Chess Analysis Using AI Methodologies

Team 32

CS7.403b : Statistical methods in AI

Introduction to Chess Analysis

- Chess: A strategic game with deep structure and rich patterns.
- Chess engines (e.g., Stockfish) provide precise numerical evaluations.
- Limitation: Lack of intuitive explanations for human players.
- Goal: Develop a human-aligned, interpretable framework for board strength evaluation.

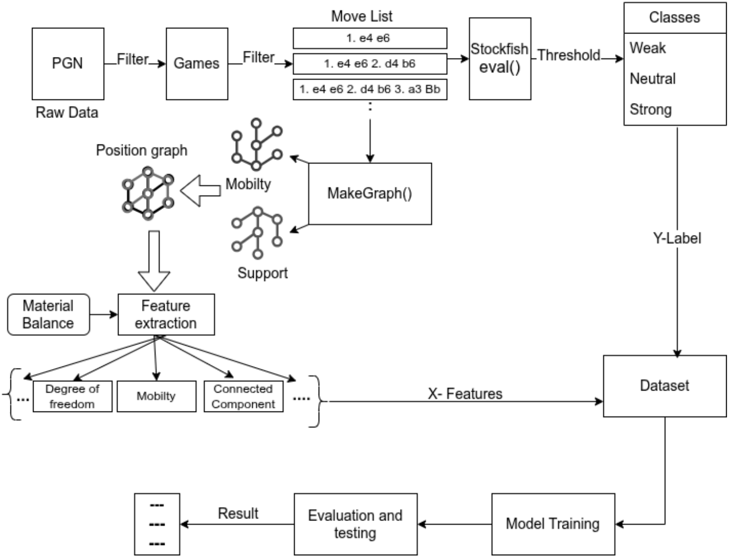
Problem Statement

- Engine evaluations are computationally robust but lack interpretability.
- During gameplay, players rely on intuition without engine access.
- Post-game analysis lacks explanations for move quality (e.g., poor piece coordination, weak structure).
- Need: A model that provides clear, human-readable metrics for strategic decision-making.

Our Approach

- Model chess board states as graphs.
- Extract ≈ 50 interpretable features (e.g., connectivity, mobility, material balance).
- Label board states as strong, neutral, or weak using Stockfish evaluations.
- Train a classification model to predict board strength.
- Enhance interpretability via feature importance, ablation studies, and error analysis.

Proposed Methodology



Pipeline Overview

- **Phase 1: Data Parsing** – Parse PGN (Lichess 2025, classical/rapid games), extract move lists until game conclusion reached.
- **Phase 2: Stockfish Evaluation** – Evaluate moves (depth 20), label positions: mate scores ± 1000 (strong/weak); non-mate thresholds: $|score| \leq 0.26$ (neutral), $0.26 < |score| \leq 1.5$ (weak), $|score| > 1.5$ (strong).
- **Phase 3: Graph Construction** – Build mobility (legal moves) and support (attack/defence) graphs, merge into position graph.
- **Phase 4: Feature Extraction** – Extract 50+ features , Material Based

  (central control, king safety, pawn structure), graph-based (degree of freedom, mobility, threat-to-support ratio).
- **Phase 5: Model Training** – Create dataset (X-features, Y-labels), split train/test, train and evaluate model.

Data Acquisition

- Source: Lichess Open Database (2025)
- Formats: Blitz, Rapid, Classical (exclude HyperBullet, Bullet for strategic focus).
- PGN to CSV conversion for efficient processing.
- Filter out unnecessary metadata (e.g., Site, Date).

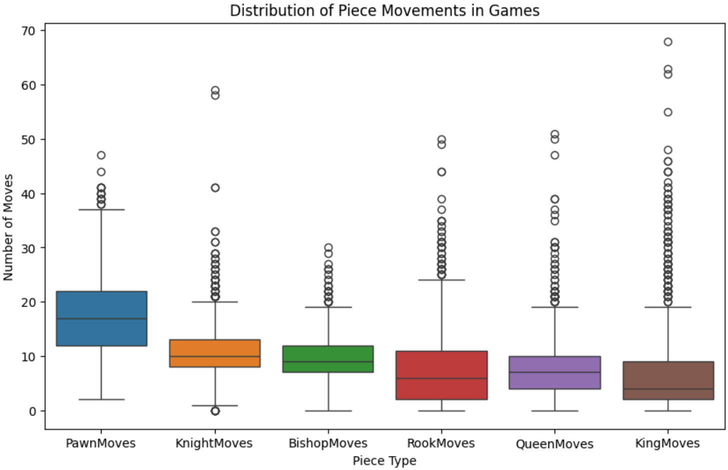
Raw Game Data

```
[Event "Rated Blitz game"]
[Site "https://lichess.org/sKojzYYV"]
[Date "2025.02.01"]
[Round "-"]
[White "ferlionrod"]
[Black "Tahafouad"]
[Result "0-1"]
[UTCDate "2025.02.01"]
[UTCTime "00:00:00"]
[WhiteElo "1671"]
[BlackElo "1608"]
[WhiteRatingDiff "-6"]
[BlackRatingDiff "+6"]
[ECO "A45"]
[Opening "Paleface Attack"]
[TimeControl "180+0"]
[Termination "Normal"]

1. d4 { [%clk 0:03:00] } 1... Nf6 { [%clk 0:03:00] } 2. f3 { [%clk 0:02:58] }
```

