Investigation of OthelloGPT World Model Learning using Mechanistic Interpretability

Untersuchung des World Model Learning von OthelloGPT mittels Mechanistic Interpretability

Jim Maar

Universitätsbachelorarbeit
zur Erlangung des akademischen Grades

Bachelor of Science
(B. Sc.)

im Studiengang
IT Systems Engineering

eingereicht am 20. October 2024 am
Fachgebiet Algorithm Engineering der
Digital-Engineering-Fakultät
der Universität Potsdam

Gutachter	Prof. Dr. Gerard De Melo
Betreuer	Prof. Dr. Gerard De Melo

Abstract

Transformer models are increasingly employed across various tasks, yet their internal computational mechanisms remain largely opaque. Recent work on OthelloGPT, a toy language model trained to produce legal moves in the board game Othello, has revealed insights into the behavior of transformer models, such as that they can have a world model. This thesis is an attempt at explaining how OthelloGPT computes its world model. We identify a Previous Color Circuit that reoccurs in multiple layers and appears to play a central role in the computation of the world model. Additionally, we propose and evaluate a Flipping Circuit Hypothesis, which suggests the use of neurons to implement game-specific tile-flipping rules, though our findings do not support this mechanism. We describe our process for developing and testing these circuit hypotheses through detailed analysis of the model's internal representations.

Zusammenfassung

Transformer-Modelle werden zunehmend in verschiedenen Aufgabenbereichen eingesetzt, doch ihre innere Funktionsweise bleibt weitgehend undurchsichtig. Jüngste Arbeiten an OthelloGPT, einem Sprachmodell, das darauf trainiert wurde, legale Züge im Brettspiel Othello zu produzieren, haben Einblicke in das Verhalten von Transformer-Modellen geliefert, etwa dass sie über ein Weltmodell verfügen können. Diese Arbeit unternimmt den Versuch zu erklären, wie OthelloGPT sein Weltmodell berechnet. Wir identifizieren einen Previous-Color-Circuit, der in mehreren Layern wiederkehrt und eine zentrale Rolle bei der Berechnung des Weltmodells zu spielen scheint. Zusätzlich schlagen wir die Flipping-Circuit-Hypothese vor und evaluieren diese. Diese Hypothese deutet auf die Verwendung von Neuronen zur Implementierung spielspezifischer Regeln zum Umdrehen von Spielsteinen hin, wobei unsere Ergebnisse diesen Mechanismus nicht bestätigen. Wir beschreiben unseren Prozess der Entwicklung und Überprüfung dieser Circuit-Hypothesen durch eine detaillierte Analyse der internen Repräsentationen des Modells.

Acknowledgments

I would like to express my sincere gratitude to Professor Dr. Gerard de Melo for his valuable feedback throughout this thesis work. I am particularly thankful for the academic freedom he granted me in selecting my research topic.

Contents

1 Introduction

The remarkable advances in Natural Language Processing (NLP) achieved through Transformer models have revolutionized the field of artificial intelligence [Bub+23]. However, despite their impressive performance, these models remain largely opaque in terms of their internal operations, often being characterized as "black boxes." Mechanistic Interpretability has emerged as a promising approach to address this challenge, offering methods to "understand the internal reasoning processes of trained neural networks and gain insight into how and why they produce their outputs" [Has].

Current research in Mechanistic Interpretability has primarily concentrated on analyzing attention layer circuits [Elh+21; Wan+22], while our understanding of Multi-Layer Perceptron (MLP) components in Transformer architectures remains limited. Two main approaches have emerged in this field:

1. **Targeted Investigation:** Addressing specific questions about language models through focused analysis [Ols+22; Wan+22]

2. **Proxy Model Analysis:** Studying simplified models that serve as effective proxies for Large Language Models (LLMs) [Elh+21]

This research adopts the second approach, building upon the work of Li et al. [Li+24a], who developed OthelloGPT—a Transformer model trained to predict legal moves in the board game Othello based on previous moves. Their research demonstrated the model's ability to track board states using nonlinear probes. Subsequent work by Nanda et al. [NLW23] revealed that these world model representations are encoded linearly (see Chapter 5). They also show that the model tracks which tiles are flipped and how the model computes, which tiles are empty. While these findings represent significant progress, a comprehensive understanding of the model's board state calculations remains elusive.

The implementation of Othello boardstate computation in transformer models presents a significant challenge. While a straightforward algorithm would require at least 60 sequential operations to copy the previous boardstate and execute piece placement and flips, OthelloGPT's 8-layer architecture suggests it must employ a more sophisticated approach.

Previous work in mechanistic interpretability has identified circuits (subgraphs within neural networks) that span multiple layers [Cam+20; Ols+22; Wan+22]. These circuits explain certain behaviors, such as simple cases of out-of-context learning, in an end-to-end manner by taking tokens as input and producing logits as output. In our investigation of OthelloGPT, we discovered that the attention layers in the early stages of the network copy the color of previous moves. We term this the Previous Color Circuit. Unlike previously discovered circuits that span multiple layers, this circuit operates within a single layer, making it a shallow" or mini" circuit. Furthermore, it differs from previous work by operating on high-level features (such as D3 is flipped") rather than raw inputs and outputs, leading us to characterize it as a "modular circuit."

This thesis is structured as follows: We begin with an overview of the transformer architecture, the game of Othello, the OthelloGPT model, and previous research conducted on OthelloGPT. We then present our initial investigations into the model's behavior, including the training and analysis of multiple linear probes. Subsequently, we analyze the behavior of early attention layers and provide an explanation for how these layers compute tile colors through the previous color circuit, supported by empirical evidence. We then examine the behavior of early MLP layers, identifying a group of neurons hypothesized to be responsible for updating tile colors according to Othello's rules—what we term the Flipping Circuit Hypothesis. However, our investigation ultimately couldn't provide conclusive evidence.

We conclude by presenting a refined understanding of how OthelloGPT computes its board state representation and discuss directions for future research.

Part I

Preliminaries

2 Othello and OthelloGPT

2.1 Othello

Othello is a strategic board game played between two players on an 8 × 8 grid. The game proceeds through alternating turns, where players place tiles of their respective colors on the board. A valid move must satisfy the following condition: the newly placed tile must create at least one straight line (horizontal, vertical, or diagonal) where opponent's tiles are completely enclosed between two tiles of the active player's color. Upon placement, all enclosed opponent tiles are converted to the active player's color. The game continues until the board is completely filled, and victory is achieved by the player with the majority of tiles in their color. Figure Figure 2.1 illustrates the initial configuration of an Othello game.

2.2 OthelloGPT Architecture

OthelloGPT is an 8-layer decoder-only transformer model trained on next-token prediction using a dataset of randomly generated legal Othello moves. The model's architecture is characterized by the following specifications:

- Input vocabulary: 60 tokens (representing each playable square, e.g., B4)
- Hidden dimension: 512

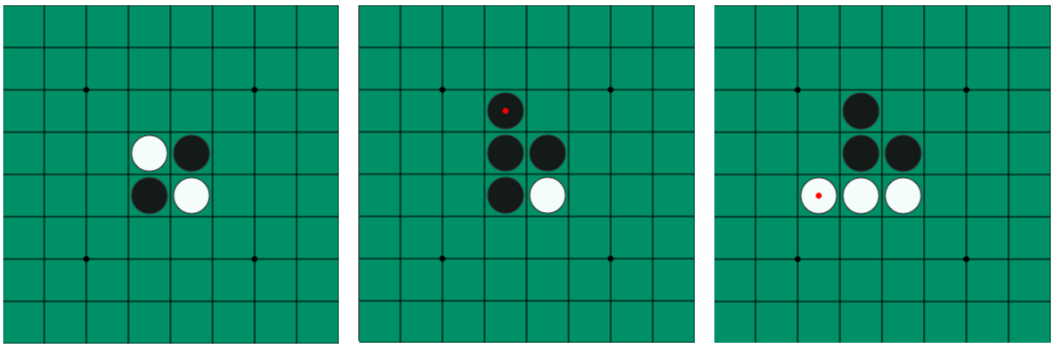


Figure 2.1: Initial Configuration of an Othello Game Board

- Attention heads per layer: 8
- MLP layer size: 2048 neurons
- Activation function: GELU [HG23]
- Training features: Attention and residual dropout

Since the training data consists of uniformly random legal moves, the model learns to generate approximately uniform positive logits for legal moves while assigning negative logits to illegal moves.

2.3 Research Significance

OthelloGPT's primary purpose is not to serve as a competitive Othello player but rather as a controlled environment for interpretability research [Li+24b]. The model employs the same architectural principles as modern Large Language Models (LLMs) while maintaining a more tractable scope for analysis. This makes it an ideal candidate for studying neural network behavior and developing interpretability techniques.

