Comparing Concept Acquisition in Human & Superhuman DeepRL Models

1st Anthony Marchiafava
dept. Computer Science
University of New Orleans
New Orleans, United States of America
amarchia@uno.edu

2nd Atriya Sen
dept. Computer Science
University of New Orleans
New Orleans, United States of America
asen@uno.edu

***Abstract*—In order to better exploit Deep Reinforcement Learning (DeepRL) systems such as DeepMind's AlphaGo & AlphaZero, it is desirable to understand how they acquire knowledge, and how human knowledge acquisition can contribute to or benefit from such an understanding. We analyze a series of DeepRL models called Maia, trained to play the board game of chess in a human–like fashion, to study if these models acquire concepts differently from self–trained DeepRL models such as AlphaZero. Our results indicate that human chess players may acquire concepts differently from self–trained models. We further discuss some of the potential consequences of such an outcome, and opportunities for future work.**

***Index Terms*—artificial intelligence**

I. Introduction

In [1] we explored the question of whether a series of deep neural networks models designed to play chess in a human–like manner, called Maia [2], learnt elementary chess concepts such as *material advantage*. We showed evidence that they indeed learnt a range of simple, hand-crafted concepts. For a simplified version of material advantage, we used linear regression to identify if the player has a material advantage, if the opponent has a material advantage, or if there is no material advantage. Subsequently, we studied an augmented version of material advantage that took piece positions into account. Whereas an accuracy score of 1.0 would be the best possible result (implying that our model was able to detect the concept perfectly from the input data) we obtained a score close to 0.7 for only the prediction of who has an advantage, and approximately 0.5 for advantage prediction using custom evaluation functions for all the versions of Maia we examined. These early results indicated that there was sufficient data in the network to say that the simple version of advantage we used was detectable.

This work builds upon our previous results: we improve our dataset, increasing the locations in the network we examine to include more activation and convolution layers than previously used, expand our methods of evaluation beyond linear regression, and change the concepts we wish to detect, to the more sophisticated concepts provided by the classical position evaluator of the chess engine Stockfish 8 [3], which is the last version of that engine which does not use neural networks to evaluate positions, but instead provides discrete evaluation scores for a range of simple concepts, which we use.

The overall goal of this work remains the same as [1]: to analyze Maia models, which are trained to play the board game of chess in a human–like fashion, to study if these models acquire concepts differently from self–trained DeepRL models such as AlphaZero.

Artificial Intelligence (AI) systems for playing the game of Chess have existed since the early development of computers, and computer chess as an idea has existed for at least seven decades [4], and has continued to see rapid development. Chess has been called the "*drosophila*" of reasoning [5], alluding to the widespread use of the *drosophila* (fruit fly) in biological research. Many advances in chess AI have advanced the state-of-the-art of AI systems generally.

A summary of this paper follows. In Section II we discuss some relevant background. Then, Section III summarizes prior & related work. In Section IV, we detail the methodology used to generate data and discuss investigative goals. In Section V we detail and discuss the results and conclusions drawn from our data and experiments. Finally, in Section VI, we indicate some further research avenues to consider in the future, as informed by our results.

II. Background

Neural networks are commonly referred to as "black boxes" [6]: their operation is not understandable or interpretable to human beings merely by observing the propagation of information through them in the form of high-dimensional and large volumes of numbers, or by directly observing learnt parameters, called *weights*, whose values have no direct or easily discernible connection to the predictive outcome. Neural networks are trained by propagating the error that results from comparing the supervised *training signal* against the error in prediction at an instance of time in the training process.

Nevertheless, neural networks can provide excellent function approximation and produce groundbreaking results on a wide variety of (especially) perceptual tasks. Explainable AI (XAI) techniques are designed to address the opacity of operation that neural networks exhibit [7]. In this work, we use a technique called *linear probing* to determine if a deep neural network model has acquired a human–interpretable concept, at an instance of it's training. This technique is explained in further detail in Section IV.

Recent chess-playing AI systems employ deep neural networks to inform their decision-making process [8] [9] [3]. These deep neural models combine neural network activation values across layers with *normalizations*, *convolutions*, and *skip connections*, in the form of an architecture known as a Residual Neural Network (ResNet) [10]. ResNet, originally developed for computer vision tasks, was used by Google DeepMind in the development of their AlphaGo system, which learned to play the game of Go, the Google DeepMind AlphaZero system which learned to play Go, Chess, and Shogi, and the Leela Chess Zero system, which plays Chess [11] [9].

Maia was chosen due to similarity of the neural network architecture to AlphaZero. The primary difference between them is training methodology: AlphaZero was trained through self-play [9] and moves were selected using the Monte Carlo Tree Search (MCTS) [12] algorithm. MCTS explores the large search space of the various games AlphaZero learned to play, by exploring the consequences of moves suggested by the randomly initialized neural network in a tree–like fashion. The search mechanism is an augmented form of tree search that exploits probabilistic sampling of the search space.

Randomly initialized weights are initially used, and eventually updated using the neural network training process, guided by the move prediction output. The Monte Carlo Tree search process is made up of a series of repeated steps. First is the selection of nodes in a search tree until a node which has had no simulations is selected. The expansion where this incomplete game's node is chosen to be played. The simulation or rollout uses the neural network in this case to choose a move play out that game. Finally, backpropagation is used to update all the nodes in the path from the played node to the root with the results of the game.

This is the technique used to guide self–play and produce training data for our neural networks: after many rollouts and the backpropagation of their results, this produces a dataset of input games and predicted results. Small amounts of noise are used to reintroduce some randomness into the network's prediction to enable it to continues to explore the search space, where it might otherwise continue to focus on a single approach and not explore and thereby learn from other yet unexplored approaches.

Each neural network is a general function approximator that learns through an algorithm known as backpropagation [6]. A neural network at its most basic is a two-state regression or classification model which takes in some input and produces one or more outputs. The network will take the input and produce derived features, which are used further to produce more derived features depending on the depth of the network, and the derived features are used to produce the final output. The derived features are produced using linear combinations of the inputs and non-linear activation functions and other operations such as *convolutions*, which occur at different layers of the network. The first few layers more closely match the structure of the initial input, but as further derived features are generated, they become increasingly abstract.

The neural networks we examine here observe the current state of the chess board, prior states of the chess board, and a number of parameters specific to the game of chess, such as the current castling status, as input, and produces two outputs: one for each of the network "heads". One of these is the *policy head*, which outputs the probability distribution of possible moves. The other is the *value head*, which output the predicted outcome of the game: a win, loss, or draw. These are based on Maia, which is in turn based on Leela Chess Zero, which is itself based on AlphaZero [2][13][9].

