Synthetic-to-Real Generalization for Chessboard Square and Board-State Classification

Abstract

Accurate recognition of chessboard states from images is a challenging visual recognition task, particularly under real-world conditions. In this work, we study the problem of synthetic-to-real generalization for chessboard square classification, where a model is trained primarily on synthetic data generated in simulation and evaluated on real images.
We formulate the task as a per-square multi-class classification problem with 13 classes corresponding to chess pieces and empty squares. Using a convolutional neural network based on a ResNet-18 backbone, we evaluate zero-shot performance on real data, fine-tuning with limited real samples, and joint training on mixed synthetic and real datasets.
Our experiments show that while zero-shot transfer achieves high overall accuracy due to class imbalance, it suffers from low macro F1-score.
Fine-tuning and mixed training significantly reduce the domain gap and improve piece-level recognition. We further analyze failure cases, architectural choices, and the impact of data quality and resolution. Our results highlight both the potential and limitations of simulation-to-real transfer for structured visual recognition tasks.

1. Introduction

1.1 Problem Description

We address the problem of chessboard square classification from a single RGB image. Given an image of a chessboard, the goal is to predict the piece occupying each of the 64 squares independently. Each square is classified into one of 13 classes: empty or one of the 12 chess pieces.

This task is a core component of full board-state reconstruction and requires accurate fine-grained visual recognition under varying conditions.

1.2 Motivation

Collecting and annotating large-scale real-world chessboard datasets is costly and time-consuming. Synthetic data generation using 3D rendering engines such as Blender offers a scalable alternative. However, models trained on synthetic data often struggle to generalize to real images due to differences in appearance, lighting, resolution, and noise.

Understanding how well a model trained in simulation generalizes to real data-and how this gap can be mitigated is the main motivation of this work.

1.3 Challenges and goals

The task of chessboard square classification under a synthetic-to-real training regime presents several practical and empirical challenges, many of which were directly observed throughout the development and evaluation of our system.

A central challenge is the **synthetic-to-real domain gap**. Although the synthetic data accurately encodes board geometry and piece placement, models trained solely on synthetic images exhibited poor generalization to real images in the zero-shot setting.
This gap manifested as high overall accuracy but very low macro F1-score on real data, indicating that the model relied on synthetic-specific visual cues that did not transfer well to real-world textures, lighting conditions, and noise patterns.

Another major challenge is **strong class imbalance**. Empty squares constitute the majority of board positions, causing accuracy metrics to be dominated by this class. In practice, this resulted in misleadingly high square-level accuracy during zero-shot evaluation, while most piece classes exhibited low recall and F1-scores. This imbalance required careful interpretation of results and motivated the use of macro F1-score as a primary evaluation metric.

Visual similarity between chess pieces further complicates classification. Pieces such as bishops, queens, and kings share similar shapes, particularly under varying illumination and resolution. This challenge was clearly reflected in the confusion matrices, where misclassifications frequently occurred between visually similar pieces, especially in the absence of real-data fine-tuning.

The system also proved **sensitive to perspective distortions and image resolution**.
Early experiments using low-resolution synthetic images resulted in degraded fine-grained feature learning and reduced precision on real data. Additionally, small inaccuracies in board detection and perspective normalization propagated to square-level crops, negatively affecting classification performance. These observations motivated improvements in synthetic data resolution and stricter preprocessing consistency.

Finally, the **limited availability of labeled real data** posed a significant constraint.
While fine-tuning on even a small real dataset substantially improved performance, the risk of overfitting was evident in training dynamics. This limitation necessitated careful training strategies, such as mixed synthetic-real training, to balance generalization and robustness within practical resource and time constraints.

In light of these challenges, the primary goals of this work are to:

1) evaluate the extent to which a model trained on synthetic data can generalize to real chessboard images.
2) quantify the impact of the synthetic-to-real domain gap using appropriate metrics beyond raw accuracy.

3) investigate training strategies such as fine-tuning and mixed synthetic-real training that mitigate these limitations while remaining feasible under practical data and time constraints.

1.4 Contributions

The main contributions of this work are as follows:

1) An end-to-end synthetic-to-real chessboard square classification pipeline, including board detection, perspective normalization, square-level extraction, and per-square multi-class classification using a convolutional neural network.

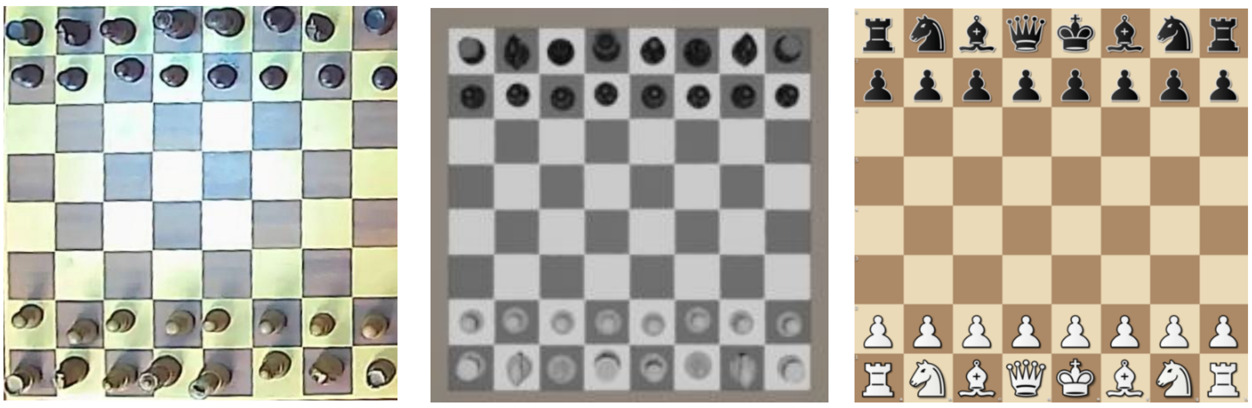
2) A systematic comparison of training strategies for sim-to-real transfer, evaluating zero-shot generalization, fine-tuning with limited real data, and mixed synthetic-real training, and analyzing their respective strengths and limitations.

3) An empirical analysis of architectural and data-generation choices, including the impact of model selection and synthetic data resolution on real-world performance.

4) A detailed examination of failure modes and practical trade-offs, highlighting the effects of class imbalance, visual similarity between pieces, and computational constraints on model design and training decisions.

1.5 Visual Examples

This section presents representative examples of the input and output data used throughout the project, including synthetic and real chessboard images and the corresponding predicted board states.



(a) (b) (c)

Figure 1: (a) Real chessboard image, (b) Synthetic chessboard image, (c) Predicted board-state visualization generated from model output.

2. Related Work

Chessboard recognition has been explored using both classical computer vision techniques and deep learning approaches. Traditional methods rely on handcrafted features and heuristics, while modern approaches employ convolutional neural networks for piece recognition and board reconstruction.

Synthetic data has been widely used in sim-to-real transfer for vision and robotics tasks. Prior work demonstrates that while synthetic data can provide strong supervision, reducing the domain gap often requires careful preprocessing, fine-tuning, or domain adaptation techniques. This work focuses on empirically analyzing these effects in the context of structured board-game recognition, with an emphasis on synthetic-to-real generalization under practical data constraints.