The potential to fully reverse engineer OthelloGPT's legal move computation mechanism could provide valuable insights into LLM operation and serve as a benchmark for automated interpretability methods, an area of active research [Con+23; Hsu+24].

3 The Transformer Architecture and Notation

We describe the architecture of the Transformer model. To facilitate understanding of the model's inner workings, we adopt an intuitive notation style inspired by Sanderson [San24], rather than using the most compact or implementation-focused notation. Following Elhage et al. [Elh+21], we introduce intermediate variables that enhance interpretability. The key components are defined as follows:

3.1 Notation Glossary

- $D = 512$: Dimension of the residual stream
- $d = 64$: Dimension within each attention head
- $m = 2048$: Number of neurons in the MLP layer
- $T = 60$: Number of tokens, each representing a playable tile[1]
- $C = 60$: Context size, indicating each tile can be played exactly once
- $W_E \in \mathbb{R}^{D \times T}$: Embedding matrix
- $t \in \{1, \ldots, T\}_{\text{one-hot encoded}}$: Input tokens in one-hot encoded format
- $x^l \in \mathbb{R}^{D \times C}$: Residual stream at layer l
- $x_i \in \mathbb{R}^D$: Residual stream at sequence position i
- $x_i^l \in \mathbb{R}^D$: Residual stream at layer l in sequence position i
- H_i: Set of attention heads in layer i
- $b_i \in \mathbb{R}^D$: Bias term after the attention layer
- MLP_i: MLP layer in layer i

1 While a special token exists for "passing" when no valid moves are available, our dataset contains no games with passing moves

3.2 Transformer Layer

Each Transformer layer i consists of a self-attention layer and a multilayer perceptron (MLP) layer. These layers modify the residual stream by adding their outputs to it:

$$\mathbf{x_0} = W_E t$$

$$\mathbf{x'_{i-1}} = \text{LayerNorm}(\mathbf{x_{i-1}})$$

$$\mathbf{x}_{\text{i\_mid}} = x'_{i-1} + \text{Att}_i(x'_{i-1})$$

$$\mathbf{x'}_{\text{i\_mid}} = \text{LayerNorm}(\mathbf{x}_{\text{i\_mid}})$$

$$x_i = \text{MLP}_i(x'_{\text{i-1\_mid}})$$

The final Transformer output is given by:

$$T(t) = W_U x_{-1}$$

3.3 Layer Normalization

Layer normalization is defined as:[2]

$$\text{LayerNorm}(x_i) = \frac{x_i - \mathbb{E}(x_i)}{\sqrt{\mathbb{V}(x_i) + \epsilon}}$$

3.4 Self-Attention Operation

The core attention operation is defined as:

$$\text{Att}_i(x'_{i-1}) = \sum_{h \in H_i} h(x'_{i-1}) + b_i$$

2 We use the OthelloGPT model from the Transformer Lens Library [NB22] with folded-in weights, where certain weights and bias terms are modified to incorporate the LayerNorm parameters, simplifying some computations

Each attention operation utilizes four key matrices:

- $h.W_Q \in \mathbb{R}^{D \times d}$: Query matrix
- $h.W_K \in \mathbb{R}^{D \times d}$: Key matrix
- $h.W_V \in \mathbb{R}^{D \times d}$: Value matrix
- $h.W_O \in \mathbb{R}^{D \times d}$: Output matrix
- M: Lower triangular matrix of $-\infty$

For an attention head h, the attention pattern $h.A$ is computed as:

$$h.A = \text{softmax}(\mathbf{x}^\top h.W_Q h.W_K^\top \mathbf{x} + M)$$

where $A_{i,j}$ represents the attention paid by token i to token j.
The output of an attention head $h(x)_i$ is:

$$h(x)_i = \sum_{j \leq i} A_{i,j} \cdot (h.W_O h.W_V x_j)$$

In the mechanistic interpretability literature, the operation $x_i^\top W_Q W_K^\top x_i$ is known as the QK Circuit, determining when information is moved between positions. Similarly, $W_O W_V x_j$ is termed the OV circuit, describing what information is read from position j and how it is written to the attending position [Elh+21].

3.5 MLP Layer

Each layer i contains an MLP with the following parameters:

- $W_{\text{in}} \in \mathbb{R}^{m \times D}$: Input weight matrix
- $b_{\text{in}} \in \mathbb{R}^m$: Input bias vector
- $W_{\text{out}} \in \mathbb{R}^{D \times m}$: Output weight matrix
- $b_{\text{out}} \in \mathbb{R}^m$: Output bias vector

The MLP operation is computed as:

$$\text{MLP}(x_i) = b_{\text{out}} + W_{\text{out}} \cdot \text{GELU}(b_{in} + W_{\text{in}} \cdot x_i)$$

This can be rewritten in terms of individual neurons:

$$a_{i,j} = \text{GELU}(b_{\text{in}\,j} + W_{\text{in}}[:, j] \cdot x_i)$$
$$\text{MLP}(x_i) = b_{\text{out}} + \sum_{j=0}^{m} W_{\text{out}}[j, :] \cdot a_{i,j}$$

where:

- GELU: GELU nonlinear activation function
- $W_{\text{in}}[j, :] \in \mathbb{R}^D$: Input weights of neuron j (j'th row of W_{in})
- $b_{\text{in}\,j} \in \mathbb{R}$: Bias of neuron j
- $W_{\text{out}}[:, j] \in \mathbb{R}^D$: Output weights of neuron j (j'th column of W_{out})
- $a_{i,j} = \text{GELU}(b_j + W_{\text{in}}[:, j] \cdot x_i) \in \mathbb{R}$: Activation of neuron j at sequence position i
- $a_i \in \mathbb{R}^m$: All neuron activations in a layer at position i

4 Mechanistic Interpretability

In this Chapter we introduce key concepts in mechanistic interpretability and establish precise terminology for our discussion.

4.1 Mechanistic Interpretability Glossary

The current paradigm in mechanistic interpretability research approaches the understanding of Neural Networks through the lens of features and circuits [BG24].

- **Feature:** "The fundamental units of how neural networks encode knowledge, which cannot be further decomposed into smaller, distinct concepts" [BG24]
- **Circuit:** "Sub-graphs within neural networks consisting of features and the weights connecting them. Circuits can be thought of as computational primitives that perform understandable operations to produce (ideally interpretable) features from prior (ideally interpretable) features" [BG24]

The Linearity Hypothesis [Elh+22] posits that features correspond to directions in space.

4.2 Terminology Framework

To enhance clarity and precision, we define the following terms:

- **Feature Name:** The semantic label assigned to a feature (e.g., "C3 is Empty")
- **Feature Direction:** The corresponding vector direction in the network's representational space
- **Feature Variable:** A boolean variable indicating feature presence in a given residual stream x_i. This is true when the residual stream has a substantial dot-product with the Feature Direction: $F \cdot x_i \gg 0$
- **Feature Logit:** The dot product of the feature direction with the current residual stream at a specific position: $F \cdot x_i$

Part II

Findings

5 Training Linear Probes

In this chapter, we explore linear probes and demonstrate their utility in revealing OthelloGPT's internal world model. Furthermore, we conduct additional linear probe experiments to test hypotheses about the model's board state computation mechanisms.

5.1 Previous Work

Nanda et al. [NLW23] demonstrated that OthelloGPT maintains linear representations of board states through linear probes. Their methodology employs a linear probe p defined as a linear projection from the residual stream:

$$p_l(x_i^l) = \text{Softmax}(Wx_i^l), W \in \mathbb{R}^{D\times 3}$$

For each tile position, a probe is trained to classify its state as EMPTY, MINE, or YOURS, relative to the current player's turn.

To illustrate this concept, consider square D3 containing a black tile across multiple moves. During move 1 (white player's turn), D3 would be classified as YOURS, while in move 2 (black player's turn), it would be classified as MINE. The linear probe can be conceptualized as three features corresponding to three directions in D-dimensional space ($W[:, 0]$, $W[:, 1]$, and $W[:, 2]$). The presence of a feature (e.g., "D3 is Yours") is indicated when the dot product between the residual stream and the corresponding feature vector is significantly positive (e.g., $x_i \cdot W[:, 2] \gg 0$).

The authors observed consistent probe accuracy improvements across layers, ranging from 90.9% in layer 0 to 99.6% in layer 6. They successfully utilized these trained linear probes for model intervention, such as modifying tile colors in the board state representation, which produced corresponding changes in the model's legal move predictions. This validates the causal relationship between the discovered internal representations and the model's behavior. Additionally, they achieved comparable success in training linear probes to identify flipped tiles, using a two-way classification probe $p_f(x_i^l) = \text{Softmax}(Wx_i^l), W \in \mathbb{R}^{D\times 2}$. This framework can be extended to probe any feature we hypothesize the model might represent.

