Project - Positional analysis of chess games

Jake Moss - s46409665

May 5, 2021

Contents

1 Introduction

1.1 Project aims

The goal of this project was to provide a new and unique visualisation of chess games. To accomplish this two main plots where developed,

- Heat map of where pieces were captured
- Histogram of when pieces were captured

The purpose of this project is to show the positional differences between different levels of play and in different time periods. This is something not often visualised within the chess community.

1.2 Obtaining data

1.2.1 Sources

Chess games are found freely on the internet from many archives such as PGN Mentor and FICS Games Database. From here I am able to download millions of games at mass. However, when considering large time periods most preserved games are of tournaments and high skill players, this data does not show the trends of average level games. To help mitigate this the `FISC` database was used to analyse a wide range of skill level games from a single year. This tool was also build with the capability to analyse a single players games given their username from Lichess and Chess.com. These games can be downloaded using their respective public API's.

To download games from Chess.com the bash command below can be used.

```
for g in $(curl -Ls https://api.chess.com/pub/player/$PLAYERNAME/games/archives | jq -rc
↪  ".archives[]") ; do curl -Ls "$g" | jq -rc ".games[].pgn" ; done >> games.pgn
```

Lichess easily allows for downloading of an entire players archive at once with a simple `curl`.

```
curl https://lichess.org/games/export/$PLAYERNAME > games.pgn
```

`DaenaliaEvandruile` has given their permission for their games to be featured with this report to demonstrate this functionality.

1.2.2 PGN Format

The most common format used to store chess games is Portable Game Notation. It is a human-readable plain text notation which stores individual moves rather than the board states. While this is considerably more efficient for both storage and transcribing it makes extracting data that isn't explicitly given difficult and costly.

The PGN standard defines a required seven tag roster and allows for optional tags such as `WhiteElo` and `BlackElo`. Each move is stored in Standard Algebraic Notation, which describes the change in board state such as captures, checks, and promotions.

SAN comes in a varsity of formats short, long, or minimal. Each variant encodes moves with different levels of detail. Unfortunately no single variant is used in all PGN databases, although it is rare for the format to change from within a database.

A standard game might look something like this

```
[Event "WCh 2018"]
[Site "London ENG"]
[Date "2018.11.09"]
[White "Caruana, Fabiano"]
[Black "Carlsen, Magnus"]
[Result "1/2-1/2"]
[WhiteTitle "GM"]
[BlackTitle "GM"]
[WhiteElo "2832"]
[BlackElo "2835"]
[WhiteFideId "2020009"]
[BlackFideId "1503014"]
[EventDate "2018.11.09"]

1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 ... 115. Ra8 1/2-1/2
```

Game tags must be in `[]`. Often there are more than the standard requires. These tags are referred to as the headers. The moves themselves are listed in SAN with the result at the end.

2 Data Processing

2.1 Tools used

This project made use of the below tools,

- python-chess Used to handle the complex move validation and PGN parsing required to step though games.
- matplotlib Standard plotting library.
- seaborn A high level plotting library built on `matplotlib`.
- numpy Standard scientific computing library. Within this project it is used for its `pandas` optimisations, and reshape function.
- pandas Data frames provide an efficient way to store and conditionally select games based on metadata.
- cPython Profiler Function call profiler.
- snakeViz In browser `cPython profiler` visualiser.

2.2 Meta Data Extraction

To extract the metadata from a game the PGN file is read using standard python methods and the game object is initialised using the `aggergate` function from the `aggergation.py` module. From the game object the headers are extracted into a dictionary and a data frame is created consisting of the headers as column titles and rows as individual games.

	SAN string	Black	White	Result	BlackElo	WhiteElo	Date	...
1	`string`	`string`	`string`	`string`	`int`	`int`	`DateTime`	...
...								

2.3 Extracting Implicit Data

To extract the "implicit data", such as piece captures, each game must be played out in its entirety with each move analysed as the format games are encoded and does not explicitly provided this information.

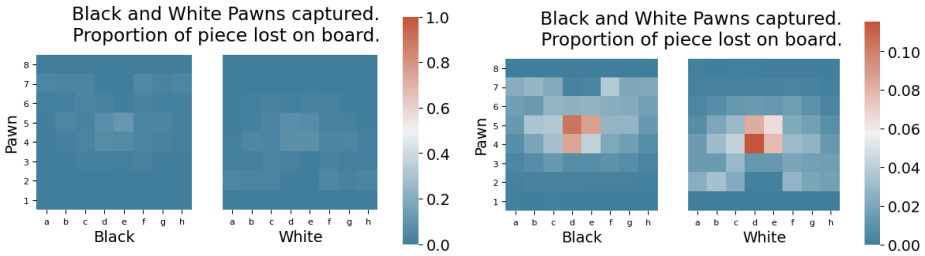
This data is then appended to the metadata data frame in another column. This allows for each conditioning of the data frame.

3 Data Analysis

3.1 Local and Global Normalisation

Within the heat map grid plots, each map shares the colour bar on the right. To accomplish this each square with each map is normalised to the sum of the local grid so the map represents proportion instead of frequency. From this we now know the maximum value is `1` and the colour bar can be calibrated to `[0,1]`. However, as the proportion of lost pieces rarely gets close to `0.5` let alone `1`, the colours become hard to differentiate.

To remedy this I set the maximum of all local maximums to the max of the colour bar. This made it so that each plot is proportional to itself and the colouring is consistent between plots.



A similar strategy was employed to ensure the histograms and kernel density plots shared the same axis.

3.2 Piece comparability

As there are 8 pawns, but only 1 queen it is not fair to directly compare proportion statistics. This was not accounted for on purpose as doing so could muddy the information and may introduce over correction. Instead it is assumed that the viewer understands the context of the pieces.

3.3 Colour schemes

Although all plots presented here feature the same colour scheme for consistency, they are interchangeable with ease through the use of `matplotlib colour palettes`. Some example can be found here. This facilitates for adjustment due to the display medium and user accommodations. A grey scale can be used for printing, or appropriate pallets for colour blindness.

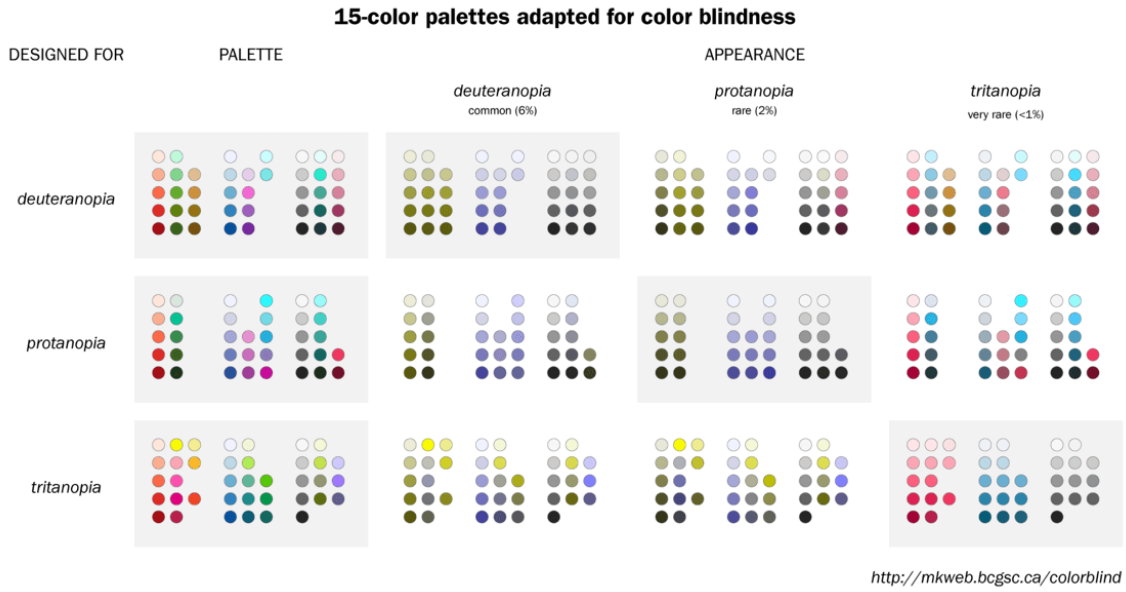
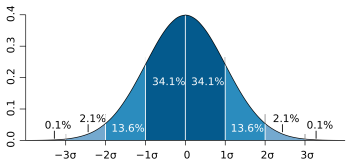


Figure 1: Image source

All plot commands support supplying a custom `cmap` as an optional argument.

4 Results

The skill level of players was assumed to be normally distributed. All ELO grid plots have their `x-axis` binned by percentiles from the normal distribution. Sigma or σ here denotes one standard deviation [2]. Date grid plots are binned by quartiles.



4.1 Plot analysis

4.1.1 ELO grid plots

This first visualisation shows positions where pieces were lost by a particular colour plotted against ELO ranking of the player. Each combination of ELO and piece is represented by an `8x8` heat map of the board from Whites perspective. The decision to make all plots from Whites POV was made in an attempt to improve comparability and consistency between plots.