The results of these games are then evaluated during the tree search, so that the move predicted is the one most likely to result in a win. The neural network guides the exploration and exploitation of the tree. After a sufficient number of games played out, these game results are then used to train the neural network. This process continues to repeat as the neural network is improved, to explore increasingly better moves. The MCTS search, based on exploring and exploiting moves selected by the neural network, results in the model choosing incrementally better moves.

AlphaZero was based upon AlphaGo, which first learned from human experts and then from self–play. The original AlphaGo was designed to only play Go and its neural network first learned to make moves which approximated human games through supervised learning. This means it would examine a given board position, and learn to generate the move that the high–level Go player made. Then, AlphaGo trained on data generated through playing against itself. This prevents the network from choosing moves based on approximating those of the human experts and instead find the aim to find the optimal sequence of moves to a win [11], in a way not constrained by human conceptual understanding and possible *mis*understandings.

Maia takes a different approach from AlphaGo and AlphaZero in that it does not select moves based on using MCTS at all. Instead, it uses a supervised learning technique similar to the early stage of AlphaGo where AlphaGo used supervised learning to train on the top moves of human go masters, but instead of choosing the best chess players in the world as its sample to learn from, the versions of Maia were each trained on groups of players coinciding with specific skill groups, based on the ELO rating, which is the universally accepted rating system in chess. So, instead of learning to approximate moves based on what the best players in the world are choosing (presumably to win), each version of Maia learns to play in such a way that it would approximate both the good moves and mistakes that players in that skill bracket would make [2]. While the neural networks that formed the various versions of Maia were similar to that of LeelaChess Zero (Lc0), which itself was based on AlphaZero's chess implementation, the Maia networks were structured with 6 convolutional blocks and 64 filters, as a compromise of performance and computational costs.

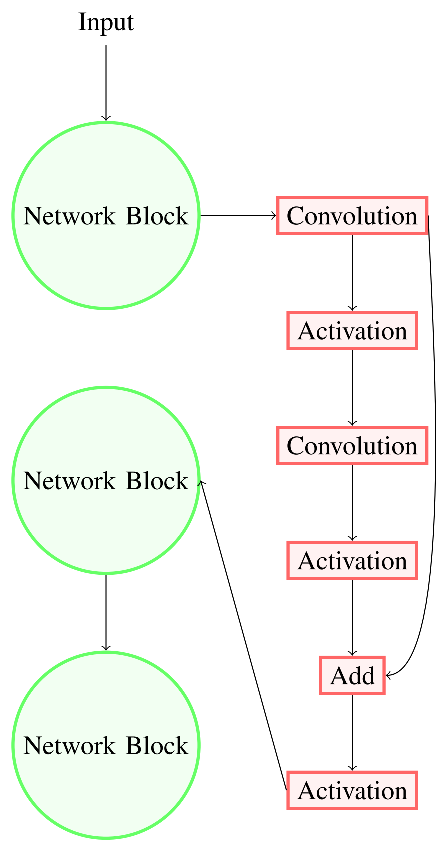


Fig. 1. Simplified view of a neural network. The area of interest in the network is made up of six network blocks, here illustrated in circles. As information passes through the network's block, the layers of interest within the blocks are here illustrated in rectangles. The skip connections are noted by the Add layer. Each network block contains many of these layers, connected sequentially. We omit the squeeze-and-excitation and batch normalization layers in our illustration.

III. Related Work

There have been analyses of AlphaZero's neural network's development [14], and neural network based versions of Stockfish [15]. However, to our knowledge, there has been no analysis of such a network designed to approximate human play as Maia. How the self–playing trained networks and the human–supervised learning trained networks differ, may provide further insight into what networks which learned from scratch like AlphaZero do, that entirely supervised learning networks such as Maia do not.

Within the field of explainable chess models there are also alternative versions of the game which are less computationally expensive to process such as minichess [16]. While these versions of chess are generally similar to normal chess, their smaller boards and restricted numbers of pieces are more computationally inexpensive. These more obscure versions of chess do not have the same level of data to support training a supervised learner in the same fashion as Maia.

In addition to linear probing, the idea of *counterfactual explanations* as an concept interpretation technique was also considered. These describe the smallest change to the world that can be made to obtain a desirable outcome, or to arrive at the closest possible world, without needing to explain the internal logic of the system [17]. Such an explanation of a chess concept could indicate what the chess playing program was considering when evaluating a position on a case–by–case basis. This amounts to considering what a chessboard that is in as close as possible a configuration would need to look like, to produce a specific move that we would select, instead of the move that was chosen in reality. This technique would allow us to better understand why particular moves are made when compared to other moves, but would not give us an indication of what broad concepts are being acquired by the network, informing the selection of moves overall.

We also considered *non-negative matrix factorization* [18]. This approach allows concept detection for concepts that we did not define and therefore provides a mechanism for unsupervised concept discovery. This is done by taking some matrix such as layer activation values and decomposing into two non-negative matrices, a matrix of features and a matrix of coefficients. The product of these two smaller matrices should approximate the original matrix. This would allow us to approximate how much a given feature contributes to the activation values in the networks, but would not indicate what the feature actually is and would therefore not help us understand what differentiates our network from Leela Chess Zero and similar networks.

There are other notable neural network architectures outside of ResNet which have proven capable of playing chess. One increasingly popular architecture is that of generative large language models, which have been applied to chess in addition to many other topics [19]. Further analysis of concepts in some of the other chess playing systems, such as like chess AI trained on language models that use transformers, may provide further insight into concept utilization. In particular, the context of previous moves have a strong influence on which move will be made in the future [19], motivating the use of a transformer or other memory–based architectures. This is in contrast with the observations of Leela, which only retains the last eight moves in the game and not the complete move history. Language models may also be used for explainable chess concepts outside of interpreting neural network models, such as building chess commentary from language models trained on human comments [20]. These language models provide an alternative to the interpretation methods we examine here in detail. However, using LLMs as an interpretability method is still in its infancy.

Currently, the performance of these language based networks falls short of the superhuman standard set by networks such as Leela Chess Zero. The twelfth major version of Stockfish and onward (after 2020) use efficiently updateable neural network evaluations (NNUE) [3] and this network architecture has been analyzed in a fashion similar to that used for AlphaZero [15]. This network design is based on work used for playing Shogi [21], and when used in chess it is a much more shallow network (with fewer but more wide layers) than AlphaZero, Leela Chess Zero, or Maia. These networks were not examined here, due to there not existing a human-play trained version of them, for comparison against.