5.2 Experimental Setup

We train various linear probes to minimize the cross-entropy loss between predictions and target labels. Our training setup consists of 300,000 input sequences (games) using the Adam optimizer [Kin14] with the following hyperparameters:

- Learning rate: 0.01
- Batch size: 256
- Weight decay: 0.1
- β_1: 0.9, β_2: 0.99

We evaluate the probes on a validation set of 10,000 games, using F1-score for classifications with sparse positive labels (e.g., flipped tiles or legal tiles) and accuracy otherwise.

5.3 Our Probes

Our reproduction of the probes in Nanda et. al. [NLW23] yielded similar results, as demonstrated in Table 5.1 and Table 5.2.

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Mine/Yours/Empty	90.8	94.7	97.1	98.2	98.9	99.3	99.4	99.3

Table 5.1: Accuracy of Linear Board State Probe Across Different Layers.

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Flipped	87.0	92.6	95.7	97.3	98.2	98.5	98.4	98.1

Table 5.2: F1-score of Flipped Probe Across Different Layers.

The Flipped Probe results suggest a potential mechanism for the model's board state computation:

Hypothesis 1. *The model maintains both the initial color played on each tile and the number of subsequent flips, enabling straightforward computation of current tile colors.*

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Num Flipped	14.5	15.1	15.2	15.4	15.5	16.0	16.1	16.0
Num Flipped Even/Odd	71.4	72.7	74.2	74.8	75.2	76.1	77.4	80.3
First Tile Mine/Yours	82.5	80.5	81.7	79.5	80.1	79.7	80.1	79.6

Table 5.3: Evaluation of probes trained to test Hypothesis 1. We trained a probe to classify how often a tile has been flipped (Num Flipped), a probe to classify if a tile has been flipped an even or odd number of times (Num Flipped Even/Odd) and a probe to classify if a tile was first placed my the current player or the other player (First Tile Mine/Yours). We use F1-score for the first probe and accuracy for the other two

To test this hypothesis, we implemented linear probes to track both flip counts and initial tile colors. However, as shown in Table 5.3, the results failed to support Hypothesis 1.

While He et. al. [He+24] identified features of the form "Tile ... Placed, Flipping Tile ... next to it" using Sparse Autoencoders on a smaller OthelloGPT model, our linear probes failed to detect similar patterns in the original OthelloGPT model (Table 5.4).

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Placed And Flipped	21.3	17.5	14.0	11.9	10.1	08.5	08.1	06.5

Table 5.4: F1-score of the placed and flipped probe, which classifies if a tile has been placed, flipping tiles in a certain direction. This is a multi-label classification setup, which can be seen as 8 binary classifications

Supporting the findings of Lin et. al. [jyl+], we confirmed that the model first identifies accessible tiles (adjacent to non-empty tiles) before determining legal moves (Table 5.5).

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Accessible	99.4	99.5	99.7	99.9	99.9	99.3	97.4	96.8
Legal	86.6	89.7	91.14	92.0	95.0	98.4	98.5	98.3

Table 5.5: F1-score of the Accessible Probe and Legal Probe

For subsequent analysis, we successfully trained probes to detect which tile has been placed. (This is not very surprising because the information is already contained in the token embedding) (Table 5.6).

Probe Name	L0	L1	L2	L3	L4	L5	L6	L7
Placed	100	100	100	100	100	100	99.9	100

Table 5.6: F1-score of the Placed Probe

5.4 Visualization and Outlook

Figure 5.1 illustrates the board state probe accuracy (Mine/Yours/Empty) across layers and sequence positions. Each layer achieves near-perfect board representation up to a position n_l, after which accuracy diminishes. This suggests that each layer might rely on the accuracy of the board state representation in the last few positions of the previous layer (see Chapter 6 for further explanation).

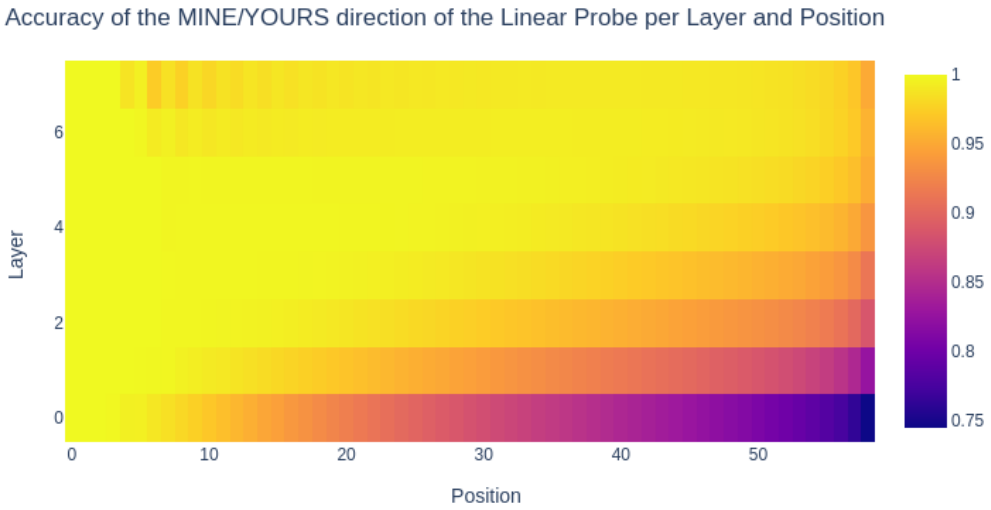


Figure 5.1: Accuracy of the Linear Probe Across Layers and Sequence Positions

Our analysis has yielded a comprehensive set of features for each tile: "tile is empty," "tile is mine," "tile is yours," "tile is flipped," "tile is placed," "tile is accessible," and "tile is legal." By applying these probes to the residual stream, we can determine active and inactive features, collectively forming OthelloGPT's world model.

6 The Attention Layer

In this chapter, we employ mechanistic interpretability and visualizations to analyze the attention layers of OthelloGPT. We demonstrate that OthelloGPT uses the Previous Color Circuit to compute the color of each tile in the board state, based on the color the tile had in previous moves. We conclude by explaining how this mechanism accounts for a substantial part of OthelloGPT's board state computations.

6.1 Basic Observations

Hazineh et al. [HZC23], in their investigation of OthelloGPT, categorize the attention heads into five groups based on their typical attention patterns:

- **Last Heads**: Primarily attend to the last sequence position.
- **First Heads**: Primarily attend to the first sequence position.
- **Mine Heads**: Focus on sequence positions that are an even number of steps away, thereby attending to prior instances when the current player took their turn.
- **Yours Heads**: Focus on sequence positions that are an odd number of steps away, attending to prior instances when the opposing player took their turn.
- **Other**

Figure 6.1 shows the distribution of Mine Heads and Yours Heads across different layers.

Examining the Mine/Yours Heads, they seem to pay less attention to moves further away, as illustrated in Figure 6.2.

We compute the empirical mean and variance of all non-zero entries in the attention patterns of each attention head, derived from 200 input sequences. The attention patterns exhibit an average empirical variance of 0.0007, suggesting that they are mostly influenced by the sequence position. This observation makes understanding the OV-Circuit more relevant than understanding the QK-Circuit.

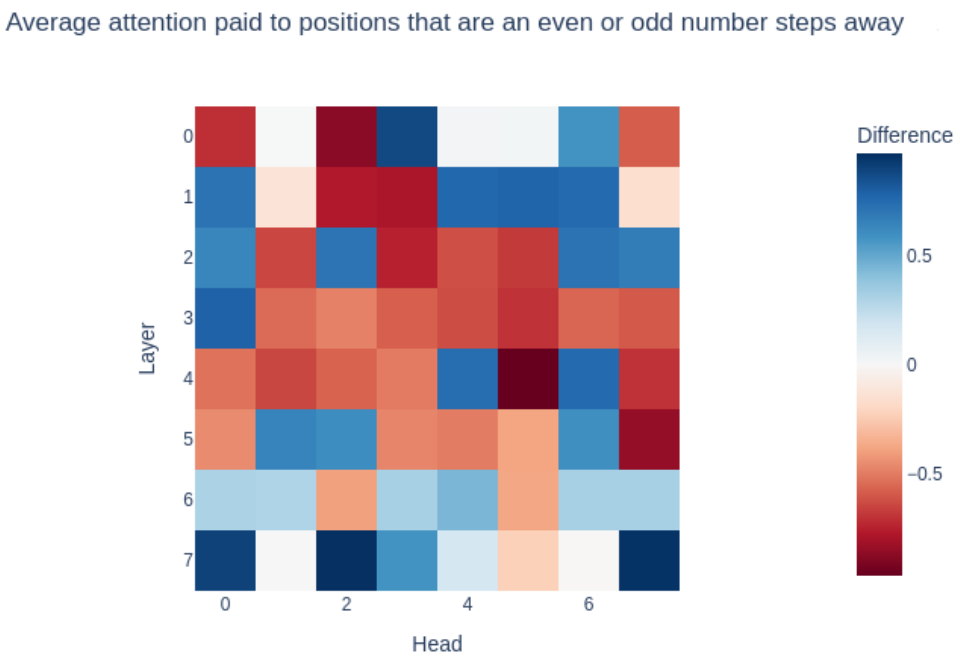


Figure 6.1: Average attention paid to positions an even number of steps away minus attention paid to positions an odd number of steps away, for each Attention Head and layer. Mine Heads are seen in blue, and Yours Heads in read. "Last," "First," and other types of heads are also visible. L4H5, for example, is a "Last" Head.

