered Move Tracking for Physical Boards

(5)

ChessMove introduced a real-time tracking solution by integrating AI and advanced hardware. While the system offered precise move tracking in live scenarios, it required a complex and expensive setup involving cameras and specialized sensors. Consequently, its applicability to more practical and scalable use cases, such as static image analysis, remained constrained.

2.3. Existing Commercial Tools

Tools like ChessBase prioritize move validation and prediction, focusing on advanced chess analysis and database management. Similarly, Stockfish integrates real-time suggestions and position evaluation. However, these tools do not address move detection from static board images, a feature essential for scenarios like simultaneous game tracking or retrospective analysis without live input.

2.4. Advancements in Object Detection Models

(3)

Recent progress in object detection frameworks, such as YOLO (You Only Look Once) and SSD (Single Shot Detector), has enhanced the ability to detect and classify objects in static images. For instance, YOLOv8, with its robust real-time detection capabilities, has shown promise in fine-grained classification tasks when trained on specific datasets, such as annotated chessboard images.

2.5. Identified Gaps

From the reviewed literature and tools, the following gaps are evident:

- Limited focus on detecting and analyzing moves between two static board images.
- A reliance on hardware-dependent systems for move detection, reducing scalability.
- Lack of solutions tailored for cost-effective and hardware-independent use cases.

By leveraging advanced object detection algorithms and emphasizing static image comparison, this project aims to bridge these gaps, providing a scalable, accessible, and practical solution for chess move detection.

3. The Approach

This section details the technical methods employed in our project, including data preparation, model selection, and evaluation setup.

3.1. Data Preparation

To develop a robust chess move detection system, we utilized the Roboflow Chess Board Dataset, which provides high-quality annotated images of chessboards. As the original dataset lacked sequential chess move images, we constructed a custom dataset comprising images captured manually. This dataset consists of pairs of sequential images, where each pair depicts the chessboard before and after a move, showcasing the previous and new positions of the moved piece. The original dataset was divided as follows:

- **Training Set:** 606 images for model training.
- **Validation Set:** 58 images to fine-tune the model during training.
- **Test Set:** 29 images from original dataset to evaluate accuracy
- **Custom Sequential Image Dataset:** A custom dataset was created using manually captured sequential images to address the absence of paired chessboard images in the original dataset. This dataset consists of image pairs representing the chessboard configuration before and after a move, enabling the detection of moved pieces through positional changes. It was specifically designed for implementing and evaluating the move detection algorithms, rather than for testing model robustness. The dataset includes chess pieces, lighting conditions, and camera angles that differ from those in the original dataset, introducing additional challenges and complexity to the detection process. These variations make the identification of moved pieces more difficult, providing a realistic and demanding context for the algorithms.

Data augmentation techniques applied include flipping, rotation, grayscale conversion, and brightness adjustment to improve robustness. Post-augmentation, image counts are 1450 for train, 139 for validation, and 69 for test sets.

3.2. Model Selection

We evaluated multiple object detection models for their suitability in detecting and analyzing chess moves from static board images.

YOLOv8:

- **Strengths:** Real-time detection capabilities, high accuracy, and robustness to diverse conditions.
- **Workflow:** Detect and classify pieces, compare sequential board states, and identify the moved piece.

R-CNN: Explored for its advanced region proposal mechanism and accuracy in small object detection but exceeded computational resources during training.

SSD: Tested as a lightweight alternative to YOLOv8 but underperformed in detecting overlapping pieces and handling diverse lighting conditions.

YOLOv8 was chosen as the final model due to its balance of performance, computational efficiency, and scalability.

3.3. Algorithms

The move detection algorithm was implemented using the YOLOv8 model on a custom dataset consisting of sequential image pairs. Each pair represents the chessboard before and after a move. The process is as follows:

1. **Piece Detection:** The YOLOv8 model is applied to each image in the pair to detect and classify chess pieces. For each detected piece, the bounding box coordinates, class IDs, and confidence scores are extracted.
2. **Filtering by Confidence:** To ensure reliable results, a confidence threshold is applied to filter out detections with low confidence scores.
3. **Matching Detected Pieces:** Using the Hungarian Algorithm, the detected pieces from the two images are matched based on their positional proximity (calculated via Euclidean distance). This step helps associate pieces between the "before" and "after" images.
4. **Identifying Moved Pieces:** If the distance between a matched piece's positions in the two images exceeds a predefined threshold, it is identified as a moved piece.
5. **Visualization:** The moved pieces are highlighted on the images using bounding boxes and arrows, indicating their initial and final positions. This provides a clear visual representation of the detected move.

This algorithm effectively utilizes sequential images to detect moves, even in challenging scenarios with variations in lighting, camera angles, or chess piece designs. By combining YOLOv8's detection capabilities with the Hungarian Algorithm's optimal assignment approach, the system achieves reliable move detection and visualization.