3. Method

3.1 System Overview

The proposed system follows a structured, modular pipeline designed to enable robust square-level classification while minimizing synthetic-to-real discrepancies. Given a single RGB image of a chessboard, the system processes the image through a sequence of geometric normalization and classification stages:

Input Image → Board Detection → Perspective Normalization → Square Extraction → CNN Classifier → Per-Square Predictions → Board State (FEN)

This design isolates the visual recognition task at the square level, allowing the classifier to focus on local appearance while relying on preprocessing to enforce global board consistency across synthetic and real images.

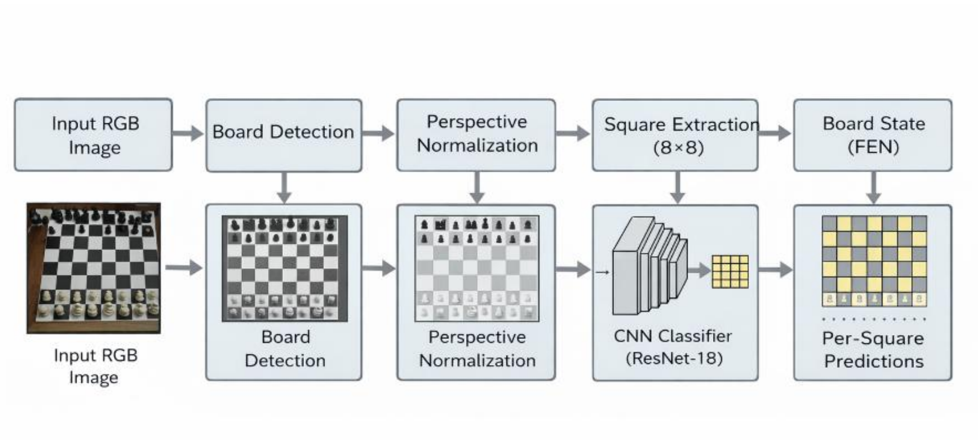


Figure 2: End-to-End pipeline for chessboard state classification.

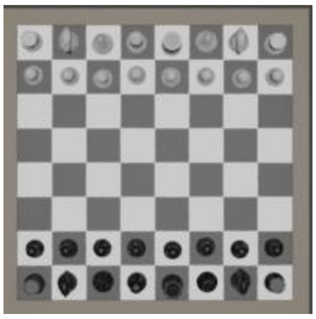
Overview of the full pipeline, from raw input image to per square classification and final board-state reconstruction (FEN).

3.2 Synthetic Data Generation

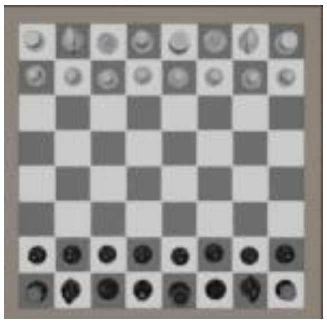
Synthetic training data is generated using a Blender-based rendering pipeline, where full chessboards are rendered directly from FEN strings. This approach provides precise square-level labels and enables large-scale data generation without manual annotation.

The generator supports controlled variation in camera viewpoint, lighting, and board configuration, while preserving exact ground-truth piece placement.

During early experimentation, synthetic images were rendered at a resolution of 720×720 to accelerate data generation and training iterations. However, this choice limited the model's ability to capture fine-grained visual details of chess pieces, resulting in reduced precision when evaluated on real images. Based on this observation, a higher-resolution synthetic dataset (1024×1024) was generated and used in later experiments, leading to improved feature learning and better real-world generalization.



(a)



(b)

Figure 3: (a) High resolution (1024) synthetic board.

(b) Low resolution (720) synthetic board.

3.3 Preprocessing

To reduce the synthetic-to-real domain gap and ensure consistent input geometry, identical preprocessing is applied to both synthetic and real images. The preprocessing pipeline consists of the following steps:

1) **Chessboard detection** to localize the board region in the input image.
2) **Perspective transformation** to obtain a canonical top-down board view.
3) **Cropping and resizing** to a fixed resolution.
4) **Square-level extraction**, producing individual square images with context aware margins.
5) **Input normalization** prior to classification.

This pipeline ensures that the classifier receives geometrically aligned and scale-consistent square images, allowing it to focus on appearance rather than global layout variations.

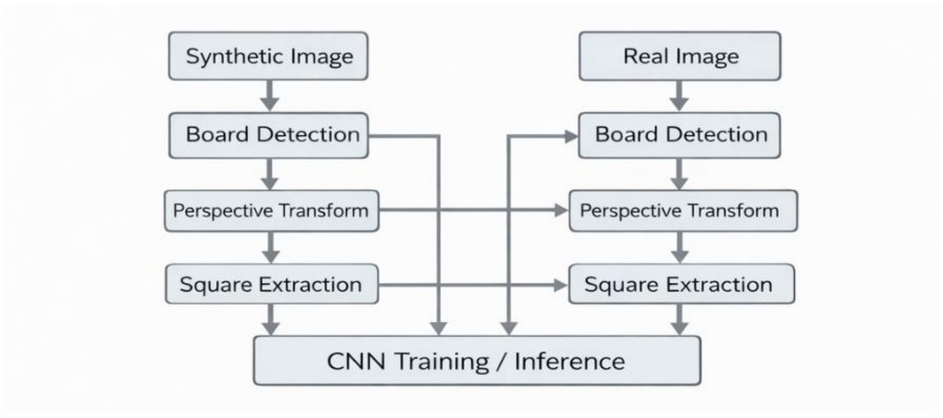
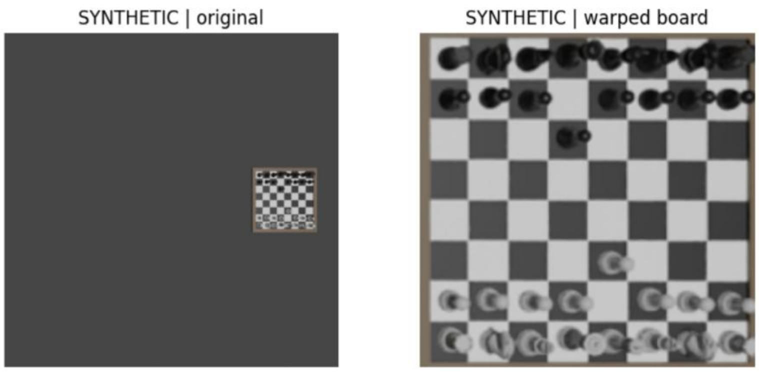


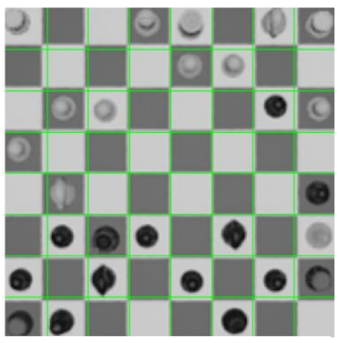
Figure 4: Image preprocessing and training pipeline.

3.3.1 Preprocessing Examples

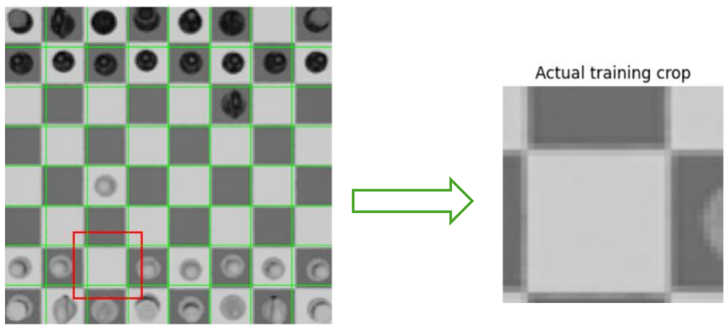
Figure 5 presents qualitative examples of the preprocessing stages applied to real images. The examples illustrate the detection and rectification of the board, grid localization, and the extraction of individual square crops used as inputs to the classifier.