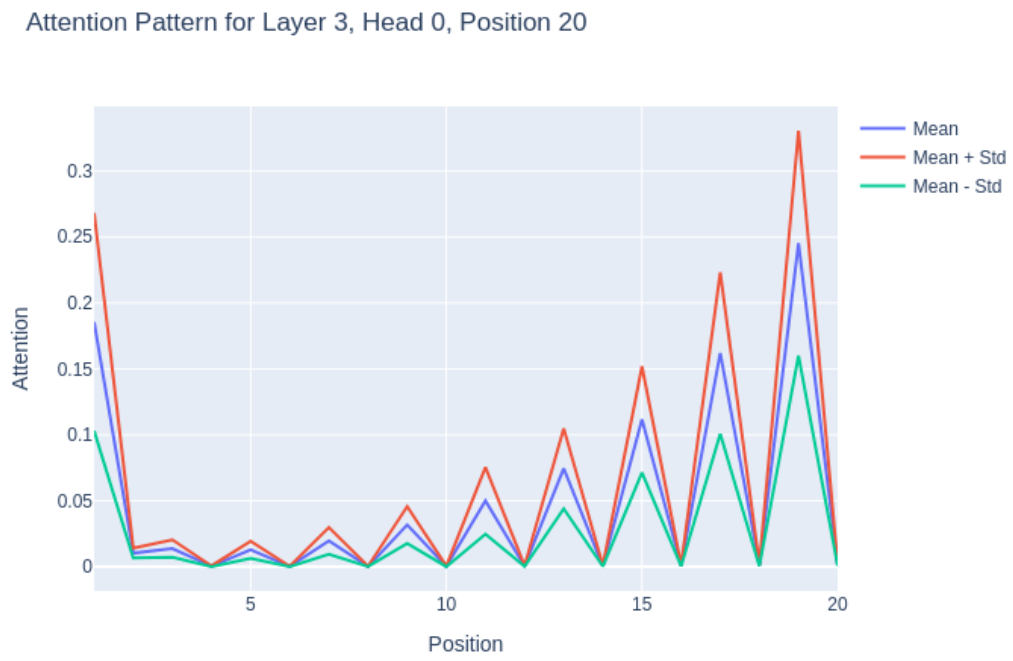


Figure 6.2: Attention Pattern for Layer 3, Head 0, Position 20

6.2 Visualization of the Iterative Refinement of the Board State

To better understand how the model computes the board state, we created a visualization of the board state representation across the entire input sequence and all layers, as shown in Figure 6.3.

The model seems to build up the correct board state incrementally, refining information across layers. For example, in Figure 6.3, at sequence position 19 in layer 1, before the Attn layer, the tile D3 is marked as Red/Yours in the board representation. After the Attn layer, the color updates to Blue/Mine, possibly because the tile flipped to Blue at sequence position 16, which was encoded in the board state representation after layer 0.

6.3 Direct Logit Attribution

Direct Logit Attribution is an interpretability technique for identifying which parts of a model contribute to specific behaviors. We apply this method to confirm our hypothesis.

Let $\mu : \mathbb{R}^D \to \mathbb{R}$ be the feature direction for "D3 is mine". To determine which part of the model encoded this feature in the residual stream, we decompose the residual stream into contributions from different layers.

$$\mu(x_{19}^{1\_\text{mid}}) = \mu(x_{19}^{0}) + \mu(\text{Att}_1(x^{0'}))$$

The contributions of the previous residual stream and the attention layer are 0.21 and 1.78, respectively. We further decompose the contributions from the attention layer into those from each head and sequence position, as shown in Figure 6.4. We observe that the largest contribution originates from position 16 through the Yours head L1H3, aligning with our expectations.

6.4 The Previous Color Circuit

Examining examples like this leads to a hypothesis regarding the role of the attention layer in the board state computation.

Hypothesis 2. *The color of each tile written by the attention layer depends on the color of the tile in previous sequence positions, with recent moves carrying more weight, as well as the color of the tile when it was last flipped.*

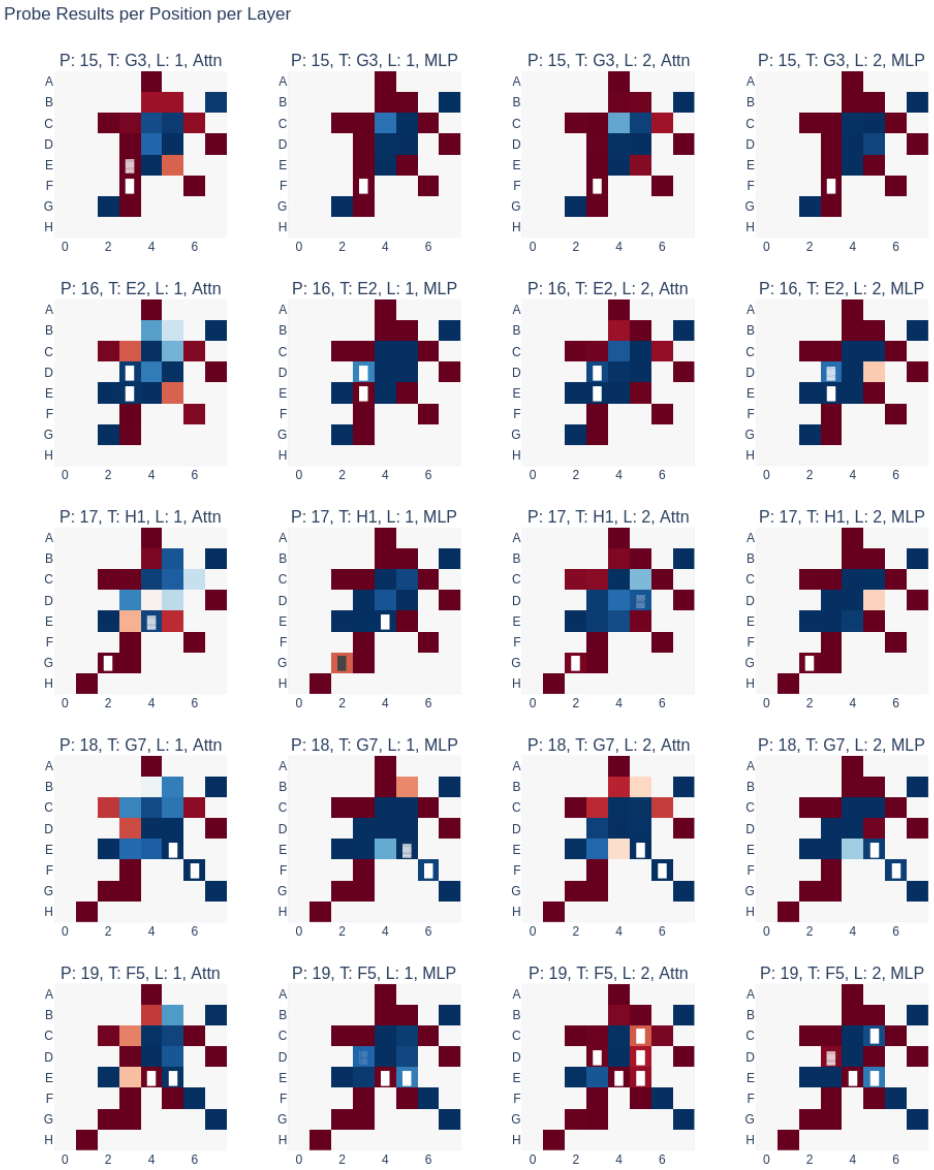


Figure 6.3: Example interface for analyzing board state computation across layers and positions. "Attn/MLP" indicates the next transformer layer. Tiles use a blue/red color scale, with blue representing black and red representing white. Tiles flipped in the board representation are marked with a white/black rectangle.

