Copy Suppression: Comprehensively Understanding an Attention Head[†]

Callum McDougall[*], Arthur Conmy[*], Cody Rushing[*], Thomas McGrath, Neel Nanda
[*]: Joint Contribution.
(Affiliations and contact omitted)

Abstract

We present a single attention head in GPT-2 Small that has one main role across the entire training distribution. If components in earlier layers predict a certain token, and this token appears earlier in the context, the head suppresses it: we call this copy suppression. Attention Head 10.7 (L10H7) suppresses naive copying behaviour which improves overall model calibration. This explains why multiple prior works studying certain narrow tasks found negative heads that systematically favoured the wrong answer. We uncover the mechanism that the Negative Heads use for copy suppression with weights-based evidence and are able to explain 76.9% of the impact of L10H7 in GPT-2 Small. To the best of our knowledge, this is the most comprehensive description of the complete role of a component in a language model to date.

One major effect of copy suppression on model behaviour is its role in self-repair. Self-repair refers to how ablating crucial model components results in downstream neural network parts compensating for this ablation, and is a major bottleneck for circuit discovery, even in non-dropout models. Copy suppression leads to self-repair: if an initial overconfident copier is ablated, then there is nothing to suppress. Through targeted causal interventions and circuit analysis, we show that self-repair is implemented by several mechanisms, one of which is copy suppression, which explains 39% of the behaviour.

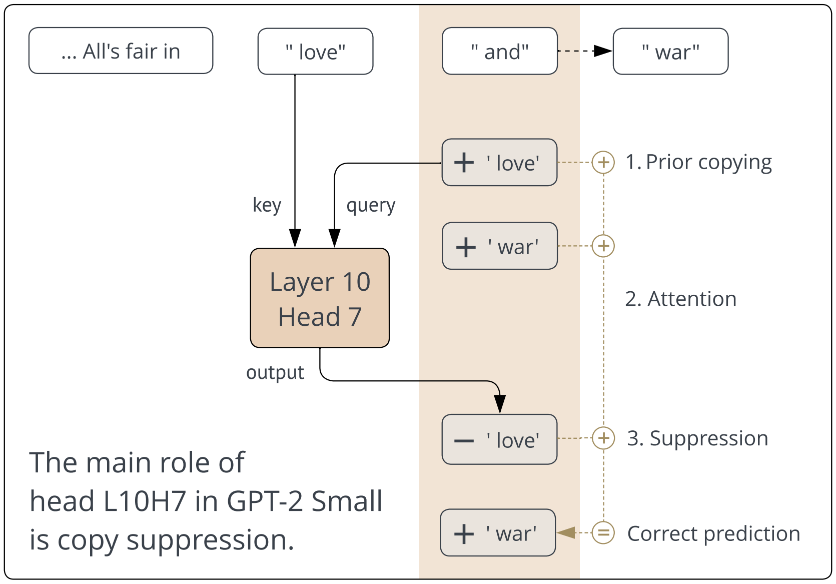


Figure 1: L10H7's copy suppression mechanism.

1 Introduction

Mechanistic interpretability research aims to reverse engineer neural networks into the algorithms that network components implement (Olah, 2022). A central focus of this research effort is the search for explanations for the behavior of model components, such as circuits (Cammarata et al., 2020; Elhage et al., 2021), neurons (Radford et al., 2017; Bau et al., 2017; Gurnee et al., 2023) and attention heads (Voita et al., 2019; Olsson et al., 2022). However, difficulties in understanding machine learning models has often limited the breadth of these explanations or the complexity of the components involved (Räuker et al., 2023), particularly when the models studied are large transformer language models (Elhage et al., 2022). In this work, we explain how a class of attention heads which we term Negative Heads (previously described as "anti-copying prefix-search" heads by (Olsson et al., 2022) and "negative name-mover heads" by Wang et al. (2023)) in GPT-2 Small (Radford et al., 2019) function on the natural language training distribution. These attention heads were observed to systematically write against the correct completion on narrow datasets. We explain this seemingly harmful effect as an instance of **copy suppression**, which accounts for a majority of the head's behavior and reduces the model's loss overall. To the best of our knowledge, we present the most comprehensive account of the function of a component in a Transformer-based language model.

In our work we use **Negative Heads** to refer to model components which primarily reduce the model's confidence in particular token completions. **Copy suppression** is defined by three steps (Figure 1):

1. **Prior copying**. Language model components in early layers directly predict that the next token is one that already appears in context, e.g that the prefix "All's fair in love and" is completed with " love".
2. **Attention**. Copy suppression heads detect the prediction of a copied token by an earlier layer and attend back to the previous instance of the copied token (" love").
3. **Suppression**. Copy suppression heads write to the model's output to decrease probability on the copied token.

By lowering incorrect logits, steps 1–3 can increase the probability on correct completions (e.g " war") and decrease model loss.[1] **Our central claim is that 76.9% of the role of attention head L10H7 on GPT-2 Small's training distribution is copy suppression**. However, we do not explain precisely when or how much copy suppression is activated in different contexts. Nevertheless, to the best of our knowledge, there is no prior work which has explained the main role of any component in a large language model in terms of its input stimulus and specific downstream effect across a whole training distribution.

Explaining language models components across wide distributions in mechanistic detail may be important for engineering safe AI systems, in addition to the scientific contribution of understanding how machine learning models work. While interpreting parts of language models on narrow distributions (Hanna et al., 2023; Heimersheim & Janiak, 2023; Wang et al., 2023) may be easier than finding complete explanations, researchers can be misled by hypotheses about model components that do not generalize (Bolukbasi et al., 2021). Mechanistically understanding models could fix problems that arise from opaque training processes or identify adversarial inputs (Goh et al., 2021; Carter et al., 2019), as mechanisms can predict behavior on all inputs (Mu & Andreas, 2020) rather than merely those that arise in training. Secondly, models can exhibit emergent capabilities (Wei et al., 2022) and internal insight could be extremely useful in predicting these abrupt changes (Nanda et al., 2023).

Mechanistic interpretability research is difficult to automate and scale (Räuker et al., 2023), and understanding negative and backup heads[2] could be crucial for further progress. Many approaches to automating interpretability use **ablations** - removing a neural network component and measuring the effect of this intervention (Conmy et al., 2023; Wu et al., 2023;

[1] We recommend using our web app `https://self-repair.streamlit.app/` to understand L10H7's behavior interactively.

[2] We define backup heads (see Section 5) as attention heads that respond to the ablation of a head by imitating that original behavior.

Bills et al., 2023; Chan et al., 2022). Ideally, ablations would provide accurate measures of the importance of model components on given tasks, but negative and backup components complicate this assumption. Firstly, negative components may be ignored by attribution methods that only find the positive components that complete tasks, artificially inflating their apparent performance. This means that these attribution methods will not find faithful (Jacovi & Goldberg, 2020) explanations of model behavior. Secondly, backup components may counteract the effects of ablations and hence cause unreliable importance measurements.

In this work we rigorously reverse-engineer Attention Head L10H7 in GPT-2 Small to show that its main role on the training distribution is copy suppression. We do not know *why* language models form copy suppression components, but in Appendices A and C we discuss ongoing research into some hypotheses. Appendix B provides evidence that copy suppression occurs in models trained without dropout.

- Section 3 shows that Negative Heads copy suppress on the entire natural language training distribution.
- Section 4 describes a mechanism by which Negative Heads copy suppress.
- Section 5 finds some connections between copy suppression and self-repair in language models.

2 Related Work

A common theme across language model interpretability research is the search for explanations for model components, such as neurons (Elhage et al., 2022), attention heads (Vig, 2019; Rogers et al., 2020) and circuits (Elhage et al., 2021; Wang et al., 2023).

1. **Neurons**. Researchers have found neurons that detect French words (Gurnee et al., 2023), neurons firing to predict " an" rather than " a" (Miller & Neo, 2023), or explanations for all neurons in GPT-2 Small by using GPT-4 (Bills et al., 2023).[3] In GPT-2 Small, each neuron is associated with 1537 parameters.
2. **Attention heads**. Prior work has identified heads with attention patterns correlated with rare words (Voita et al., 2019), previous tokens (Vig, 2019) as well as induction heads (Olsson et al., 2022). In GPT-2 Small, attention heads are associated with 196,800 parameters each.
3. **Circuits**. End-to-end circuits are subsets of computational graphs of models that explain model behavior on narrow distributions (Lieberum et al., 2023; Wang et al., 2023; Hanna et al., 2023; Heimersheim & Janiak, 2023) or on toy models (Nanda et al., 2023; Elhage et al., 2021). These circuits have contained up to millions of parameters.

Rigorous interpretations. Cammarata et al. (2021) rigorously interpreted a set of curve detecting neurons on the full data distribution of an image classifying convolutional network. We consider Gurnee et al. (2023); Olsson et al. (2022) to come the closest to comprehensively understanding a language model component (contextual neurons and induction heads, respectively), but neither goes as far as analysing them on the full data distribution. Indeed Goldowsky-Dill et al. (2023) showed that induction heads sometimes perform tasks other than induction, demonstrating the importance of rigorous analysis.

Quantifying explanation strength. Geiger et al. (2021) introduce causal abstraction analysis to formalize explanations of model components, similarly Chan et al. (2022) quantify explanation strength with the loss recovered metric. Our metric for explaining the effect of L10H7 (Section 4.3) is most similar to Bills et al. (2023)'s ablation score metric except we use KL divergence instead (Section 3 discusses this decision).

Iterative inference. Greff et al. (2017) propose that neural networks layers iteratively update feature representations rather than recomputing them, in an analysis specific to

[3] In fact, the researchers found a neuron that appears responds to prediction of " from" by suppressing it (though this observation required manual inspection): `https://openaipublic.blob.core.windows.net/neuron-explainer/neuron-viewer/index.html#/layers/44/neurons/307`

Prompt	**Source token**	**Incorrect completion**	**Correct completion**
... Millions of **Adobe** users picked easy-to-guess ~~**Adobe**~~ **passwords** ...	**“ Adobe”**	**“ Adobe”**	**“ passwords”**
... tourist area in **Beijing**. A university in ~~**Beijing**~~ **Northeastern** ...	**“ Beijing”**	**“ Beijing”**	**“ Northeastern”**
... successfully stopped **cocaine** and ~~**cocaine**~~ **alcohol** ...	**“ cocaine”**	**“ cocaine”**	**“ alcohol”**

Table 1: Dataset examples of copy suppression.