IV. Methodology

To explore concept usage by the neural network models in question, we assembled a dataset of unique chess positions and produced Stockfish 8 classical evaluations for each of those positions. We then matched the output of each activation layer in each neural network model to the corresponding Stockfish 8 evaluation for that position. We were able to generate regression functions using three linear regression techniques, and produce metrics that show how predictable our outputs are from our data.

```
for all unique chess positions do
    $StockfishList$ ←StockfishEvaluate(position)
end for
for all versions of Maia do
    for all layers in each version of Maia do
        $LayerEvalList$ ←LayerOutput(position)
    end for
end for
for each set of activations in LayerOutputs do
    $x \leftarrow activations$
    $y \leftarrow stockfishevaluations$
    $RegressionFunction$ ←LinearRegression(x,y)
    $RegressionFunction$ ←LassoRegression(x,y)
    $RegressionFunction$ ←RidgeRegression(x,y)
end for
```

A. Data

We selected the games from the Maia dataset used to train the 1500 ELO version of Maia. There were between 15000 and 20000 unique game boards used for analysis in the random sample we drew from the initial dataset. Each game was converted into a format suitable for use in Maia using the same Trainingdata-tool used for generating Maia data [22]. Trainingdata-tool takes complete games in Portable Game Notation (PGN), after extraction with pgn-extract [23], and converts them into the binary data format that the Maia networks use. This conversion was done in the same manner that was used to train the Maia networks initially. This dataset was paired down to only unique game positions based on the position of pieces on the chess board by that board's unique FEN (Forsyth-Edwards Notation). The FEN format was necessary because it allowed us to input these positions into Stockfish 8's classical evaluator. We detected the active player and mirrored the board as necessary, because Maia (and by extension Lc0) requires the input to be in a certain orientation, while FEN always initially positions the black pieces on top and white on the bottom. Therefore, it was necessary to produce a way of translating from the Maia to Stockfish 8 formats.

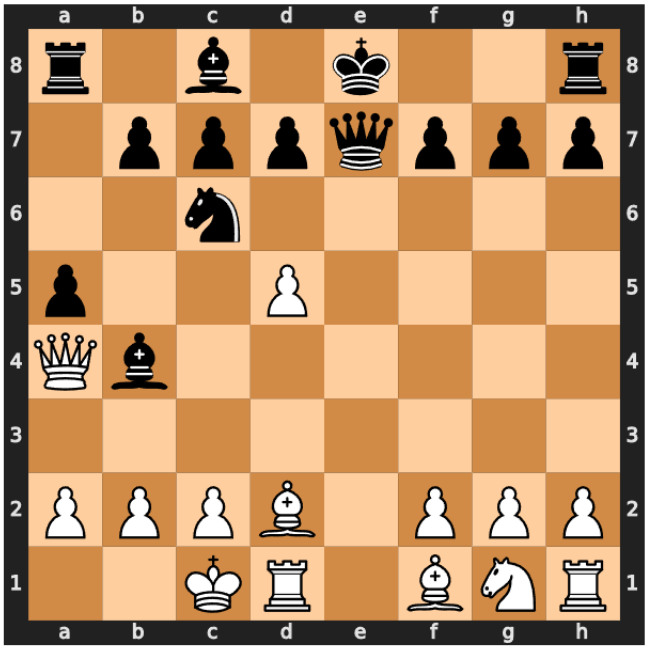


Fig. 2. Example Chess Position from out dataset.

B. Stockfish 8 Evaluations

After adjusting for these formatting changes, we compiled a set of Stockfish's classical evaluations. We chose to evaluate each position as though it were the *middle game*, as the binary data format that Maia uses does not encode the move number. Stockfish 8 produces both a middle game and an early game evaluation and as the line between the opening and middle game can be unclear and arbitrary, we chose to use the middle game evaluations for all evaluations used here. The concepts with a classical evaluation were material imbalance, the relative value of the following pieces: pawns, knights, bishops, rooks, queens, mobility (how many legal moves pieces have), king safety, threats (which pieces are currently under attack or weakly protected), passed pawns (blocked or unblocked passed pawns), and space (squares controlled by each player's pieces). Each of these produces a numerical value that estimated a value that the concept contributed to a given board position. These are concepts that the Stockfish developers chose as important, to evaluate a given chess position. The evaluations are a single value per concept for the black and a value for the white player, and the net of these two values as a total. These Stockfish 8 evaluations are then the concepts we aimed to probe for.

In the example shown in Figure 2 Stockfish 8 evaluates its middle game scores for total material score as -0.05, Knights score of 0.06, Rooks score of 0.19, Queens score of -0.14, Mobility score of -0.01, King Safety score of -0.48, threats score of 0.86, space score of 0.10, and zero for the other metrics evaluated.

C. Neural Network Architecture

The Maia neural network is structured as a *convolutional residual network* (ResNet). It is composed of a series of six convolutional blocks, where each block is made up of convolutions, batch normalizations, and activations using rectified linear units (ReLU). The input to each of the network blocks also has skip a connection to the end of its block, where the two are added together, and the combined output is then used to feed into the next block. The structure of this network uses 6 blocks and 64 filters, where choice of the number of blocks and filters was informed by computational costs. This is smaller than the typical number of blocks (10 to 24) and filters (128 to 320) found in a network like Leela Chess Zero [13].

As information is transformed by successive layers of the neural network, the input is processed from a series of bitmaps to represent piece positions on the board into increasingly abstract numerical values through convolutions, batch normalization, and unit activation up to the output layer. This activation data is used by our linear probing model, described as follows.

D. Linear Probing

The linear probing experiment was based on a similar process used to analyze AlphaZero and a more recent version of Stockfish, where the output of a given activation layer is used as input to a linear regression model to examine if the activation is sufficient to predict the output of the concept that is being probed for [14][15]. These probes are based on *concept activation vectors* [24][25], which tie user–defined concepts to neural network activation results: their values indicate how strongly a concept influenced the output of the network. This technique was modified to be applicable to concepts and topics outside of computer vision domains. The process requires isolating the concept(s) we wish to examine, recalling the abstract structure illustrated in Figure 1. We created a version of the network which terminates at a given layer. For example, instead of sending the first convolution layer's output to the first activation layer we stored that output as our regression model's input.

E. Logistic Regression Model

As a baseline we trained a binary classifier using logistical regression, to predict the concept of *material advantage greater than 3*, across activation layers 1 through 9 and 11 through 17, defined as a Stockfish evaluation of material advantage as $>= 3$, with anything < 3 indicating the absence of significant material advantage. The results were similar to those in [1]. In this paper we focus on the more difficult task of predicting the exact real–valued Stockfish–generated concept scores, using linear regression.

