Rapport de Mi-Projet: Assistant de Jeu d'Échecs Basé sur un LLM

ChessLLM:
Membres: Briac Six, Jason Perez, Samy Hadj-said

November 15, 2024

Contents

1 Project Description

1.1 Objective

The aim of this project is to train an LLM on PGN-formatted chess games, enabling it to analyze positions and suggest optimal moves. By treating chess moves as a sequence prediction problem, this project bridges the domains of structured logic and natural language processing, exploring how LLMs can be adapted to follow strict game rules while exhibiting strategic reasoning.

1.2 Context and Relevance

Chess, being a game of both strategy and computation, has long served as a benchmark for artificial intelligence. Unlike traditional chess engines, which rely on brute-force search and handcrafted evaluation functions, our project focuses on leveraging the inherent pattern-recognition capabilities of LLMs. These models, trained on massive amounts of data, excel at identifying context and generating outputs that mirror human-like reasoning.

1.3 Innovation and Applications

One key innovation of this project lies in its approach to training. By framing chess as a sequential decision-making task, the LLM can learn patterns in move progression, capturing not only the rules of chess but also the strategies embedded within millions of historical games. This approach is particularly useful in emulating human intuition and decision-making, making it ideal for applications like chess education, game analysis, and simulating diverse opponents.

1.4 Challenges

Adapting LLMs to the rigid rules of chess presents unique challenges. Unlike free-form text generation, chess requires strict adherence to legal move sets and strategic coherence. The model must learn to avoid illegal moves, such as placing the king in check, while simultaneously proposing moves that align with long-term strategic goals.

Key Challenge: Ensuring that the LLM accurately models chess rules and strategy, especially in positions requiring deep tactical calculations.

2 Project Background

2.1 Existing Research

Our project builds on innovative work showing how language models, traditionally designed for text tasks, can adapt to strategy-based games like chess. One notable example is Nicholas Carlini's exploration of GPT-3.5-turbo-instruct, which could play human-like chess despite lacking chess-specific training.

2.2 Challenges in Existing Approaches

Carlini observed that GPT-3.5 could track board states by sequentially processing each move, implying "world modeling" within the model's internal states. However, since GPT-3.5 is optimized for language rather than chess, it occasionally favored plausible moves over optimal ones, highlighting its limitations in competitive gameplay.

2.3 How Our Project Differs

Our project fine-tunes or trains a model on curated PGN data, customizing it for consistent, high-quality move recommendations. This approach builds on Carlini's insights and Google's findings by optimizing the fine-tuning process for chess and emphasizing real-time applications through an interactive web-based chessboard interface.

3 Project Steps

3.1 Data Collection and Preprocessing

- Gather a large dataset of PGN-formatted games from online chess repositories (e.g., Lichess).
- Clean the data and ensure consistent formatting.
- Extract input-output sequences for training.

3.2 Model Selection and Training

- Choose a pre-trained GPT-like model as the base architecture or train a model from scratch.
- Fine-tune the pre-trained model on extracted game sequences to generate accurate and legal moves.

3.3 Evaluation and Benchmarking

- Develop evaluation metrics to measure adherence to chess rules and move quality.
- Compare model performance against established chess engines.

3.4 Interface Development

- Build an interactive chessboard interface using web technologies.
- Integrate the model to enable real-time move suggestions.

4 First Results

4.1 Preprocessing and Training

Initial experiments involved preprocessing 5000 chess games from PGN files. We removed unnecessary data such as move numbers and added a prefix before each game to enable the model to generate the first move.

4.2 Model Performance

For the fine-tuning process, we selected the GPT-2 small model. Although we have a database containing several million chess games, we trained the model on a subset of 5000 games. Training this subset takes approximately 4 hours per epoch. After only 20 minutes of training, the model achieved a promising loss of 0.5.

4.3 Challenges and Future Work

- Extend training to larger datasets to improve loss and prediction accuracy.
- Address limitations in strategic reasoning and adherence to rules.

Output Example:

Input Sequence: "1. e4 e5 2. Nf3 Nc6 3. Bb5"
Predicted Move: "a6"

5 Additional Content

5.1 Github Repository

Github Repo: `https://github.com/samy-hadj/chessLLM`

5.2 Images Placeholder

Figure 1: Example of an interactive chessboard.