(a)



(b)



(c) **(d)**

Figure 5: (a) Board detect and crop algorithm (real image -> wrapped image ready to crop).

(b) Grid detector.

(c) Square crop + margins.

(d) Single square crop.

3.4 Model Architecture

Square classification is performed using a **ResNet-18** convolutional neural network.
Each input square image is independently processed by the network, which outputs a probability distribution over **13 classes** corresponding to the 12 chess pieces and the empty square.

The ResNet-18 architecture consists of an initial convolutional layer followed by four residual stages with increasing channel depth, global average pooling, and a fully connected classification head. Residual connections enable stable training and effective gradient flow, which proved important given the limited amount of real training data.

Several alternative architectures, including EfficientNet and deeper ResNet variants, were evaluated during development. While some models showed comparable performance, ResNet-18 offered the best balance between accuracy, training stability, and computational efficiency, enabling extensive experimentation within practical time constraints.

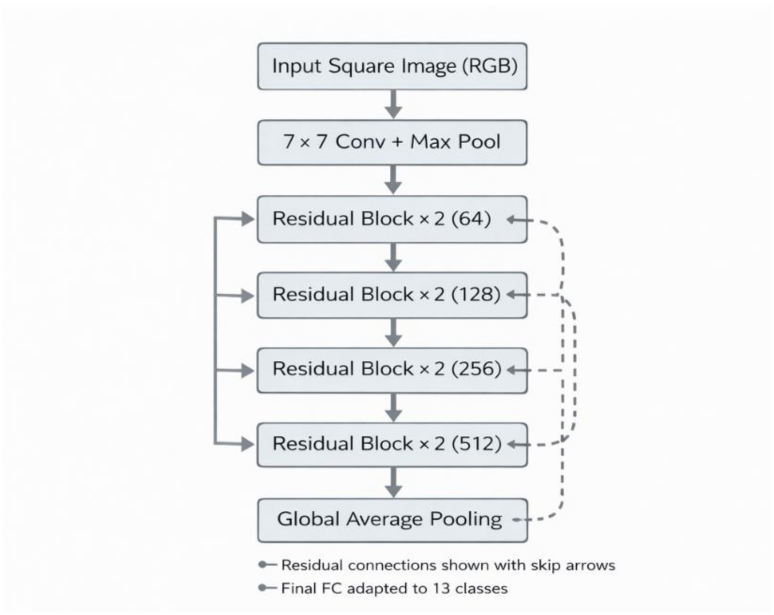


Figure 6: Res-Net-18 architecture used for square classification.

ResNet-18 backbone adapter for per-square chess piece classification.

3.5 Training Procedure

We trained the square classifier under three regimes aligned with the project requirements:

1) **synthetic-only training** for zero-shot evaluation.
2) **fine-tuning on real data.**
3) **mixed synthetic-real training** with imbalance-aware sampling and loss design.

In all settings, the task is formulated as 13-way classification (12 chess pieces and the empty square). Training was performed using cross-entropy-based objectives optimized with AdamW, and best checkpoints were selected according to validation macro F1-score.

Input configuration:

For all experiments, chessboard images are first rectified to a canonical top-down representation at 512×512 ("warp size"). Individual square inputs are then extracted as 128×128 RGB crops. We use context-aware cropping (as described in Section 3.3): during training, the crop scale factor k is sampled from a range, while at evaluation it is fixed.

Experiment A: Synthetic-only Training (Zero-shot Setup)

To measure synthetic-to-real generalization, we trained exclusively on synthetic squares and evaluated directly on the real test set without using any real training data.

- **Batch size:** 16 (GPU) / 8 (CPU)
- **Epochs:** 5
- **Learning rate:** 3×10^{-4}
- **Weight decay:** 1×10^{-4}
- **Dropout:** 0.2
- **Label smoothing:** 0.06
- **Gradient accumulation:** 4 steps
- **Training samples per epoch:** 30,000 square crops
- **Validation samples:** 3,000 square crops (subset for faster feedback)

This configuration enabled rapid iteration while maintaining stable optimization. The resulting model was evaluated in the **zero-shot** setting on the real test split.

Experiment B: Fine-tuning on Real Data

To reduce the domain gap, we fine-tuned the model on labeled real data. Due to strong class imbalance (with empty squares dominating), we used a **class-weighted** cross-entropy loss, where weights were computed from the real training split.

- **Warp size:** 512, **Square crop size:** 128
- **Context crop scale:** $k \sim U(1.4, 2.0)$ during training, $k = 1.6$ at evaluation
- **Batch size:** 128 (GPU) / 32 (CPU)
- **Epochs:** 5
- **Learning rate:** 1×10^{-4}
- **Weight decay:** 1×10^{-4}

- **Dropout:** 0.2
- **Label smoothing:** 0.04
- **Optimizer:** AdamW
- **Loss:** class-weighted cross-entropy

Model selection was performed using validation macro F1-score, and the best checkpoint was saved accordingly.

Experiment C: Mixed Synthetic-Real Training

To achieve robust performance across domains, we trained on a mixed dataset containing both synthetic and real images. This regime explicitly addresses domain shift and class imbalance using (1) **weighted sampling at the square level** and (2) a **weighted focal loss** optimized for macro F1-score.

Training configuration

- **Warp size:** 512, **Square crop size:** 128
- **Context crop scale:** $k \sim U(1.35{,}2.0)$ during training, $k = 1.6$ at evaluation
- **Batch size:** 128 (GPU) / 32 (CPU)
- **Epochs:** 15
- **Learning rate:** 3×10^{-4}
- **Weight decay:** 1×10^{-4}
- **Dropout:** 0.2
- **Optimizer:** AdamW
- **Training samples per epoch:** 32,000 square crops (sampled with replacement)
- **Mixed validation subset:** 7,000 square crops

Imbalance-aware sampling:

A **WeightedRandomSampler** was used to alter the effective training distribution:

- **Real board upweight:** real boards sampled x**2.0** more often than synthetic boards.
- **Non-empty square upweight:** occupied squares weighted x**3.0** relative to empty squares.
- **Rare piece boost:** additional weight of **0.5** for rare pieces (applied to kings, queens, bishops, and knights of both colors).

This strategy reduces the dominance of empty squares and increases the frequency of rare pieces during training.

Loss function:

Experiment C uses a **weighted focal loss**:

- **Focal gamma:** 1.7
- **Label smoothing:** 0.02
- **Class weights:** computed from the real training split.

Training-time Augmentation:

Data augmentation was applied only during training via the transformation pipeline. Augmentations were designed to improve robustness to illumination changes, blur, and small geometric variations, and were not applied at inference time.

4. Experiments

4.1 Datasets

Experiments were conducted using two domains:

1) Synthetic dataset rendered with Blender from FEN strings. The synthetic set contains 9,725 board images (across 71 games), split into 8,450 train (63 games), 510 validation (4 games), and 765 test (4 games).
2) Real dataset, split into training, validation, and test sets. The final evaluation is performed on REAL_TEST_FULL, which contains 11,520 square samples (64 squares per board).