F. Linear Regression Model

Like in [14], we trained a regression model from activation z^l at a given layer l to our Stockfish concept s using a linear predictor for the continuous concepts in question. We trained a regression function from the output of the l^{th} activation layer of the network to predict the Stockfish concept s. We analyzed the set of continuous concepts reported by Stockfish 8 by using Linear Regression, LASSO, and Ridge Regression on a training dataset of activation values and their matching concepts, to generate three linear models, as follows:

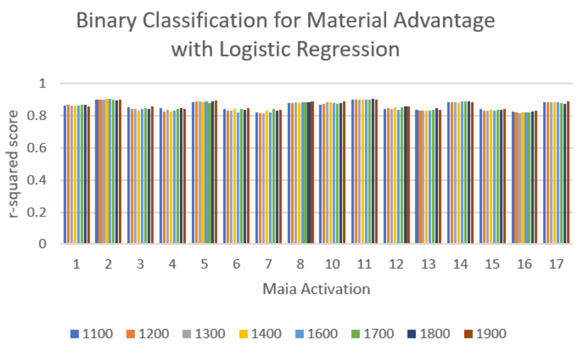


Fig. 3. Binary classification r-squared scores for different versions of Maia across the network.

- *Linear Regression* or ordinary least squares linear regression fits a linear model to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.
- *Lasso* is a linear model trained that does both regularization and variable selection, and uses L1 regularization. It uses the standard linear regression model and modifies this using the sum of the absolute values of the parameters being estimated multiplied by some tuning parameter. When the number of tuning parameters is large, more coefficients are driven to zero so as to remove features which are not useful.
- *Ridge regression* minimizes the squared error with L2 regularization in the same way as previous research has shown [15].

V. Results

Starting with baseline logistic regression, our scores conform to our expectations and previous results [1]. These are shown in Figure 3. They indicate that the binary concept of *material advantage greater than 3 points* is quite accurately classifiable from the feature vectors of the pre–trained Maia networks, showing that, across the breadth of the network, each version of Maia learns representations conducive to easily predicting if a given player has such an advantage. Similar results using linear regression to probe for a real–valued concept would indicate that the exact Stockfish concept score is predictable in the same manner: a more challenging task.

We sampled over 10000 unique chess positions to generate our dataset, and for each linear regression split out data into 4 equally sized folds. Each fold was combined using k-fold cross validation to ensure that our sampling was not affecting the performance of our model. We then performed a hyper–parameter search over the alpha values (which are used as the L2 term multiplier) using values of 0.1, 1.0, 10.0, 50.0, 100.0, 500.0, and 1000.0. Finally, we used the r-squared score of the held-out testing fold to indicate the regression accuracy. After producing these scores for each fold, we calculated the mean of the folds and we have reported these values in this section, for each activation and for each layer.

As shown in Figure 4, across the various options for alpha, none of our results attained r-squared scores above 0.30. These results were duplicated across every version of Maia we examined. A r-squared score close to 1.0 would indicate that we were able to generate a model which can perfectly predict the concept manifestation from the activation data. A result of zero or negative indicates that the input was not sufficient to predict the output (if the model was constant and always predicted the output without regard to the input the result would be 0.0), and models could be arbitrarily negative in their resulting score.

Figure 5 shows that nearly all of the concepts under examination have a negative r-squared score, for all versions of the network. The overall pattern of the r-squared scores for these networks are not notably dissimilar. The later network activation values are similar.

For all of the versions of Maia and layer activation values we examined (i.e., Maia for ELO 1100, 1200, 1300, 1400, 1600, 1700, 1800, and 1900, and activation 1 through 8: see Figure 5, and activation values 10 to 17: see Figure 8), returned similar r-squared scores. At best they were very mildly positive when using the higher alpha values. None were near even 0.5, instead scoring near 0.2 at best. Each version of Maia performed similarly for all concepts identified. Our expectation was that the lower rated versions of Maia would have less conceptual understanding while the higher rated versions would have superior conceptual understanding. However, our data indicates that the level of understanding was similarly poor for all versions of Maia, at least with the concepts under examination.

We also examined some of the outputs of the convolutional layers, as a potential source of concept regression: these are illustrated in Figure 6.

VI. Discussion & Analysis

While previous work [1] indicated that our hand–crafted preliminary human–interpretable concepts could be accurately detected within the network with r-squared scores of up to 0.7 out of 1.0, here our conclusions differ. The concepts we explored previously were simple in nature: indicating if it was possible to regress which player had an advantage or if none was present (but not a mathematical evaluation of their advantage) and used a dataset which included duplicate positions. After removing those duplicated positions to better match similar datasets in similar research [15] and increasing the complexity of the concepts under investigation, the scores of the evaluation metrics dramatically decreased.

This is not to say that our present claims are incompatible with our previous results. Originally, we asked if we can probe for *whether or not* there is a *significant* material advantage present, as a binary classification problem. Our present work

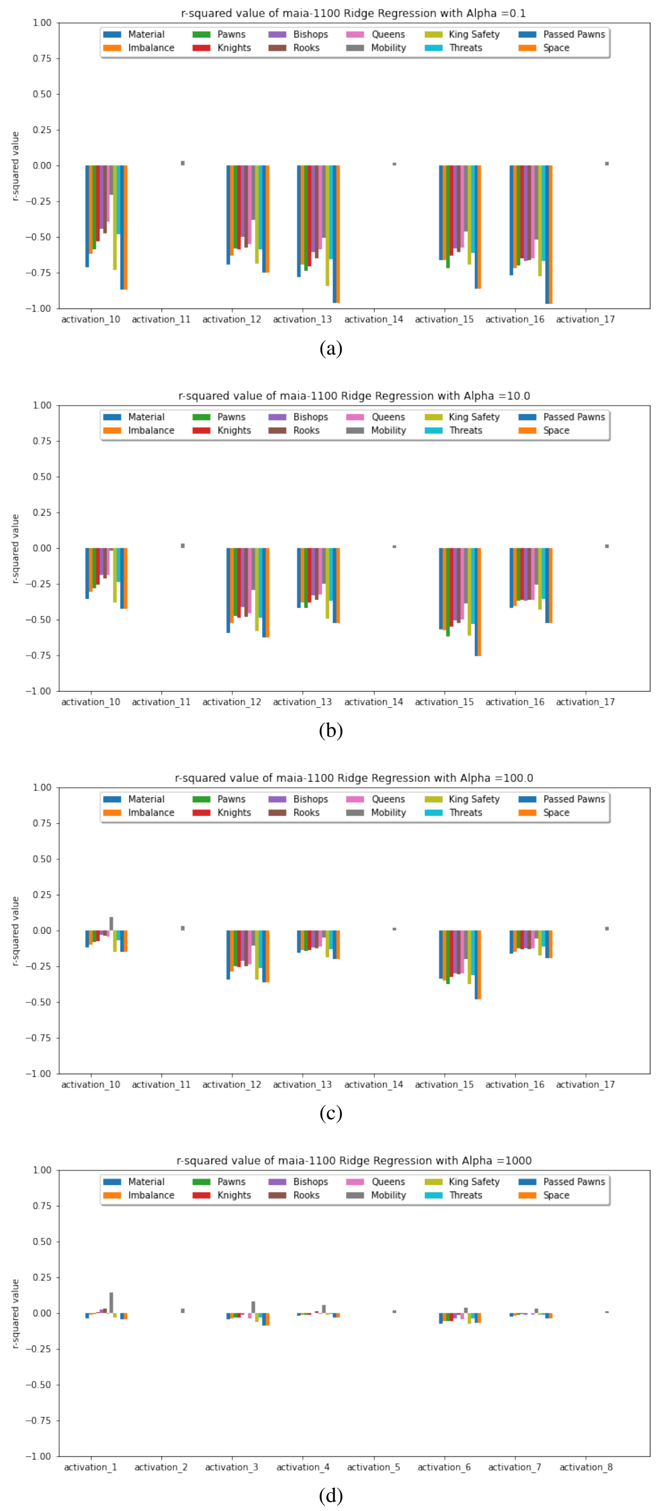
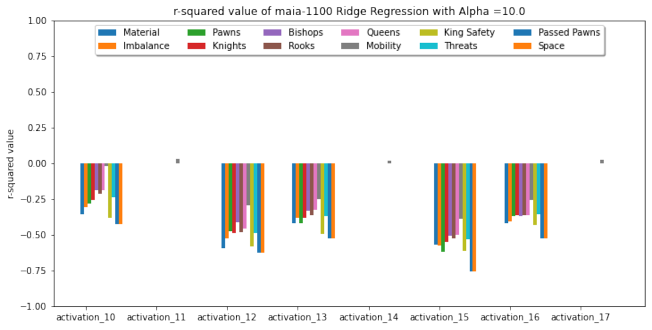
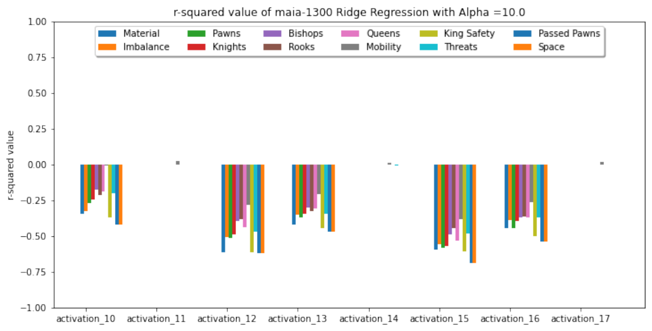