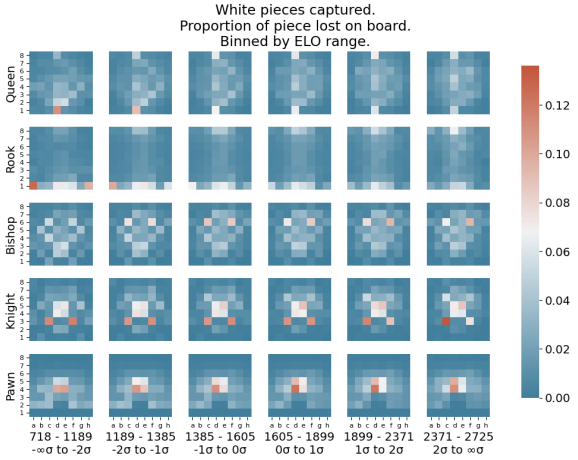
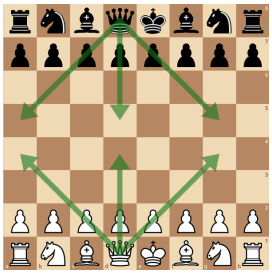


Figure 2: All captured White pieces binned by ELO

The unique pattern for each piece shows the most common lines played by White.

Queens typically move through the `C,D,E` pawns which are the easiest lines out of the starting position.



As ELO increases all pieces trend towards the centre as it is a highly advantageous position. The `A1` and `H1` Rooks (`A8` and `H8` for Black) loses almost disappear in higher ELO's, this is likely due to higher usage, and the utilisation of castling. The highest ELO players show a significant increase in spread of loses indicating that the Rooks are more active through a wider range of centre files. The decrease in loses on positions that are common from castling does not indicate a decrease in castling as the heat maps only show losses, as higher ELO players are more likely to play better positions, they are less likely to castle into a position where their Rook will be lost.

A similar spread increase can be seen in the Queens. Higher ELO players tend to lose their Queens in a much more even manner. This may be a symptom of sacrificing which is common in high level play.

Bishops, Knights, and Pawns do not display a pattern change between ELOs suggesting most players know common and effective positioning. These position are common knowledge and often taught. Although there is negligible change in the patterns, the frequency at which they are lost becomes more concentrated. This may indicate that higher ELO player are more likely to keep these pieces near positions where they are strongest.

A similar plot as `Figure 1` is attached in the Appendix which instead shows Black pieces. There is no note worthy pattern difference between the Black and White plot. All treads are present but with 180° rotation.

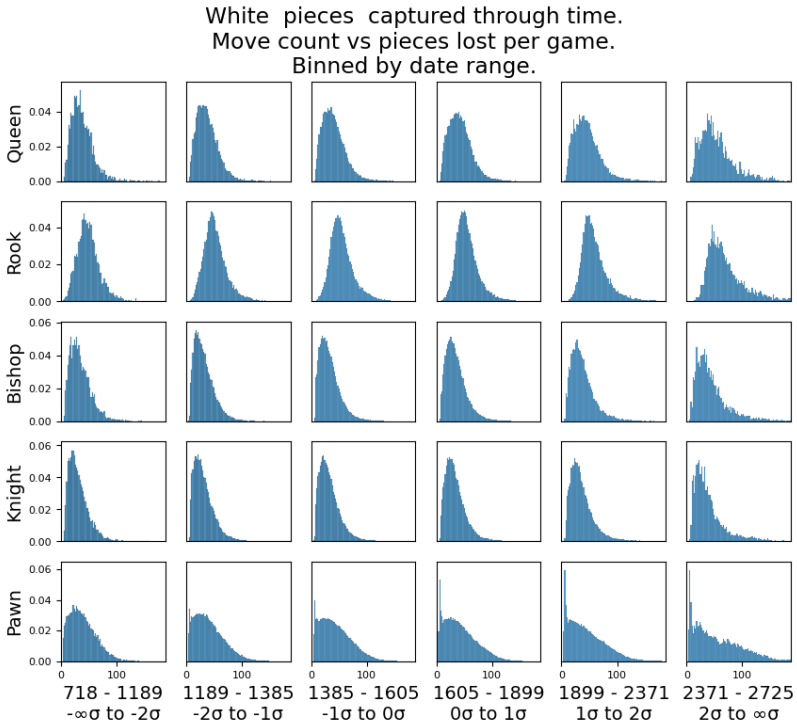


Figure 3: When White pieces where captured binned by ELO

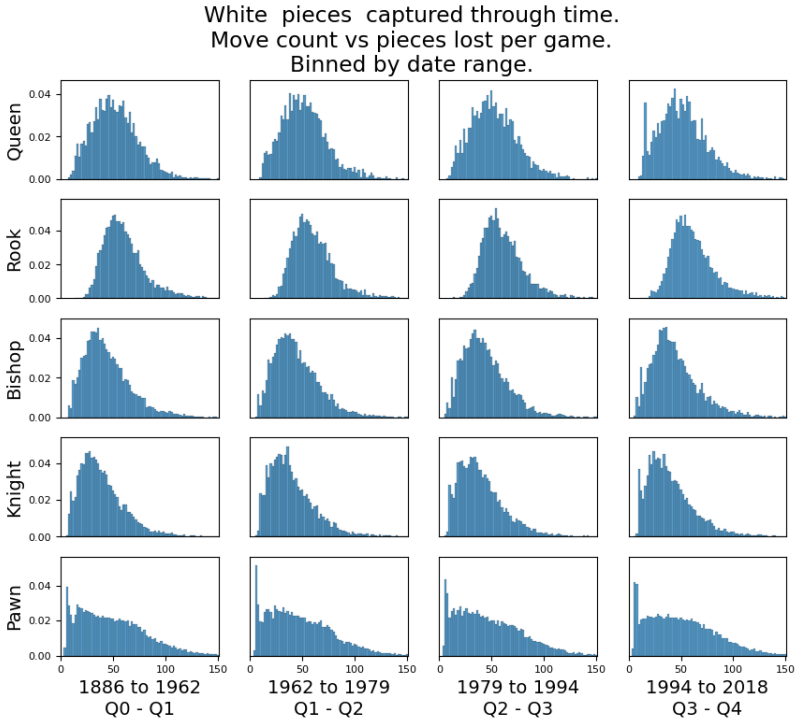
Here the number of captured pieces is divided by the number of games in the respective bin. This is to account for the different game counts due to the binning of players based on the assumption ELO is normally distributed.

Although there are a lower number of games in the more extreme percentiles (`col 1` and `col 6`) some conclusions can still be drawn. A common trend among must all pieces is the gradual spreading effect of pieces lost per game. This is probably due to higher skill players playing longer games. There is often a large spike in Pawn losses around move numbers `1-6` demonstrating the trading that is common at the beginning of games while competing for control of the centre of the board. This is corroborated by `Figure 1` where the Queen side Pawn is most commonly lost.

Further plots are included in the appendix.

4.1.2 Date grid plots

A different database was used for date binning as the `FISC` database only had games from `2000`, instead a database comprising of `Tournaments`, `Candidates Interzonals`, and `World Championships` games were used. (These games were acquired from PGN mentor.)

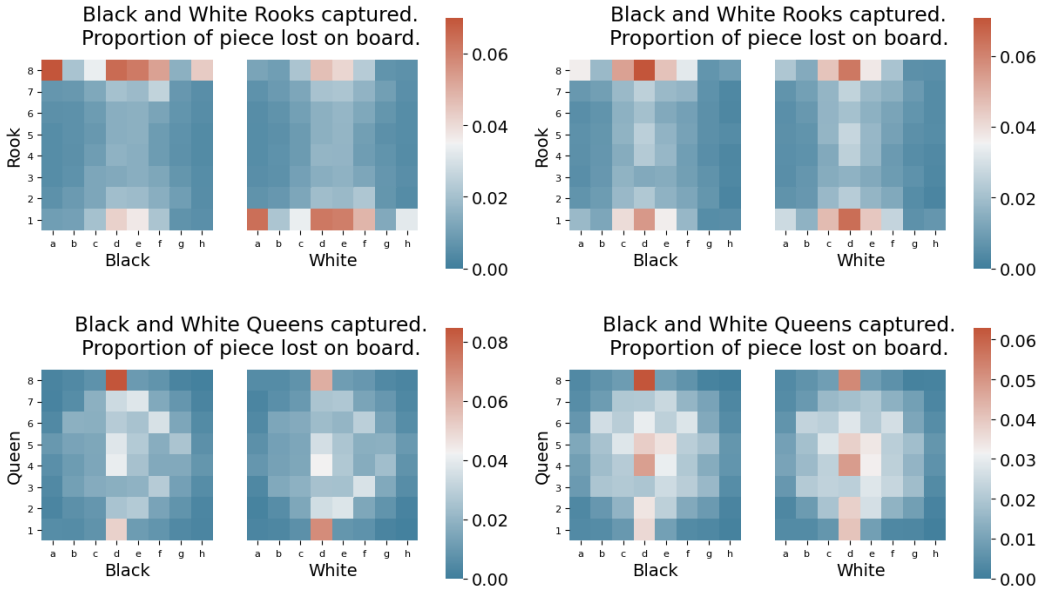


Unfortunately binning by date reveals no interesting patterns or trends. However, it does show that chess has not changed significantly over the centuries.