To characterize board quality and filtering behavior, we computed a board-level score distribution by domain. Across available boards, synthetic images show a relatively tight distribution (mean ≈ **0.805**, std ≈ **0.064**), while real images exhibit higher variance (mean ≈ **0.776**, std ≈ **0.233**), reflecting real-world variability in capture conditions. We additionally identified 24 “bad” boards, all of which originated from the real domain, consistent with the higher variance and occasional preprocessing failures observed in real

4.2 Evaluation Metrics

We report:

1) Square-level accuracy, measuring the fraction of correctly classified squares.
2) Macro F1-score, computed as the average F1 across all 13 classes.
3) Per-class precision, recall, and F1-score, to diagnose class-specific behavior.

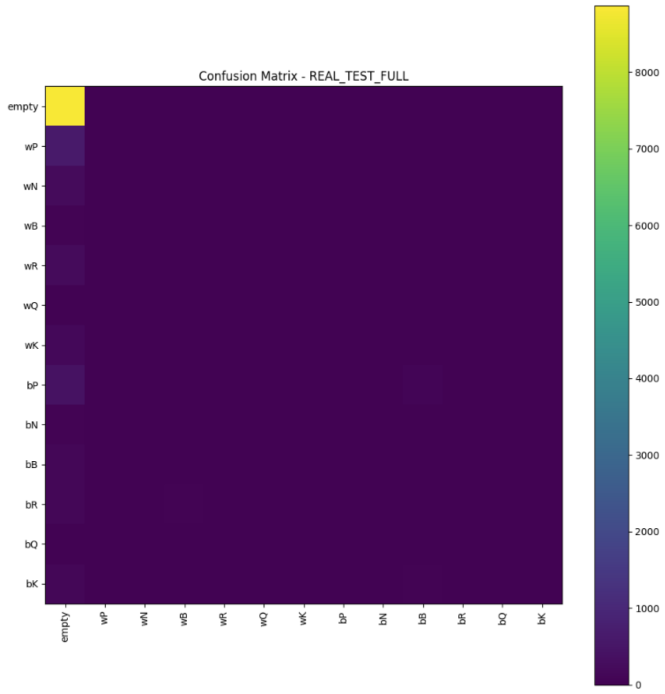
Macro F1-score is emphasized because the dataset is highly imbalanced: empty squares dominate, so a model that predicts empty frequently can achieve high accuracy while performing poorly on minority piece classes. This effect is clearly observed in the zero-shot setting.

4.3 Baseline and Zero-Shot Generalization

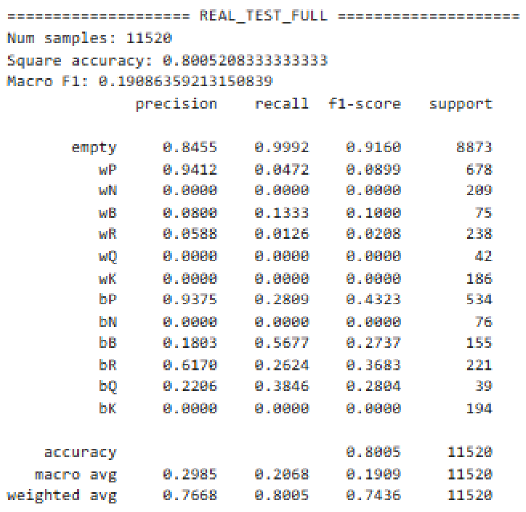
Baseline definition. Our baseline is the synthetic-only model evaluated directly on real images without any real-data training. This setting measures raw synthetic-to-real transfer.

In the zero-shot setting, the model achieves relatively high square accuracy (≈ 0.80) but a low macro F1-score (≈ 0.19). This gap indicates that most correct predictions come from the majority empty class, while piece classes have low recall.

Figure 7(a) shows the normalized confusion matrix on the real test set. The matrix is dominated by a strong empty-square diagonal, while many piece classes are confused with each other or predicted as empty. Figure 7(b) further confirms this through per-class metrics, where minority classes exhibit substantially lower F1-scores than the empty class.



(a)



```
=================== REAL_TEST_FULL ===================
Num samples: 11520
Square accuracy: 0.8005208333333333
Macro F1: 0.19086359213150839
             precision    recall  f1-score   support

       empty     0.8455    0.9992    0.9160      8873
          wP     0.9412    0.0472    0.0899       678
          wN     0.0000    0.0000    0.0000       209
          wB     0.0800    0.1333    0.1000        75
          wR     0.0588    0.0126    0.0208       238
          wQ     0.0000    0.0000    0.0000        42
          wK     0.0000    0.0000    0.0000       186
          bP     0.9375    0.2809    0.4323       534
          bN     0.0000    0.0000    0.0000        76
          bB     0.1803    0.5677    0.2737       155
          bR     0.6170    0.2624    0.3683       221
          bQ     0.2206    0.3846    0.2804        39
          bK     0.0000    0.0000    0.0000       194

    accuracy                         0.8005     11520
   macro avg     0.2985    0.2068    0.1909     11520
weighted avg     0.7668    0.8005    0.7436     11520
```

(b)

Figure 7:

(a) Confusion matrix for zero-shot evaluation on real test data and experiment data results.

(b) Per-class precision, recall, and F1-score for zero-shot evaluation on the real test set.

As shown in Figure 8, training loss decreases steadily and training accuracy improves rapidly. however, validation accuracy saturates early and remains significantly lower. This behavior indicates that successful optimization on synthetic data does not translate to improved real-world performance, highlighting the synthetic-to-real domain gap.

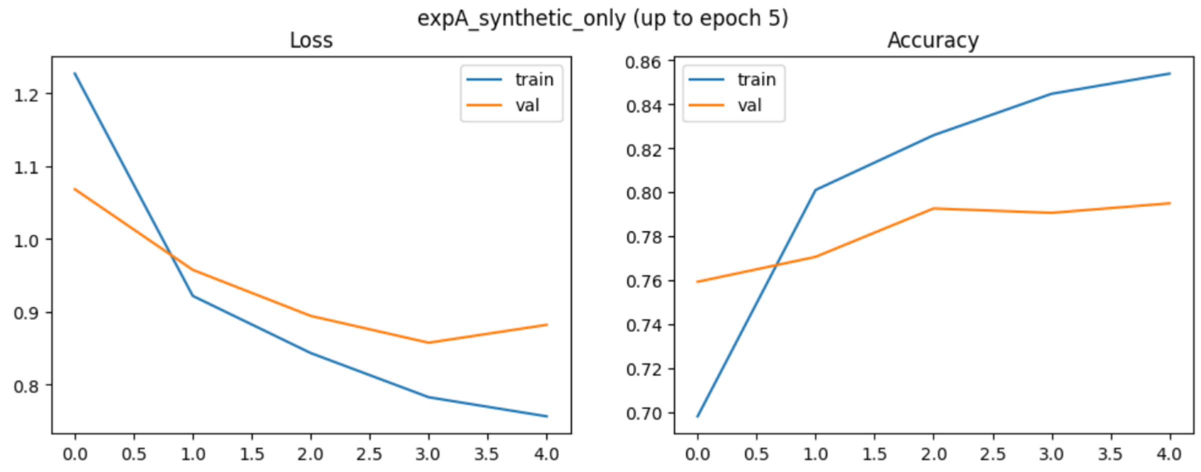
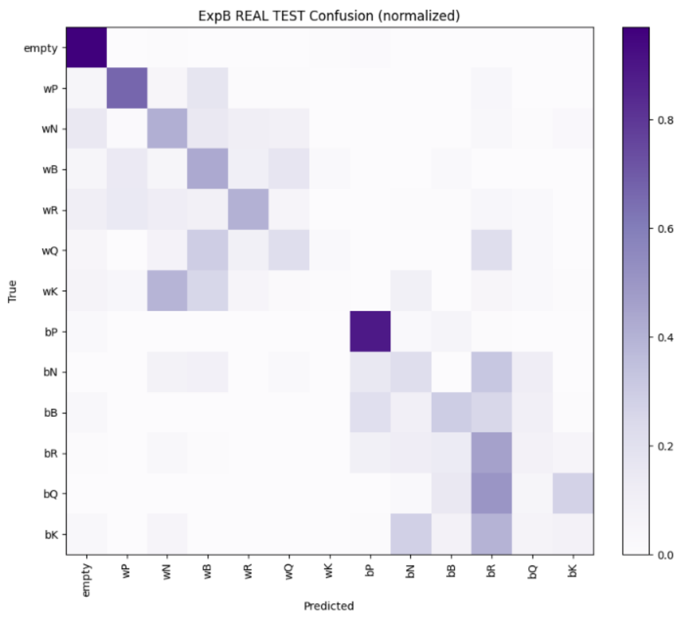