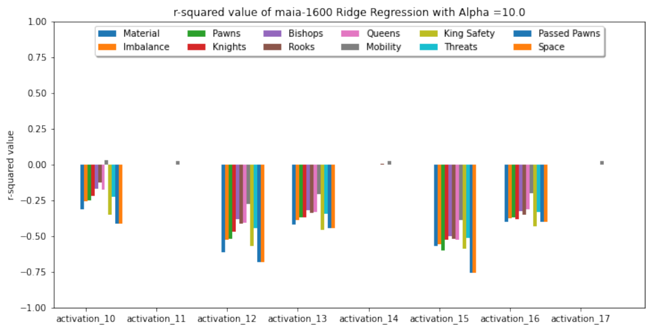
Fig. 4. Graph Showing the r-squared scores of the concepts examined across alpha 0.1, 10.0, and 1000.0.



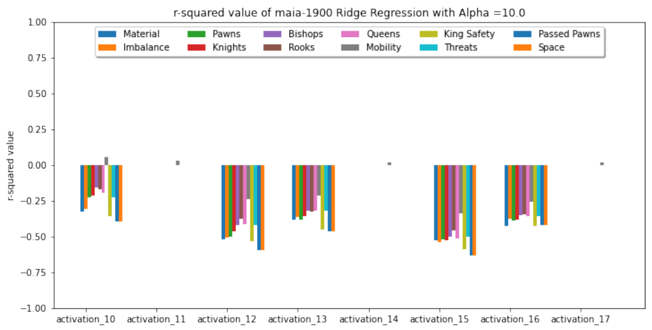
(a)



(b)

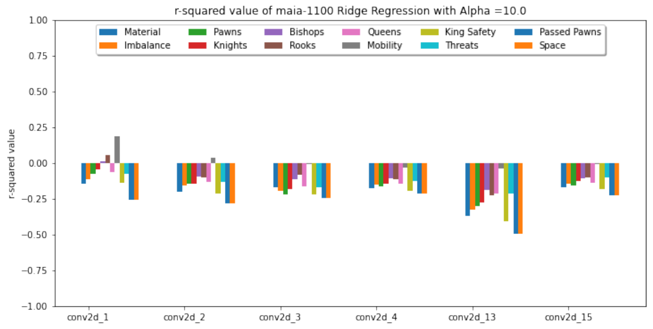


(c)

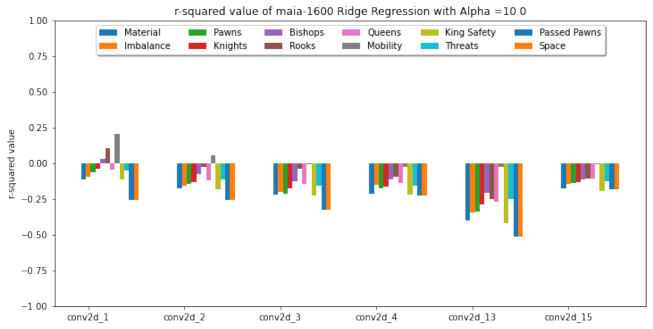


(d)

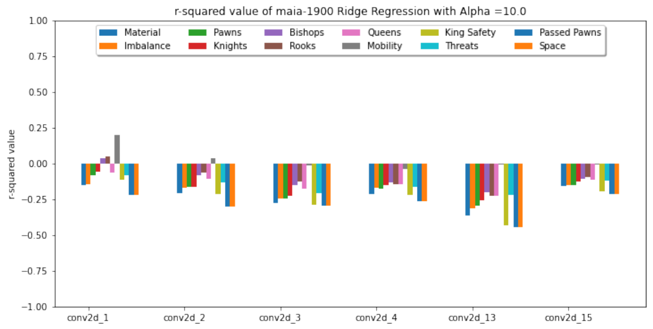
Fig. 5. Graph Showing the r-squared scores of the concepts examined across multiple versions of Maia.