LSTMs and Highway Networks. Several works have found that transformer language model predictions are iteratively refined (Dar et al., 2022; nostalgebraist, 2020; Belrose et al., 2023; Halawi et al., 2023), in the sense that the state after intermediate layers forms a partial approximation to the final output, though no connections have yet been made to Negative Heads.

3 Negative Heads Copy Suppress

In this section we show that Negative Head L10H7 behaviourally suppresses copying across GPT-2 Small's training distribution. We show that these examples explain most of L10H7's role in the model, and defer quantitative evaluation of our mechanism for copy suppresion to Section 4.3.

Logit Lens. We can measure which output predictions different internal components push for by applying the Logit Lens method (nostalgebraist, 2020). Given model activations, such as the state of the residual stream or the output of an attention head, we can multiply these activations by GPT-2 Small's unembedding matrix. This measures the direct effect (ie not mediated by any downstream layers) that this model component has on the output logits for each possible token in the model's vocabulary (sometimes called direct logit attribution). The Logit Lens method allows us to refer to the model's predictions at a given point in the network.

Mean ablation. Mean ablation refers to replacing a model's output with the mean output calculated over a distribution.

Behavioural Results. We selected 5% of OpenWebText completions[4] where mean ablating L10H7 worsened cross entropy loss by the most (on a per-token basis), to get a representative sample of the cases where this head improves model performance. We observed that **84% of examples sampled in this way satisfied all three copy suppression steps** (Section 1):

1. A source token S found earlier in the context was confidently predicted at the destination token position D in the residual stream just before attention layer 10 (it was in the top 10 predictions of tokens in-context as measured by the Logit Lens).
2. The destination token D where L10H7 had a large effect on the next token prediction attended strongly to some source token S (the total attention paid to S tokens was in the top 10 amongst all in-context tokens).
3. At the D position, Head L10H7 wrote negatively in the unembedding direction for token S (S was one of the 10 in-context tokens that L10H7 wrote most negatively against).

Qualitative examples can be found in the Table 1. These results and more can also be explored on our interactive web app (`https://self-repair.streamlit.app/`).

[4] OpenWebText (Gok) is an open source replication of GPT-2's pretraining distribution

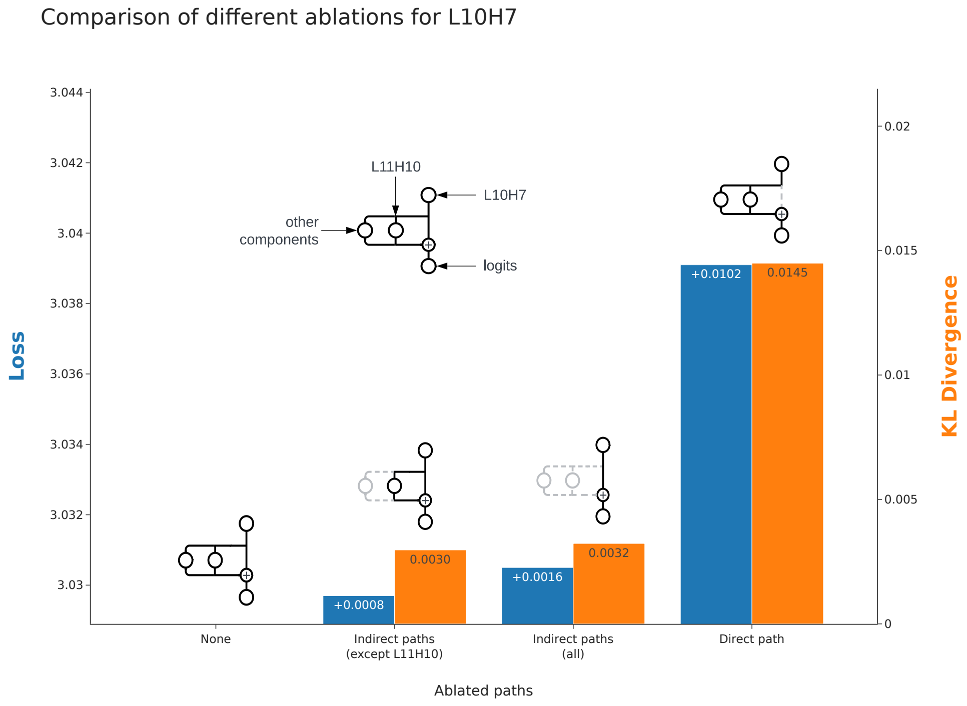


Figure 2: Loss effect of L10H7 via different paths. Grey paths denote ablated paths.

3.1 How Does L10H7 Affect the Loss?

To investigate whether this copy suppression mechanism explains the majority of how head L10H7 affects the model's loss, we first decompose the effect of head L10H7 into a set of different paths through the model (Elhage et al., 2021; Goldowsky-Dill et al., 2023). We find that most of the effect on loss (averaged over OpenWebText) was via the direct path from L10H7's output to the final logits, with the exception of the path from L10H7 through head L11H10 (Figure 2). This path is special because both heads are performing copy suppression, which is a self-repair mechanism: once a predicted token is suppressed, it is no longer predicted, and therefore does not activate future copy suppression components. This means that ablating head L10H7 will often result in it being backed up by head L11H10. We also plot the effect of ablating these paths on the average KL divergence of the resultant token distributions from the original model's distributions.

The results of Figure 2 suggest that a) we should try to understand the direct effect of head L10H7 on the final logit distribution since this has the largest impact on model outputs, and b) the KL divergence of model outputs obtained from ablation experiments from the model's original outputs is correlated with the increase in loss of ablated outputs. Our goal is to show that our copy suppression mechanism faithfully reflects L10H7's behaviour (Section 4.3). Therefore in the rest of our main text, we minimize KL divergence: in the limit, this would result in mechanisms that give identical output distributions to the model. On the other hand, loss could be minimized with unfaithful mechanisms that sometimes get better performance than the model. In Appendix H.1, we discuss this choice further and report results on how our mechanisms affect loss, too.

4 How Negative Heads Copy Suppress

In this section, we show that copy suppression explains 76.9% of L10H7's behavior on OpenWebText. To reach this conclusion, we first describe a concrete mechanism for how head L10H7 works.

1. In Section 4.1, we analyse the OV circuit, and show that the head suppresses the prediction of any token which it attends to.
2. In Section 4.2, we analyse the QK circuit, and show that the head attends to any token which the model is currently predicting.
3. In Section 4.3, we define a form of ablation (CSPA) which deletes all of L10H7's functionality except 1. and 2., and preserves 76.9% of its effect.

CSPA projects the OV circuit outputs onto a single direction, and filters for cases where attention would be large according to our QK mechanism.[5]

To understand both the QK and OV circuit, we use a refinement of GPT-2 Small's embedding matrix we call the **effective embedding** matrix. In short, the effective embedding matrix W_{EE} has the same dimensions as the embedding matrix W_E but more faithfully reflects how the model represents tokens in latent space.

Effective embedding definition and motivation. GPT-2 Small uses the same matrix in its embedding and unembedding layers, which may change how it learns certain tasks.[6] Prior research on GPT-2 Small has found the counter-intuitive result that at the stage of a circuit where the input token's value is needed, the output of MLP0 is often more important for token predictions than the model's embedding layer (Wang et al., 2023; Hanna et al., 2023). To account for this, we define the **effective embedding matrix** W_{EE} as the linear map whose rows are the values in the residual stream after applying the first block of the transformer, with the position-dependent model components (the positional embeddings and the attention layer) zero-ablated. Note that this remains purely a function of the input token, with no leakage from other tokens in the prompt, as the attention is ablated.

4.1 OV Circuit

To understand L10H7's output, we compute the OV circuit (Elhage et al., 2021) with our effective embedding refinement:

$$W_U^\top W_{OV}^{\text{L10H7}} W_{EE} \in \mathbb{R}^{d_{\text{vocab}} \times d_{\text{vocab}}} \tag{1}$$

Each column of this matrix is the vector of logits which will be added to the final logits at destination token D, if it only paid attention to source token S (and if we make the simplifying assumption that the residual stream at S only holds the effective embedding of token S). If head L10H7 is suppressing the tokens that it attends to, then we would expect to find that the diagonal elements of this matrix (i.e the effect on the logits of the token which is attended to) are consistently the most negative elements in their respective columns. The results support this: for 84.70% of the tokens in GPT-2 Small's vocabulary, the diagonal element of the OV-circuit is one of the top 10 most negative values in its column, and only 571 tokens weren't in the bottom 5% in their columns. This suggests that L10H7 is copy-suppressing almost all of the tokens in the model's vocabulary.

Moreover we can zoom in on the cases where the rank of diagonal elements in their columns are least negative to find the tokens which L10H7 suppresses the least, according to the OV-circuit, and we find that the errors are interpretable. We find that the top 22 failure cases (as measured by rank of the token in its column) are all **function words**, including

[5] In ongoing work, we are extending our results to projecting QK circuit inputs (Section 4.2, Appendix K.2 and K.3 discuss some important QK subspaces).

[6] As a concrete example, Elhage et al. (2021) show that a zero-layer transformer with tied embeddings cannot perfectly model bigrams in natural language.

“ of”, “ at”, “ their”, “ most”, “ as” and “ this”. These are common, generic words and so we speculate that there is no need for copy suppression as it is normal for them to occur multiple times. This shows that copy-suppression is not generically used for every token in the vocabulary, but differentially activated depending on the type of word which has been copied. We discuss more failure cases, as well as possible interpretations and implications in Appendix F.

Finally, does this OV circuit reflect the function of head L10H7 in practice? We filtered for examples in OpenWebText where L10H7’s had a large attention weight to certain tokens, and in 78.24% of these cases, we observed that the attended-to token was among the ten most suppressed tokens, from the direct effect of L10H7 (full experimental details in Appendix E). Along with the success of the ablation in Section 4.3, this suggests that negative copying describes L10H7 well.

4.2 QK Circuit

To understand L10H7’s attention patterns, we compute the QK circuit (Elhage et al., 2021), again with our effective embedding refinement:

$$W_U W_{QK}^{\text{L10H7}} W_{EE}^{\top} \in \mathbb{R}^{d_{\text{vocab}} \times d_{\text{vocab}}}. \qquad (2)$$

This is a bilinear form, which computes attention scores from destination to source token when the residual stream contains an unembedding at the destination token position, and an effective embedding at the source token position. Our theory of copy suppression suggests that L10H7 attends from destination tokens D where some token S is confidently predicted, back to previous instances of the token S. So we expect to find that the largest elements of each row are the diagonal elements of this matrix.

