An Informal Investigation of Indirect Object Identification in Mistral GPT2-Small Battlestar

Chris Mathwin

Interpretability Hackathon Report

Apart Research

PIs: Esben Kran, Neel Nanda, Fazl Barez

Date: 13th November, 2022

Abstract

This report represents an informal investigation of an IOI circuit within the Mistral GPT2-Small x49 Battlestar transformer model, inspired by the work performed by Wang et al. [1] wherein they identify the IOI circuit in GPT2-Small and methods outlined by Neel Nanda [2, 4, 6]. This report does not take the rigorous approach detailed by Wang et al. [1], that conforms to a criteria of Faithfulness, Completeness and Minimality. Instead, this is intended as an informal 'first pass' wherein a number of techniques are used to speculatively identify attention heads that may contribute towards analogous *Negative Name Mover Heads, Name Mover Heads, Backup Name Mover Heads* and *Previous Token Heads.* These preliminary findings were identified using Direct Logit Attribution, Attention Patterns and Ablation techniques provided in notebooks distributed by Redwood Research [5] and Neel Nanda [2, 4, 6]. Future work will attempt to solidify these findings and identify which attention heads may also be present within the aforementioned groups, and potentially within the *Duplicate Token Heads, Induction Heads* and *S-Inhibition Heads* groups. This work intends to conform to the three criteria proposed by Wang et al. [1] and to make use of both Activation and Path Patching.

Acknowledgements

This report was written while participating in Apart Research's Interpretability Hackathon, it incorporates significant parts of work conducted by Neel Nanda and Redwood Research. I would like to sincerely thank them for the wealth of resources they've provided in support of mechanistic interpretability research. Additionally, I would like to sincerely thank Apart Research for hosting such an enjoyable hackathon.

Disclaimer

This report is primarily intended as a self-learning tool, for this reason and by the nature of a Hackathon, it is likely to contain numerous errors and omissions. These errors are entirely of my own, they do not reflect upon any of the resources that I have used.

Accompanying Code

https://github.com/cmathw/Interpretability-Hackathon

Introduction

Indirect Object Identification (IOI) refers to the ability to infer correct subject/indirect object understanding in a sentence. This concept is most clearly illustrated by an example, "John and Mary went to the cinema, while driving home John asked [token]", outputting the "Mary" token would signify a correct Indirect Object Identification (IOI). Wang. et al [1] showed that GPT2-Small (a comparatively small autoregressive decoder-only transformer with 117M parameters) could display this behavior and successfully identified the associated circuit within this model. This circuit is illustrated within Figure 1, where 7 subcomponents constructed from 26 attention heads (labeled via layer, then head within the subcomponent graph in Figure 1) were found to be implementing the following human-understandable algorithm:

1. Find all previous names within the sequence.
2. Remove all instances of a name that have a duplicate.
3. Output remaining name.

Not only was the model capable of generating this algorithm independently but it also exhibited robustness, wherein if attention heads (in this case, *Name Mover Heads*) associated with outputting the Indirect Object token (in the above case, "Mary") were knocked out, this ability was retained by the *Backup Name Mover Heads.*

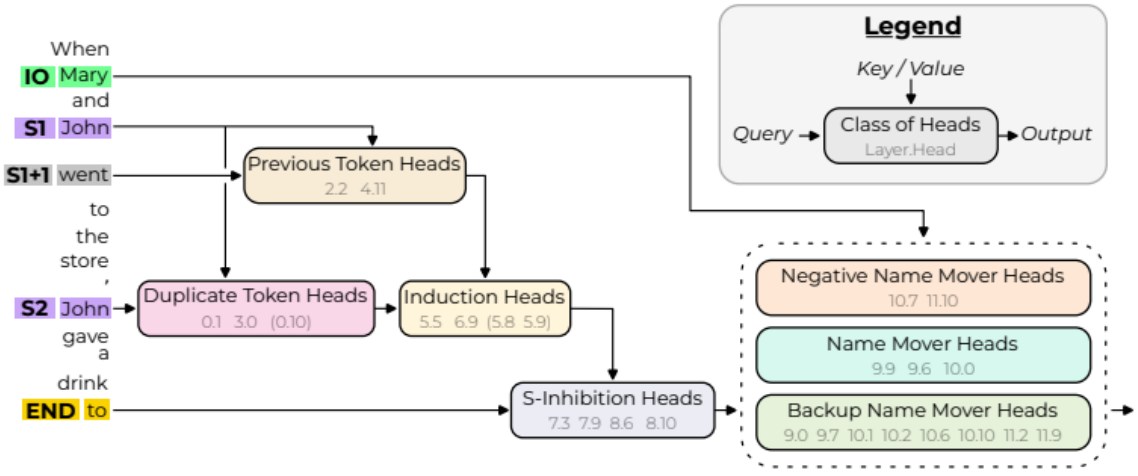

Figure 1: GPT2-Small IOI Circuit from Wang et al. [1]