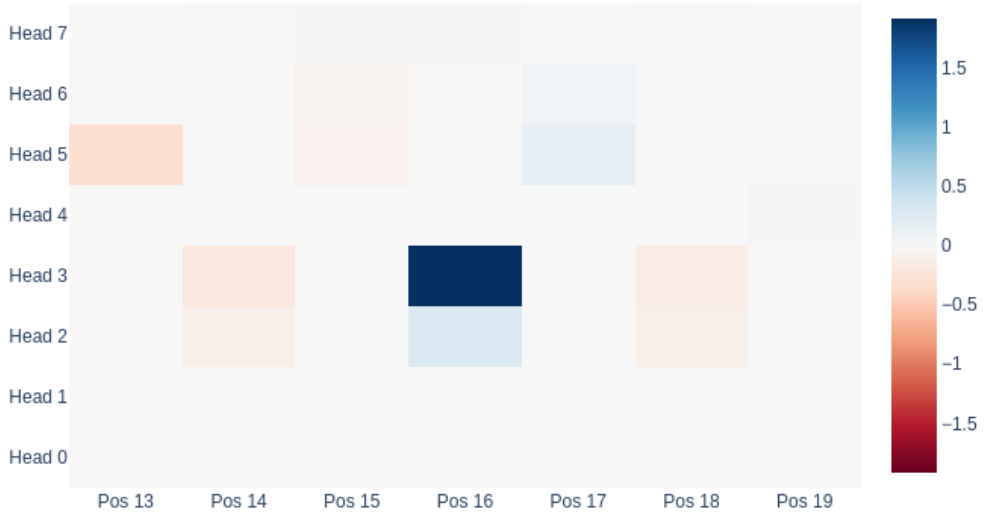


Figure 6.4: Direct logit attribution to "D3 is mine" at layer 1, position 19 for each attention head and sequence position.

Specifically, we propose that the attention layer functions as shown in Table 6.1, with Mine Heads and Yours Heads writing the correct color for each tile.

Mine Heads		Yours Heads	
reads	**writes**	**reads**	**writes**
tile is mine	tile is mine	tile is mine	tile is yours tile is not flipped
tile is yours	tile is yours tile is not flipped	tile is yours	tile is mine
tile is flipped	tile is yours tile is not flipped	tile is flipped	tile is mine

Table 6.1: Overview of the actions performed by the OV circuits of attention heads involved in the previous color circuit.

6.5 The OV Circuit

We analyze the OV circuit to validate whether the Attention Layers exhibit behavior consistent with the patterns described in Table 6.1.

6.5.1 Experimental Setup

Let us define the following feature directions for some tile in layer i:

- μ_i: "tile is flipped"
- $\overline{\mu_i}$: "tile is not flipped"
- ω_i: "tile is placed"
- ϕ_i: "tile is yours"
- ψ_i: "tile is mine"

For a given attention head in layer i, let W_O and W_V denote its output and value matrices, respectively. We define $\mu^*_{i-1} = (W_O W_V \mu_{i-1})$, which is the direction written to the residual stream when a tile is flipped at a previous position.

The attention heads behavior can be characterized as follows:

- It writes "tile is yours" when $\textbf{cosine\_similarity}(\mu^*_{i-1}, \phi_i - \psi_i) \gg 0$

- It writes "tile is mine" when **cosine_similarity**$(\mu^*_{i-1}, \phi_i - \psi_i) \ll 0$

We compute the average cosine similarity for each input feature $(\mu_i, \omega_i, \phi_i, \psi_i)$ across all tiles from layers 1 to 6, analyzing Mine Heads and Yours Heads separately. Additionally, we evaluate **cosine_similarity**$(\mu^*_{i-1}, \mu_i - \overline{\mu_i})$ to examine the effect on the "tile is flipped" versus "tile is not flipped" direction. This is particularly relevant as tiles can only be flipped from "mine" to "yours," not vice versa, suggesting that when "tile is yours" is written, "tile is not flipped" should also be written.

6.5.2 Results

The results presented in Table 6.2 demonstrate that the "flipped" feature exhibits the strongest effect. The overall patterns largely align with Table 6.1, although some discrepancies exist in the effects on the "tile is flipped" versus "tile is not flipped" direction.

head_kind	feature_in	yours vs mine	flipped vs not flipped
mine	flipped	0.54	-0.26
mine	placed	0.14	-0.11
mine	yours	0.04	0.07
mine	mine	0.10	-0.05
yours	flipped	-0.45	0.08
yours	placed	-0.06	0.00
yours	yours	-0.33	0.04
yours	mine	0.27	0.02

Table 6.2: Cosine similarities of different features after the OV circuit of Mine/Yours Heads

6.5.3 Visualization

To better understand the underlying mechanisms, we visualize each attention head's written output to tiles when the tile has previously been flipped. Given that μ_i and ϕ_i exhibit some overlap, we define: $\mu'_i = \mu_i - \frac{\mu_i \cdot \phi_i}{\phi_i \cdot \phi_i} \cdot \phi_i$, where μ'_i represents the "tile is flipped" feature direction orthogonal to the "tile is yours" feature direction.

For each attention head, we compute $\mu^*_i = (W_O W_V \mu'_{i-1})$ and visualize

$$\textbf{cosine\_similarity}(\mu^*_i, \phi_i - \psi_i)$$

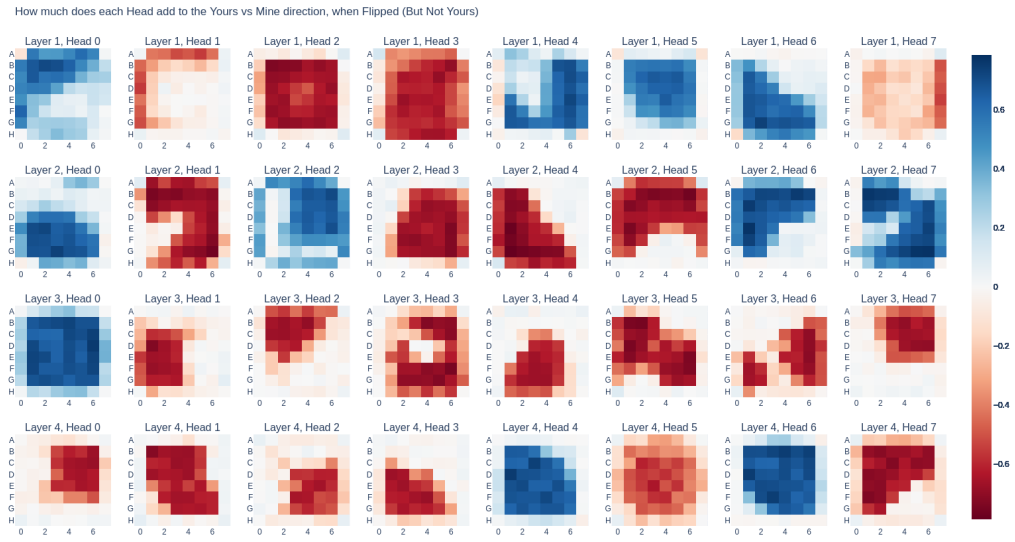


Figure 6.5: Contribution of each Attention Head to the "Tile is Yours" - "Tile is Mine" direction, when the head payes attention to a previous sequence position where the tile was flipped

The visualization in in Figure 6.5 reveals that Mine Heads write to the “tile is yours” direction while Yours Heads write to the “tile is mine” direction, with different attention heads focusing on specific board regions.

6.6 Evaluating the Previous Color Circuit

6.6.1 Experimental Setup

To assess how much this circuit really explains, we formalize it in Section 6.6.2. The circuit outputs feature logits for "tile is yours" and "tile is mine," which we evaluate by comparing them to the feature logits produced by the actual attention layer. Specifically, when our circuit outputs a higher logit for "tile is yours" than "tile is mine," we label it as predicting "tile is yours," and vice versa. We compute accuracy by comparing this prediction with the actual values for all non-empty tiles across 64,000 input sequences.

6.6.2 Formalization

For each tile, we have a set of features $\{\mu_i, \phi_i, \psi_i\}$ representing "tile is flipped", "tile is yours" and "tile is mine". The average residual stream is calculated as $\mathbf{x}_{\text{avg}} = \frac{1}{200} \cdot \sum_{g=0}^{200} \mathbf{x}_\mathbf{i}^\mathbf{g}$, where $\mathbf{x}_\mathbf{i}^\mathbf{g}$ refers to the residual stream of game g in the training dataset. Using Gram-Schmidt orthogonalization, we derive a new set of orthogonal vectors $\{\mu_i^*, \phi_i^*, \psi_i^*, \mathbf{x}_{\text{avg}}^*\}$ spanning the same space as $\{\mu_i, \phi_i, \psi_i, \mathbf{x}_{\text{avg}}\}$.