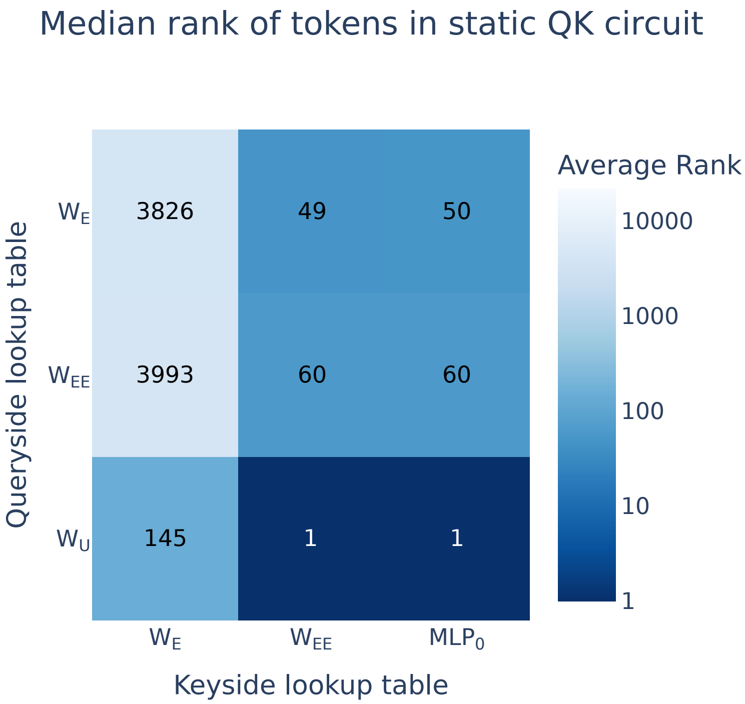


Figure 3: The median ranks of tokens when they are used as queryside vectors.

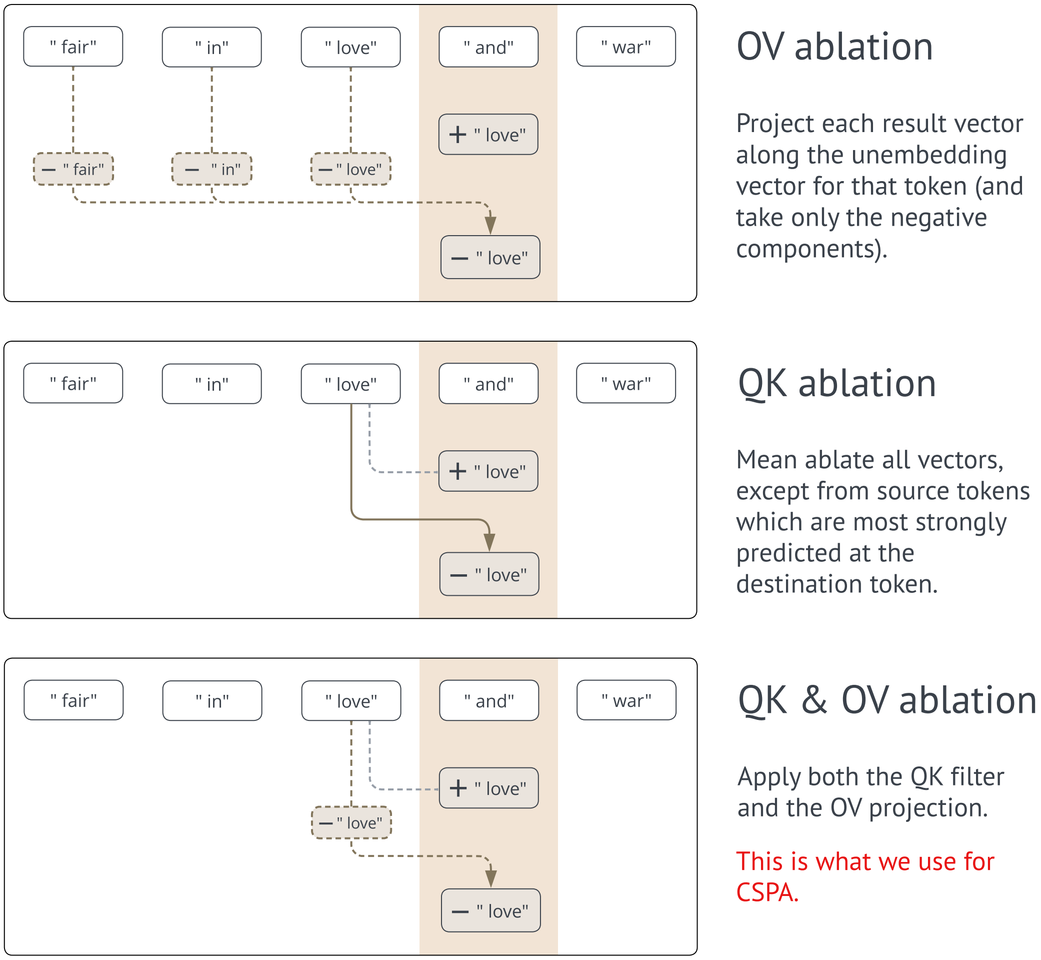
We performed an equivalent experiment to that from the previous section, measuring the rank of the diagonal elements in each row. We found that 95.72% of diagonal values in this matrix were the largest in their respective rows.[7] To test alternative mechanisms (and provide a baseline), we also try replacing the queryside and keyside lookup tables W_{EE} and W_U with other (un)embeddings to test alternative mechanisms. We computed the median ranks over GPT-2 Small’s vocabulary that we found in Section 4.1. The results can be found in Figure 3, and full details of these experiments can be found in Appendix K.1. We find that

1. Head L10H7 has greatest token self-attention when the queryside vector is an unembedding vector.
2. Head L10H7 has greatest token self-attention when the keyside vector involves MLP0.

[7] We ignore bias terms in the key and query parts (as we find that they do not change results much in Appendix K), and our experimental setup allows us to ignore LayerNorm (see Appendix G for all details).

Head L10H7 attends to embeddings most when the queryside vector is an unembedding vector, as across all three different keyside embeddings, the median rank is closest to 1st when the queryside lookup table is W_U. Additionally, the two ranks closest to 1st across all queryside and keyside lookup tables occur when we use either MLP0's output or the effective embedding, which includes MLP0. In Appendix K.1 we show that in a Duplicate Token Head the greatest average ranks do not occur when the queryside vector is an unembedding vector, suggesting that L10H7 has a qualitatively different QK-circuit to naive matching.

Does this mechanism represent how Attention Head L10H7 copy suppresses on OpenWebText? Unfortunately, we found that these approximations were not as faithful as we thought, and projecting key and query inputs onto the unembedding and effective embeddings directions did not capture full model performance (Appendix K.3 and K.2). Additionally, the CSPA mechanism (Section 4.3) works surprisingly well on many model heads, though it works best of all on 10.7 (Appendix H.3). Despite these limitations, we show that copy suppression nevertheless captures 76.9% of L10H7's behaviour, and in ongoing research we hope to explain the query- and key-side inputs more completely.



Project each result vector along the unembedding vector for that token (and take only the negative components).

Mean ablate all vectors, except from source tokens which are most strongly predicted at the destination token.

Apply both the QK filter and the OV projection.

This is what we use for CSPA.

Figure 4: Illustration of three different kinds of ablation: just OV, just QK, and both. The results presented in Section 4.3.2 section use both.

4.3 How much of L10H7's behavior have we explained?

In this section, we perform an ablation which deletes all functionality of L10H7's OV and QK circuits, except for the mechanisms described in Section 4.1 and Section 4.2 respectively.

We refer to this as **Copy Suppression-Preserving Ablation** (CSPA). In Section 4.3.1 we explain exactly how each part of CSPA works, and in Section 4.3.2 we present the result of the ablation.

4.3.1 Methodology

Ablation metric. We measure the amount of L10H7's behavior that we have explained by comparing our ablation to a baseline that mean ablates L10H7's direct effect. Formally, if the model's output token distribution on a prompt is π and the distribution under an ablation Abl is π_{Abl}, then we measure the KL divergence $D_{\text{KL}}(\pi||\pi_{\text{Abl}})$. We average these values over OpenWebText for both ablations we use, defining $\overline{D_{\text{CSPA}}}$ for CSPA and $\overline{D_{\text{MA}}}$ for the mean ablation baseline. Finally, we define the effect explained as

$$1 - \left(\overline{D_{\text{CSPA}}} \middle/ \overline{D_{\text{MA}}}\right). \tag{3}$$

For example, if CSPA was identical to mean ablation of L10H7's direct effect, the effect explained would be 0%, and if CSPA preserved L10H7 exactly, 100% of effect would be recovered. Appendix H motivates and describes the limitations of KL divergence further, compares to loss recovery metrics and considers performance baselines.

OV ablation. The output of an attention head at a given destination token D can be written as a sum of result vectors from each source token S, weighted by the attention probabilities from D to S. We can project each of these vectors onto the unembedding vector for the corresponding source token S, and only keep the negative components.

QK ablation. We mean ablate the result vectors from each source token S, except for the k source tokens which are predicted with highest probability at the destination token D (using the logit lens). Note that (for small values of k) this part of the ablation also tests the hypothesis of **sparsity** - that in normal operation, the majority of the head's effect comes from a small number of source tokens.

As an example of how the OV & QK ablations work in practice, consider the opening example "All's fair in love and war". In this case the destination token D is " war". The predicted token S is " love", and so we would take the result vector from this token and project it onto the unembedding vector for " love". This will capture the way head L10H7 suppresses the " love" prediction.

4.3.2 Results

Performing OV and QK ablation leads to 76.9% effect explained. Since the QK and OV ablations are modular, we can apply either of them independently and measure the effect recovered and performing only the OV ablation leads to 81.1% effect explained and only using QK leads to 95.2% effect explained.

To visualize the performance of CSPA, we plot $\overline{D_{\text{CSPA}}}^{(i)}$ against $\overline{D_{\text{MA}}}^{(i)}$ for $0 \leq i < 100$, the averaged values over points which are in the i-th percentile of D_{MA} values (Figure 5). We observe that for larger values of i our copy suppression mechanism is relatively better at explaining the head's behaviour than for smaller i. For example, amongst the completions in the quantile $i = 99$, where mean ablation increased loss by at least 0.19, CSPA explained 88.1% of L10H7's effect.