Further plots are included in the appendix.

4.1.3 Individual piece plots

Left hand side plots are from the `FISC` database. Right hand side from the `Tournements` database.

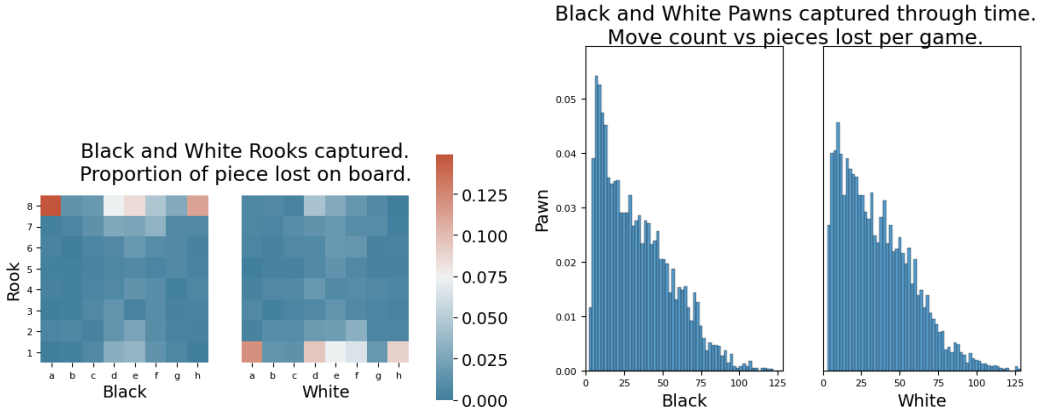


As the `FISC` database contains varying skill level players, it is fair to assume the most prominent trends are those of average skill players, similar to the left side of `Figure 1`. The `Tournement` plots exhibit patterns similar to those of higher ELOs in `Figure 1` as expected.

High level players lose their Queens in the same positions regardless of colour. This could be a consequence of the first move advantage.

4.1.4 Individual player plots

All previously shown plots are also capable of focusing on a single player. Here we have positions where `DaenaliaEvandruile`'s Rooks were lost, and a histogram of when their Pawns were lost. This allows the user to compare their specific play patterns to common trends.



4.2 Conclusion

Overall this project has successfully visualised an uncommon statistic of chess which may provide insight into future trends or an individual players style.

While the `Date` binning did not reveal and trends and patterns initially hoped for it was successful in showing the common positions do not often change over centuries.

All code and resources used are open source and is available at this repository.

4.2.1 Areas for improvement

- Performance and memory requirements prevented me from analysing the databases I hoped to. A custom PGN parse and a rewrite of the main processing module in a language with more control over memory and data structures such as Rust or Haskell might provide the improvements required.
- More positional statistics such as, positions that give and receive checks and checkmates may be interesting to visualise. This is possible with the current implementation requiring only a few more functions due to the features of the `python-chess` library and composability of my program.

4.2.2 Flaws and disappointments

- I found myself struggling to manage the code base due to significant my lack of planning and documentation. In an effort to mitigate this I conducted 3 rewrites, each adding or changing significant portions of code.
  - The first implementation had no storage of meta data or any way to filter game. Games where stored in a list and every plot had to play every game every time some new piece of information of required.
  - Second implementation used `5-6` lists to store meta data and filter the games. However managing 5 or 6 independent lists based on index alone wasn't easy to work with or scalable.
  - Final refactor added data frames and smarter processing. However, this had significant performance costs due to the magnitude of information that had to persist the life time of program and required a large portion of code to be rewritten to be compatible with data frames.

5 Reflection

As the program was designed with modularity and composability in mind it is highly general, and able to process any game database format and group based on any meta data field with a few changes. Binning based on location or opening would only require a few lines to group the games and the rest can remain unchanged.

No pre-processing is required by the user, all plots are non-specific to a single database. Everything is self contained making it easy to work with. Analysing another database requires a single filename change. While writing this program my goal was not to visualise a single data set but rather to write a tool that allowed visualisation to be made with ease. This added a lot of complexity and time requirements, however, I believe this was for the best as it forced me to focus on the process itself rather than manipulate a single database.

While this program can create interesting and new visualisations, as far as I can see there is little strategical use for these plots. Single player can be created to show changes in piece positioning at current and previous rankings. However, plain positioning is unlikely to get you far in a game, complex theory and strategies play a much more important role.

An alternative project maybe to visualise evaluations of moves over time using an chess engine and table base. This could show differences in move accuracy and how the accuracy of different ELO's have changed.

Due to the complexity, design approach, and extent of this project, I believe it is deserving of a 7.

References

[1] FIDE. Laws of chess, 2008.

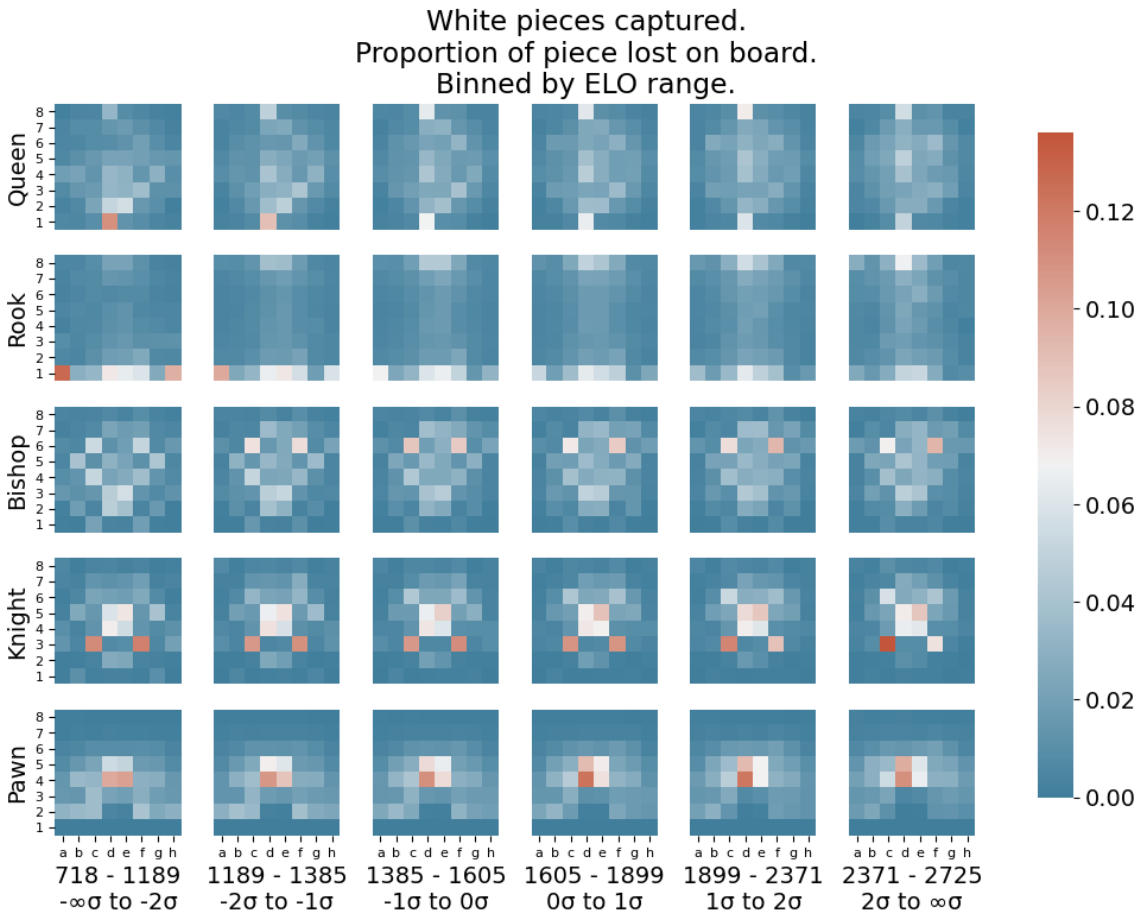
[2] M. W. Toews. Percentile - wikipedia, 2005.