(a)



(b)

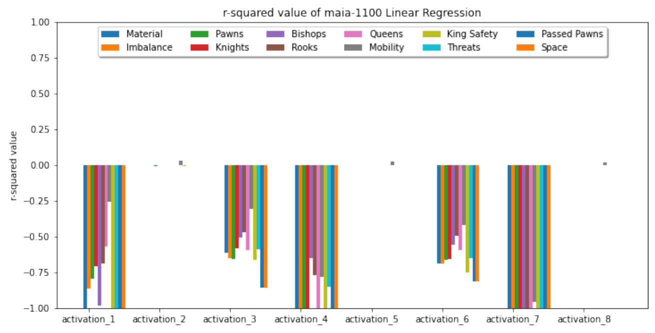


(c)

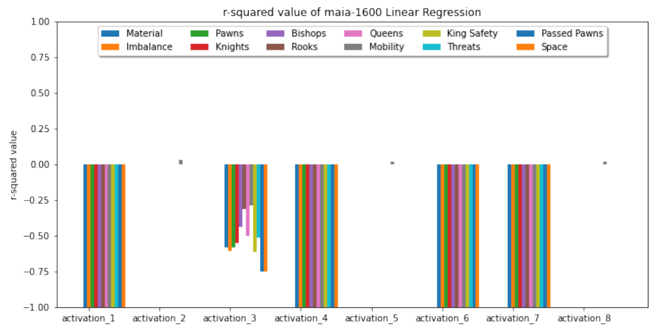
Fig. 6. Graph Showing the r-squared scores of the concepts examined across multiple versions of Maia for the initial and final convolution layers.

attempts to distinguish between the abilities of versions of Maia to predict the *precise scores* of the concepts examined. That is: can our linear probing of different versions of Maia predict not only if a concept is present or absent, but also the precise value of that concept? Our results here indicate that not enough information is being distilled by the Maia networks for linear probing to succeed at this more challenging task.

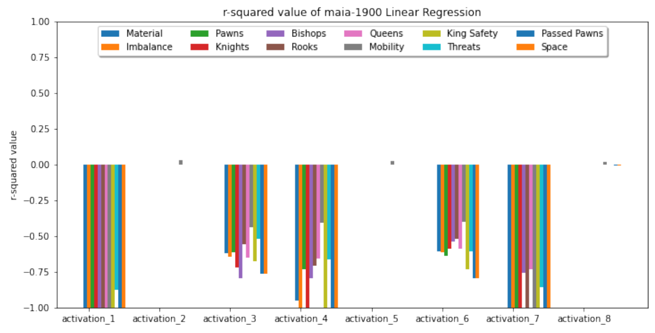
The more sophisticated concepts we examined here better reflect meaningful conceptual differences between various versions of Maia for the purpose of determining what, if anything, differentiates them in ways that human chess players are able to understand and draw conclusions from. Our initial belief was that the various Maia models, like a human player, would learn many of the concepts we examined. For example, it was intuitive to believe that the 1100 ELO version of Maia would have a much lower r-squared score for evaluating *king safety* compared to the 1900 ELO version of Maia. However,



(a)

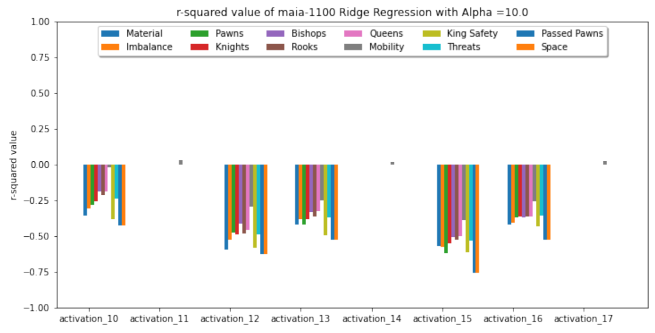


(b)

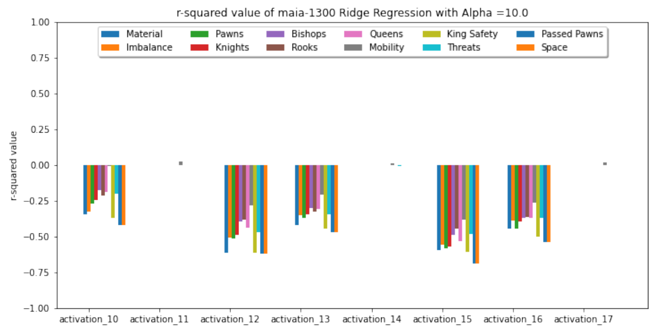


(c)

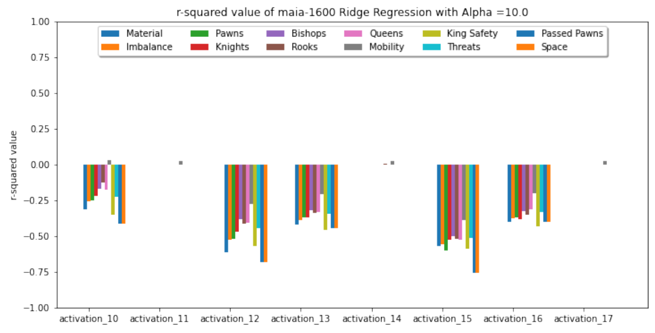
Fig. 7. Linear Regression.



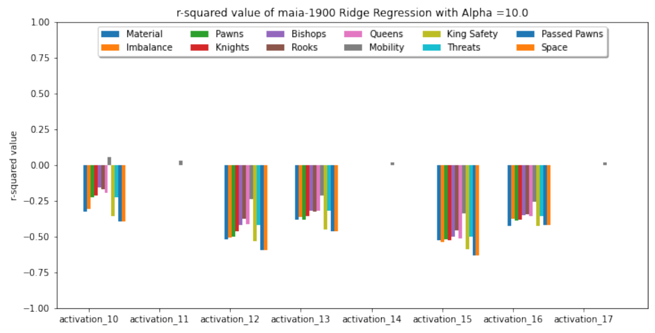
(a)



(b)



(c)



(d)

Fig. 8. r-squared scores of concepts examined across multiple versions of Maia, in the later activation layers.

our results indicate that this is not the case. Our regressed models were not able to accurately score above 0.3: which does not lend itself to the conclusion that this concept is being used at that point in the network.

As deeper layers of the network are examined, this conclusion does not change. At no point do these concepts become detectable to a significant degree. Further into the network it would seem likely that some of the more complex concepts would emerge, but our results indicate that this is not the case. The concepts that one would expect to emerge early and then get lost in the network, such as like Stockfish's *material score*, the more abstract concepts like king safety and scores related to the positional value of the other pieces do not improve further in the network. One notable concept is that of *piece mobility*. This concept has much higher scores than the others, but to be more confident in saying that our values are meaningful and that the networks are detecting and using mobility would require a score, in our estimation, above 0.50, which is not the case.

Instead the data indicates that the sophisticated concepts examined here are not detectable across any version of Maia inside the portions of the neural networks examined. While their play sufficiently differs from one another, the concepts which they use do not match up to the concepts we examined. Our results indicate that the way that Maia understands the game, across each version of itself, roughly match up similarly when compared to the concepts examined here in that the data necessary to detect the concepts in question was insufficiently present. The closest concept that was able to be detected with some measure of confidence is mobility.