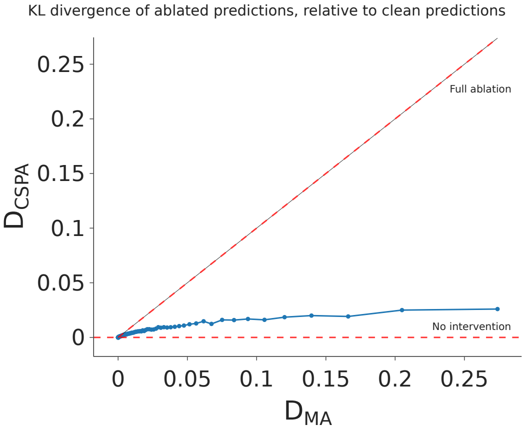


Figure 5: KL divergence to the original model under different ablations.

5 Copy Suppression and Self-Repair

In this section we focus on a specific distribution rather than the whole natural language training distribution in order to study self-repair and its relation to copy suppression.

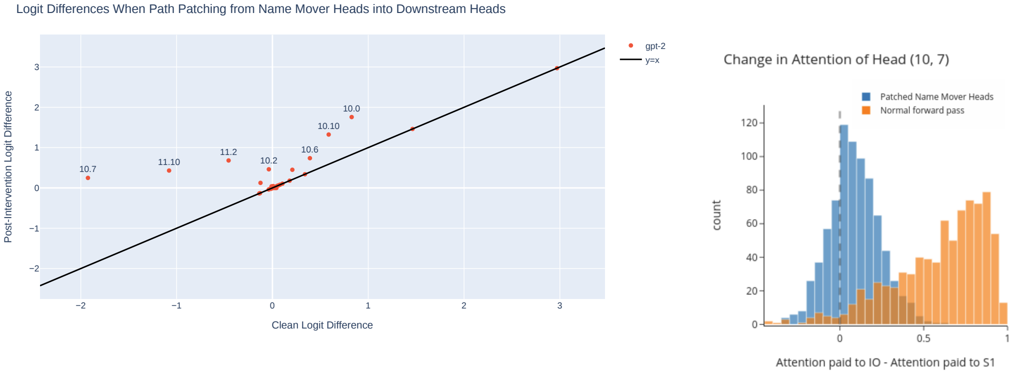
Indirect Object Identification (Wang et al., 2023, **IOI**) is a phenomenon in natural language where sentences such as "When John and Mary went to the store, Mary gave a bottle of milk to" are completed with the indirect object " John". GPT-2 Small has a circuit for choosing the name " John" over the name " Mary" on a narrow distribution of similar sentences (the **IOI Distribution**). We refer to the indirect object token (" John") as **IO** and the first instance of the subject token (" Mary") as **S1**, and frequently measure the logit difference between the final logits of the IO and S1 token. Wang et al. (2023) also used an **ABC distribution** with no repeated names to perform patching experiments.

In Wang et al. (2023), the Negative Heads were introduced without an explanation of their behavior. Copy Suppression provides a simple explanation: responding to the earlier layer's prediction of the IO token, the Negative Heads suppress it. But a more important consequence of Copy Suppression is that it helps explain portions of the Self-Repair (McGrath et al., 2023) within the IOI distribution.

In the context of the IOI task, ablating an upstream **Name Mover Head** - responsible for writing the IO token to the final token position - causes some downstream heads to 'repair' the lost logit difference created by the Name Mover Head. We focus on the following key self-repair heads in GPT2-Small:

- Backup Heads L10H2, L10H6, L10H10 and L11H2, and Name Mover Head L10H0 - characterised by responding to the ablation of a head by imitating the original behavior
- Negative Heads L10H7 and L11H10 - characterised by responding to the ablation of a head by performing less of a behavior that opposes the effect of the ablated head

In this section, we aim to gain insights into how GPT-2 Small communicates signals relevant to self-repair: **the copy suppression phenomena suggests that the unembedding directions may be an important signal for self-repair**. We investigate the extent to which this general hypothesis (Section 5.2), as well as copy suppression specifically (Section 5.1), explain self-repair in the context of the direct effects on the output logits.



(a) Ablating the Name Mover Heads in Layer 9 causes a change in the direct effects of all the downstream heads. Plotting the Clean Logit Difference vs the Post-Intervention Logit Difference for each head highlights whether a head performed self-repair or not, depending on if it is above the $y = x$ line.

(b) Measuring attention paid to names when editing the input Negative Heads receive from Name Mover Heads

Figure 6: Self-repair and copy suppression in the IOI task.

5.1 Relating Copy Suppression to Self-Repair

Upon ablation of the Name Mover Heads, many downstream heads perform self-repair: Figure 6a depicts a variety of heads recovering the lost logit difference from the Name Mover Heads when path patching from a sample ablated Name Mover Head into a downstream head. Recall that while many of these heads lead to the same outcome - recovered logit difference - the actual mechanism by which this occurs may differ.

Copy Suppression explains the mechanism behind the self-repair performed in the Negative Heads: the upstream Name Mover Heads reduces copying of the indirect object (IO) token, causing less copy-suppression behavior within the Negative Heads. Indeed, we edited the input that the Negative Heads receive from the Name Mover heads by replacing it with an activation from the ABC distribution. We then measured the difference between the attention that the negative head paid to the IO token compared to the S token. We found that the Negative Heads now attended equally to the IO and the S1 token, as the average IO attention minus S1 attention was just 0.08 for Head L10H7 and 0.0006 for Head L11H10 (Figure 6b).

To get a quantitative measurement to the extent Copy Suppression explains self-repair, we observe changes in logit difference between the IO and S1 token within individual components in the model. By sample ablating the IOI task with the ABC distribution in the Name Mover Heads, we measure the ratio between the change in logit difference in the Negative Heads to the *all* downstream heads. The Negative Heads account for 39% of all of the change in downstream behavior, indicating that Copy Suppression helps explain over a third of the self-repair in the model.

What sort of mechanisms may enable self-repair? Given that Copy Suppression accounts for a significant portion of the self-repair in the Negative Heads, this suggests that the Negative Heads are primarily responding to the copied IO embeddings from the Name Movers. However, this is one possible way out of many that two heads could cause Anti-Erasure. To capture more general forms of Anti-Erasure (and self-repair), we introduce the Static Backup Identity Score.

The SBIS between two attention heads signifies the reactivity of one head to another head operating on a specific token. Specifically, the SBIS between an attention head with OV matrix W_{OV} (the OV-head) and a second head with QK matrix W_{QK} (the QK-head) is a measure of how close

$$W_{EE}W_{QK}^{\top}W_{OV}W_{EE}^{\top} \tag{4}$$

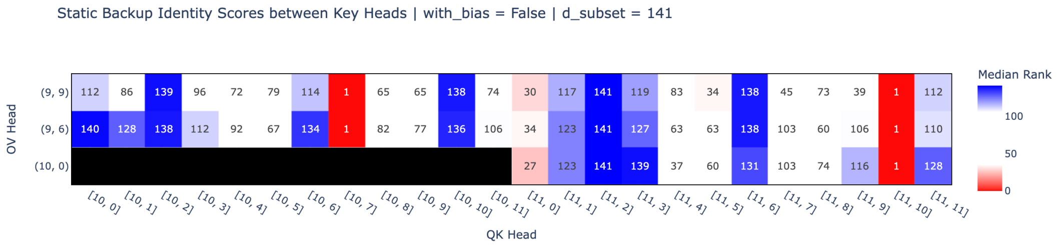


Figure 7: A graph of the Static Backup Identity Scores between the Name Mover Heads (on the OV side) and Layer 10 and 11 Heads

is to the identity or negative identity matrix.[8] This $\mathbb{R}^{d_{\text{subset}} \times d_{\text{subset}}}$ matrix combines an OV and QK circuit with a $W_{EE} \in \mathbb{R}^{d_{\text{subset}} \times d_{\text{model}}}$ matrix of the extended embeddings of various names. We approximate the matrix's similarity to the identity by looking at the median rank of the diagonal elements in their column. The resulting SBIS ranges from 1 to d_{subset} and indicates the similarity of the matrix to the positive or negative identity, respectively.

Intuitively, the $d_{\text{subset}} \times d_{\text{subset}}$ matrix represents attention scores between pairs of names. Each attention score is calculated between a query-side vector multiplied by the W_{OV} matrix for one head, dotted with the key-side embedding vector. For pairs of query-side token A and key-side token B , a low attention score indicates that at a specific token position, if the OV-head attends to A then the QK-head is unlikely to attend to B as a result; conversely, a high score indicates that the QK-head is more likely to attend to B if the OV-head attended to A. Thus, SBIS scores close to one indicate that the matrix is similar to the positive identity, and further indicate the QK-head is likely to attend to the name X given that the OV-head attended to X.

Figure 7 shows these scores between pairs of these heads. The Negative Heads L10H7 and L11H10 have very small SBIS with upstream Name Mover Heads in comparison to other heads.

Importantly, when combining SBIS with whether or not the downstream head has a positive or negative identity OV circuit (Section 4.1), we see that the self-repair mechanism in backup and negative heads are qualitatively different.

Head Type	**Response to Name Movers predicting T**	**Effect of attending to T**
Negative	More attention to T	Decrease logits on T
Backup	Less attention to T	Increase logits on T

Table 2: Qualitative Difference between Negative and Backup Heads

The "anti-erasure" approach towards self-repair, where a head reduces a negative behavior upon a changed output of an upstream head, can be implemented by a model in many ways. However, interestingly, the Negative Heads are the only heads with a small SBIS with the Name Mover Heads. This indicates that copy-suppression may qualitatively explain the "anti-erasure" occuring within the model, rather than another general mechanism the model may have learned.

5.2 Complicating the Story: Component Intervention Experiments

While copy suppression seems to explain an important mechanism behind self-repair in the Negative Heads, it cannot explain the full picture of self-repair. As introduced at the end of Section 5.1, the self-repair performed in Backup Heads is qualitatively different to self-repair in Negative Heads.

[8] We omit output query and key value biases for simplicity.

Copy suppression suggests that rather than the copy suppression mechanism specifically, perhaps all of self-repair can be explained by changes in the unembedding more generally. However, we present two pieces of evidence to highlight how the unembedding explains only part of the self-repair in the model.

First, we intervened on the output of the Name Mover Heads and L10H7,[9] and path patched the resulting changes into the queries of downstream heads. The intervention was either a projection *onto* or *away from* the IO unembedding. We also freeze LayerNorm.