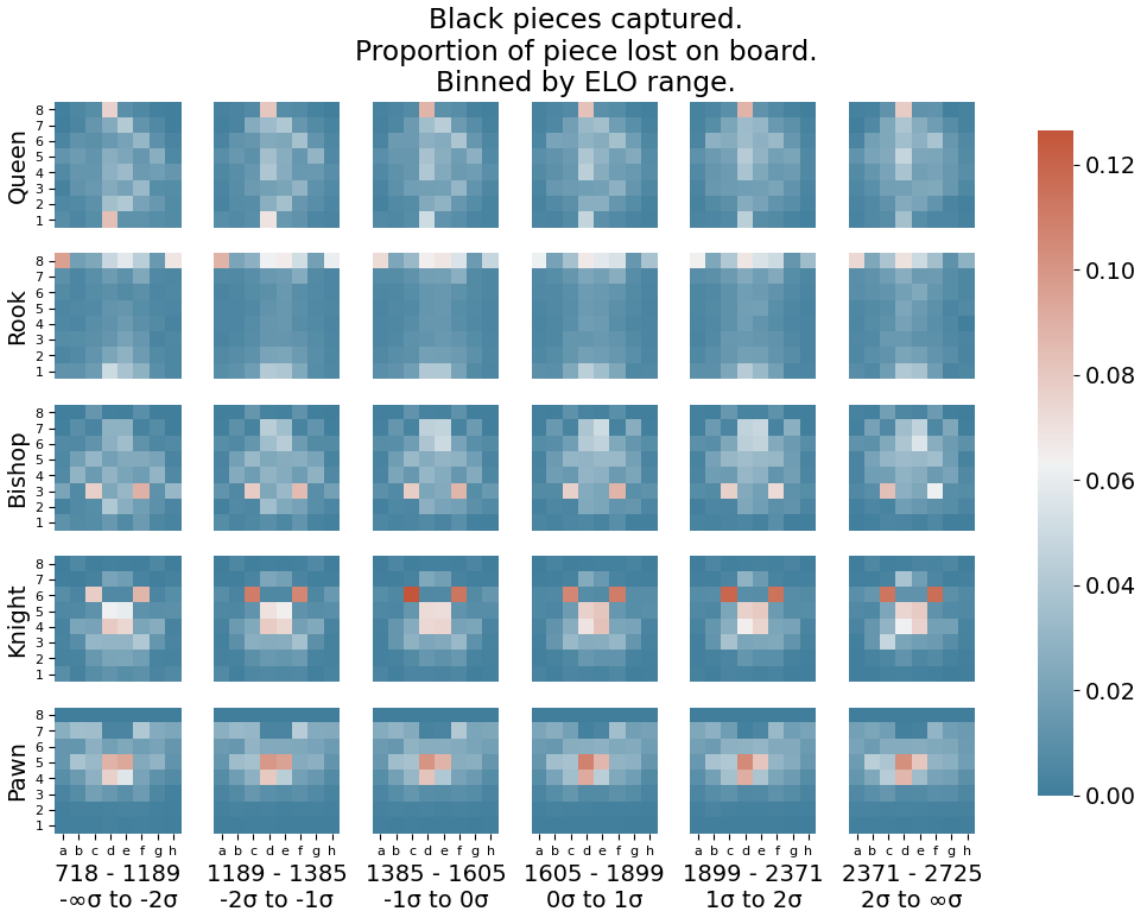
[3] Unknown. Soviet calendar - wikipedia, 2021.

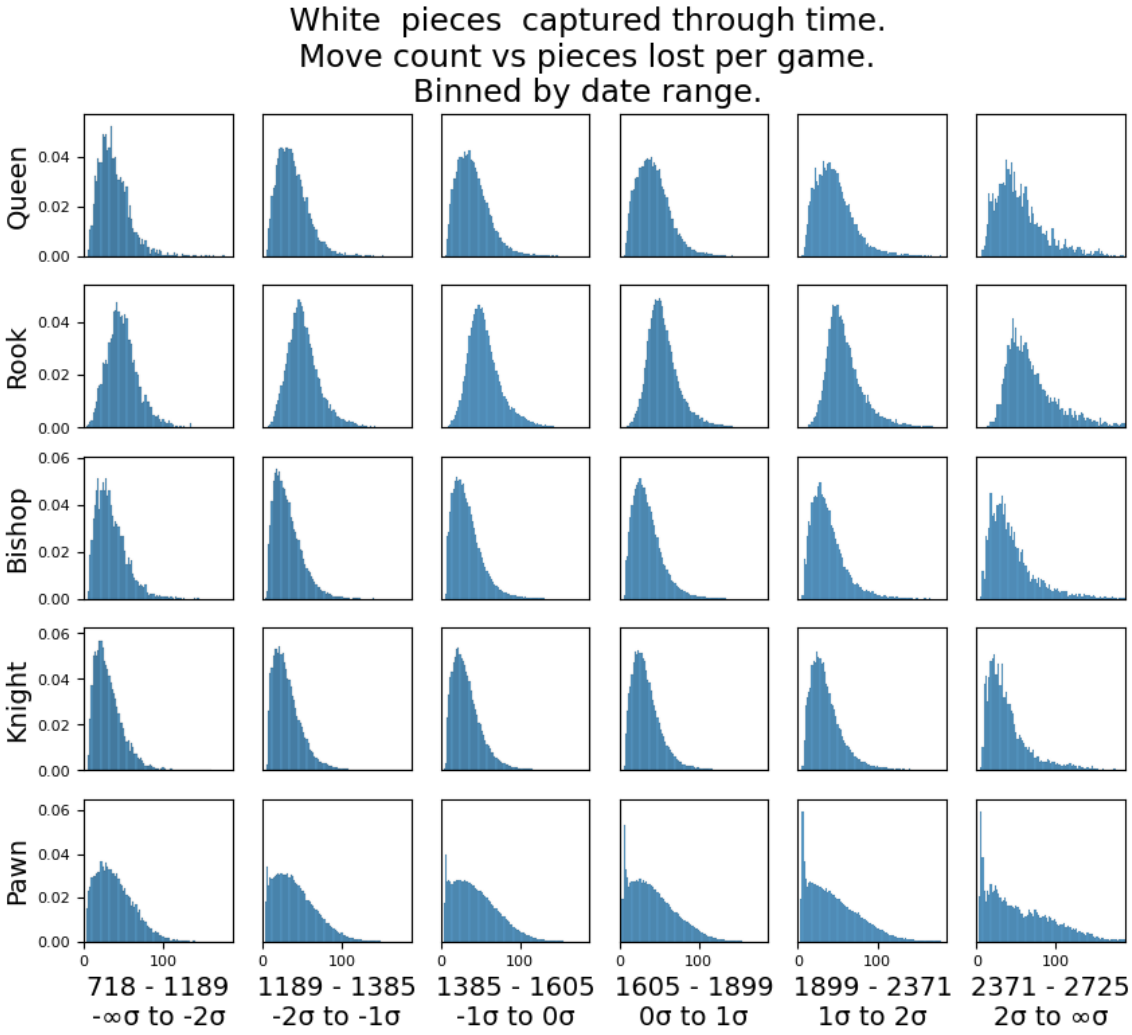
6 Appendix

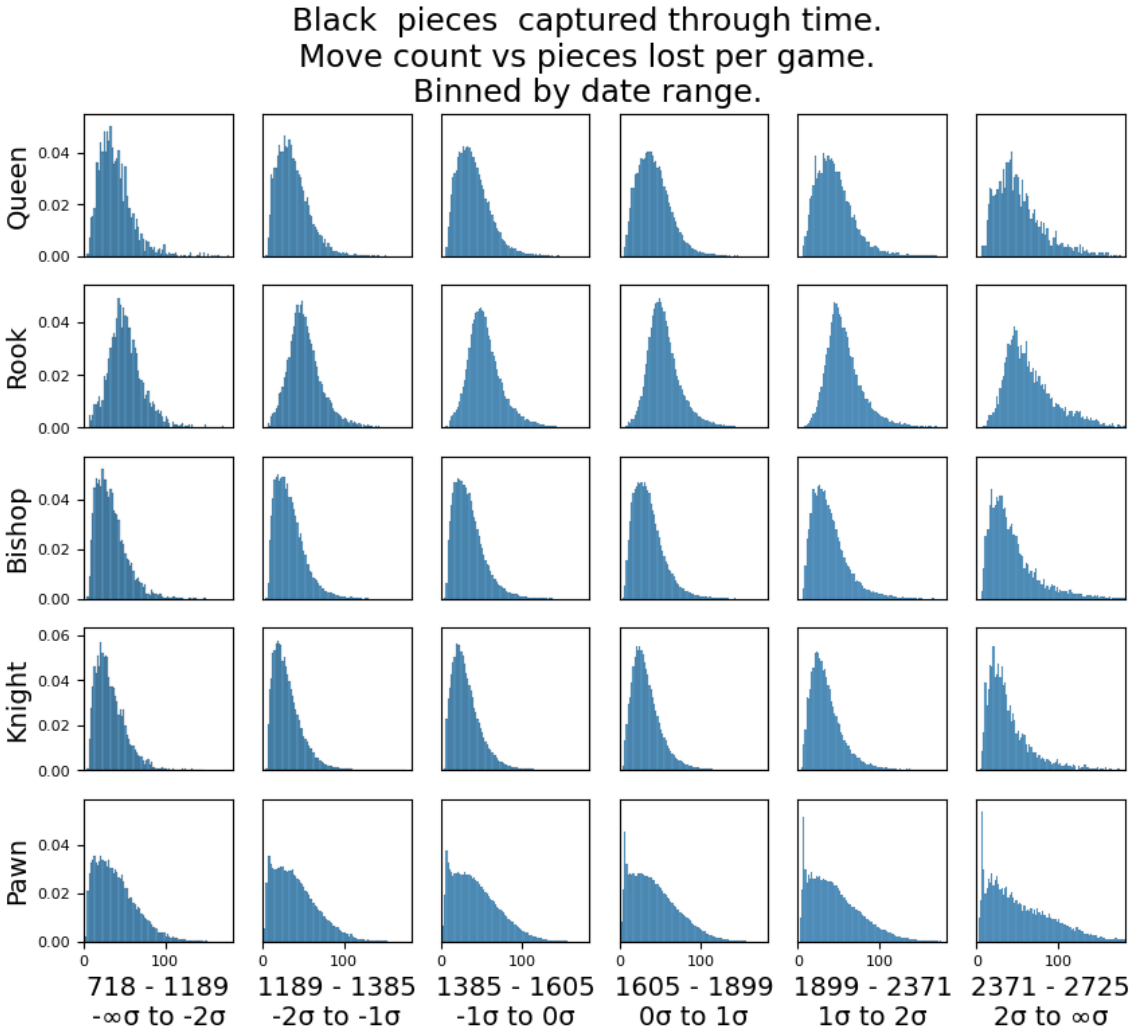
This repository is the best place to view these images, under the `Images/` folder as well as the source code for this project. `main.py` contains all plots shown here.