Given a residual stream x_i. We can project the residual stream on to the space spanned by $\{\mu_i^*, \phi_i^*, \psi_i^*\}$, with

$$x_i^{\text{proj}} = \frac{\langle x_i, \mu_i^* \rangle}{\langle \mu_i^*, \mu_i^* \rangle} \cdot \mu_i^* + \frac{\langle x_i, \phi_i^* \rangle}{\langle \phi_i^*, \phi_i^* \rangle} \cdot \phi_i^* + \frac{\langle x_i, \psi_i^* \rangle}{\langle \psi_i^*, \psi_i^* \rangle} \cdot \psi_i^*$$

.

We can then decompose the residual stream as the sum of the projected residual stream and a remainder x_i^*, such that $x_i = x_i^{\text{proj}} + x_i^*$.

In our circuit, we assume that the remainder has minimal influence on the actual result and approximate it using the average residual stream $\mathbf{x}_{\text{avg}}^*$[3], which is orthogonal to the feature directions.

The attention layer can be decomposed into contributions from each head and sequence position:

$$\begin{aligned}
\phi(\text{Att}_i(x'_{i-1})) &= \sum_{h \in H_i} \phi(h(x'_{i-1})) + \phi(b_i) \\
&= \sum_{h \in H_i} \sum_{j \le i} h.A_{i,j} \cdot \phi_i(h.W_O h.W_V x'_j) + \phi_i(b_i) \\
&= \sum_{h \in H_i} \sum_{j \le i} h.A_{i,j} \cdot \phi_i(h.W_O h.W_V x_j^*) + \phi_i(h.W_O h.W_V x_i^{\text{proj}})) + \phi_i(b_i) \\
&\approx \sum_{h \in H_i} \sum_{j \le i} h.A_{i,j} \cdot (\phi_i(h.W_O h.W_V x_{\text{avg}}) + \phi_i(h.W_O h.W_V x_i^{\text{proj}})) + \phi(b_i)
\end{aligned}$$

The output of this formula is a predicted logit for "tile is yours." Analogously, we can compute the logit for the "tile is mine" feature. As noted, we can also approximate the attention pattern A with an average approximation $A_{\text{avg}} = \frac{1}{200} \cdot \sum_{g=0}^{200} A$.

3 We average the residual stream for each sequence position independently

Figure 6.6: Average accuracy of the previous color circuit for layer 2 over all sequence positions split into tiles on the rim and tiles in the middle of the board

6.6.3 Results

The accuracy over all layers and sequence positions is 0.804. Using the original attention pattern, not the approximation, we get an accuracy of 0.818. But the accuracy varies a lot depending on the layer, sequence position, and whether the tile is on the rim or not. Figure 6.6 shows the accuracy at layer 2 for all sequence positions and Figure 6.7 shows the accuracy at sequence position 15 over all layers. We can see that for some combination of layer, sequence position and tile, the previous color circuit has an accuracy of > 0.97 which is very good.

We know that the board state in the first few sequence positions has very high accuracy after layer 0. This is not too surprising, because much of that can be memorized. Now, if we assume that after each attention layer the MLP layer flips the correct tiles to Yours and newly placed tiles are also set to Yours, then this would already be enough to compute the board state for all sequence positions. Because the board state is correct after layer 0 in the first few positions, it will be

Figure 6.7: Average accuracy of the previous color circuit at sequence position 15 over all layers, split into tiles on the rim and tiles in the middle of the board

correct after layer 1 in the next few positions, and so on. In the next section, we will test hypotheses about how the MLP layer might compute which tiles are flipped.

7 The MLP Layer

In this Section we use mechanistic interpretability techniques and visualizations to analyze the MLP layers of OthelloGPT. We find a familily of Flipping Neurons, that seem responsible for flipping tiles according to rules of Othello. We call this the Flipping Circuit Hypothesis, but fail to give conclusive evidence for it.

7.1 Basic Observations

Examining Figure 6.3, we observe that at position 19 in layer 1, following the Attention Layer's output of "D3 is mine", the MLP correctly writes "D3 is flipped" and "D3 is yours". We can employ Direct Logit Attribution to identify the contributing neurons. Let μ represent the feature "D3 is flipped"; then we can decompose the MLP layer's contribution into individual neuron contributions (See Chapter 3):

$$\mu(\text{MLP}(x_i)) = \mu(b_{\text{out}}) + \sum_{j=0}^{m} \mu(W_{\text{out}}[j,:] \cdot a_j)$$

This decomposition is illustrated in Figure 7.1.

The analysis reveals that a single neuron, L1N1411, predominantly contributes to writing "D3 is flipped". To understand this neuron's function, we project its input and output weights onto the probe directions (See Figure 7.2). The neuron appears to activate when a tile is placed at H7, G6, or F5, triggering a sequence of flips extending to C2 (which is already Yours). The neuron subsequently writes both "D3 is flipped" and "D3 is yours".

7.2 The Case for Monosemantic Neurons

While the Mechanistic Interpretability literature frequently documents polysemantic neurons [Elh+22; Ola+20], these studies primarily focus on vision and language models that are relatively underparameterized for their tasks. In contrast, OthelloGPT, with its 25 million parameters dedicated to predicting legal moves in Othello, may have sufficient capacity for neurons to specialize in single functions. Looking

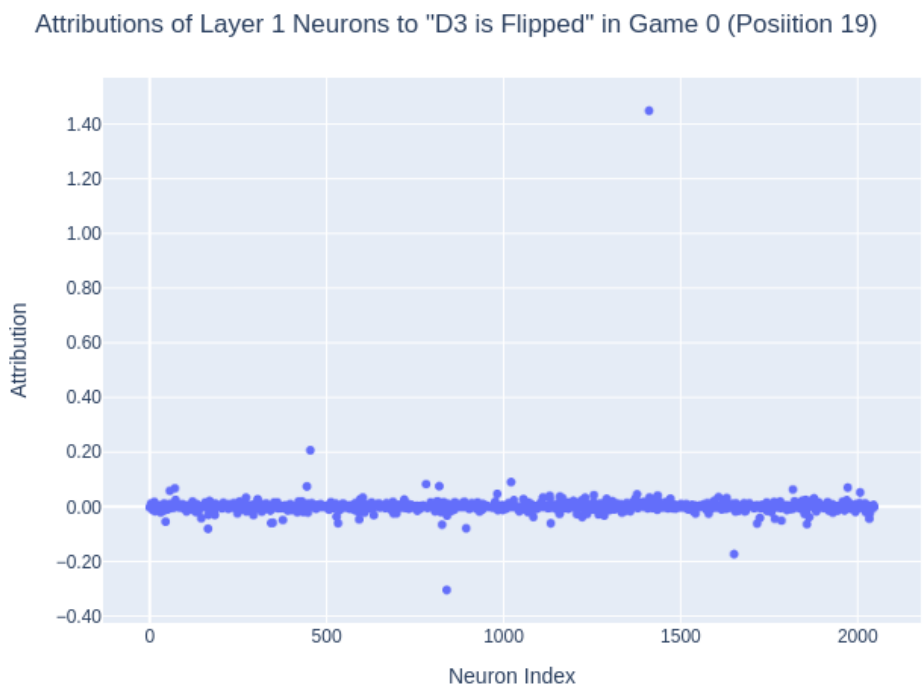


Figure 7.1: Direct Logit Attributions of Neurons in Layer 1 to "D3 is Flipped"

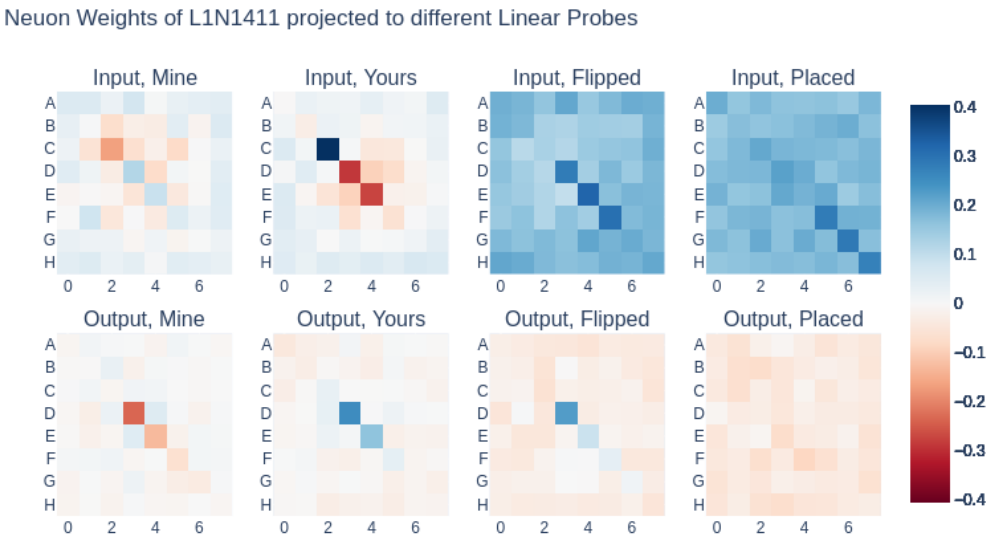


Figure 7.2: Neuron weights of L1N1411 projected to different linear probes. The top row displays input weight projections, while the bottom row shows output weight projections

at visualizations similar to Figure 7.2 and corresponding dataset examples seems to support this hypothesis.