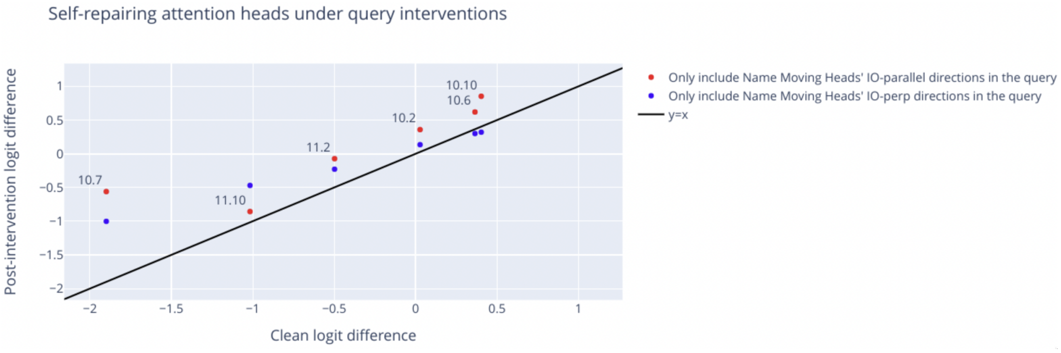


Figure 8: Path Patching from Name Mover Heads and L10H7 into the queries of downstream heads.

Figure 8 shows that while there are hints of self-repair that occur when when removing the IO direction from the outputs of the Name Mover Heads and L10H7, there is more self-repair when constraining the NMH outputs to the IO directions. As these backup heads respond less to the loss of the IO direction than the IO-perpendicular direction, this indicates that there is important information in the IO-perpendicular direction which helps control for self-repair outside of the unembedding.

To complement this analysis, we also broke the attention score (a quadratic function of query and key inputs) down into terms and again found the importance of the perpendicular direction (Appendix J).

While we currently do not understand this perpendicular direction, we are not confident that it is meaningfully different than the mechanism used for copy suppression. It could potentially be related to *semantic* copying of related but different tokens (Appendix I) or to a communication mechanism between specific heads that we do not yet understand.

Ultimately, we conclude that while fragments of copy suppression exist within these heads, the self-repair occurring across both the Erasure and Backup Heads is more general.

6 Conclusion

Write the conclusion after seeing the strength of the projection etc. results (ongoing research).

Author Contributions

Write Author Contributions after we're done.

Acknowledgments

Corrective Components were conceived carefully by Callum Conmy and Cody, collectively. Combining creative capabilities with code, these compiled collection of component conquers concerns and critical confusions and Connor commented with Joseph Jett Jonathan Jake

[9] We also ablate the output of L10H7 due to self-repair that occurs between L11H10 and L10H7.

Joseph (much MATS and McGrath mentorship, McDougall mentions) and Atticus Geiger which doesn't alliterate :-(Make this sensible after we've done with the writeup.

References

Jay Alammar. The illustrated gpt-2, 2019. URL http://jalammar.github.io/illustrated-gpt2/. Online; accessed 23. Aug. 2023.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations, 2017.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens, 2023.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Tolga Bolukbasi, Adam Pearce, Ann Yuan, Andy Coenen, Emily Reif, Fernanda Viégas, and Martin Wattenberg. An interpretability illusion for bert. *arXiv preprint arXiv:2104.07143*, 2021.

Nick Cammarata, Shan Carter, Gabriel Goh, Chris Olah, Michael Petrov, Ludwig Schubert, Chelsea Voss, Ben Egan, and Swee Kiat Lim. Thread: Circuits. 2020. doi: 10.23915/distill.00024. https://distill.pub/2020/circuits.

Nick Cammarata, Gabriel Goh, Shan Carter, Chelsea Voss, Ludwig Schubert, and Chris Olah. Curve circuits. *Distill*, 2021. doi: 10.23915/distill.00024.006. https://distill.pub/2020/circuits/curve-circuits.

Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas. *Distill*, 4(3):e15, 2019.

Lawrence Chan, Adria Garriga-Alonso, Nix Goldowsky-Dill, Ryan Greenblatt, Jenny Nitishinskaya, Ansh Radhakrishnan, Buck Shlegeris, and Nate Thomas. Causal scrubbing: A method for rigorously testing interpretability hypotheses. Alignment Forum, 2022. URL https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability, 2023.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. Analyzing transformers in embedding space. *arXiv preprint arXiv:2209.02535*, 2022.

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. URL https://transformer-circuits.pub/2021/framework/index.html.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks, 2021. URL https://arxiv.org/abs/2106.02997.

Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. Multimodal neurons in artificial neural networks. *Distill*, 6(3):e30, 2021.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching, 2023.

Klaus Greff, Rupesh K. Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation, 2017.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing, 2023.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. Overthinking the truth: Understanding how language models process false demonstrations, 2023.

Michael Hanna, Ollie Liu, and Alexandre Variengien. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model, 2023.

Stefan Heimersheim and Jett Janiak. A circuit for Python docstrings in a 4-layer attention-only transformer, 2023. URL https://www.alignmentforum.org/posts/u6KXXmKFbXfWzoAXn/a-circuit-for-python-docstrings-in-a-4-layer-attention-only.

Mengting Hu, Zhen Zhang, Shiwan Zhao, Minlie Huang, and Bingzhe Wu. Uncertainty in natural language processing: Sources, quantification, and applications, 2023.

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness?, 2020.

Tom Lieberum, Matthew Rahtz, János Kramár, Neel Nanda, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. Does circuit analysis interpretability scale? evidence from multiple choice capabilities in chinchilla, 2023.

Thomas McGrath, Matthew Rahtz, Janos Kramar, Vladimir Mikulik, and Shane Legg. The hydra effect: Emergent self-repair in language model computations, 2023.

Joseph Miller and Clement Neo. We found an neuron in gpt-2. *AI Alignment Forum*, Feb 2023. URL https://www.alignmentforum.org/posts/cgqh99SHsCv3jJYDS/we-found-an-neuron-in-gpt-2.

Jesse Mu and Jacob Andreas. Compositional explanations of neurons. *CoRR*, abs/2006.14032, 2020. URL https://arxiv.org/abs/2006.14032.

Neel Nanda and Joseph Bloom. Transformerlens, 2022. URL https://github.com/neelnanda-io/TransformerLens.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

nostalgebraist. interpreting gpt: the logit lens, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Chris Olah. Mechanistic interpretability, variables, and the importance of interpretable bases. https://www.transformer-circuits.pub/2022/mech-interp-essay, 2022.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment, 2017.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works, 2020.

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks, 2023.

Jesse Vig. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 37–42. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-3007. URL `https://aclanthology.org/P19-3007`.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned, 2019.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=NpsVSN6o4ul`.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022.

Zhengxuan Wu, Atticus Geiger, Christopher Potts, and Noah D. Goodman. Interpretability at scale: Identifying causal mechanisms in alpaca, 2023.

A Anti-induction

As one example of behaviors which copy-suppression seems to explain outside the context of IOI, we present the phenomenon of **anti-induction**. Attention heads have been discovered in large models which identify repeating prefixes and suppressing the prediction of the token which followed the first instance of the prefix, in other words the opposite of the induction pattern (Olsson et al., 2022). Analysis across different model architectures revealed a strong correlation between attention heads' copying scores on random sequences of repeated tokens (i.e. the induction task) and their copy-suppression scores on the IOI task, in the quadrant where both scores were positive. For example, head L10H7 in GPT-2 Small ranked higher than all other attention heads in both copy-suppression and negative induction score.

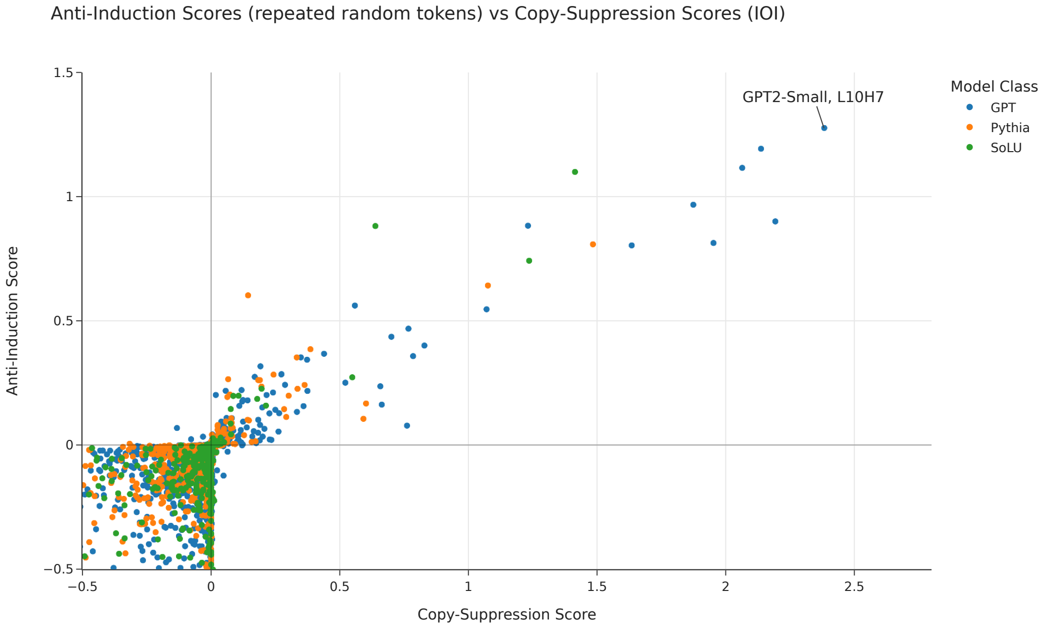


Figure 9: Anti-induction and copy suppression on the IOI task compared.

Importantly, since the induction task involves a repeated sequence of random tokens, this graph strongly suggests that the negative behavior displayed by certain heads on the IOI task is not task-dependent. We believe this holds a generalisable lesson for mechanistic interpretability - **certain components can appear to be using task-specific algorithms, but are actually implementing a more general pattern of behaviour.**

B Copy Suppression in other models

We have performed the experiment in Section 4.2 on all heads in GPT-2 Medium: Figure 10. We found that the two heads most prominently recovered were 2/3 of the most negative heads on the IOI task in GPT-2 Medium (Figure 11).

We also find instances of copy suppression (though weaker) in the Pythia models that were trained without copy suppression (Figure 9).

C Entropy and Calibration

A naive picture of attention heads is that they should all reduce the model's entropy (because the purpose of a transformer is to reduce entropy by concentrating probability mass in the few most likely next tokens). We can calculate a head's direct contribution to entropy by measuring (1) the entropy of the final logits, and (2) the entropy of the final logits with the head's output subtracted. In both cases, the negative head L10H7 stands out the most, and the other negative heads L11H10 and L8H10 are noticeable.