We conclude that the Maia networks are not distilling representations with *sufficient direct informational content about our hand–crafted concepts*, to predict them on a real–valued scale via linear probing, from the feature vectors of the network. It is possible that a more sophisticated probing technique, such as using a neural network, may result in better predictability. Alternatively, a *larger* version of the Maia networks with more parameters may allow for the distillation (given sufficient training data) of a wider range of conceptual information, with the same result. Both of these possibilities suggest that the internal representations of Maia may more directly encode *other concepts*, perhaps unknown to human experts, or not considered fundamental/pedagogical/intuitive to the game by them. These concepts *may indirectly* predict or align with the concepts that *are* frequently described in chess literature, but requiring further neural information processing to uncover that alignment. In short: humans and Maia–like networks may be looking at chess through different conceptual "lenses", despite similar behavior in terms of playing similar moves (most of the time). This suggests the possibility that the concepts humans learn and understand may not be the most efficient representations from a predictability perspective, even solely to mimic human–like play. There may be underlying assumptions and patterns in our behavior that we do not explicitly formulate or understand, and our human concepts may reflect mere imperfect "shortcuts" or heuristics to achieving the desired outcome of a win, which a neural learning process would have no cause to explicitly learn to mimic.

Two contributing factors are proposed. The first is a question of the means by which Maia was trained. Instead of using MCTS like Lc0 or AlphaZero, the training for Maia was entirely based on human games and emulating human behavior. This may have resulted in a lower amount of exploration when compared to MCTS. The neural network would train itself to make moves a human would make in a given situation, but appears not be evaluating the position in the way the human who would make that move would. For example, material advantage would be foundational to a human's examination of the game but Maia does not seem to learn this concept.

Secondly, Maia's network architecture itself may also contribute to the low scores reported here. It may be a challenge for the model to separate the concepts linearly, due to the smaller size of the Maia network compared to Lc0. While Maia was sufficient for modeling human behavior with a smaller network than Lc0 [2], this smaller network may provide more challenges for our conceptual analysis here. However, this does lend evidence to the idea that a smaller neural network capable of producing human-like moves in a given situation may not be learning concepts that humans. Instead, like the example of AlphaGo before the MCTS and self–play, it is merely learning what moves a human would make, but not the cognitive processes that motivate such a choice of move, from a human perspective. The scope of our work did not include the construction of formal belief models of human and artificial agents playing the games, which are key to human decision processes. The next section outlines an approach for addressing this concern.

Maia is able to learn to play like a human being without internalizing human–like chess concepts. AlphaZero has been shown to use many of the concepts explored above [14], yet Maia does not. Our hypothesis that versions of Maia would learn these concepts to meaningful degrees at various levels of training was not validated in our results: the architecture chosen to sufficiently encode human–like play does not require the use and internalization of the hand crafted human–like concepts explored. Instead of a distribution of these concepts across Maia to varying degrees based on the ELO of the network, it is shown that these concepts were unnecessary for effectively emulating a human–like chess–playing agent.

The goal of the Maia networks was not necessarily to learn to play better chess. Instead, the goal was to make moves that approximate the skill level of human players across a range of skill levels. The data necessary to understand and evaluate a chess position in the way that humans expect, using something like Stockfish's classical evaluations is, according to our results, unnecessary. We hypothesize that the Maia networks are learning different concepts from the ones probed for in this work. In the next section, we discuss some avenues for investigating the truth of this hypothesis.

VII. Conclusion and Future Work

The various versions of Maia are designed to replicate moves humans would make in a similar situation, even going as far as to intentionally include blunders, but not necessarily to learn what is necessary to win. This is unlike Leela Chess and AlphaZero, where the goal is to create the best possible chess–playing agent. Therefore, the concepts that Maia must learn may be different from the concepts that the previously mentioned agents learned.

Future work may focus on what unique concepts Maia does use, to better differentiate it from the other networks discussed, and how Maia is able to achieve the goal of human–like moves without the same level of depth and filter size that the other networks use. Unsupervised techniques to detect concepts that Maia does use, and to which degree they are used, and to associate the before-unlabeled concepts to human concepts is another avenue for further work. These concepts detected through unsupervised means would need to be associated to human concepts through manual examination and may provide insight into what is more important to the inner workings of a human–like model.

Once such interpretability technique is a *saliency map* (or *pixel attributions*), which help indicate what part of the input (usually an image) contribute to the final classification of that image [26]. This technique would allow us to examine the factors that influence the decision of our network on a single example by examine basis. Generating a sufficient number of notable examples and attempting to interpret the decisions of Maia and possibly Leela and similar models, on the same examples, remains posited as future work.

A comparison featuring Maia–like networks with drastically different skill levels, such as 400 ELO vs. 2400 ELO, has the potential to provide additional insight, due to the sharp contrasts expected in concept acquisition. This might enable synthetic concepts to be constructed by a secondary model, that a 400 ELO model is guaranteed to not learn, but a 2400 ELO model is guaranteed *not* to learn. However, this study remains to be conducted, because of our original hypothesis that the versions of Maia that are already trained would show the sort of differences we expect.

Training deep neural networks of the depth of Maia from randomly initialized weights is a time consuming process, given the large datasets required for training, testing, and validation [2] of Maia, due to the associated memory requirements. Without the time and resources necessary to do so, alternative approaches are required, such as using pre–trained models. This is the approach we have taken, but as a necessary drawback, our work here is constrained by existing models. If we were training "from scratch" a version of Maia that precisely matches other similar chess playing networks, we could adjust our network architecture to maximize chances of concept retrieval. This is a general drawback of using pre–trained models, and the raison d'être of post–hoc explainability techniques, such as the linear probing we used in this work.

Differences in concept overlap and acquisition order may contribute to the stylistic differences attributed to self–playing systems. A study of optimal style characterization may advance the goal of effective concept and style *transfer* between humans and deepRL systems.

Our post–hoc approach enables concepts to be represented symbolically (as opposed to probabilistically), enabling formal specification of concept relationships and hierarchies, and computational inference of their deductive closure. Knowledge representation challenges for concepts and concept hierarchies warrant further research, as well as the associated deductive processes, as an important intermediate goal to concept-guided deepRL policy synthesis. It remains to study the use of formal logics, possibly "cognitive" modal logics (in the form of epistemic modal logics) to represent concepts leveraging arbitrary-order beliefs, with the objective of constraining or guiding deepRL systems with symbolic constraints that are either learnt post-hoc or inferred by automated reasoning. This may be seen as a neuro-symbolic approach to concept-guided deepRL. It may be necessary to design an efficient *description logic* reasoner for concept deduction in these modal logics.