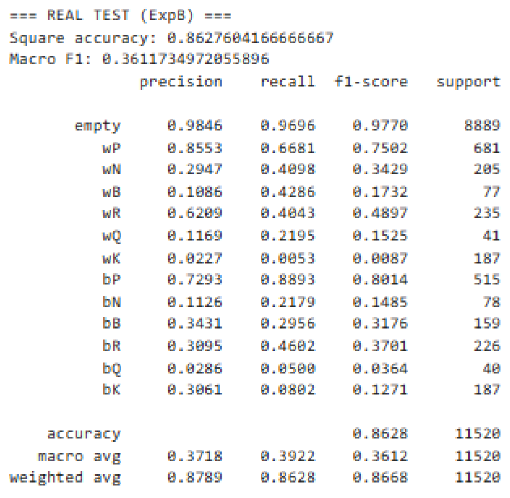
Figure 8: Training and validation loss and accuracy across epochs for the ResNet-18 model.

4.4 Fine-Tuning on Real Data

To reduce domain shift, we fine-tuned the synthetic-pretrained model using labeled real training data. Fine-tuning improves performance substantially, increasing square accuracy to approximately 0.86 and macro F1-score to approximately 0.36. This indicates that even limited real supervision significantly improves minority-class recognition.

As shown in Figure 9(a), fine-tuning yields a clearer diagonal structure in the confusion matrix, reducing misclassification of pieces as empty and improving separation between visually similar classes. Figure 9(b) shows that per-class recall and F1 improve for most pieces, consistent with reduced domain mismatch.





```
=== REAL TEST (ExpB) ===
Square accuracy: 0.862760416666667
Macro F1: 0.3611734972055896
              precision    recall  f1-score   support

       empty     0.9846    0.9696    0.9770      8889
          wP     0.8553    0.6681    0.7502       681
          wN     0.2947    0.4098    0.3429       205
          wB     0.1086    0.4286    0.1732        77
          wR     0.6209    0.4043    0.4897       235
          wQ     0.1169    0.2195    0.1525        41
          wK     0.0227    0.0053    0.0087       187
          bP     0.7293    0.8893    0.8014       515
          bN     0.1126    0.2179    0.1485        78
          bB     0.3431    0.2956    0.3176       159
          bR     0.3095    0.4602    0.3701       226
          bQ     0.0286    0.0500    0.0364        40
          bK     0.3061    0.0802    0.1271       187

    accuracy                         0.8628     11520
   macro avg     0.3718    0.3922    0.3612     11520
weighted avg     0.8789    0.8628    0.8668     11520
```

(a) (b)

Figure 9:

(a) Confusion matrix after fine-tuning the synthetic-pretrained model on limited real data and experiment data results.

(b) Per-class precision, recall, and F1-score after fine-tuning on real data.

Figure 10 shows that while training accuracy increases rapidly during fine-tuning, validation accuracy exhibits limited improvement and fluctuates across epochs. This gap suggests overfitting due to the small size of the real training set, motivating the exploration of mixed training strategies.

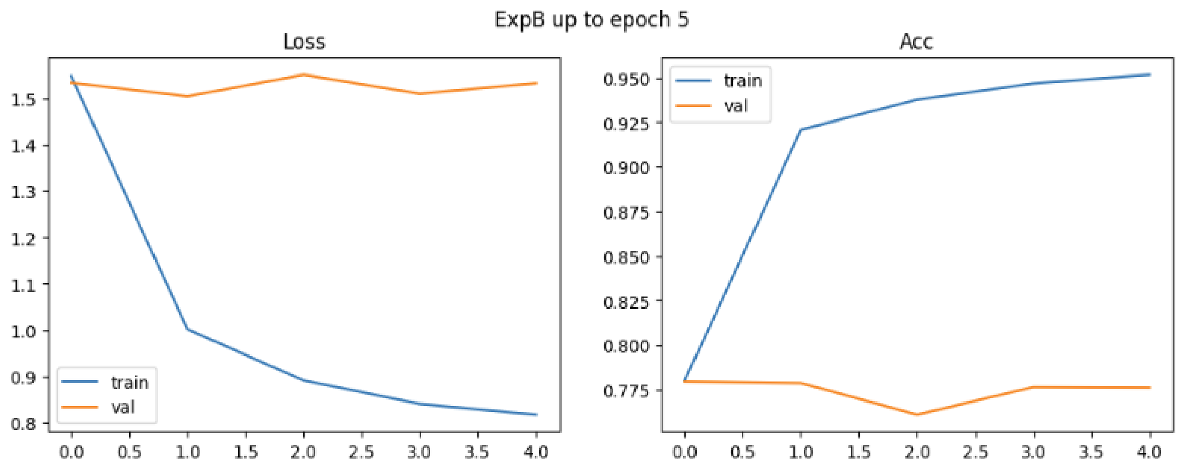


Figure 10: Training and validation loss and accuracy across epochs for the ResNet-18 model.

<u>4.5 Mixed Synthetic and Real Training</u>

We further trained a model on a mixed synthetic-real dataset to achieve balanced generalization across domains. Mixed training achieves square accuracy of approximately 0.82 and macro F1-score of approximately 0.37 on real test data, while also preserving strong performance on synthetic test data.

Mixed training used imbalance-aware sampling and loss design (Section 3.5), including:

1) Real board upweighting
2) Non-empty square upweighting
3) Rare-piece boosting
4) Weighted focal loss

Figure 11 shows confusion matrices on both real and synthetic test sets, demonstrating improved recognition on real data while maintaining strong diagonal dominance on synthetic data. Figure 12(a-b) provides per-class metrics, showing more balanced performance than the zero-shot baseline and competitive performance relative to fine-tuning.