Example game from the Lichess February 2025 (PNG Format)

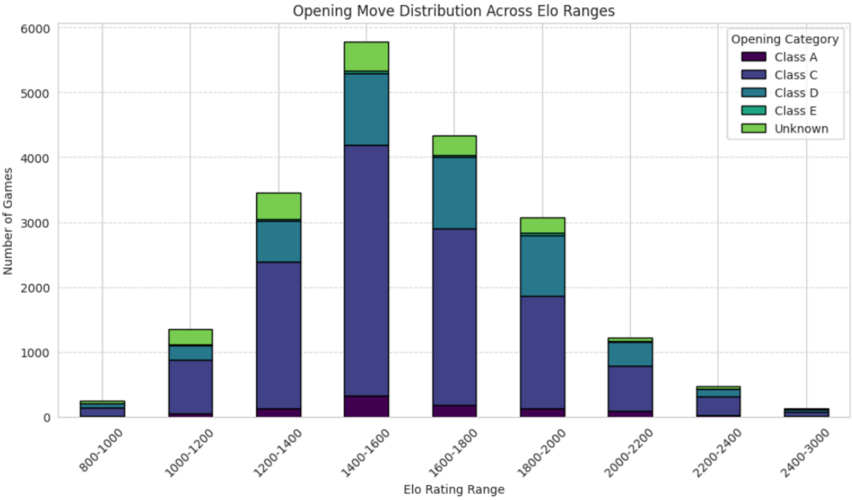
Understanding Data : Chess Piece Movement Distribution



Piece Movement Distribution

- Boxplot of move counts per piece type (Pawn, Knight, Bishop, Rook, Queen, King).
- Observations:
  - Pawns: High activity.
  - Kings: Low movement (protective role).
- Insight: Piece movement correlates with board strength and tactical advantage.

Understanding Data : Opening Move Distribution



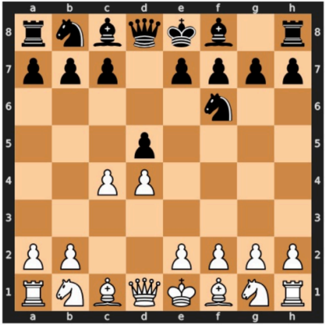
Opening Move Distribution

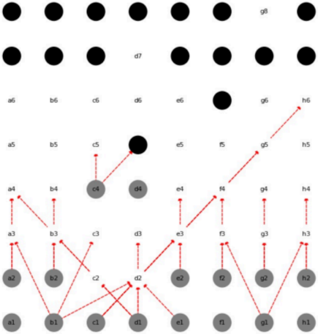
- Stacked bar plot of ECO opening classes (A-E, Unknown).
- A (Flank), B (Semi-Open), C (Open, French), D (Closed/Semi-Closed), E (Indian Defenses).
- **Network Impact**: Openings shape early topology (centralization, mobilisation, expansion).
- **Metrics Affected**: Clustering coefficient, centrality, path diversity.
- **Insight**: Openings influence early graph topology, such as : centrality, connectivity ,Clustering etc.

Feature Extraction

- Graph-based representation to capture spatial and relational board structure.
- Two approaches:
  - Structural/mobility features (e.g., node degree, clustering).
  - Domain knowledge (e.g., material balance, piece effectiveness).
- Two curated feature sets:
  - Structural knowledge only (e.g., node degree, clustering).
  - Combined structural and domain knowledge (e.g., material balance, piece effectiveness).
- Example: Knights excel in clustered structures, bishops in open diagonals.

Mobility Network



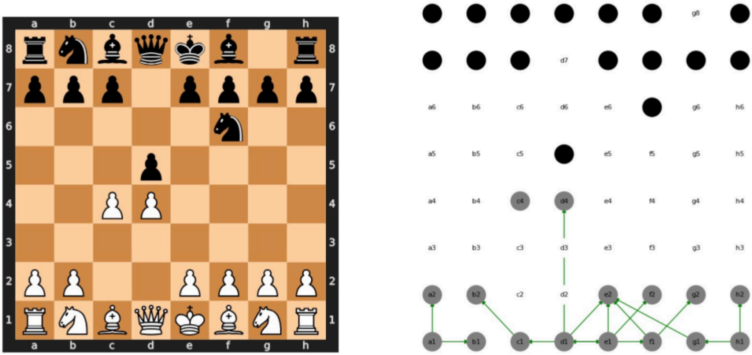


Board State and its corresponding Mobility Network

Mobility Network

- Nodes: Squares (occupied or empty).
- Edges: Legal moves or captures from occupied squares.
- Visualization: Shows potential moves/captures for each piece.

