Project Title: "Chess Player Identification and Style-Based Move Prediction Across Time Controls"

Nakul Goyal
Student number: 22072855
Supervisor: Professor Nicolas Gold

1. Aims and Objectives:

Aims:
- To develop a system that can identify chess players based on their playing style
- To understand how playing style varies across different time controls
- To find similar top-level players based on style of any given player
- To create a chess engine that can adapt its play based on player style

Objectives:
1. Research and analyze existing machine learning approaches to chess player identification/chess stylometry
2. Filter, collect and preprocess game data from the lichess.org open database into usable format and split into blitz, rapid and classical time controls and further into training, reference and test sets.
3. Design and implement a model architecture to extract player style embeddings from chess games
4. Train separate models for different time controls (blitz, rapid, classical)
5. Evaluate identification accuracy, given one game of a player by comparing with reference games
6. Create a system for finding stylistically similar players
7. Develop a modified chess engine from an existing engine that incorporates style information
8. Evaluate and compare performances across different time controls and player skill levels

Key questions to answer during evaluation and analysis:
How does identification accuracy change:
- Across different numbers of games
- Across time controls
- For seen vs unseen players (a seen player is a player whose games were part of training)

Style analysis:
- Is play style for an individual consistent across time controls/do predictions get harder or easier? (the expected answer is that it gets harder for classical since the players rely lesser on intuition and more on raw-calculation)
- How does style change across time controls?
- Clustering of similar styles together + can we tell closest top player to a non-top player's style?

Engine Performance:
- Compare moves suggested by style engine vs base engine
- Can it predict moves better than a chess engine that tries to predict solely based on elo strength (this will be tested against Maia Chess)?

2. Expected Outcomes/Deliverables:

1. A trained model capable of:
   - Identifying players from their games
   - Generating style embeddings
   - Finding similar players (will require some tweaks to primary model)
2. The analysis of style consistency and accuracy of identification task across time controls and corresponding analysis code

3. A modified chess engine incorporating style information
4. Results and analysis from evaluation of engine move prediction and corresponding code
5. Design specification of my model and a guide to using the model
6. Project report with detailed methodology and findings

3. Work Plan Timeline:
Term 1:
Weeks 1-5:
- Literature review
- Understanding existing approaches
- Setting up development environment
- Initial data collection and processing

Reading Week:
- Submit Project Plan
- Complete ethics assessment (likely minimal as using public chess data)

Weeks 6-10:
- Model architecture design
- Initial implementation
- Training data and pipeline setup

Term 2:
Weeks 1-2:
- Complete base model implementation (likely needs 2-4 iterations)
- rain time control specific models
- Submit Interim Report

Weeks 3-5:
- Develop similarity analysis system
- Start base engine modifications

Reading Week:
- Begin final identification evaluation experiments
- Begin report writing

Weeks 6-10:
- Complete engine implementation (likely needs 2-4 iterations)
- Run full engine evaluation experiments
- Create video preview

Term 3:
Weeks 1-2:
- Complete all experiments
- Finalize report writing
- Submit final report

4. Ethics Review:
This project will use publicly available chess game data from Lichess, which is released under a Creative Commons license. No personal user data beyond game moves and timestamps and potentially usernames will be used. The project does not involve:
- Human participants

- Sensitive personal information
- Potential for harm

5. Resources Required:
- Access to Lichess API for game data
- Python development environment
- Neural network libraries (PyTorch + Numpy preferred over Tensorflow)
- Chess libraries (python-chess)
- Base chess engine (likely Stockfish)
- Computing resources for training models

6. Potential Risks:
1. Data collection scale might need adjustment based on processing capacity and results
2. Model training time might be longer than expected
3. Engine modification complexity might require scope adjustment
4. Performance across time controls might vary significantly

7. Risk Mitigation:
1. Start with smaller dataset, scale up if feasible and/or required
2. Plan for multiple training iterations with increasing complexity
3. Have fallback options for engines needed for engine modifications
4. In case of this, continue to only use time controls with good performance and mention the differences in the report