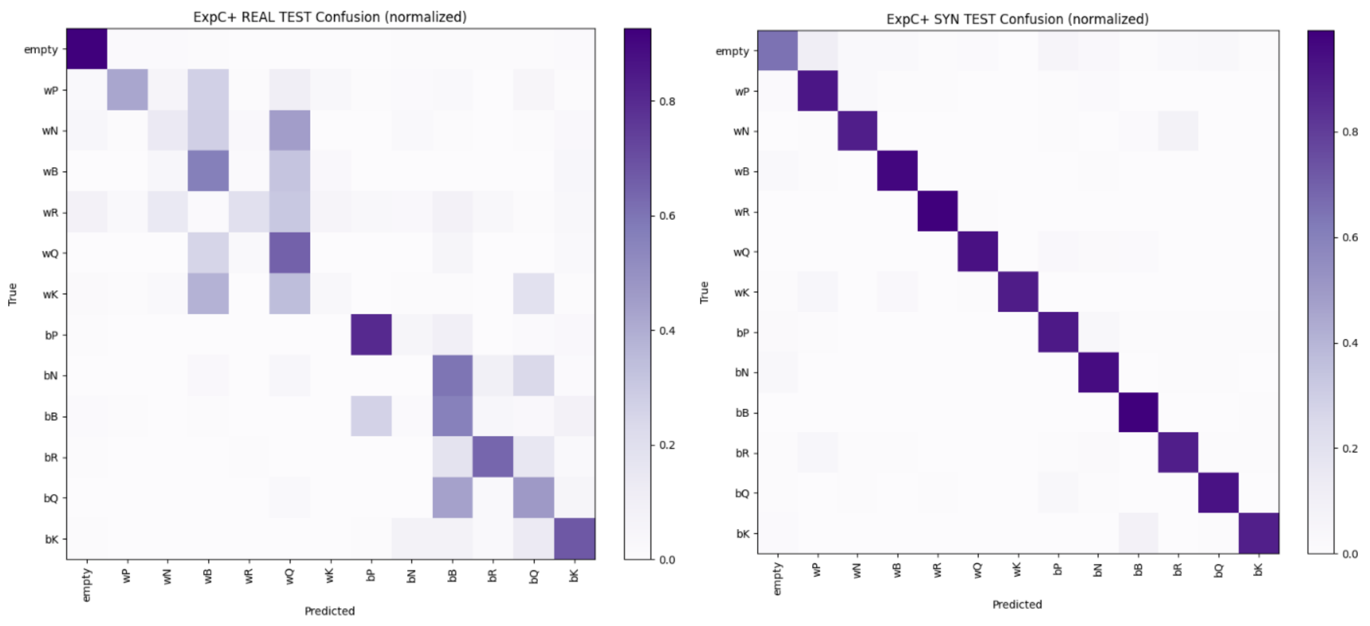
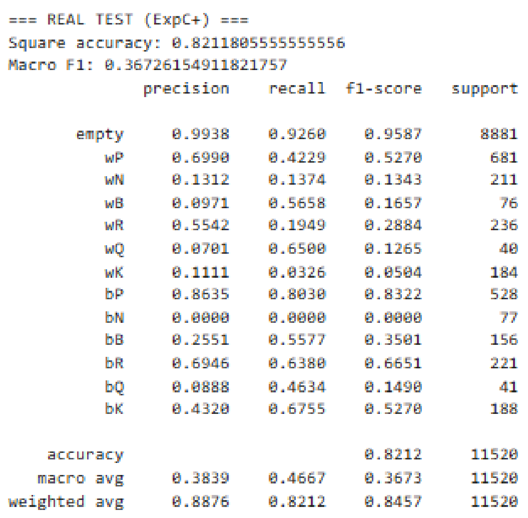


Figure 11: Confusion matrices for mixed synthetic-real training evaluated on real test data and synthetic test data.

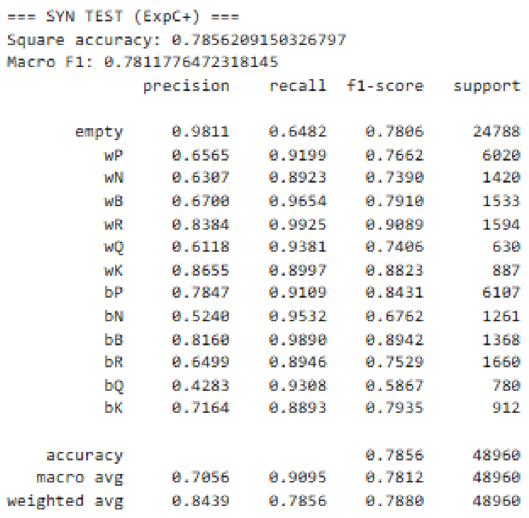


```
=== REAL TEST (ExpC+) ===
Square accuracy: 0.8211805555555556
Macro F1: 0.36726154911821757
              precision    recall  f1-score   support

       empty     0.9938    0.9260    0.9587      8881
          wP     0.6990    0.4229    0.5270       681
          wN     0.1312    0.1374    0.1343       211
          wB     0.0971    0.5658    0.1657        76
          wR     0.5542    0.1949    0.2884       236
          wQ     0.0701    0.6500    0.1265        40
          wK     0.1111    0.0326    0.0504       184
          bP     0.8635    0.8030    0.8322       528
          bN     0.0000    0.0000    0.0000        77
          bB     0.2551    0.5577    0.3501       156
          bR     0.6946    0.6380    0.6651       221
          bQ     0.0888    0.4634    0.1490        41
          bK     0.4320    0.6755    0.5270       188

    accuracy                         0.8212     11520
   macro avg     0.3839    0.4667    0.3673     11520
weighted avg     0.8876    0.8212    0.8457     11520
```

(a)



```
=== SYN TEST (ExpC+) ===
Square accuracy: 0.7856209150326797
Macro F1: 0.7811776472318145
              precision    recall  f1-score   support

       empty     0.9811    0.6482    0.7806     24788
          wP     0.6565    0.9199    0.7662      6020
          wN     0.6307    0.8923    0.7390      1420
          wB     0.6700    0.9654    0.7910      1533
          wR     0.8384    0.9925    0.9089      1594
          wQ     0.6118    0.9381    0.7406       630
          wK     0.8655    0.8997    0.8823       887
          bP     0.7847    0.9109    0.8431      6107
          bN     0.5240    0.9532    0.6762      1261
          bB     0.8160    0.9890    0.8942      1368
          bR     0.6499    0.8946    0.7529      1660
          bQ     0.4283    0.9308    0.5867       780
          bK     0.7164    0.8893    0.7935       912

    accuracy                         0.7856     48960
   macro avg     0.7056    0.9095    0.7812     48960
weighted avg     0.8439    0.7856    0.7880     48960
```

(b)

Figure 12:

(a) Performance of mixed synthetic-real training on the real test set.

(b) Performance of mixed synthetic-real training on the synthetic test set.

As shown in Figure 13, mixed training results in smoother validation curves and improved stability compared to fine-tuning alone. Although training accuracy continues to increase, validation accuracy remains more consistent across epochs, indicating improved generalization across domains.