Support Network

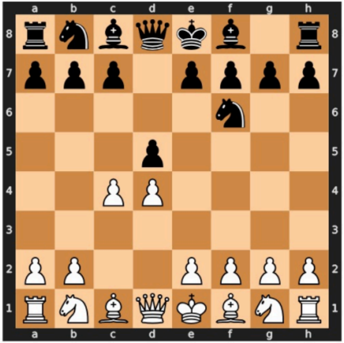


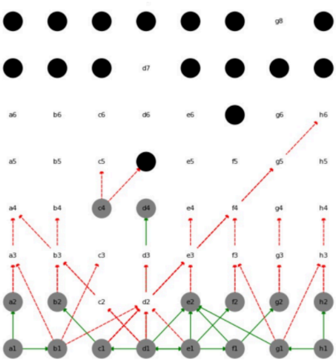
Board State and its corresponding Support Network

Support Network

- Nodes: Pieces.
- Edges: Attacks (opposing colors) or defenses (same color).
- Visualization: Highlights piece coordination and threats.

Position Network





Board State and its corresponding Position Network

Position Network

- Union of Mobility and Support Networks.
- Nodes: All squares.
- Edges: Attacks, defenses, potential moves.
- Captures comprehensive board dynamics.

Model Development

- Framework: Scikit-learn **Random Forest** (150 trees, *max_features* = sqrt).
- Features: Graph-based (structural, mobility) + domain knowledge (material balance).
- Labels: Strong, neutral, weak (based on Stockfish thresholds).
- Explainability: **SHAP** for feature importance.

Model Performance

- Accuracy: 74%.
- F1-scores:
  - Strong: 0.83
  - Neutral: 0.73
  - Weak: 0.68
- Strong positions identified most accurately; neutral/weak differentiation needs improvement.
- Best performance using combined structural and domain features

Model Performance

[20]:

```
print("Random Forest Classification Report:")
print(classification_report(y_test, y_pred))
print("Random Forest Accuracy:", accuracy_score(y_test, y_pred))
```

```
Random Forest Classification Report:
              precision    recall  f1-score   support

     neutral       0.76      0.70      0.73       350
      strong       0.81      0.85      0.83       350
        weak       0.67      0.68      0.68       350

    accuracy                           0.74      1050
   macro avg       0.74      0.74      0.74      1050
weighted avg       0.74      0.74      0.74      1050

Random Forest Accuracy: 0.7447619047619047
```

Model Performance on Combined Feature Set.

Feature Importance

Random Forest SHAP Feature Importances (sampled):

	feature	shap_importance
12	player_pawn_pst	0.025716
48	support_nodes	0.024785
18	opponent_pawn_pst	0.019950
49	support_edges	0.019449
0	player_pawns	0.015942
6	opponent_pawns	0.012155
32	opponent_queen_mobility	0.011313
1	player_knights	0.009610
8	opponent_bishops	0.009450
10	opponent_queens	0.009273
31	opponent_rook_mobility	0.009149
26	player_rook_mobility	0.008225
46	support_avg_degree	0.007498
7	opponent_knights	0.007352
27	player_queen_mobility	0.007211
29	opponent_knight_mobility	0.007130
9	opponent_rooks	0.006701
2	player_bishops	0.006562
22	opponent_queen_pst	0.006463
50	support_centrality	0.006022
19	opponent_knight_pst	0.005857
47	support_clustering	0.005807
45	opponent_pawn_chains	0.005806
25	player_bishop_mobility	0.005763
14	player_bishop_pst	0.005666
41	player_pawn_chains	0.005393
51	mobility_avg_degree	0.005156
20	opponent_bishop_pst	0.005114
55	mobility_centrality	0.005059

Relative Feature Importance Using SHAP

Ablation Study

- Structural + mobility features (only graph-based) ≈ 50% accuracy
- Adding material balance: ≈ 74% accuracy

Domain knowledge boosts performance.

Conclusion

- **Key Findings**: Graph-based model (mobility, support, position graphs) achieves 50% accuracy; combined features (structural + domain) outperform structural-only (74%).
- **Insights**: SHAP highlights pawn structure, support nodes, mobility as key; reflects classical chess principles (pawn play, piece coordination).
- **Player Benefits**: Quantifies positional factors and offers data-driven guidance to improve play (e.g., pawn structure, connectivity).
- **Limitations**: Reliance on Stockfish scores may undervalue long-term strategies, especially in complex middlegames due their inherent in board strength,
- **Future Work**: Integrate human annotations, multi-engine scores; explore neural graph models, real-world validation.

Thank You