Chess Recognition

The **Chess 2D Board Recognition** algorithm focused on developing a hybrid computer vision framework capable of detecting and classifying chess pieces across 2D digital boards using AI techniques. This required the creation of a custom-trained convolutional neural network (CNN) specifically designed to interpret flat, stylized chessboard images. The primary objective was to integrate these two paradigms into a unified system capable of automatically adapting its detection strategy based on the visual characteristics of the input image.

To enable accurate 2D recognition, the dataset was constructed by cropping high-resolution chessboard images into 64 individual squares corresponding to the standard 8×8 grid. Each square was manually labeled into one of thirteen possible categories: *empty*, *white_pawn*, *white_rook*, *white_knight*, *white_bishop*, *white_queen*, *white_king*, *black_pawn*, *black_rook*, *black_knight*, *black_bishop*, *black_queen*, and *black_king*. The dataset was then divided into training and validation subsets following an 80/20 split to ensure generalization and prevent overfitting.

The convolutional neural network was implemented using the ResNet-18 architecture, chosen for its balance between performance and computational efficiency. The final classification layer was replaced with a fully connected layer corresponding to the thirteen chess piece classes. The model was trained for twenty epochs on 64×64 pixel images of the cropped squares. During training, the network learned to accurately differentiate between empty and occupied squares and to classify each chess piece type. Upon completion, the trained model demonstrated high classification accuracy and was exported as **chess_cnn.pth** for integration into the main pipeline.

This trained CNN was then incorporated into a broader hybrid visual reasoning system that also includes **YOLOv8x** for 3D piece detection, **CLIP** for contextual label refinement, **Places365** for scene classification, and **RelTR** for relational triplet extraction (subject–predicate–object representations). A lightweight mode classifier based on image variance, entropy, and edge density automatically determines whether an input image corresponds to a 2D digital board or a 3D real-world setup, ensuring that the appropriate model—YOLO or the CNN—is used for detection.

For 2D boards, the CNN divides the chessboard into 64 regions, classifies each square, and maps detected pieces to their respective coordinates (e.g., *white_pawn on a2*, *black_queen on d8*). The system then expresses these detections as structured semantic triplets such as
{ "subject": "white_pawn", "predicate": "on", "object": "a2" },
which can be directly used in downstream reasoning or knowledge graph generation modules.

Overall, the CNN reliably recognized all major chess pieces across a variety of digital board styles, achieving strong consistency in both position mapping and class identification. The complete hybrid system now dynamically switches between YOLO and the CNN depending on

whether the input represents a real-world or digital environment, thereby achieving robust, context-aware chessboard understanding across both visual domains.

Methodology with examples:

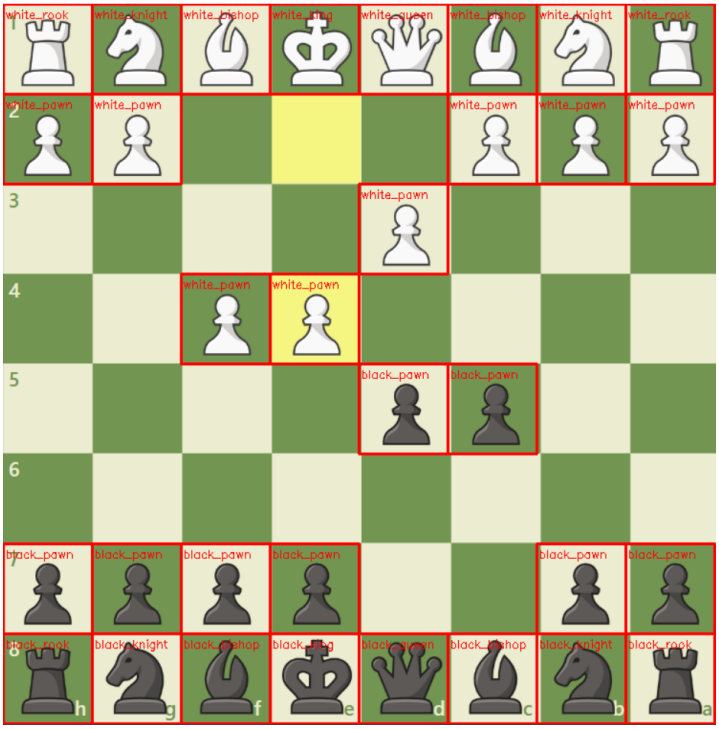
1. Images of chessboards from different websites were saved and cropped into 64 regions

Examples of resulting crops can be seen below:

2. A simple CNN model was trained using 475 training images and 132 validation images.

```
✅ Found 475 training images and 132 validation images.
Classes: ['black_bishop', 'black_king', 'black_knight', 'black_pawn', 'black_queen', 'black_rook', 'empty', 'white_bishop', 'white_king', 'white_knight', 'w
hite_pawn', 'white_queen', 'white_rook']
Using device: cpu
Epoch 1/20 | Train Acc: 68.63% | Val Acc: 89.39% | Loss: 0.3280
Epoch 2/20 | Train Acc: 86.95% | Val Acc: 95.45% | Loss: 0.1258
Epoch 3/20 | Train Acc: 90.11% | Val Acc: 93.94% | Loss: 0.5277
Epoch 4/20 | Train Acc: 91.16% | Val Acc: 97.73% | Loss: 0.0876
Epoch 5/20 | Train Acc: 92.42% | Val Acc: 98.48% | Loss: 0.0657
Epoch 6/20 | Train Acc: 93.89% | Val Acc: 89.39% | Loss: 0.2033
Epoch 7/20 | Train Acc: 95.79% | Val Acc: 98.48% | Loss: 0.0825
Epoch 8/20 | Train Acc: 97.05% | Val Acc: 99.24% | Loss: 0.0328
Epoch 9/20 | Train Acc: 98.74% | Val Acc: 95.45% | Loss: 0.2073
Epoch 10/20 | Train Acc: 96.84% | Val Acc: 97.73% | Loss: 0.0804
Epoch 11/20 | Train Acc: 97.05% | Val Acc: 100.00% | Loss: 0.0093
Epoch 12/20 | Train Acc: 95.37% | Val Acc: 93.94% | Loss: 0.3129
Epoch 13/20 | Train Acc: 94.53% | Val Acc: 95.45% | Loss: 0.2553
Epoch 14/20 | Train Acc: 93.68% | Val Acc: 100.00% | Loss: 0.0091
Epoch 15/20 | Train Acc: 97.05% | Val Acc: 100.00% | Loss: 0.0032
Epoch 16/20 | Train Acc: 95.37% | Val Acc: 100.00% | Loss: 0.0021
Epoch 17/20 | Train Acc: 98.74% | Val Acc: 100.00% | Loss: 0.0013
Epoch 18/20 | Train Acc: 99.37% | Val Acc: 100.00% | Loss: 0.0004
Epoch 19/20 | Train Acc: 99.58% | Val Acc: 100.00% | Loss: 0.0015
Epoch 20/20 | Train Acc: 99.16% | Val Acc: 100.00% | Loss: 0.0019
```

3. The model was then tested on various chessboard images from different websites.


white_rook
white_knight
white_bishop
white_king
white_queen
white_bishop
white_knight
white_rook
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
white_pawn
black_pawn
black_pawn
black_pawn
black_pawn
black_pawn
black_pawn
black_pawn
black_pawn
black_rook
black_knight
black_bishop
black_king
black_queen
black_bishop
black_knight
black_rook