Title Page

# Abstract

Brief summary of the project, its objectives, methodologies, and key findings.

# Introduction

# Literature Review

Overview of Existing Chess Analysis Tools

Importance of Chess Opening Strategy

Previous Work on Chess Game Analysis

# Methodology

Description of the Data Acquisition (Chess.com and Lichess.org APIs)

Explanation of the Opening Analysis Process

Details on the Use of Stockfish for Position Validation

Comparative Analysis of Parallel Processing and Itemization Strategies

# System Architecture

Overview of the System Design

Data Flow Diagram

Interaction Between Components

# Implementation

## Prototype

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

Automatisch generierte Beschreibung

Figure 1: Prototype Flowchart

The flowchart outlines the initial solution to our problem. Through an interface, the Lichess API is accessed, allowing users to select criteria for the games they wish to analyze. The specified games are retrieved via the API and saved as PNG files.

These PNG files are then loaded into the PreProcessing class and modified. Only the content relevant to the opening analysis is retained, with games truncated to a 15-move sequence and the opening names cleaned up. The resulting dataframe is then saved as a CSV for simplicity.

This CSV is subsequently loaded into the Analysis class, where Stockfish is configured with specific strength and depth settings. The higher these parameters, the longer the chess engine will take to analyze each position. In this initial approach, both strength and depth are set to 15, resulting in a reasonably feasible quantitative analysis. However, this configuration takes precisely 2 hours, 1 minute, and 40 seconds to analyze 1,000 games completely. The positions extracted from the original CSV are stored in a new dataframe called game\_analysis. Some aggregations are performed on this file to augment the original CSV with statistics like "game\_accuracy," which represents the average accuracy of all moves in a given game from the player's perspective.

Ein Bild, das Text, Reihe, Screenshot, Diagramm enthält.

Automatisch generierte BeschreibungBoth dataframes are then output as a JSON to the Markdown visualizer, which serves as an important initial tool for visualizing the data. This provides a clear indication of the complexity involved in representing such a wide variety of openings in a heuristic manner.

Ein Bild, das Text, Schwarzweiß, monochrom, Origami enthält.

Automatisch generierte Beschreibung

## Indexing

This optimization step is supposed to increase the analaysis speed of the games. The chess games always have a FEN annotation, which describes in a string the current position of the chess game. Since the game has less differnt kind of possible positions in the early stage and many postions repeat through interpositions, the idea is to evaluate those positions only once and then save the position in an indexed database table. So that if the position reoccurs, we do not have to run stockfish over the postion, but can only take the saved value from the indexed table using the FEN postion as index.

When approaching this optimization step, there are evaluations which will be done, before the expensive and long process of filling the database can happen. First a mathematical estimation of how many postions must be saved and an empiric one, using big lichess Files containing hundred millions of games.

#### Mathematical Estimation

For the mathematical estimation, we can use Shannon’s calculation, which is the most referenced estimation on total possible unique chess games. The average branching factor for each played move is in average 35, which explains why the grwoth of possible postions grows exponetionally.

|  |  |
| --- | --- |
| **Number of plies (half-moves)** | **Number of possible positions** |
| 1 | 20 |
| 2 | 400 |
| 3 | 8,902 |
| 4 | 197,281 |
| 5 | 4,865,609 |
| 6 | 119,060,324 |
| 7 | 3,195,901,860 |
| 8 | 84,998,978,956 |
| 9 | 2,439,530,234,167 |
| 10 | 69,352,859,712,417 |
| 11 | 2,097,651,003,696,806 |
| 12 | 62,854,969,236,701,747 |
| 13 | 1,981,066,775,000,396,239 |
| 14 | 61,885,021,521,585,529,237 |
| 15 | 2,015,099,950,053,364,471,960 |

<https://en.wikipedia.org/wiki/Shannon_number>

#### Empric Estimation

The numbers displayed in the table above are quite initimidating, but do neither represent game theory, nor human behaviour. Firstly, there are objective better chess openings which are played much more than known bad openings. Secondly most player tend to play in a similar style and not swap to much their opening repertoire. So even in case that a player does play an uncommon opening, which is not in our index database yet, after adding those games most likely not too many new opening should follow after, making the indexed table useful again.

### Tested indexed table

## Stockfish Evaluations

### Parallel Processing

## Api optimization

**Problem:**

Technical Details of the Implementation

Description of the Algorithm and Codebase

Infrastructure and Technologies Used

# Results and Discussion

Analysis of the Findings

Comparison of Different Methodologies (Parallel Processing vs. Itemization)

Interpretation of the Results

Visualization of the Data on the Dashboard

# Evaluation

Evaluation of the System Performance

Analysis Speed and Accuracy

User Feedback (if applicable)

# Conclusion

Summary of the Findings

Implications of the Study

Limitations and Future Work