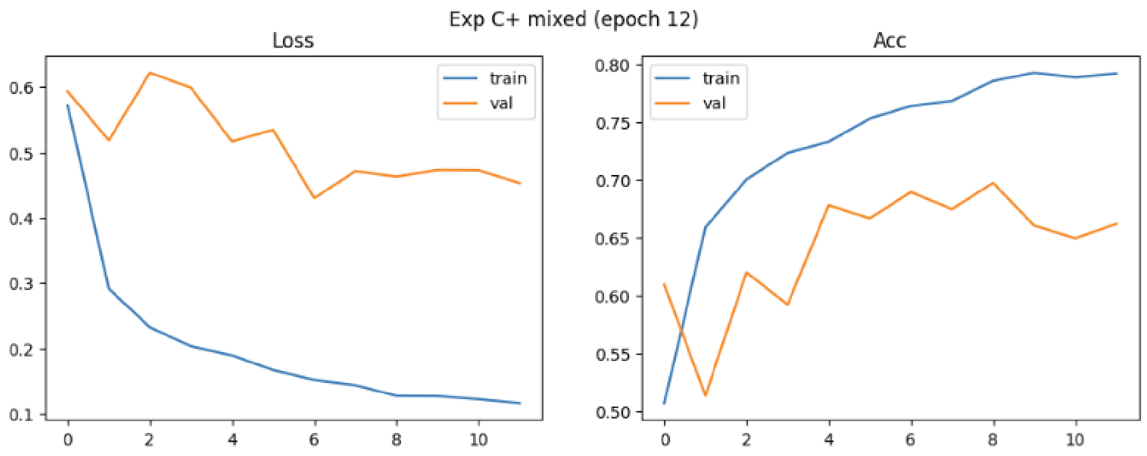


Figure 13: Training and validation loss and accuracy across epochs for the ResNet-18 model.

In addition to quantitative results, Figure 14 provides a qualitative example of the final square-based model applied to a real input image. The visualization shows that mixed synthetic-real training enables pretty decent accurate per-square predictions under real-world imaging conditions, including variations in lighting and perspective.

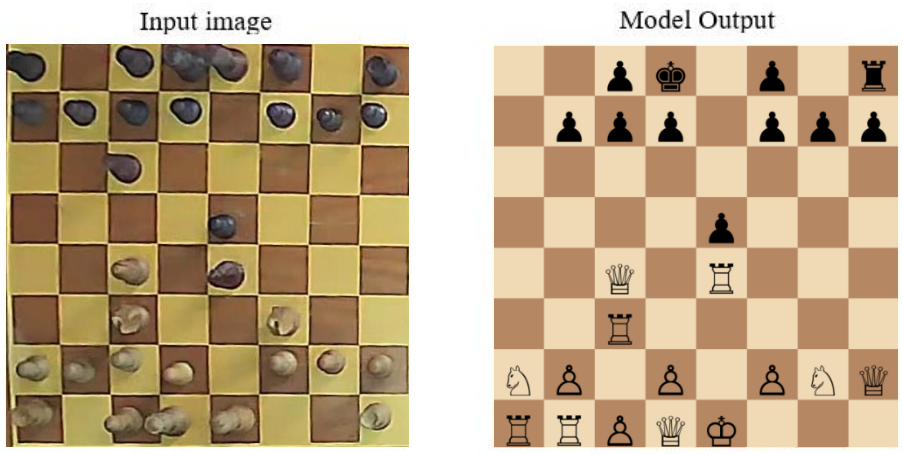
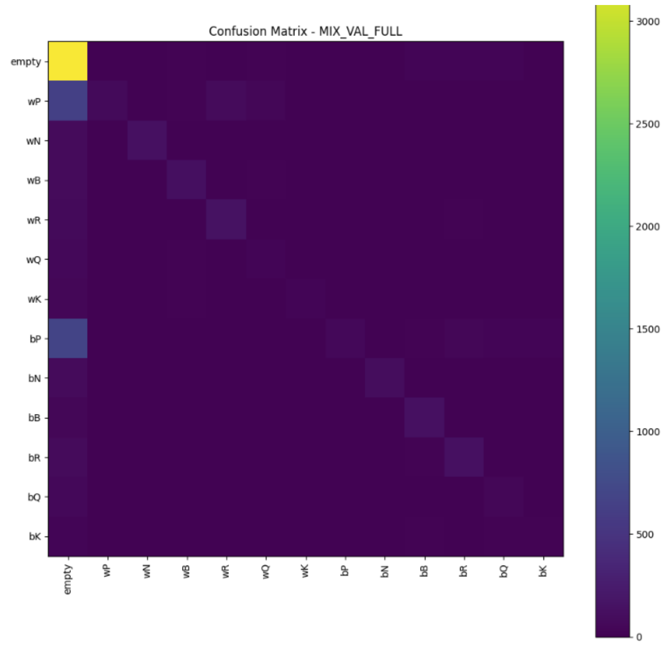


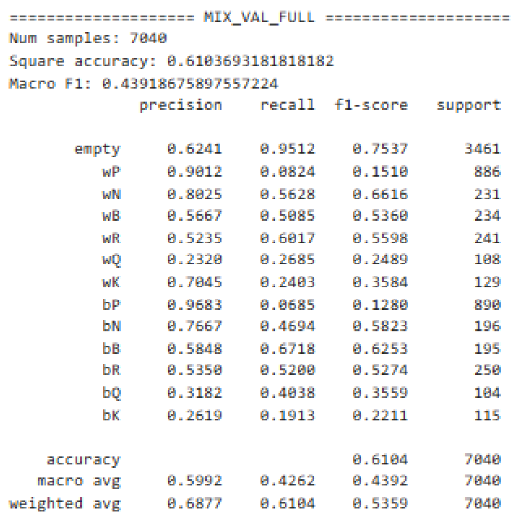
Figure 14: Qualitative example of the final square-level classification pipeline.

4.6 Validation Analysis

Model selection for mixed training was performed using REAL validation macro F1-score. Figure 15(a) shows the confusion matrix on the mixed validation set, and Figure 15(b) summarizes per-class metrics. Validation error patterns closely match those observed on the real test set, suggesting that the validation split provides a reliable proxy for real-world performance.



(a)



```
==================== MIX_VAL_FULL ====================
Num samples: 7040
Square accuracy: 0.6103693181818182
Macro F1: 0.43918675897557224
              precision    recall  f1-score   support

       empty     0.6241    0.9512    0.7537      3461
          wP     0.9012    0.0824    0.1510       886
          wN     0.8025    0.5628    0.6616       231
          wB     0.5667    0.5085    0.5360       234
          wR     0.5235    0.6017    0.5598       241
          wQ     0.2320    0.2685    0.2489       108
          wK     0.7045    0.2403    0.3584       129
          bP     0.9683    0.0685    0.1280       890
          bN     0.7667    0.4694    0.5823       196
          bB     0.5848    0.6718    0.6253       195
          bR     0.5350    0.5200    0.5274       250
          bQ     0.3182    0.4038    0.3559       104
          bK     0.2619    0.1913    0.2211       115

    accuracy                         0.6104      7040
   macro avg     0.5992    0.4262    0.4392      7040
weighted avg     0.6877    0.6104    0.5359      7040
```

(b)

Figure 15:

(a) Confusion matrix on the mixed validation set used for model selection.

(b) Performance on the mixed validation set.

4.7 Summary of Experimental Results

This section summarizes the quantitative results of the different training strategies evaluated in this work. While previous subsections presented detailed per-class metrics and confusion matrices, Table 1 provides a consolidated comparison of overall performance across training regimes.

The results clearly demonstrate the limitations of zero-shot transfer, which achieves relatively high square accuracy but poor macro F1-score due to strong class imbalance. Fine-tuning on limited real data substantially improves performance, particularly for minority piece classes. Mixed synthetic-real training achieves the most balanced performance, maintaining strong generalization to real data while preserving performance on synthetic data.