We can also examine each attention head's effect on the model's calibration. Hu et al. (2023) use **calibration curves** to visualise the model's degree of calibration. From this curve, we can define an **overconfidence metric**, calculated by subtracting the perfect calibration curve from the model's actual calibration curve, and taking the normalized L_2 inner product between this curve and the curve we get from a perfectly overconfident model (which only ever makes predictions of absolute certainty). The L_2 inner product can be viewed as a measure of similarity of functions, so this metric should tell us in some sense how overconfident our model is: the value will be 1 when the model is perfectly overconfident, and 0 when the model is perfectly calibrated. Figure 13 illustrates these concepts.

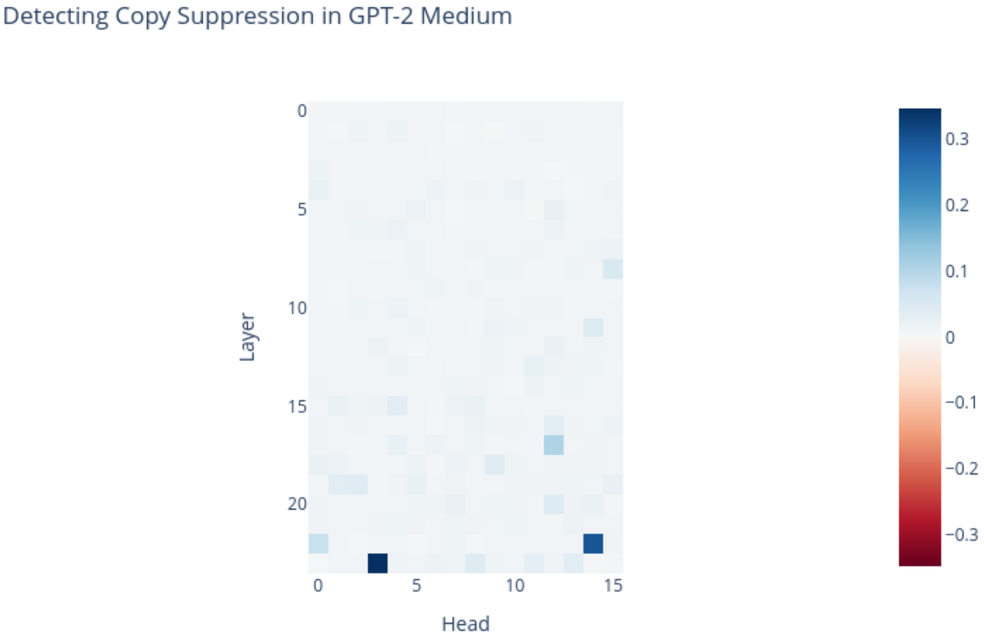


Figure 10: Repeating the experiment in Section 4.2 (with W_{EE} keyside and W_U queryside) on GPT-2 Medium.

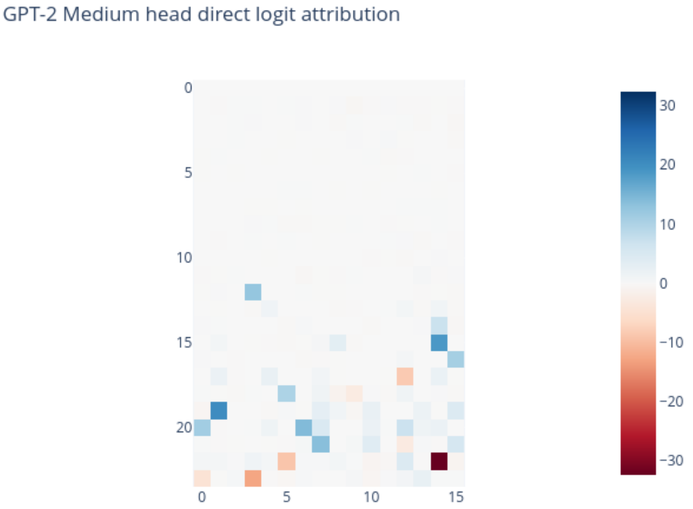
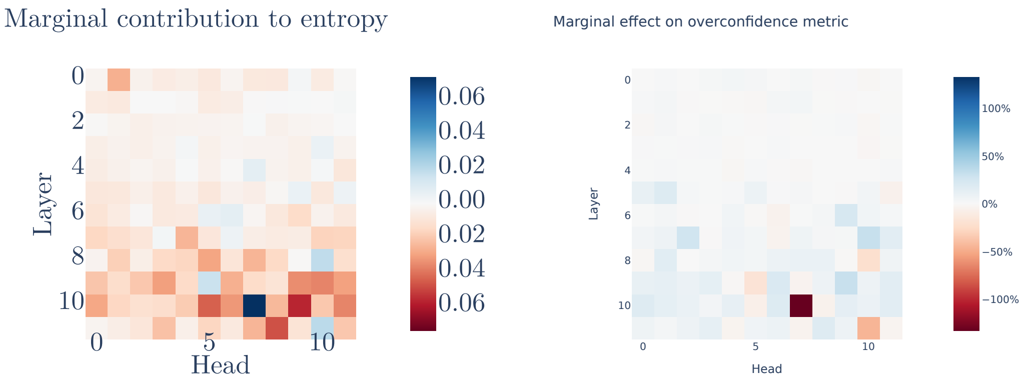


Figure 11: Finding the direct logit attribution for different heads in GPT-2 Medium on the IOI task.



(a) Entropy contribution per head. L10H7 increases entropy (as do other negative heads like L11H10); most other heads decrease it.

(b) Marginal effect on overconfidence metric per head. L10H7 decreases overconfidence; most other heads increase it.

Figure 12: Effect of attention heads on entropy & calibration.

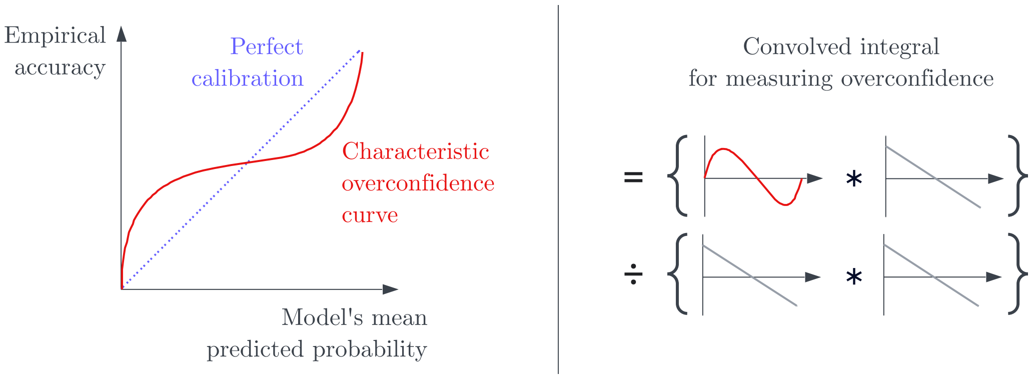


Figure 13: Illustration of the calibration curve, and overconfidence metric.

We can then measure the change in overconfidence metric from ablating the direct effect of an attention head, and reverse the sign to give us the head's direct effect on overconfidence. This is shown in the figure below, with the change shown relative to the model's original overconfidence (with no ablations). Again, we see that head L10H7 stands out, as do the other two negative heads. Interestingly, removing the direct effect of head L10H7 is enough to push the model from net over-confident to net under-confident.

What are we to interpret from these results? It is valuable for a model to not be overconfident, because the cross-entropy loss will be high for a model which makes high-confidence incorrect predictions. One possible role for negative heads is that they are reducing the model's overconfidence, causing it to make fewer errors of this form. However, it is also possible that this result is merely incidental, and not directly related to the reason these heads form. For example, another theory is that negative heads form to suppress early naive copying behaviour by the model, and in this case they would be better understood as copy-suppression heads rather than "calibration heads". See the next section for more discussion of this.

D Why do negative heads form? Some speculative theories

This paper aimed to mechanistically explain what heads like L10H7 do, rather than to provide an explanation for why they form. We hope to address this in subsequent research. Here, we present three possible theories, present some evidence for/against them, and discuss how we might test them.

- **Reducing model overconfidence.**
  - **Theory**: Predicting a token with extremely high confidence has diminishing returns, because once the logprobs are close to zero, any further increase in logits won't decrease the loss if the prediction is correct, but it will increase loss if the prediction is incorrect. It seems possible that negative heads form to prevent this kind of behaivour.
  - **Evidence**: The results on calibration and entropy in Appendix C provide some evidence for this (although these results aren't incompatible with other theories in this table).
  - **Tests**: Examine the sequences for which this head decreases the loss by the most (particularly for checkpointed models, just as the negative head is forming). Are these cases where the incorrect token was being predicted with such high probability that it is in this "diminishing returns" window?
- **Suppressing naive copying.**
  - **Theory**: Most words in the English language have what we might term the "update property" - the probability of seeing them later in a prompt positively updates when they appear. Early heads might learn to naively copy these words, and negative heads could form to suppress this naive behaviour.
  - **Evidence**: The "All's fair in love and love" prompt is a clear example of this, and provides some evidence for this theory.
  - **Tests**: Look at checkpointed models, and see if negative heads form concurrently with the emergence of copying behaviour by other heads.
- **Suppressing next-token copying for tied embeddings.**
  - **Theory**: When the embedding and unembedding matrices are tied, the direct path $W_U W_E$ will have large diagonal elements, which results in a prediction that the current token will be copied to the next sequence position. Negative heads could suppress this effect.
  - **Evidence**: This wouldn't explain why negative heads appear in models without tied embeddings (although it might explain why the strongest negative heads we found were in GPT-2 Small, and the Stanford GPT models, which all have tied embeddings).
  - **Tests**: Look at attention patterns of the negative head early in training (for checkpointed models, with tied embeddings). See if tokens usually self-attend.

While discussing these theories, it is also important to draw a distinction between the reason a head forms during training, and the primary way this head decreases loss on the fully trained model - these two may not be the same. For instance, the head seems to also perform semantic copy suppression (see Appendix I), but it's entirely possible that this behaviour emerged after the head formed, and isn't related to the reason it formed in the first place.

E Experiment details for OV-Circuit in practice.