7.3 Classifying Flipping Neurons

We observe numerous neurons exhibiting behavior similar to L1N1411, each responsible for flipping different tiles from various directions. We classify these as Flipping Neurons and develop a systematic approach to identify them through rule-based testing.

7.3.1 Experimental Setup

Let R denote the set of rules. Each rule $r \in R$ is defined by:

- A tile $t \in \{A0, A1, ..., H7\}$ marked as "Yours"
- A flipping direction $F \in \{\text{UP}, \text{UP-RIGHT}, \text{LEFT}, ...\}$
- The number of Mine tiles to be flipped $n_m \in \{1, 6\}$

A rule $r(x_i)$ evaluates to true for a residual stream x_i when a tile placement triggers flips in the specified direction, affecting exactly n_m "Mine" tiles before reaching tile t, which is Yours.

For example, L1N1411 corresponds to the rule:

$$
\begin{aligned}
&(C2 \text{ Yours} \wedge D3 \text{ Mine} \wedge E4 \text{ Placed}) \quad \vee \\
&(C2 \text{ Yours} \wedge D3 \text{ Mine} \wedge E4 \text{ Flipped} \wedge F5 \text{ Placed}) \quad \vee \\
&(C2 \text{ Yours} \wedge D3 \text{ Mine} \wedge E4 \text{ Flipped} \wedge F5 \text{ Flipped} \wedge G6 \text{ Placed}) \quad \vee \\
&(C2 \text{ Yours} \wedge D3 \text{ Mine} \wedge E4 \text{ Flipped} \wedge F5 \text{ Flipped} \wedge G6 \text{ Flipped} \wedge H7 \text{ Placed})
\end{aligned}
$$

For each rule-neuron pair (r, j), we calculate the mean activation difference between cases where the rule is active versus inactive. Given G games (we use $G = 10,000$), let:

- $T = \{x_i^g | 0 \le g \le G \wedge 0 \le i \le 59 \wedge r(x_i^g)\}$ be the set of true instances
- $F = \{x_i^g | 0 \le g \le G \wedge 0 \le i \le 59 \wedge \neg r(x_i^g)\}$ be the set of false instances

The mean activation difference is defined as:

$$\textbf{mean\_activation\_difference}(r, j) = \frac{1}{|T|} \sum_{x_i^g \in T} a_{i,j}^g \cdot \mathbb{1}_{r(x_i^g)} - \frac{1}{|F|} \sum_{x_i^g \in F} a_{i,j}^g \cdot (1 - \mathbb{1}_{r(x_i^g)})$$

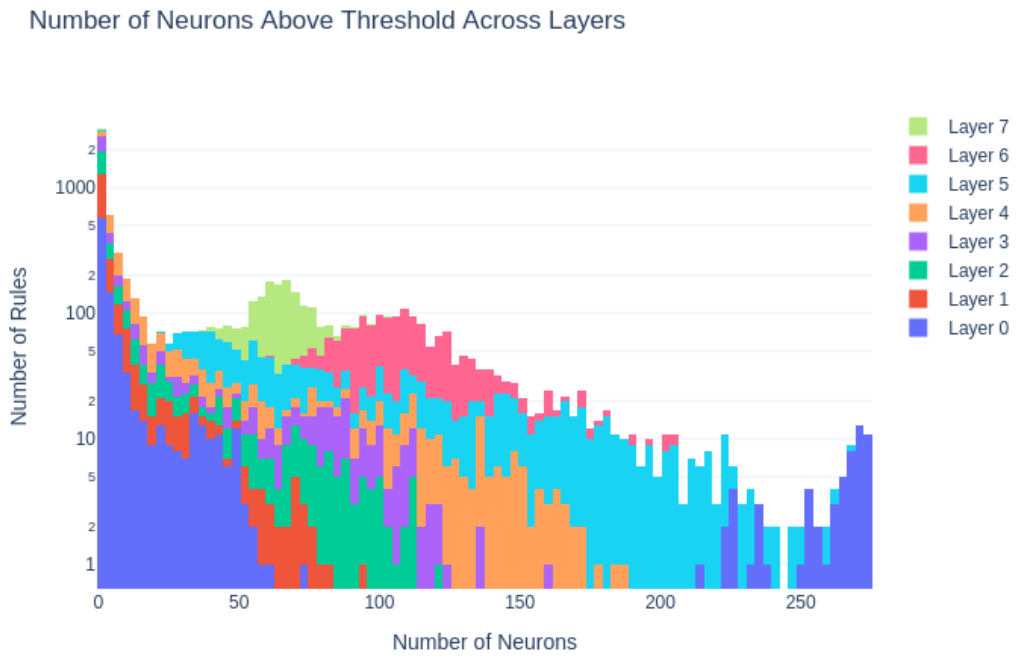


Figure 7.3: Histogram with number of neurons on the x-axis and number of rules with that many corresponding neurons on the y-axis, for each layer. y-axis is log scale. Bin Size of 3

Where $a^{g}_{i,j}$ represents neuron j's activation at position i in game g. Given the GELU activation function's minimum of approximately -0.17, a mean activation difference exceeding 0.17 suggests the neuron consistently activates positively when the rule is true and negatively otherwise.

7.3.2 Results

Figure 7.3 shows the number of rules with a given number of corresponding neurons. We can see that in the early layers 0 to 4, most rules have only a few corresponding neurons and in later layers 5 to 7, many rules have a lot of corresponding neurons. But overall, a lot of these rules have corresponding neurons, which seems to indicate that a lot of neurons follow these rules.

7.4 Evaluating the Flipping Circuit

7.4.1 Experimental Setup

To assess whether the identified flipping neurons sufficiently explain the MLP layer's tile-flipping behavior, we conducted a systematic evaluation using mean ablation [Wan+22]. For each residual stream $x'_{\text{i\_mid}}$, we:

1. Identify active rules $R_c = \{r \in R \mid r(x'_{\text{i\_mid}})\}$
2. Determine corresponding neurons

$$N_c = \{j \in \{1, \ldots, 2048\} \mid \exists_{r \in R_c} \textbf{mean\_activation\_difference}(r, j) > 0.17\}$$

3. Apply mean ablation to all neurons except those in N_c

We define the average activation $a_{i,j}^{\textbf{avg}} = \frac{1}{1000} \sum_{g=0}^{1000} a_{i,j}^{g}$ [4] and create edited activations:

$$a_{i,j}^{*} = \begin{cases} a_{i,j} & \text{if } j \in N_c \\ a_{i,j}^{\textbf{avg}} & \text{otherwise} \end{cases}$$

Our evaluation focuses on cases where the model changes its prediction regarding a tile's flipped status. Specifically, we examine instances where:

$$\text{argmax}(p_f(x_i^{l-1})) \neq \text{argmax}(p_f(x_i^{l}))$$

or

$$\text{argmax}(p_f(x_i^{l-1})) \neq \text{argmax}(p_f(x_i^{l*}))$$

Here, p_f represents the probe for detecting flipped tiles (introduced in Chapter 5). For our baseline comparison, we compute:

$$x^{\text{base}} = x_{\text{i\_mid}} + a_i^{\textbf{avg}} \cdot W_{\text{out}} + b_{\text{out}}$$

We additionally evaluate an alternative setup where we approximate neuron activations using the mean activation on active cases. For neurons $j \in N_c$, we set:

$$a_{i,j}^{*} = \begin{cases} \frac{1}{|T|} \sum_{x_i^g \in T} a_{i,j}^{g} & \text{if } j \in N_c \\ a_{i,j}^{\textbf{avg}} & \text{otherwise} \end{cases}$$

4 We are using games from the train set for this

7.4.2 Results

Results are presented in Figure 7.4 and Table 7.1, with Table 7.2 showing the average number of neurons involved per layer. The flipping circuit exceeds baseline performance only after layer 4, suggesting that early-layer flipping neurons like L1N1411 might be rarer than initially hypothesized. Further investigation of failure cases is needed to understand why the circuit's performance doesn't exceed the baseline overall.