Training Regime	Square Accuracy	Macro F1-score
Zero-shot (Synthetic to Real)	~0.80	~0.19
Fine-tuned on Real Data	~0.86	~0.36
Mixed Synthetic + Real Training	~0.82	~0.37

Table 1: Overall performance comparison of zero-shot, fine-tuned, and mixed training strategies evaluated on the real test set.

5. Ablation Study

To assess the contribution of individual components in the proposed pipeline, we perform a set of ablation experiments. Each ablation removes or alters a single design choice while keeping the rest of the system unchanged. Performance is evaluated on the real test set using square accuracy and macro F1-score.

Ablation Results:

Configuration	Square Accuracy	Macro F1-score
Full method (mixed synthetic-real training)	~0.82	~0.37
No synthetic pretraining (real-only training)	~0.79	~0.31
Low-resolution synthetic data (720 instead of 1024) *	~0.72	~0.14
No fine-tuning / adaptation (zero-shot)	~0.80	~0.19
No mixed training (fine-tuning only)	~0.86	~0.36

Table 2: Ablation study evaluating the impact of key components in the proposed training pipeline. All results are reported on the real test set.

(*) Results obtained from early experiments using low-resolution (720) synthetic data.

Analysis of Ablations:

1) **Synthetic pretraining:** Removing synthetic pretraining and training solely on real data leads to unstable training and reduced performance due to limited real-data availability. This confirms the importance of synthetic data as an initial source of supervision.

2) **Synthetic data resolution:** Training with low-resolution synthetic images significantly degrades performance, particularly for visually similar pieces. This highlights the importance of high-resolution synthetic data for learning fine-grained features that transfer to real images.

3) **Fine-tuning / adaptation:** Omitting real-data adaptation (zero-shot evaluation) results in a substantial drop in macro F1-score despite relatively high accuracy, demonstrating that synthetic-only training is insufficient to bridge the domain gap.

4) **Mixed training strategy:** Replacing mixed training with fine-tuning alone yields comparable accuracy but slightly lower robustness and stability. Mixed training provides a better balance between real and synthetic domains, improving generalization while preserving performance on synthetic data.

Overall, the ablation study demonstrates that each component of the proposed pipeline contributes to performance, and that removing any single component leads to measurable degradation.

6. What Did Not Work

During development, several alternative approaches were explored but ultimately abandoned due to limited effectiveness or practical constraints. These negative results played an important role in shaping the final system design.

First, we experimented with **larger network architectures**, including EfficientNet-B0 and deeper ResNet variants (e.g. ResNet-50). While these models increased representational capacity, they consistently required longer training times and higher memory usage, without providing reliable improvements in macro F1-score. Given the limited project timeframe and the need for extensive experimentation, these architectures were deemed impractical compared to ResNet-18.

Second, early experiments were conducted using **low-resolution synthetic data** to accelerate dataset generation and training. Although this setup produced reasonable square-level accuracy, it failed to capture fine-grained visual cues necessary to distinguish visually similar pieces (e.g. bishops vs. pawns). This limitation became evident in poor precision and confusion patterns on real data, motivating the transition to higher-resolution synthetic images.

Finally, we explored **full-board classification**, training a model to predict the entire board state directly rather than classifying individual squares. While this approach yielded surprisingly reasonable qualitative predictions in some cases, it suffered from poor scalability, increased model complexity, and error propagation across the board. As a result, the square-level classification formulation was preferred for its modularity, interpretability, and robustness.

Overall, these unsuccessful attempts highlighted key challenges related to domain shift, data resolution, and model complexity, and directly informed the design choices adopted in the final pipeline.

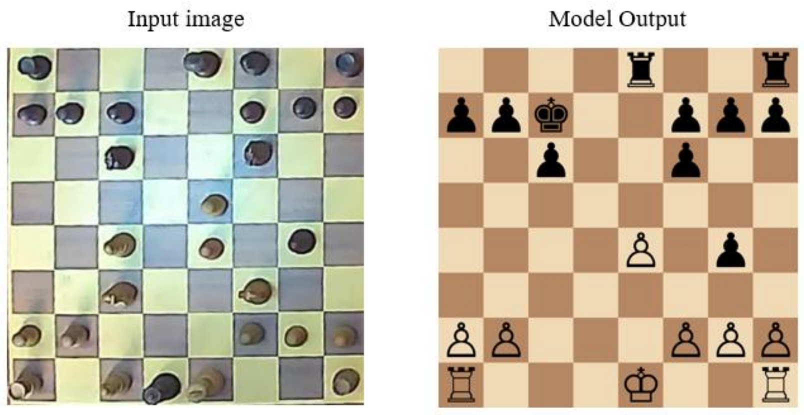


Figure 16: Example input and output of the full-board classification model explored during development. While predictions are qualitatively reasonable, the approach was ultimately abandoned due to robustness and scalability limitations.

7. Discussion and Limitations

A key consideration throughout this project was the trade-off between model performance and practical constraints such as training time, dataset size, and architectural complexity. While larger models and longer training schedules could potentially improve results, they significantly increase computational cost and limit the ability to iterate and evaluate multiple design choices within a fixed project timeline.

ResNet-18 was selected as a balanced architecture that enabled stable training, repeated experimentation, and systematic analysis across different training regimes. Similarly, training duration and dataset size were chosen to prioritize methodological improvements-such as fine-tuning strategies, mixed-domain training, and imbalance handling-over brute-force scaling.

Several failure cases reflect these limitations. In particular, visually similar pieces remain challenging under certain lighting conditions or partial occlusions, and class imbalance continues to impact minority-piece precision. The initial use of low-resolution synthetic data further illustrates this trade-off: while it accelerated early experimentation, it limited fine-grained feature learning and degraded performance on real images.

Given additional time and resources, future improvements could include generating more diverse and photorealistic synthetic data, extending training schedules, incorporating explicit domain adaptation techniques, or exploring more expressive architectures. These directions offer promising avenues for further reducing the synthetic-to-real gap while maintaining robustness and scalability.

8. Conclusion

This work presents a systematic study of synthetic-to-real generalization for chessboard square classification. Through extensive experimentation, we show that while zero-shot transfer from synthetic data remains limited, fine-tuning and mixed synthetic-real training substantially reduce the domain gap and improve piece-level recognition. Our results highlight the critical role of data quality, resolution, and training strategy, and demonstrate that strong sim-to-real performance can be achieved using a relatively lightweight model when design choices are carefully balanced under practical constraints.

9. References

1) J. P. P. Ramalho et al., *Determining Chess Game State from an Image*, arXiv preprint arXiv:2104.14963, 2021. Available at: https://arxiv.org/abs/2104.14963
2) PyTorch Contributors, *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, https://pytorch.org.
3) Blender Foundation, *Blender - Open Source 3D Creation Suite*, https://www.blender.org.

4) Course lectures slides and teaching materials, Intro to Deep Learning course, Ben-Gurion University, 2026.

10. Links

Github:

https://github.com/OmerYaish/Chess-Project#

Google Drive:
https://drive.google.com/drive/folders/190msdqfN2ncLaeTgLDZP4yoPD1drtCcN?usp=drive_link

Project webpage:
https://omeryaish.github.io/Chess-Project-Webpage/