6.1 ELO Grids (FISC)

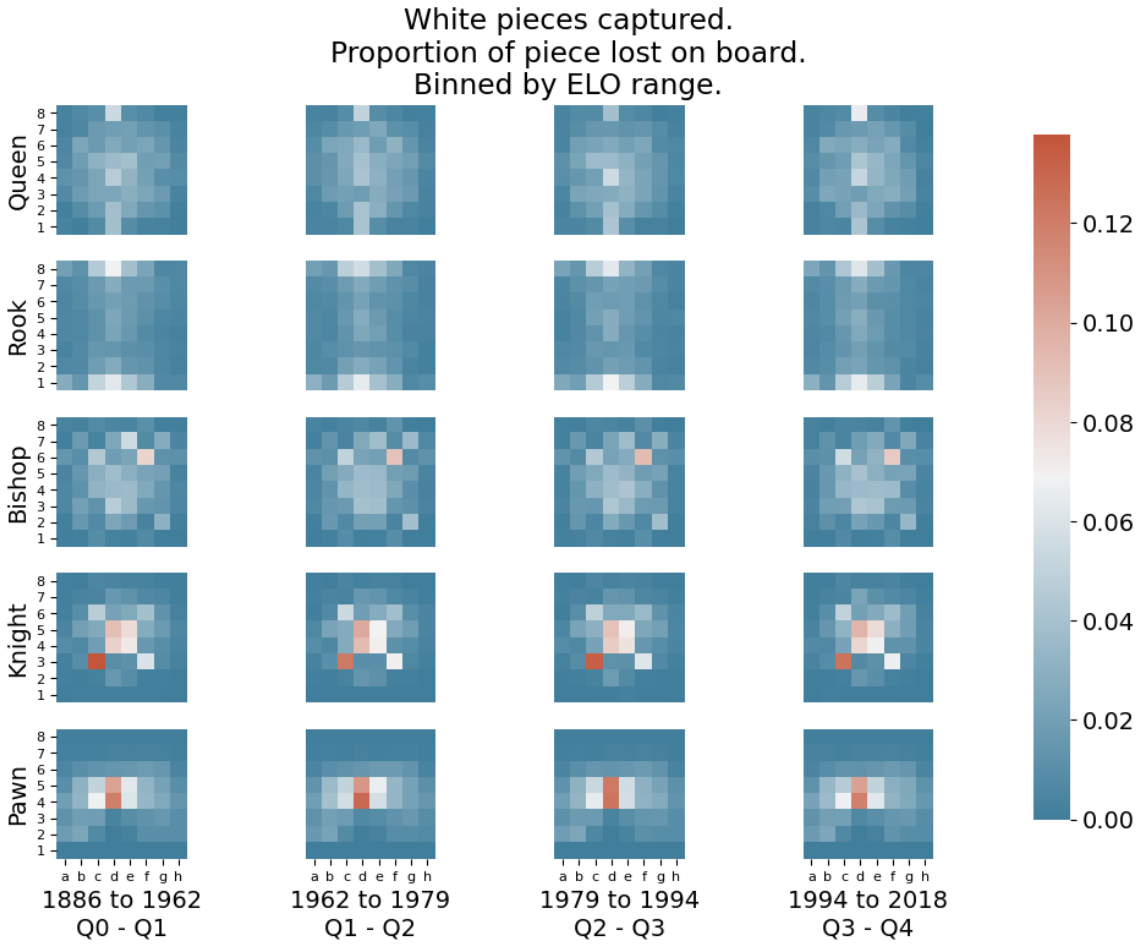


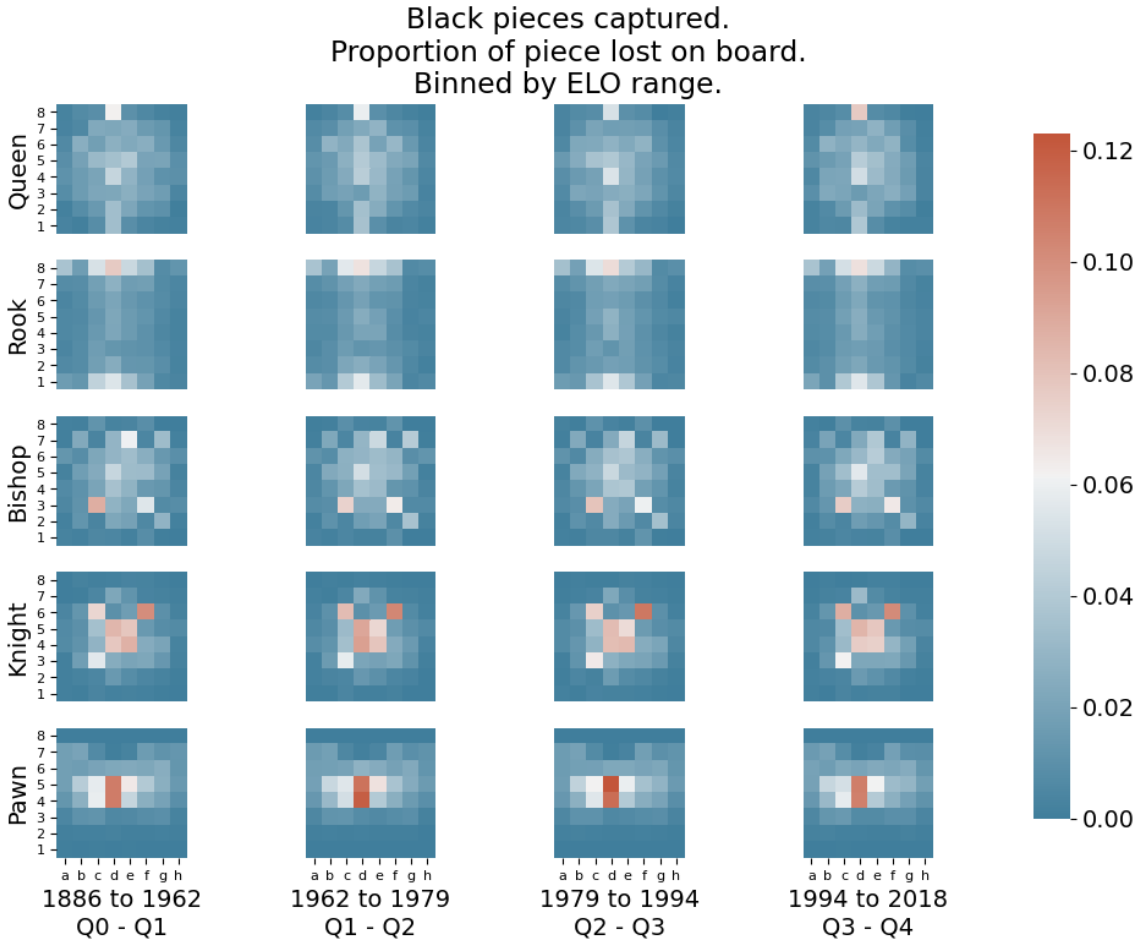

Black pieces captured.
Proportion of piece lost on board.
Binned by ELO range.
Queen
Rook
Bishop
Knight
Pawn
a b c d e f g h
718 - 1189
-∞σ to -2σ
1189 - 1385
-2σ to -1σ
1385 - 1605
-1σ to 0σ
1605 - 1899
0σ to 1σ
1899 - 2371
1σ to 2σ
2371 - 2725
2σ to ∞σ
0.12
0.10
0.08
0.06
0.04
0.02
0.00