The goal of this report will be to perform an analogous, albeit far less rigorous and far more informal study of one of the Mistral GPT2-Small models, GPT2-Small x49 Battlestar

(henceforth, referred to as Battlestar) [3]. This work will rely heavily upon that conducted by both Wang et al. [1] at Redwood Research and Neel Nanda [2, 4, 6]. The approach that Wang et al. [1] presented was substantially more rigorous than this approach will take. In particular, Wang et al. [1] evaluated the validity of their argument based on three criteria:

- Faithfulness - Does the isolated IOI circuit perform the task of IOI as well as the model does?
- Completeness - Does the isolated IOI circuit contain the entirety of what the model uses to perform the IOI task?
- Minimality - Is the isolated IOI devoid of any extraneous aspects of the model that are not needed to perform the IOI task, a similar but inverse criteria to Completeness.

The approach taken within this report will not aim to satisfy these criteria at any level, instead this report will act as an informal attempt at first identifying potential areas in which this circuit may exist.

An Attempt to *informally* identify an IOI Circuit

One of the key contributions that Wang et al. [1] made was presenting a rigorous approach to identifying circuits within transformer models. They did this by walking back through the model and progressively applying *Path Patching t*o identify important aspects of the model. I don't yet fully understand how to perform path patching, instead I have used various other tools to informally identify potentially important features of Battlestar. These tools are based on code from some of Neel Nanda's Colab Notebooks [2, 4] and both Neel Nanda [5] and Redwood Research's Easy Transformer [6] code bases.

Step 0: Verifying that the model can do IOI

A sensible first step involves verifying that Battlestar can indeed do IOI in the first place, this will be done by passing in 5 examples and recording the token with highest and second highest probability.

Example Prompt	First and Second Highest Scored Tokens (Probability)
"While John and Mary were walking their dog, Mary handed the leash to"	"John" (58.68%) "the" (5.14 %)
"While John and Mary were walking their	"Mary" (77.07%)

dog, John handed the leash to”	“his” (4.94%)
"Shortly after Bob and Ross went fishing, they were packing up when Ross nudged"	“them” (29.59%) “him” (16.80%) “Bob” (12.32%)
"While Anne was shopping with Mark, Mark persuaded"	“her” (30.19%) “Anne” (15.58%)
"After Kate and Doug came back from the beach, Doug prompted"	“Kate” (24.22%) “them” (11.09%)

Table 1: Battlestar’s Highest and Second Highest Probabilities for a set of example IOI prompts

It is clear that Battlestar can do IOI well, for every prompt the highest scored token was the IO (barring the third and fourth example, wherein suitable swaps to general descriptors were made first followed by the appropriate name)**.** An improved way to measure this performance is to take the difference in logit scores between the IO (in the first example, this is “John”) and the subject (in the first example, this is “Mary”). This directly compares the model’s preference to either the IO name or Subject name. The same set of prompts are used from Table 1 to generate this data.

Example Prompt	Logit Difference (Positive and Higher is Better) (Relative Probabilities*)
“While John and Mary were walking their dog, Mary handed the leash to”	John Logit - Mary Logit = 2.99 (~20x)
"While John and Mary were walking their dog, John handed the leash to”	Mary Logit - John Logit = 5.46 (~235x)
"Shortly after Bob and Ross went fishing, they were packing up when Ross nudged"	Bob Logit - Ross Logit = 3.70 (~40x)
“While Anne was shopping with Mark, Mark persuaded”	Anne Logit - Mark Logit = 0.92 (~2x)
"After Kate and Doug came back from the beach, Doug prompted"	Kate Logit - Doug Logit = 2.34 (~10x)

Table 2: Logit Differences for a set of example IOI prompts

*We can also think of the logit difference as being the ratio of their probabilities, for example, for the first example the logit difference is 2.99 and $e^{2.99}$ is about 20, so if Mary had a probability of 1% being outputted, John would have a probability of 20%.