- Run a forward pass on a sample of OpenWebText,
- Filter for all (source, destination) token pairs where the attention from destination to source is above some threshold (we chose 10%),
- Measure the direct logit attribution of the information moved from each of these source tokens to the corresponding destination token,
- Perform the same experiment as we did in the static analysis section (measuring the rank of the source token in this direct logit attribution).

We found that the results approximately matched our dynamic analysis (with slightly more noise). The proportion of (source, destination) token pairs where the source token is in the top 10 most suppressed tokens is 78.24% (which is close to the static analysis result of 84.70%).

F Function Words

In Section 4.1 we found that a large fraction of the tokens which failed to be suppressed were function words. The list of least copy suppressed tokens are: [‘ of’, ‘ Of’, ‘ that’, ‘ their’, ‘ most’, ‘ as’, ‘ this’, ‘ for’, ‘ the’, ‘ in’, ‘ to’, ‘ a’, ‘Their’, ‘ Its’, ’When’, ‘ The’, ‘ its’, ‘ these’, ‘The’, ‘Of’, ‘ it’, ‘ nevertheless’, ‘ an’, ‘`<|endoftext|>`, ’Its’, ‘ have’, ‘ some’, ‘ By’]. Sampling randomly from the 3724 tokens other than 92.59% that are copy suppressed, many are also connectives (and rarely nouns): [‘ plainly’, ‘ utterly’, ‘ enhance’, ‘ obtaining’, ‘ entire’, ‘ Before’, ‘eering’, ‘.)’, ‘ holding’, ‘ unnamed’].
It is notable that this result is compatible with all three theories which we presented in the previous section.

- **Reducing model overconfidence**. The unembedding vectors for function words tend to have smaller magnitude than the average token in GPT-2 Small. This might lead to less confident predictions for function words than for other kinds of tokens.
- **Suppressing naive copying**. There would be no reason to naively copy function words, because function words don't have this ”update property” - seeing them in a prompts shouldn't positively update the probability of seeing them later. So there is no naive copying which needs to be suppressed.
- **Suppressing next-token copying for tied embeddings**. Since function words' unembedding vectors have smaller magnitudes, the diagonal elements of $W_U W_E$ are small anyway, so there is no risk of next-token copying of function words.

G Model and General Experiment Details

All of our experiments were performed with Transformer Lens (Nanda & Bloom, 2022). They can be found at `https://github.com/callummcdougall/SERI-MATS-2023-Streamlit-pages` and we note that we enable all weight processing options,[10] which means that transformer weight matrices are rewritten the internal components are much simpler (though the output probabilities are identical). For example, our Layer Norm functions only apply normalization, with no centering or rescaling (this particular detail significantly simplifies our Logit Lens experiments).

[10] That are described here: `https://github.com/neelnanda-io/TransformerLens/blob/main/further_comments.md#weight-processing`

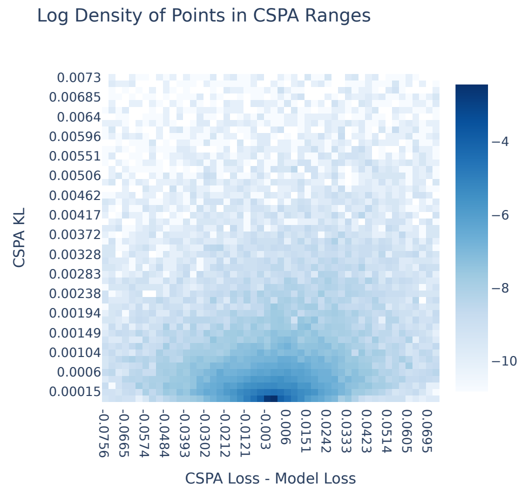


Figure 14: Log densities of dataset examples with loss change due to CSPA (x axis) and KL divergence due to CSPA (y axis). The x axis range is between -1 and $+1$ standard deviation of loss changes due to CSPA, and the y axis range is between 0 and $+1$ standard deviation of CSPA KL.

H CSPA Metric Choice

H.1 Motivating KL Divergence

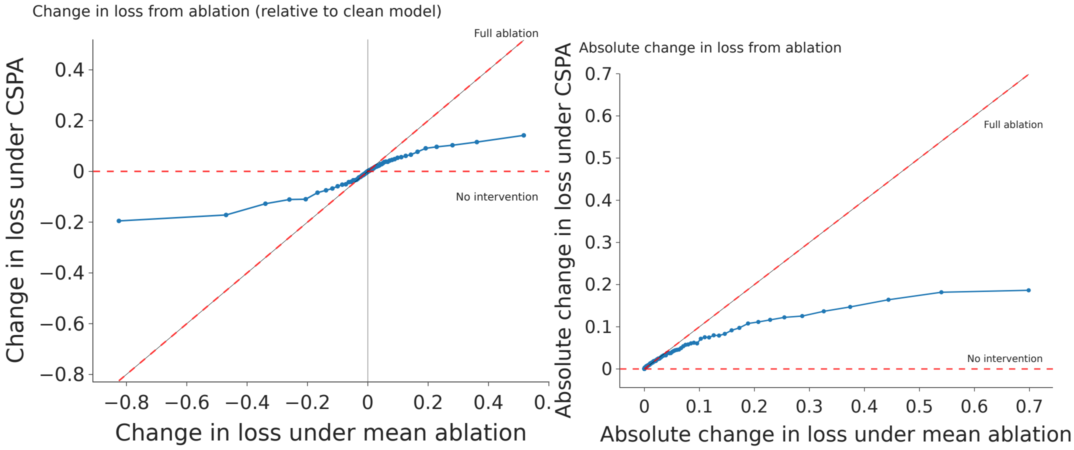
To measure the effect of an ablation, we primarily focused on the KL divergence $D_{KL}(P\|Q) = \sum_i p_i \log p_i/q_i$, where P was the clean distribution and Q was the distribution after our ablation had been applied. Conveniently, a KL Divergence of 0 corresponds to perfect recovery of model behavior, and it is linear in the log-probabilities $\log q_i$ obtained after CSPA.

There are flaws with the KL divergence metric. For example, if the correct token probability is very small, and a head has the effect of changing the logits for this token (but not enough to meaningfully change the probability), this will affect loss but not KL divergence. Our copy suppression preserving ablation on L10H7 will not preserve situations like these, because it filters for cases where the suppressed token already has high probability. Failing to preserve these situations won't change how much KL divergence we can explain, but it will reduce the amount of loss we explain. Indeed, the fact that the loss results appear worse than the KL divergence results is evidence that this is happening to some extent.Indeed empirically, we find that density of points with KL Divergence close to 0 but larger change in loss is greater than the opposite (change in loss close to 0 but KL larger) in Figure 14, as even using two standard deviations of change on the x axis leads to more spread acrosss that axis. In Appendix H.2 we present results on loss metrics to complement our KL divergence results, and we compare these metrics to baselines in Appendix H.3.

H.2 Comparing KL Divergence and Loss

In Figure 2, we use two different metrics to capture the effect and importance of different model components. Firstly, the amount by which ablating these components changes the average cross-entropy loss of the model on OpenWebText. Secondly, the KL Divergence of the ablated distribution to the model's ordinary distribution, again on OpenWebText. In essence, the first of these captures how useful the head is for the model, and the second captures how much the head affects the model's output (good or bad). In Section 4.3 we only reported the recovered effect from KL divergence. We can also compute analogous quantities to Equation (3) for loss, in two different ways.

Following the ablation metric definition in Section 4.3.1, suppose at one token completion GPT-2 Small usually has loss L, though if we ablate of L10H7's direct effect has loss L_{Abl}. Then we could either measure $L_{\text{Abl}} - L$ and try and minimise the average of these values over



(a) Average change in loss effect.

(b) Absolute change in loss effect recovered.

Figure 15: Studying CSPA under metrics other than KL Divergence.

the dataset, or we could instead minimize $|L_{\text{Abl}} - L|$. Either way, we can compare CSPA (Abl = CSPA) to the baseline of mean ablation (Abl = MA), by a similar ratio calculation as Equation (3). We get 82% effect recovered for the net loss effect and 45% effect recovered for the absolute change in loss. Despite these differing point values, the same visualisation method as Section 4.3.2) can be used to see where Copy Suppression is not explaining L10H7 behavior well (see Figure 15). We find that the absolute change in loss captures the majority of the model's (73.3%) in the most extreme change in loss percentile (Figure 15b, far right), which shows that the heavy tail of cases where L10H7 is not very useful for the model is likely the reason for the poor performance by the absolute change in loss metric.

Also, surprisingly Figure 15a's symmetry about $x = 0$ shows that there are almost as many completions on which L10H7 is harmful as there are useful cases. We observed that this pattern holds on a random sample of OpenWebText for almost all Layer 9-11 heads, as most of these heads have harmful direct effect on more than 25% of completions, and a couple of heads (L8H10 and L9H5) are harmful on the majority of token completions (though their average direct effect is beneficial).

H.3 Does Equation (3) accurately measure the effect explained?

If Equation (3) is a good measure of the copy suppression mechanism, it should be smaller for heads in GPT-2 Small that aren't negative heads. We computed the CSPA value for all heads in Layers 9-11 in Figure 16.[11] We also ran two forms of this experiment: one where we projected OV-circuit outputs onto the unembeddings (right), and one where we only kept the negative components of OV-circuit outputs (left).

While we find that CSPA recovers more KL divergence L10H7 than all other heads, we also find that the QK and OV ablations (Section 4.3.1) lead to large ($> 50\%$) KL divergence recovered for many other heads, too. In ongoing experiments, we're looking into projection ablations on the QK circuit that will likely not recover as much KL divergence for other heads.

I Semantic Similarity

42.00% of (source, destination) pairs had the source token in the top 10 most suppressed tokens, but not the most suppressed. When we inspect these cases, we find a common theme: the most suppressed token is often semantically related to the source token. For our

[11] All attention heads in Layers 0-8 have small direct effects: the average increase in loss under mean ablation of these direct effects is less than 0.05 for all these heads, besides 8.10. However heads in later layers have much larger direct effects, e.g 10/12 attention heads in Layer 10 (including L10H7) have direct effect more than 0.05.