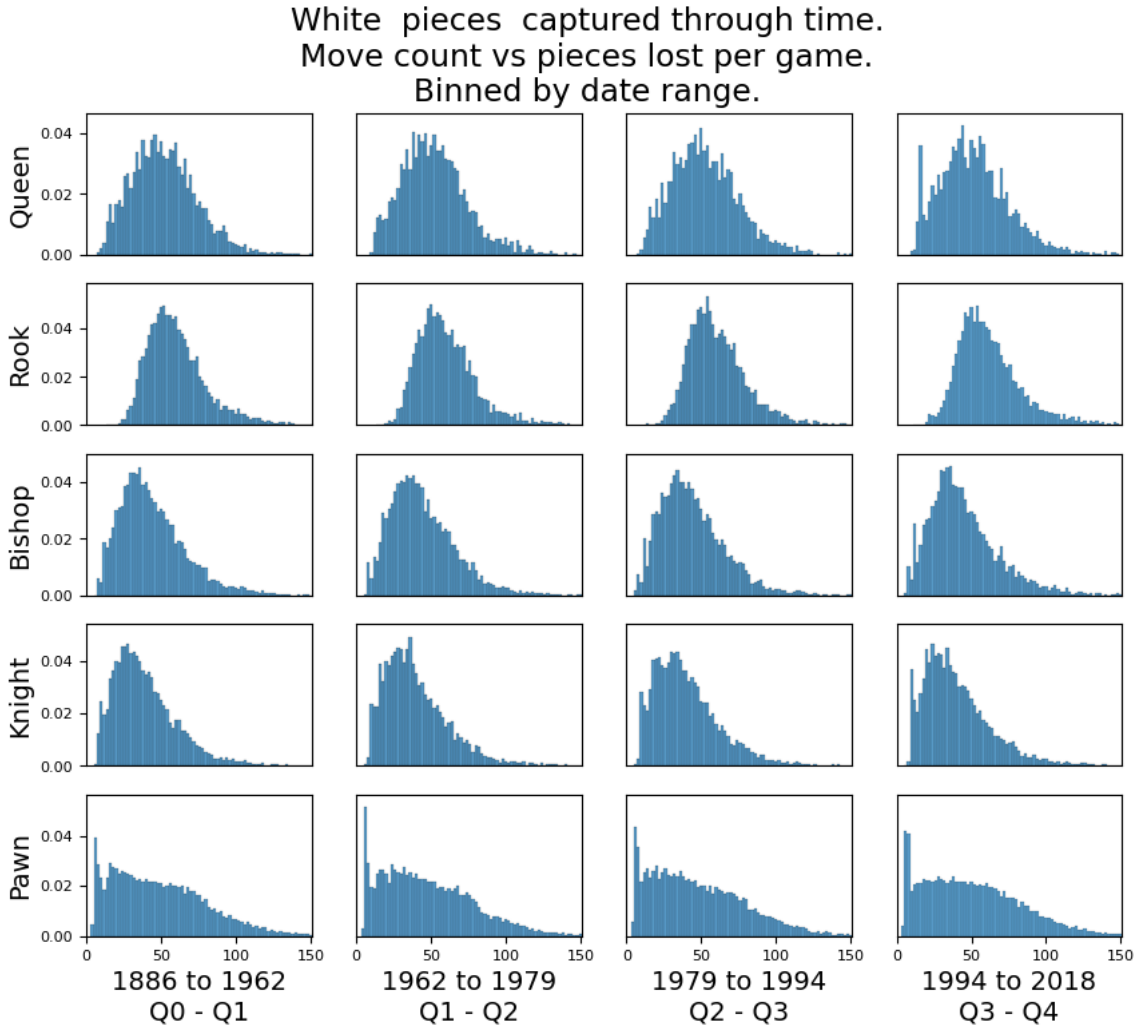

White pieces captured through time.
Move count vs pieces lost per game.
Binned by date range.
Queen
Rook
Bishop
Knight
Pawn
0.00
0.02
0.04
0.06
0
100
718 - 1189
-∞σ to -2σ
1189 - 1385
-2σ to -1σ
1385 - 1605
-1σ to 0σ
1605 - 1899
0σ to 1σ
1899 - 2371
1σ to 2σ
2371 - 2725
2σ to ∞σ

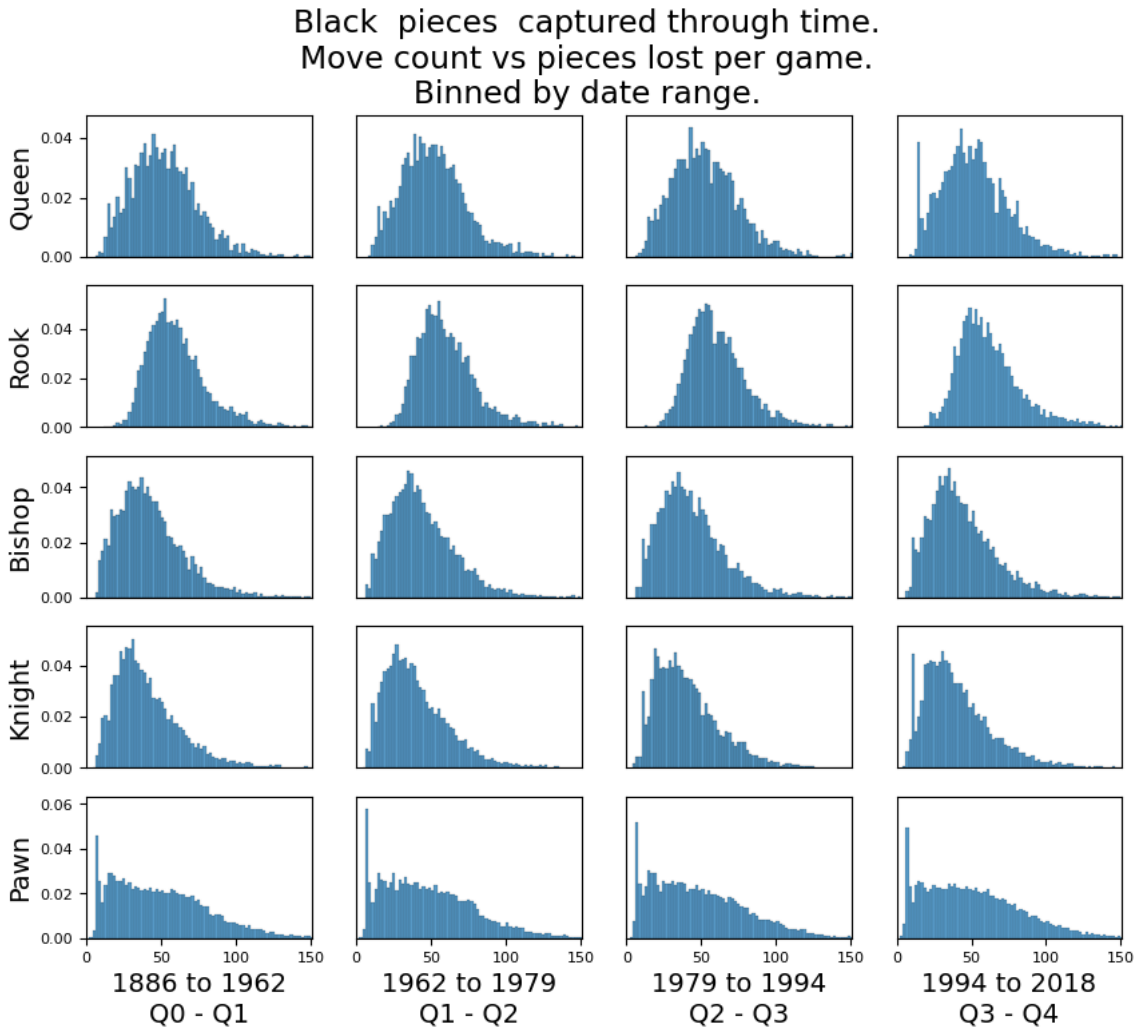

Black pieces captured through time.
Move count vs pieces lost per game.
Binned by date range.
Queen
Rook
Bishop
Knight
Pawn
718 - 1189
-∞σ to -2σ
1189 - 1385
-2σ to -1σ
1385 - 1605
-1σ to 0σ
1605 - 1899
0σ to 1σ
1899 - 2371
1σ to 2σ
2371 - 2725
2σ to ∞σ