So, this is essentially measuring the difference between Battlestar choosing the correct answer and the wrong answer, a larger positive difference is evidence that the model is much more likely to have chosen the correct answer than the wrong one. We can thus say that Battlestar is quite good at IOI.

Step 1: Direct Logit Attribution

Direct Logit Attribution can be used to help identify aspects of the last modules in the IOI circuit, namely the *Name Mover, Negative Name Mover* and *Back up Name Mover* heads (Figure 2).

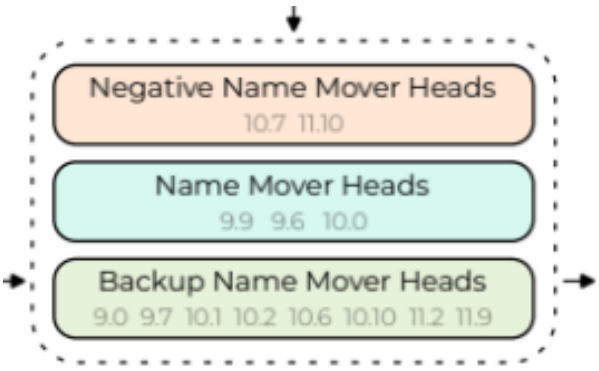

Figure 2: Last Module in the GPT2-Small IOI Circuit from Wang et al. [1]

Direct Logit Attribution is used to find how much each head in each layer affects the model's logits, this is done by using the fact that the logits are a linear transformation of the outputs of every head in every layer. We can input an example text and record the effect that each head in each layer had on the logits outputted in response to this example text.

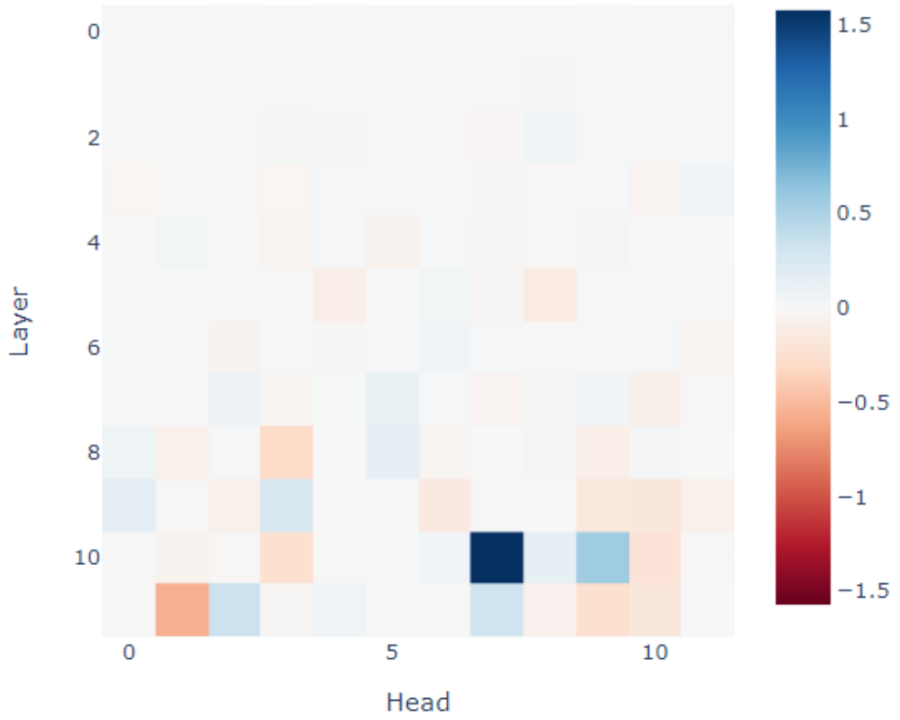

Figure 3: Direct Logit Attribution for Battlestar

In Figure 3, we have used Direct Logit Attribution with the first prompt from Table 1: "While John and Mary were walking their dog, Mary handed the leash to". We can then speculate that following heads may play a role as either *Name Mover Head* or *Back up Name Mover Heads* due to their comparatively large positive attribution scores:

- Layer 10, Head 7
- Layer 10, Head 9

We can also speculate that the following heads could be associated with *Negative Name Mover Heads* due to their comparatively large negative attribution scores*:*

- Layer 11, Head 1

Step 2: Using Attention Patterns

We can use attention patterns to try and speculate what other heads in the model may be contributing towards the IOI circuit, for example, Neel Nanda [4] provided an example of using attention patterns to look for heads that mostly attend to the previous token. By using attention

patterns in this way, we can get a sense for where the *Previous Token* heads may be within the model (Figure 4).