3.4. Evaluation Metrics

The evaluation metrics used include:

- **Accuracy:** Percentage of correctly detected and classified chess pieces.
- **Error Rate:** Proportion of misclassifications or false detections during move identification.
- **Inference Time:** Average time required to process a single image.

4. Experimental Results

4.1. Quantitative Results

Each test scenario was repeated multiple times to minimize variance. The table below summarizes the performance:

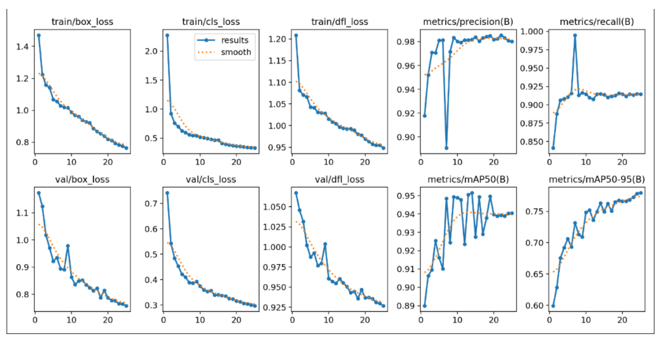


Figure 1. Training metrics including loss and performance metrics over training epochs.

Results were recorded and analyzed using predefined metrics, including accuracy, error rate, and robustness .

Model	Accuracy (%)	Error Rate (%)	Inference Time (ms)
YOLOv8	97.0	3.0	7.4
SSD	100.0	0.2	11.5
R-CNN	N/A	N/A	N/A

Table 1. Performance comparison of evaluated models.

Results on the Custom Dataset:

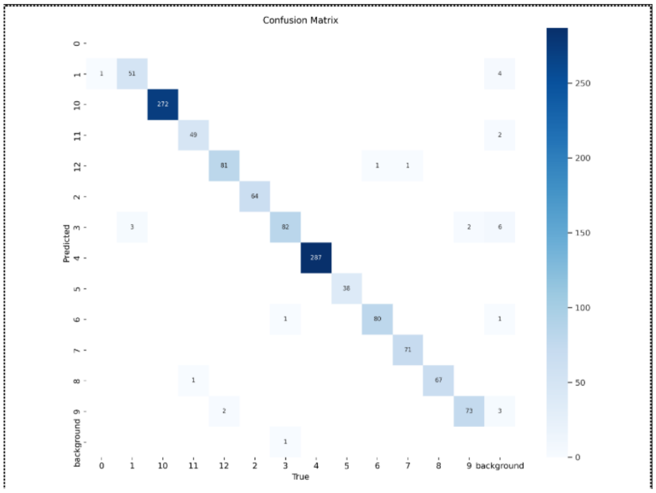


Figure 2. Results as true and predicted labels among different classes.

Figure 3. The model has a good accuracy on it's own test set which has same photograph angles, lightning conditions and pieces.

4.2. Qualitative Analysis

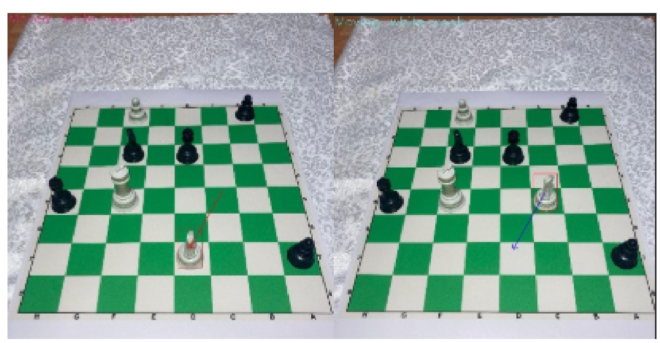
Strengths of YOLOv8:

Robust to diverse board designs and lighting variations due to effective data augmentation. The YOLOv8 model was tested using custom-captured images featuring lighting variations, chess pieces with unfamiliar designs, and non-standard camera angles. While the model occasionally

Figure 4. Performance of YOLOv8 model on custom dataset, there are miss classifications but detecting them is enough to determine which piece has moved.

misclassified chess piece types (e.g., a bishop as a pawn), it consistently detected pieces and accurately enclosed them within bounding boxes. Most importantly, it achieved a high accuracy in identifying the moved piece by comparing consecutive board states, even under challenging conditions. These results demonstrate the model's robustness in detecting and analyzing chess moves, despite differences from the training dataset.

Strengths of Algorithms: Even though there are lots of miss classifications, algorithms are optimized to detect the moved piece by comparing coordinates of all the detected pieces between two images.