6.2 Date Grids (Tournament)

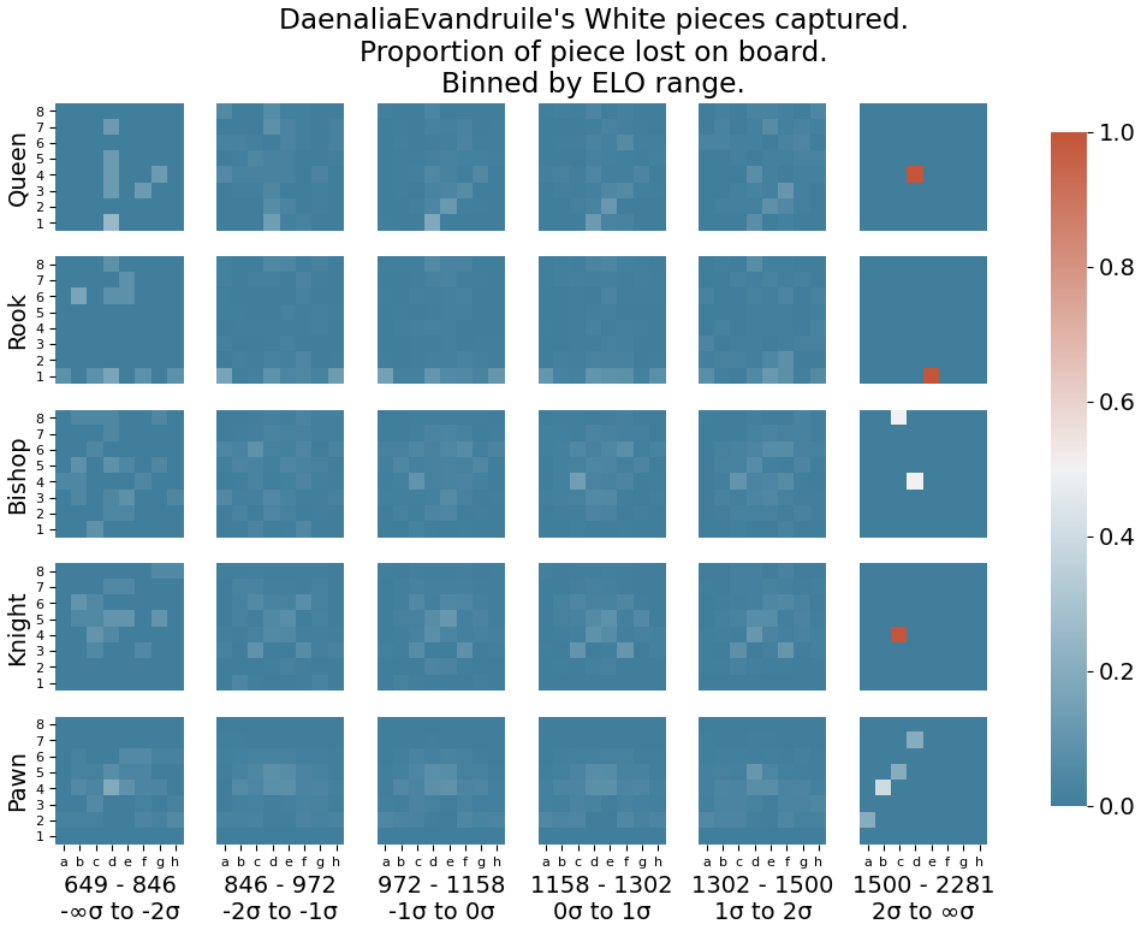



Black pieces captured.
Proportion of piece lost on board.
Binned by ELO range.
Queen
Rook
Bishop
Knight
Pawn
1886 to 1962
Q0 - Q1
1962 to 1979
Q1 - Q2
1979 to 1994
Q2 - Q3
1994 to 2018
Q3 - Q4
0.12
0.10
0.08
0.06
0.04
0.02
0.00

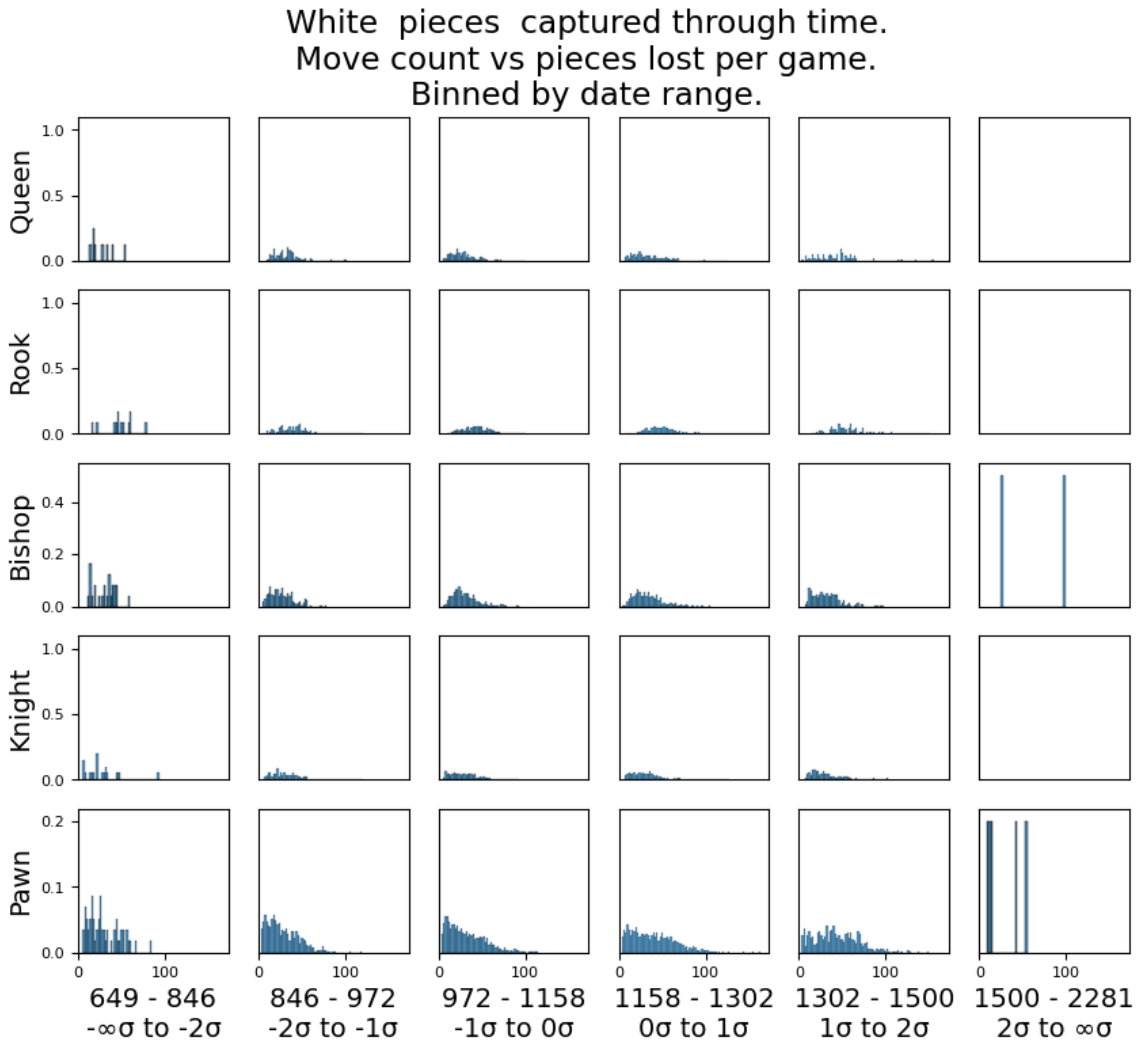

White pieces captured through time.
Move count vs pieces lost per game.
Binned by date range.
Queen
Rook
Bishop
Knight
Pawn
0.04
0.02
0.00
0
50
100
150
1886 to 1962
Q0 - Q1
1962 to 1979
Q1 - Q2
1979 to 1994
Q2 - Q3
1994 to 2018
Q3 - Q4