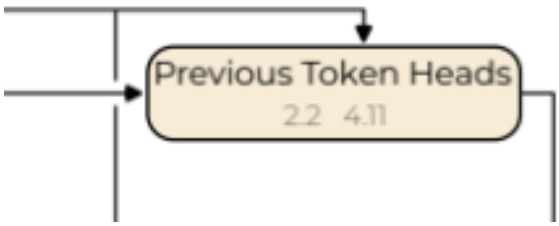

Figure 4: Previous Token Heads Module in GPT2-Small IOI Circuit from Wang et al. [1]

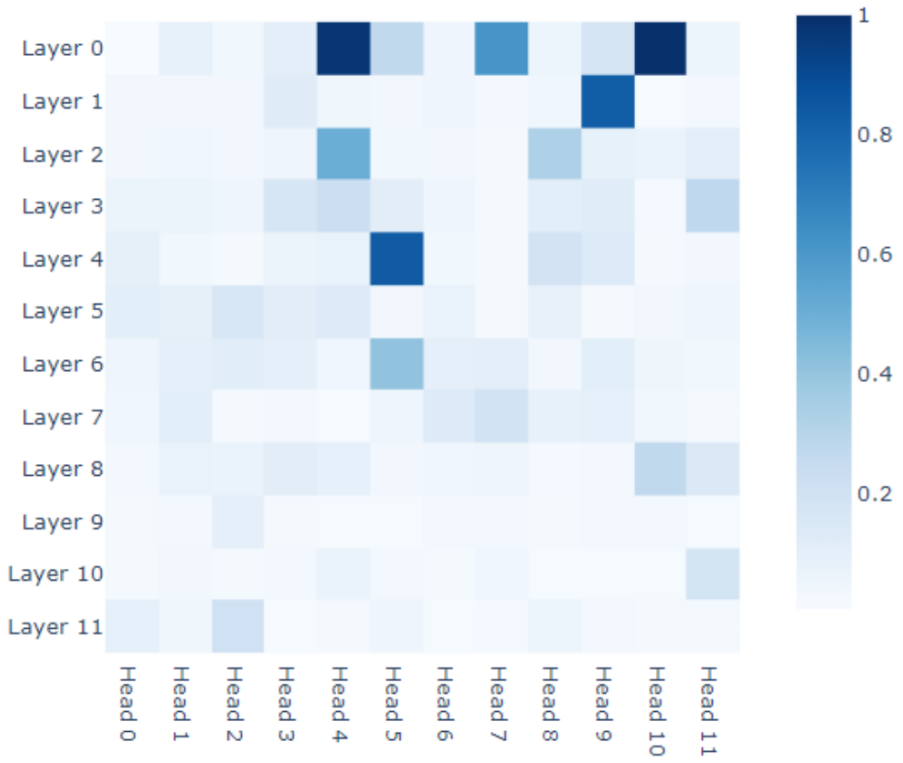

Figure 5: Previous Token Attention Pattern for Battlestar

Figure 5 represents the attention pattern for Battlestar when inputting a long text sentence, in this case I used Asimov's Three Laws of Robotics as the input. From this figure, we can speculate that the following heads may play a role in the *Previous Token* module:

- Layer 0, Head 10
- Layer 0, Head 4
- Layer 1, Head 9
- Layer 4, Head 5

Step 3: Using Ablation Techniques

In order to test our hypothesis thus far, we can employ ablation techniques to knock-out particular heads and observe whether their removal corresponds with a loss in the function we expect them to support. For example, if we were to knock-out all heads associated with the suspected *Name Mover* and *Name Mover Backup* heads, we would expect performance on IOI tasks to noticeably decrease.

We speculated that Layer 10, Head 7 acted as a *Name Mover Head* and thus we would expect that if we ablated this head, we would see a significant dip in IOI performance. This ablation is achieved by using functionality from within Neel Nanda's Easy Transformer notebook [4]. Using the following example prompt: "While John and Mary were walking their dog, Mary handed the leash to", the differences between before and after ablating this head is shown in Figure 6.

