Dataset Description: Chess Positions for DQN Training & Evaluation

Devin Williams

Overview
This project will use two main datasets:

1. **Self-play experience** generated by my training agent (primary reinforcement learning data).
2. **Public human game records (PGN files)** used for supervised pretraining and evaluation.

Together, these provide both a continuous stream of state–action–reward tuples for learning and grounded examples of expert play to stabilize early training and benchmark performance.

Sources & Licensing

- **Self-Play:** Generated by my agent using *python-chess*; fully under my control.
- **Public Games:** Downloaded from open repositories such as Lichess, which provides millions of standard chess games freely available for research.
- **Optional Engine Data:** A small subset of positions may be labeled using Stockfish for analysis, not direct training.

Data Contents
Each chess state will be represented as an **8×8×12 tensor**, with one plane per piece type and color. Each record includes the FEN state, legal-move mask, chosen move, reward, and next state. For human games, SAN moves will be converted to UCI format, and results standardized (win = 1, draw = 0, loss = –1).

Acquisition & Processing
Public PGNs will be parsed with *python-chess*, filtered to include only standard classical or rapid games. Each game will be verified for legality, converted to FEN, and stored in a structured format (JSONL or Parquet). Self-play data will be generated automatically during training and periodically saved for analysis.

Cleaning & Filtering

- Remove incomplete or corrupted PGN files.
- Exclude non-standard variants (e.g., Chess960, bullet).
- Normalize results to standard outcomes.
- Deduplicate identical positions or games.
- Convert all positions to a consistent tensor representation with legal moves and results.
- Optionally, verify a subset with Stockfish for quality assurance.

Splits & Usage
Human PGNs will be divided into **train/validation/test** sets (80/10/10) by game. Self-play data will be used continuously for training, with periodic evaluation checkpoints against fixed baselines.

This dataset provides a robust foundation for training and evaluating a chess-playing DQN, balancing realistic expert data with self-play exploration to promote effective learning and strategy emergence.