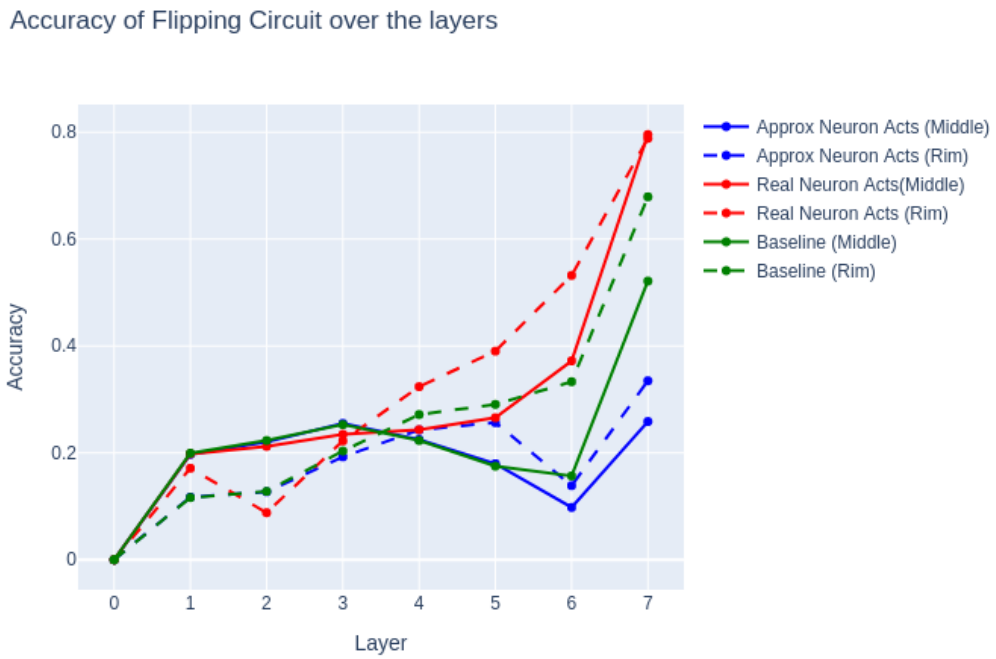


Figure 7.4: Accuracy of flipping circuit across neural network layers. Green represents the baseline, red the standard setup, and blue the setup with additional approximation of positive neuron activations. Solid lines indicate accuracy for tiles in the board's center, while dashed lines represent accuracy for tiles on the board's rim.

Flipping Circuit	Flipping Circuit (Approx. Neuron Acts)	Baseline
15.4	13.2	14.2

Table 7.1: Accuracy Comparison of flipping Circuit Variants Against the Baseline

L0	L1	L2	L3	L4	L5	L6	L7
148	21	48	86	199	291	448	97

Table 7.2: Average Number of Neurons in Flipping Circuit per Layer

8 Conclusions & Outlook

8.1 Better Understanding of OthelloGPT

Our analysis revealed that the previous color circuit demonstrates how attention layers in OthelloGPT propagate tile colors from previous sequence positions to current positions, maintaining board state consistency when the previous representation is accurate. However, this circuit's predictive accuracy diminishes notably in later sequence positions, indicating additional underlying mechanisms yet to be discovered. Furthermore, the precise computational process for determining tile flips remains unclear. While we hypothesize this computation occurs within the MLP layers and involves the identified flipping neurons, further investigation is required to validate this theory.

8.2 Future Research Directions

8.2.1 Further Investigation of the Flipping Circuit Hypothesis

Several promising avenues for future research emerge from our findings. A primary focus should be investigating cases where the flipping circuit hypothesis fails to accurately predict model behavior, potentially revealing additional mechanisms. Specifically, we propose beginning with an analysis of direct logit attributions of neurons, as illustrated in Figure 7.1, followed by examining the involved neurons using methods similar to those demonstrated in Figure 7.2.

8.2.2 Ablation Studies

Our theory suggests that the model's board state representation relies on early layers, with each layer being crucial for specific sequence positions. To validate this hypothesis, we propose conducting mean ablation studies on these layers across non-relevant sequence positions. If OthelloGPT maintains high performance under these conditions, it would constitute compelling evidence supporting our understanding of the model's internal computational mechanisms.

Bibliography

[BG24] Leonard Bereska and Efstratios Gavves. **Mechanistic Interpretability for AI Safety–A Review**. *arXiv preprint arXiv:2404.14082* (2024) (see page 11).

[Bub+23] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. *Sparks of Artificial General Intelligence: Early experiments with GPT-4*. 2023. arXiv: 2303.12712 [cs.CL]. URL: https://arxiv.org/abs/2303.12712 (see page 1).

[Cam+20] Nick Cammarata, Gabriel Goh, Shan Carter, Ludwig Schubert, Michael Petrov, and Chris Olah. **Curve Detectors**. *Distill* (2020). DOI: 10.23915/distill.00024.003 (see page 2).

[Con+23] Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. **Towards automated circuit discovery for mechanistic interpretability**. *Advances in Neural Information Processing Systems* 36 (2023), 16318–16352 (see page 6).

[Elh+21] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. **A Mathematical Framework for Transformer Circuits**. *Transformer Circuits Thread* (2021) (see pages 1, 7, 9).

[Elh+22] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. **Toy Models of Superposition**. *Transformer Circuits Thread* (2022) (see pages 11, 33).

[Has] Sarah Hastings-Woodhouse. *Introduction to Mechanistic Interpretability*. URL: %5Curl%7Bhttps://aisafetyfundamentals.com/blog/introduction-to-mechanistic-interpretability/#:~:text=Mechanistic%20Interpretability%20is%20an%20emerging,1%5D%7D (see page 1).

[He+24] Zhengfu He, Xuyang Ge, Qiong Tang, Tianxiang Sun, Qinyuan Cheng, and Xipeng Qiu. **Dictionary learning improves patch-free circuit discovery in mechanistic interpretability: A case study on othello-gpt**. *arXiv preprint arXiv:2402.12201* (2024) (see page 17).

[HG23] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2023. arXiv: 1606.08415 [cs.LG]. URL: https://arxiv.org/abs/1606.08415 (see page 6).

[Hsu+24] Aliyah R Hsu, Yeshwanth Cherapanamjeri, Anobel Y Odisho, Peter R Carroll, and Bin Yu. **Mechanistic Interpretation through Contextual Decomposition in Transformers**. *arXiv preprint arXiv:2407.00886* (2024) (see page 6).

[HZC23] Dean S Hazineh, Zechen Zhang, and Jeffery Chiu. **Linear Latent World Models in Simple Transformers: A Case Study on Othello-GPT**. *arXiv preprint arXiv:2310.07582* (2023) (see page 19).

[jyl+] jylin04, JackS, Adam Karvonen, and Can. *OthelloGPT learned a bag of heuristics*. https://www.alignmentforum.org/posts/gcpNuEZnxAPayaKBY/othellogpt-learned-a-bag-of-heuristics-1. Accessed: 2024-10-01 (see page 17).

[Kin14] Diederik P Kingma. **Adam: A method for stochastic optimization**. *arXiv preprint arXiv:1412.6980* (2014) (see page 16).

[Li+24a] Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. *Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task*. 2024. arXiv: 2210.13382 [cs.LG]. URL: https://arxiv.org/abs/2210.13382 (see page 1).

[Li+24b] Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. *Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task*. 2024. arXiv: 2210.13382 [cs.LG]. URL: https://arxiv.org/abs/2210.13382 (see page 6).

[NB22] Neel Nanda and Joseph Bloom. *TransformerLens*. 2022 (see page 8).

[NLW23] Neel Nanda, Andrew Lee, and Martin Wattenberg. *Emergent Linear Representations in World Models of Self-Supervised Sequence Models*. 2023. arXiv: 2309.00941 [cs.LG]. URL: https://arxiv.org/abs/2309.00941 (see pages 1, 15, 16).

[Ola+20] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. **Zoom In: An Introduction to Circuits**. *Distill* (2020). https://distill.pub/2020/circuits/zoom-in. DOI: 10.23915/distill.00024.001 (see page 33).

[Ols+22] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. **In-context Learning and Induction Heads**. *Transformer Circuits Thread* (2022) (see pages 1, 2).

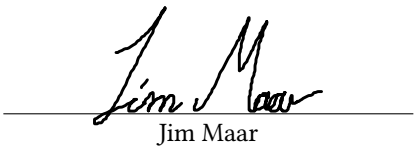
[San24] Grant Sanderson. *Attention in transformers, visually explained | Chapter 6, Deep Learning*. Apr. 2024. URL: https://www.youtube.com/watch?v=eMlx5fFNoYc (see page 7).

[Wan+22] Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. **Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small**. *arXiv preprint arXiv:2211.00593* (2022) (see pages 1, 2, 38).

Declaration of Authorship

I hereby declare that this thesis is my own unaided work. All direct or indirect sources used are acknowledged as references.

Potsdam, November 1, 2024



Jim Maar