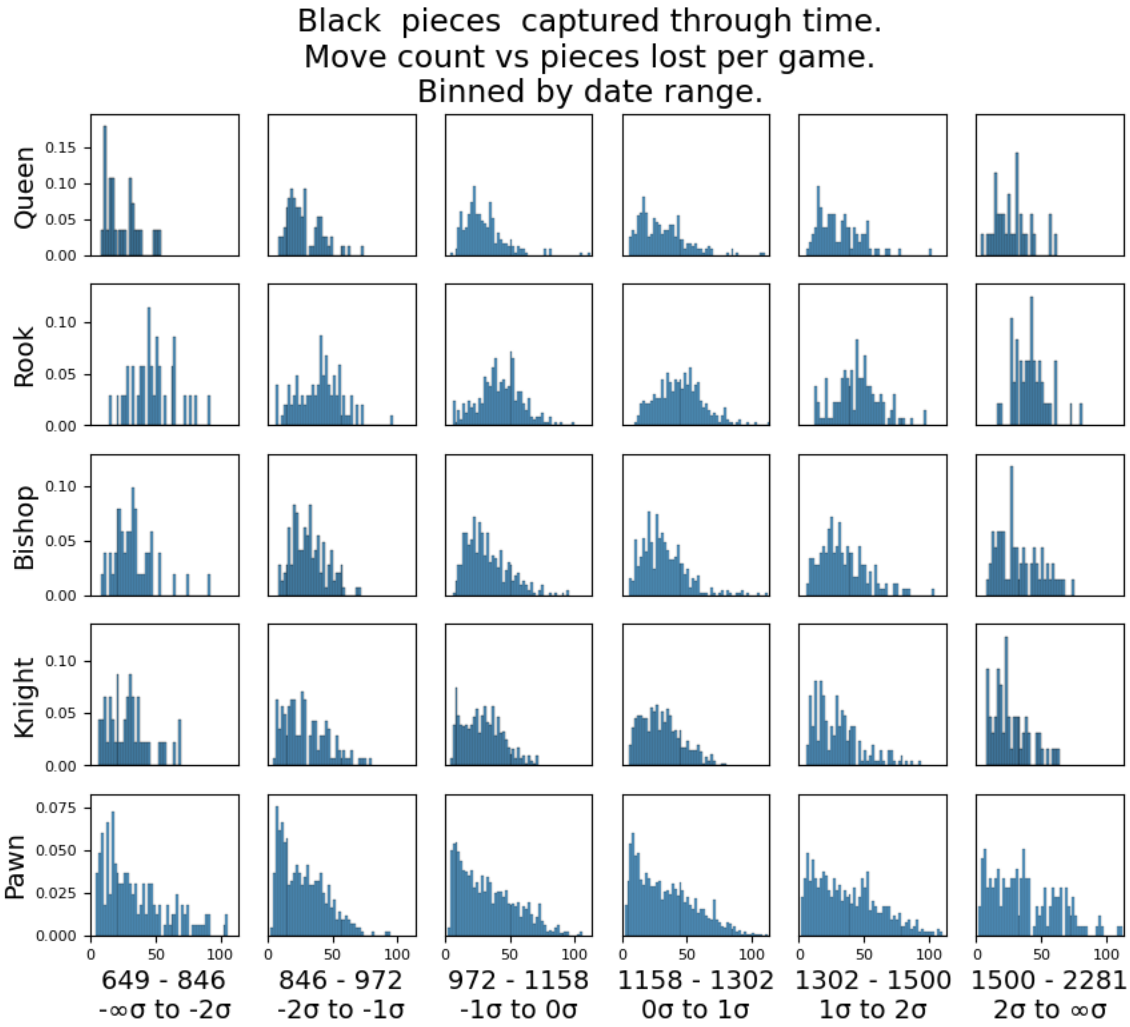

Black pieces captured through time.
Move count vs pieces lost per game.
Binned by date range.
Queen
Rook
Bishop
Knight
Pawn
0.00
0.02
0.04
0.06
0
50
100
150
1886 to 1962
Q0 - Q1
1962 to 1979
Q1 - Q2
1979 to 1994
Q2 - Q3
1994 to 2018
Q3 - Q4

6.3 DaenaliaEvandruile

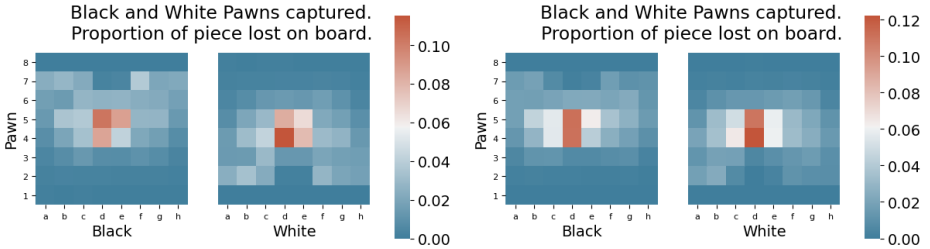


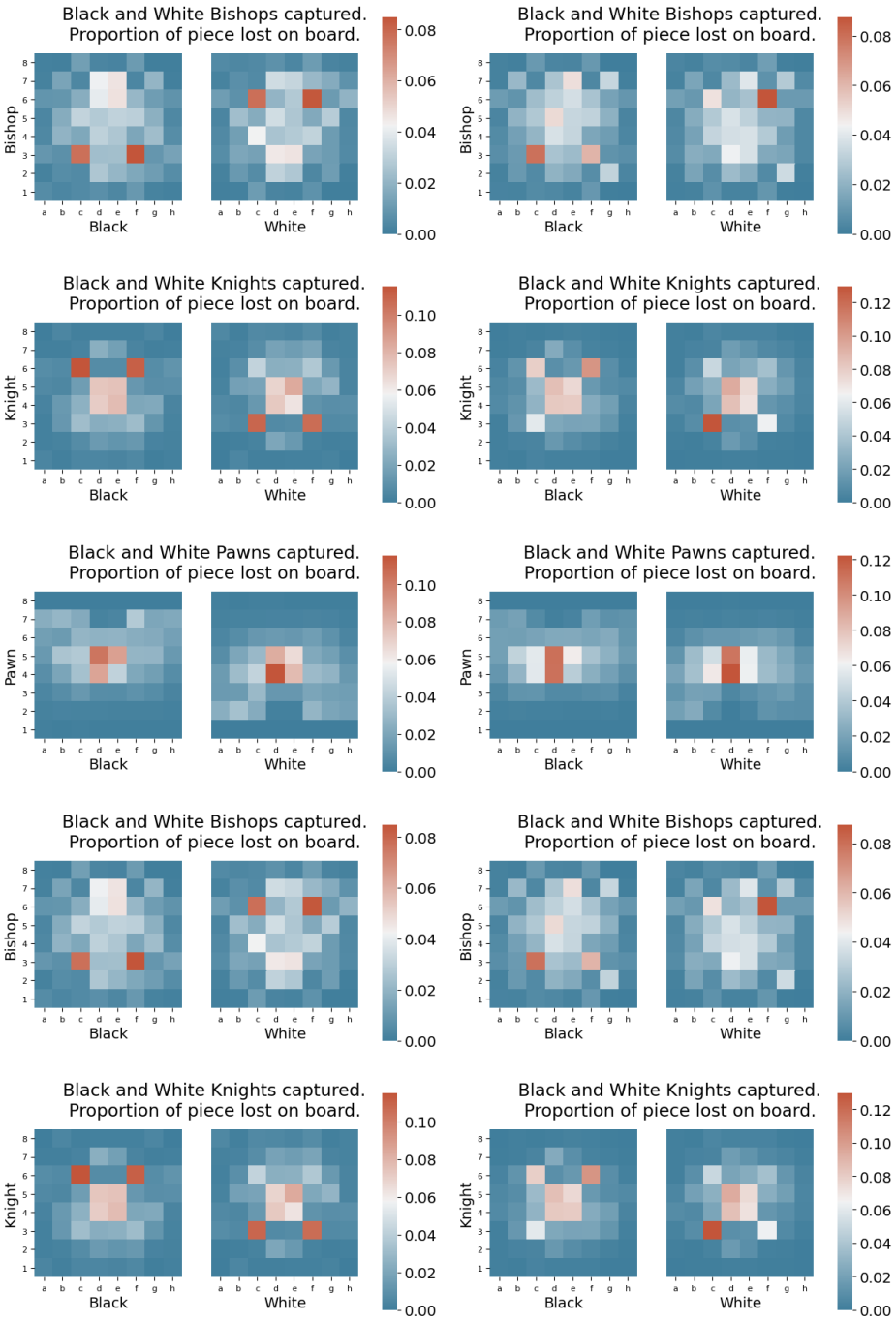

DaenaliaEvandruile's Black pieces captured.
Proportion of piece lost on board.
Binned by ELO range.
Queen
Rook
Bishop
Knight
Pawn
649 - 846
-∞σ to -2σ
846 - 972
-2σ to -1σ
972 - 1158
-1σ to 0σ
1158 - 1302
0σ to 1σ
1302 - 1500
1σ to 2σ
1500 - 2281
2σ to ∞σ
0.25
0.20
0.15
0.10
0.05
0.00


White pieces captured through time.
Move count vs pieces lost per game.
Binned by date range.
Queen
Rook
Bishop
Knight
Pawn
649 - 846
-∞σ to -2σ
846 - 972
-2σ to -1σ
972 - 1158
-1σ to 0σ
1158 - 1302
0σ to 1σ
1302 - 1500
1σ to 2σ
1500 - 2281
2σ to ∞σ



6.4 Individual piece plots




Black and White Bishops captured.
Proportion of piece lost on board.
Bishop
Black
White
Black and White Knights captured.
Proportion of piece lost on board.
Knight
Black and White Pawns captured.
Proportion of piece lost on board.
Pawn