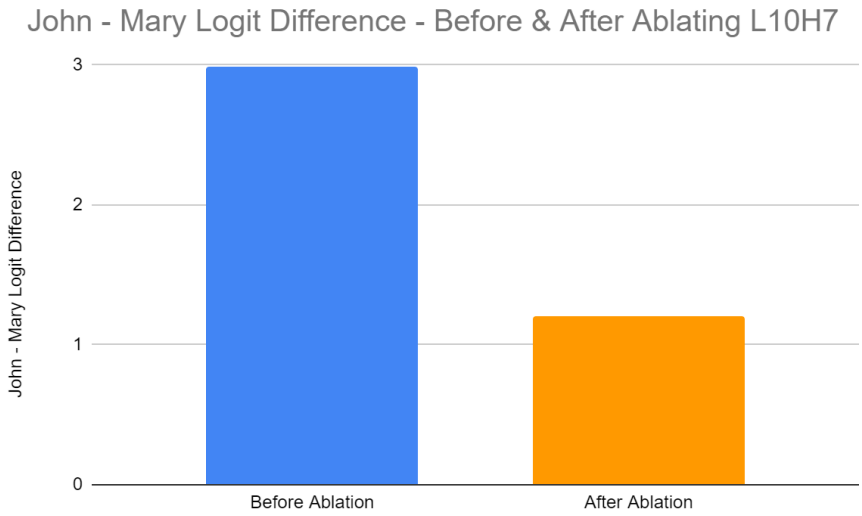

Figure 6: Effect of Ablating Layer 10, Head 7 on John - Mary Logit Difference

It does indeed appear that Layer 10, Head 7 does exist within Battlestar's IOI circuit. To obtain a stronger intuition of the effect size of removing any one attention head (vs. removing this one specifically), Figure 7 represents the effect of removing any single attention head within the model and its effect on performance of the IOI task described in the previous paragraph.

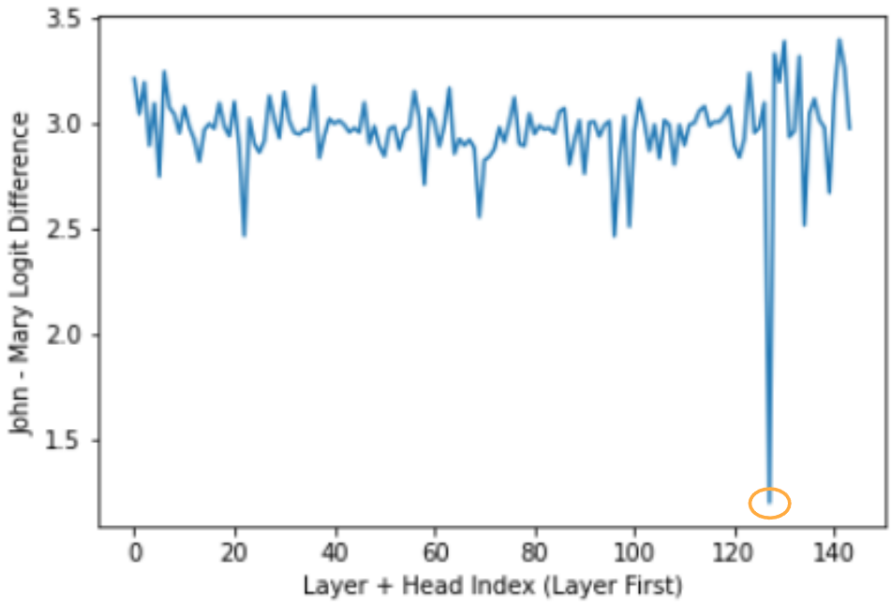

Figure 7: IOI Performance Following Ablation of each Head

The "Layer + Head Index" simply refers to any particular attention head, found by running through each layer, then each head (Layer + Head Index: Layer 0, Head 0 = 1, Layer 0, Head 1 = 1, … Layer 11, Head 11 = 143). The purpose of this graph isn't to find the performance drop associated with the ablation of any particular attention head, but to instead illustrate the significance of removing Layer 10, Head 7 on IOI performance (this attention head is circled orange on Figure 7).

Step 4: Using Activation Patching and Path Patching

A powerful technique that has not been included within this report is Activation Patching and Path Patching. Activation Patching involves roughly the following:

1. Using a trained model that produces an incorrect response to the IOI task
2. Pick a small aspect of the model (a single head, or perhaps a layer), potentially via the techniques outlined thus far
3. Copy the activations from that selected region from a model that provides a correct responses to the IOI task
4. Observe whether this modification improves the model in Step 1 towards the correct answer (ie. decreases the negative logit difference or causes the subject and indirect object logits to switch).
5. If Step 4 is successful, this is suggestive of this aspect of the model being potentially a component of the IOI circuit.