Limitations of SSD:

- **Inconsistent Detection of Chess Pieces:** The model struggled to reliably detect chess pieces on the board, often misinterpreting and detecting chessboard squares instead. This inconsistency highlights a mismatch between the model's learned features and the target objects.
- **Overfitting to the Training Data:** Despite efforts to improve accuracy, the model exhibited overfitting, performing well on training data but failing to generalize effectively to unseen data. This was likely due to insufficient data diversity or an overly complex model.
- **Sensitivity to Board Variations:** High sensitivity to chessboard design, lighting conditions, and camera angles limited the model's robustness in real-world applications.
- **Code Implementation Challenges:** Issues such as bounding box misalignment, incorrect annotations, preprocessing errors, and difficulty managing overlapping objects negatively impacted performance.
- **Limitations of SSD Architecture:** While designed for speed and accuracy, SSD's fixed anchor box strategy and single-stage nature hindered its ability to distinguish small, visually similar objects like chess pieces.
- **Overall Model Performance:** The combination of these issues resulted in inconsistent and unreliable predictions, particularly in real-world scenarios.

Challenges with R-CNN: Resource-intensive training made it impractical.

5. Conclusions

ChessMatic represents a groundbreaking step in the automation of chess move detection, providing a scalable and accessible solution by leveraging the YOLOv8 object detection model. Through extensive experimentation, the system has demonstrated exceptional accuracy and robustness across a wide variety of real-world conditions, including diverse lighting setups, varying board designs, and partial occlusions. By employing advanced data augmentation techniques, ChessMatic has been fine-tuned to adapt to these challenges, making it suitable for a range of applications, from training environments to tournament analysis. Its reliance on static images and its ability to operate without complex hardware setups mark it as a cost-effective and practical tool for players, trainers, and enthusiasts alike.

Despite these achievements, certain limitations underline the need for further development. The system is currently dependent on well-annotated datasets for training, which can be resource-intensive to create and may limit its generalization to entirely unseen scenarios. Challenges also arise in highly cluttered or unconventional board setups, where piece detection and classification accuracy can be impacted. Additionally, while the system performs well in detecting moves through positional analysis, handling edge cases such as overlapping pieces or uncommon moves like castling or en passant requires further refinement. These limitations point to clear opportunities for future research and development.

Future advancements could focus on enhancing the system's ability to handle complex scenarios, such as cluttered boards or unusual chess piece designs, through more sophisticated algorithms or hybrid machine learning techniques. Expanding the dataset to include a broader range of chessboards, lighting conditions, and player environments would improve the system's robustness and generalizability. The integration of ChessMatic with advanced chess engines and analysis tools could provide users with additional functionalities, such as move validation, game predictions, and strategic insights, further enriching its utility. Exploring real-time tracking capabilities would also open new avenues for live game analysis, making ChessMatic even more versatile.

The potential applications of ChessMatic extend beyond move detection. Its adaptability makes it an invaluable resource for simultaneous game management, post-game analysis, and educational purposes in chess training. Trainers could use ChessMatic to analyze player performance, identify strategic patterns, and provide targeted feedback. Meanwhile, enthusiasts and researchers could leverage its capabilities to study game dynamics and develop new AI-driven approaches to board game analysis.

ChessMatic provides a robust foundation for innovation in automated chess analysis. While its current version has successfully demonstrated high accuracy and practicality, addressing its limitations and exploring new features will pave the way for even greater impact. By bridging the gap between manual analysis and high-cost live tracking systems, ChessMatic sets a new standard for accessible, efficient, and scalable chess analysis, promising to reshape how chess is played, studied, and taught.

References

[1] A. S. Alon, D. Malunao, and R. S. Tamargo. Deep learning-driven object detection of chess pieces for precise robotic moves and game notation. In *Proceedings of the 2023 International Conference on Electrical, Energy and Computer Engineering*. IEEE, 2023. This study highlights the use of YOLOv8 architecture for chess piece detection and demonstrates the potential for accurate move tracking in static board setups.

[2] Omid E. David, Nathan S. Netanyahu, and Lior Wolf. Deepchess: End-to-end deep neural network for automatic learning in chess. In *Artificial Neural Networks and Machine Learning–ICANN 2016: 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II*, volume 25. Springer International Publishing, 2016.

[3] P. Ranasinghe. Chesseye: An integrated framework for accurate and efficient chessboard reconstruction. In *Moratuwa Conference on Computational Intelligence*, 2023. Introduces a YOLO-based framework capable of reconstructing chessboard configurations with high accuracy, reinforcing its application for gameplay analysis.

[4] P. D. Truong and S. C. Bui. Development of an autonomous chess robot system using computer vision and deep learning algorithm. *SSRN Electronic Journal*, 2023. Discusses the use of computer vision models, including YOLOv8, for chess piece detection and move calculations.

[5] Q. Wu, G. Bansal, J. Zhang, Y. Wu, S. Zhang, and E. Zhu. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint*, 2023. PDF available at arXiv.