Prior work has aimed to explain individual decisions in terms of acquired conceptual knowledge. Yet, in most normative domains, decisions are viewed as a sequence of steps to a goal, i.e., a plan [27]. The selective employment of conceptual knowledge in plan generation may be seen as embodying a strategy, subjectively requiring (computational) creativity. Conceptual knowledge may therefore be seen as informing and constraining strategic planning in sequential decision-making systems (such as deepRL systems). This motivates a comparison of how humans and machines exploit conceptual understanding in their decision making, with the goal of plan explainability. Humans and machines may differ in their *use* of acquired conceptual knowledge in plan formulation. Using the self-playing and human-games-trained models of Chess and Go games, it may be possible to design metrics to compare concept manifestation in sequential decision-making, based on quantitatively different plan outcomes in those games. Concept usage differences may enable explainability of human-generated plans in terms of machine-acquired concepts, and vice versa.

The scope of application of our work is not restricted to chess and other games played by AlphaZero-like systems, but is imminently relevant to a range of societal domains that may be modelled as a game between parties and requiring sequential decision-making, such as what has been called "nuclear chess", i.e., nuclear deterrence, and economic and trade strategies & policies.

Further, the ability to explain & interpret deepRL-acquired political and economic strategies in terms of the conceptual understanding of a deep learning model trained on historical data – representing actual strategies from human history – may be invaluable in understanding the ways in which deep reinforcement learning strategies carry the threat of violating human ethical norms, and in building a path to ensuring that these systems comply with them.

References

[1] A. Marchiafava and A. Sen, "Toward comparing knowledge acquisition in deeprl models," in *Proceedings of 2022 IARIA DIGITAL*, Special Track, 2022.

[2] R. McIlroy-Young, S. Sen, J. Kleinberg, and A. Anderson, "Aligning superhuman ai with human behavior: Chess as a model system," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20, Virtual Event, CA, USA: Association for Computing Machinery, 2020, pp. 1677–1687, ISBN: 9781450379984. DOI: 10.1145/3394486.3403219. [Online]. Available: https://doi.org/10.1145/3394486.3403219.

[3] *Introducing nnue evaluation*. [Online]. Available: https://stockfishchess.org/blog/2020/introducing-nnue-evaluation/.

[4] C. E. Shannon, "Xxii. programming a computer for playing chess," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 41, no. 314, pp. 256–275, 1950. DOI: 10.1080/14786445008521796. eprint: https://doi.org/10.1080/14786445008521796. [Online]. Available: https://doi.org/10.1080/14786445008521796.

[5] G. Kasparov, "Chess, a drosophila of reasoning," *Science*, vol. 362, no. 6419, pp. 1087–1087, 2018. DOI: 10.1126/science.aaw2221. eprint: https://www.science.org/doi/pdf/10.1126/science.aaw2221. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aaw2221.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer Series in Statistics). New York, NY, USA: Springer New York Inc., 2001.

[7] C. Molnar, *Interpretable Machine Learning, A Guide for Making Black Box Models Explainable*, 2nd ed. 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book.

[8] J. Schrittwieser, I. Antonoglou, T. Hubert, *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, pp. 604–609, 2019.

[9] D. Silver, T. Hubert, J. Schrittwieser, *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018. [Online]. Available: http://science.sciencemag.org/content/362/6419/1140/tab-pdf.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[11] D. Silver, A. Huang, C. Maddison, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, Jan. 2016. DOI: 10.1038/nature16961.

[12] R. Coulom, "Efficient selectivity and backup operators in monte-carlo tree search.," in *Computers and Games*, H. J. van den Herik, P. Ciancarini, and H. H. L. M. Donkers, Eds., ser. Lecture Notes in Computer Science, vol. 4630, Springer, 2006, pp. 72–83, ISBN: 978-3-540-75537-1.

[13] Pascutto, Gian-Carlo and Linscott, Gary, *Leela chess zero*. [Online]. Available: http://lczero.org/.

[14] T. McGrath, A. Kapishnikov, N. Tomašev, *et al.*, "Acquisition of chess knowledge in alphazero," *Proceedings of the National Academy of Sciences*, vol. 119, no. 47, e2206625119, 2022. DOI: 10.1073/pnas.2206625119. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.2206625119. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.2206625119.

[15] A. Pálsson and Y. Björnsson, "Unveiling concepts learned by a world-class chess-playing agent," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, E. Elkind, Ed., Main Track, International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 4864–4872. DOI: 10.24963/ijcai.2023/541. [Online]. Available: https://doi.org/10.24963/ijcai.2023/541.

[16] P. Hammersborg and I. Strümke, *Reinforcement learning in an adaptable chess environment for detecting human-understandable concepts*, 2022. arXiv: 2211.05500 [cs.LG].

[17] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the gdpr," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.

[18] P. Paatero and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values," *Environmetrics*, vol. 5, no. 2, pp. 111–126, 1994. DOI: https://doi.org/10.1002/env.3170050203. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/env.3170050203.

[19] D. Noever, M. Ciolino, and J. Kalin, "The chess transformer: Mastering play using generative language models," *CoRR*, vol. abs/2008.04057, 2020. arXiv: 2008.04057. [Online]. Available: https://arxiv.org/abs/2008.04057.

[20] H. Jhamtani, V. Gangal, E. Hovy, G. Neubig, and T. Berg-Kirkpatrick, "Learning to generate move-by-move commentary for chess games from large-scale social forum data," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 1661–1671. DOI: 10.18653/v1/P18-1154. [Online]. Available: https://aclanthology.org/P18-1154.

[21] Y. Nasu, "Efficiently updatable neural-network-based evaluation functions for computer shogi," *The 28th World Computer Shogi Championship Appeal Document*, vol. 185, 2018.

[22] D. Uranga, *Trainingdata-tool: A tool for lc0 training data operations*. [Online]. Available: https://github.com/DanielUranga/trainingdata-tool.

[23] D. Barnes, *Pgn-extract:a portable game notation (pgn) manipulator for chess games*. [Online]. Available: https://www.cs.kent.ac.uk/people/staff/djb/pgn-extract/.

[24] B. Kim, M. Wattenberg, J. Gilmer, *et al.*, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav).," in *ICML*, J. G. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 2673–2682. [Online]. Available: http://dblp.uni-trier.de/db/conf/icml/icml2018.html#KimWGCWVS18.

[25] G. Alain and Y. Bengio, *Understanding intermediate layers using linear classifier probes*, 2018. arXiv: 1610.01644 [stat.ML].

[26] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image clas-

sification models and saliency maps," in *Workshop at International Conference on Learning Representations*, 2014.

[27] S. Sreedharan, U. Soni, M. Verma, S. Srivastava, and S. Kambhampati, *Bridging the gap: Providing post-hoc symbolic explanations for sequential decision-making problems with inscrutable representations*, Mar. 19, 2022. DOI: 10.48550/arXiv.2002.01080. arXiv: 2002.01080[cs]. [Online]. Available: http://arxiv.org/abs/2002.01080 (visited on 06/11/2023).