A more specialized, robust and targeted approach to Activation Patching is Path Patching. This is one of the primary techniques that Wang et al. [1] used when finding the IOI circuit. This will be an area of future work for this preliminary and informal investigation of Battlestar.

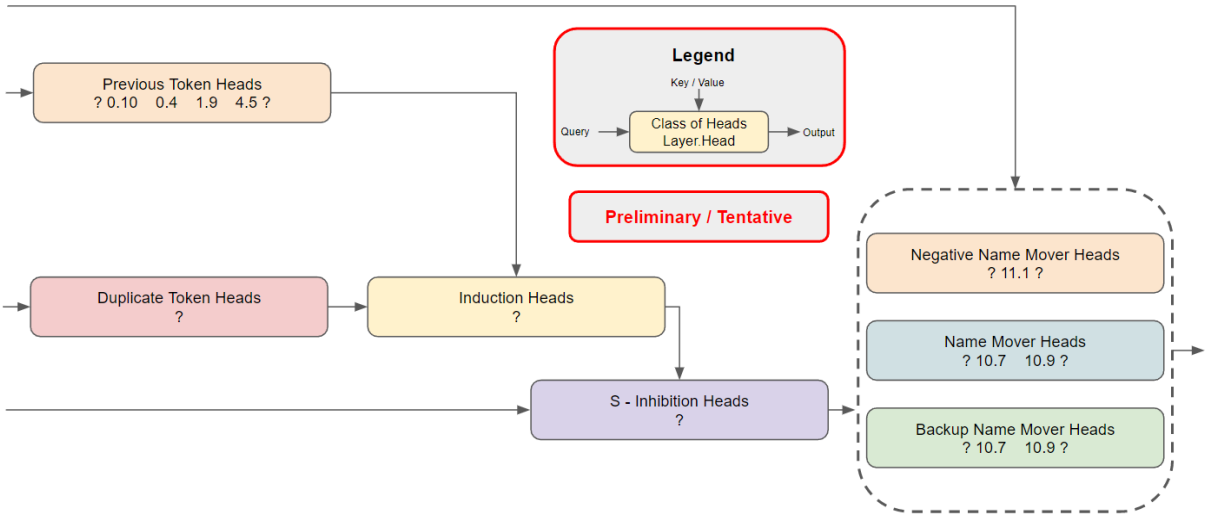

Figure 8: Preliminary (and incomplete) Battlestar IOI Circuit (inspired by Wang et al. [1])

To help illustrate the attention heads that were speculated in playing an analogous role as those identified in Wang et al. [1], an analogous diagram was produced displaying these tentative and preliminary findings. As mentioned earlier, future work will be performed on solidifying the validity of these claims (by attending to the criteria of Faithfulness, Completeness and Minimality proposed by Wang et al [1]) and by investigating the areas that have not yet been understood, namely the *Duplicate Token Heads, Induction Heads* and *S-Inhibition Heads* via Activation and Path Patching.

Conclusion

This report presents an initial and informal investigation of Battlestar's IOI circuit by using various mechanistic interpretability techniques, namely, Direct Logit Attribution, Attention Patterns and Ablation techniques. Although concrete assignments of particular heads were not made, a number of tentative claims were made as to some of attention heads that could exist in analogous IOI circuit's *Negative Name Mover Heads, Name Mover Heads, Backup Name Mover Heads* and *Previous Token Heads.* Further research effort will be employed to solidify these claims using the criteria proposed by Wang et al. [1] and in identifying the attention heads that may also occupy these modules, as well as the modules, *Duplicate Token Heads, Induction Heads* and *S-Inhibition Heads.* This research effort will make use of the Easy Transformer libraries produced by Redwood Research [5] and Neel Nanda [6], specifically by employing such techniques as Path Patching.

References

1. Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, Jacob Steinhardt: "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small", 2022; arXiv:2211.00593.
2. Neel Nanda: "SERI_MATS_IOI_Demo", 2022; https://colab.research.google.com/drive/1mL4KlTG7Y8DmmyIlE26VjZ0mofdCYVW6
3. Stanford University. "Mistral", 2022. https://github.com/stanford-crfm/mistral
4. Neel Nanda: "Easy Transformer Demo", 2022: https://colab.research.google.com/github/neelnanda-io/Easy-Transformer/blob/main/Easy Transformer_Demo.ipynb
5. Redwood Research: "Easy Transformer", 2022: https://github.com/redwoodresearch/Easy-Transformer
6. Neel Nanda: "Easy Transformer", 2022: https://github.com/neelnanda-io/Easy-Transformer