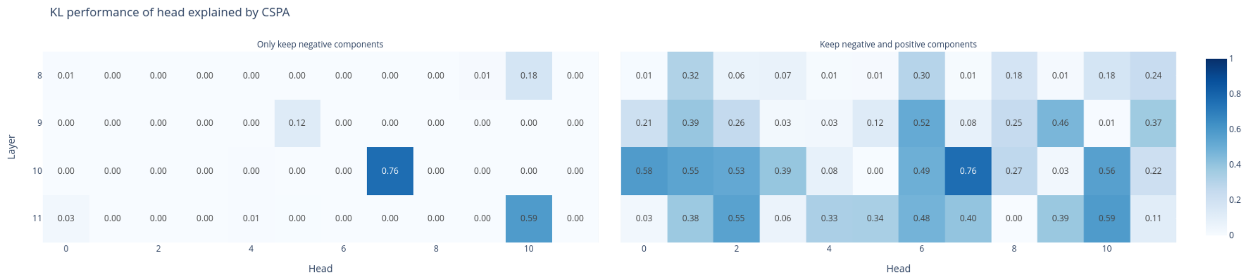


Figure 16: Calculating CSPA (with KL divergence) for all Layer 9-11 heads in GPT-2 Small.

purposes, we define **semantically related** as an equivalence relation on tokens, where if tokens S and T are related via any of the following:

- Capitalization (e.g. " pier" and " Pier" are related),
- Prepended spaces (e.g. " token" and "token" are related),
- Pluralization (e.g. " device" and " devices" are related),
- Sharing the same morphological root (e.g. "drive", "driver", "driving" are all related)
- Tokenization (e.g. " Berkeley" and "keley" are related, because the non-space version "Berkeley" is tokenized into ["Ber", "keley"]).

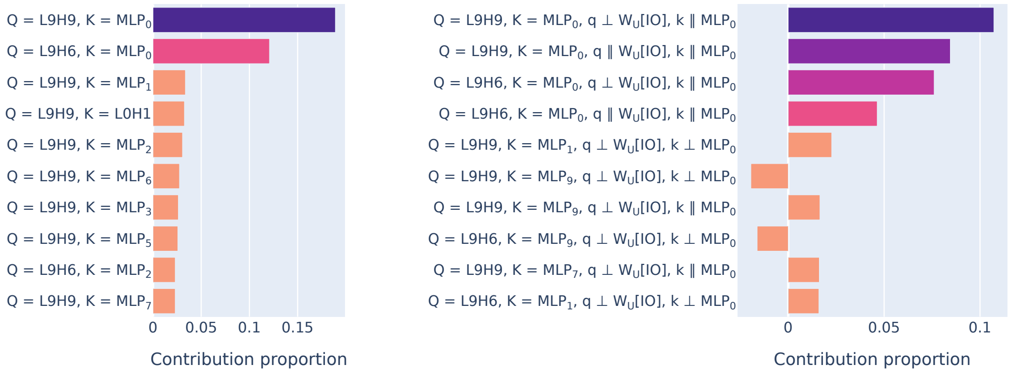
We codify these rules, and find that in 90% of the aforementioned cases, the most suppressed token is semantically related to the source token. Although part of this is explained by the high cosine similarity between semantically related tokens, this isn't the whole story (on this set of examples, the average cosine similarity between the source token and the semantically related most suppressed token was 0.520). We speculate that the copy suppression algorithm is better thought of as **semantic copy suppression**, i.e. all tokens semantically related to the source token are suppressed, rather than **pure copy suppression** (where only the source token is suppressed). The figure below presents some OpenWebText examples of copy suppression occurring for semantically related tokens.

Prompt	**Source token**	**Incorrect completion**	**Correct completion**	**Form of semantic similarity**
...America's private **prisons** ... the biggest private ~~**prison**~~ **-** ...	**" prisons"**	**" prison"**	**"-"**	Pluralization
...Steam**VR** (formerly known as Open**VR**), Valve's alternate ~~**VR**~~ **reality** ...	**"VR"**	**" VR"**	**" reality"**	Prepended space
...Ber**keley** to offer course ... university of ~~**Berkeley**~~ **California** ...	**"keley"**	**" Berkeley"**	**" California"**	Tokenization
...**Wrap** up the salmon fillets in the foil, carefully ~~**wrapping**~~ **sealing** ...	**" Wrap"**	**" wrapping"**	**" sealing"**	Verb conjugation & capitalization

Table 3: Dataset examples of copy suppression, with semantic similarity.

J Breaking Down the Attention Score Bilinear Form

In Section 5, we observed that Negative Heads attend to IO rather than S1 due to the outputs of the Name Mover heads. We can use QK circuit analysis (Section 4.2) in order to



(a)

(b)

Figure 17: Decomposing the bilinear attention score. 17a: decomposing by all model components. 17b: decomposing by all model components, and further by terms in the MLP0 direction (keyside) and terms in the IO unembedding direction (queryside). Terms involving name movers and MLP0 are highlighted.

understand what parts of L10H7's query and key inputs cause attention to IO rather than S1.

As a gentle introduction to our methodology in this section, if an attention score was computed from an incoming residual stream vector q at queryside and k at queryside, then mirroring Equation (2) we could decompose the attention score

$$s = q^\top W_{QK}^{\text{L10H7}} k \tag{5}$$

into the query component from each residual stream component[12] (e.g MLP9, the attention heads in layer 9, ...) so $s = q_{\text{MLP9}}^\top W_{QK}^{\text{L10H7}} k + q_{\text{L9H0}}^\top W_{QK}^{\text{L10H7}} k + \cdots$. We could decompose the keyside input.

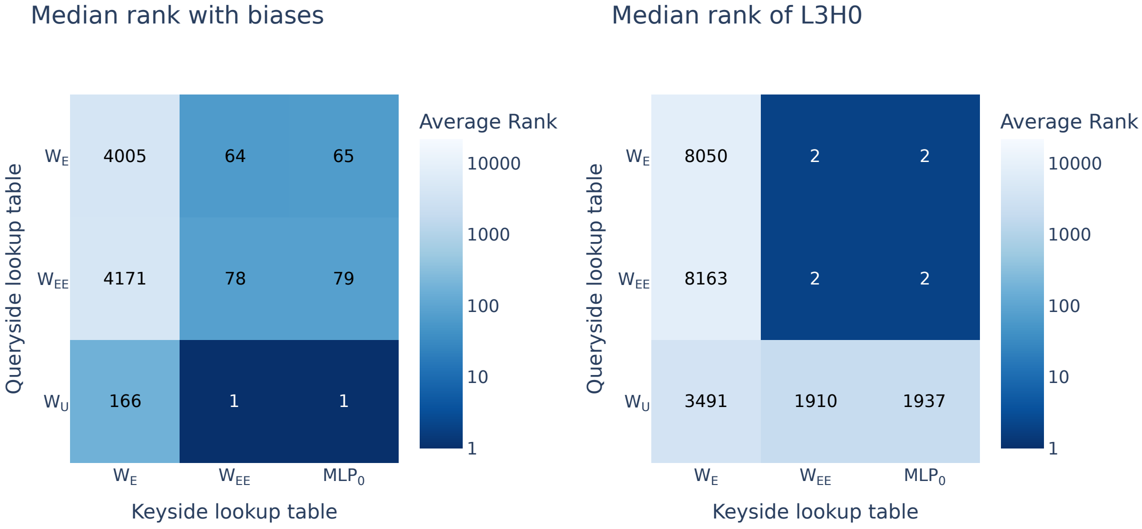
In this appendix, we're actually interested in the difference between how the model attends to IO compared to S, we actually decompose the attention score difference

$$\Delta s := q^\top W_{QK}^{\text{L10H7}} k^{\text{IO}} - q^\top W_{QK}^{\text{L10H7}} k^{\text{S1}} = q^\top W_{QK}^{\text{L10H7}} (k^{\text{IO}} - k^{\text{S1}}). \tag{6}$$

Since Δs is in identical form to Equation (5) when we take $k = k^{\text{IO}} - k^{\text{S1}}$ and q taken from the END position in the IOI task, we can decompose both the query inputs and key inputs of Δs. Under this decomposition, we find that the most contributions are from L9H6 and L9H9 queryside and MLP0 keyside (Figure 17a), which agrees with our analysis throughout the paper. Further, we can test the hypotheses in Section 4.1 and Section 4.2

that copy suppression is modulated by an unembedding vector in the residual stream, by further breaking up each of the attention scores in Figure 17a into 4 further components, for the queryside components parallel and perpendicular to the unembedding direction, as well as the keyside components parallel and perpendicular to the MLP0 direction (Figure 17b). Unfortunately the direction perpendicular to IO is slightly more important than the parallel direction, for both name movers. This supports the argument in Section 5 that self-repair is more general than the simplest possible form of copy suppression described in Section 4.2.

[12] As in Equation (2), we found that the query and key biases did not have a large effect on the attention score difference computed here.



(a) Figure 3 but including biases before multiplying query and key vectors.

(b) Figure 3 but for L3H0 (a Duplicate Token Head).

Figure 18: Repeating Figure 3 while adding biases (Figure 18a) and on a different head (Figure 18b)

K L10H7's QK-Circuit

K.1 Details on the QK-Circuit experiments (Figure 3).

We normalize the query and key inputs to norm $\sqrt{d_{\text{model}}}$ to simulate the effect of Layer Norm. Also, MLP0 in Figure 3 refers to taking the embeddings for all tokens and feeding this through MLP0 (so is identical to effective embedding besides not having W_E added). Actually, key and query biases don't affect results much so we remove them for simplicity of Equation (2). Results when we uses these biases can be found in Figure 18a. Additionally, the median ranks for other attention heads do not show the same patterns as Figure 3: for example, Duplicate Token Heads (Wang et al., 2023) have a 'matching' QK circuit that has much higher median ranks when the queryside lookup table is an embedding matrix (Figure 18b). Additionally, most other attention heads are different to copy suppression heads and duplicate token heads, as e.g for Name Mover Heads across all key and queryside lookup tables the best median rank is 561.

K.2 Making a more faithful keyside approximation

Is our minimal mechanism for Negative Heads faithful to the computation that occurs on forward passes on dataset examples? To test this, we firstly select some important key tokens which we will measure faithfulness on. We look at the top 5% of token completions where L10H7 was most useful (as in Section 3) and select the top two non-BOS tokens in context that have maximal attention paid to them. We then project L10H7's key input onto a component parallel to the effective embedding for the key tokens, and calculate the change in attention paid to the selected key tokens. The resulting distribution of changes in attention can be found in Figure 19.

We find that the median attention change is -0.09, with lower quartile -0.19. Since the average attention amongst these samples is 0.21, this suggests that the effective embedding does not faithfully capture the model's attention.

To use a more faithful embedding of keyside tokens, we run a forward pass where we set all attention weights to tokens other than BOS and the current token to 0. We then measure the state of the residual stream before input to Head L10H7, which we call the **context-free residual state**. Repeating the experiment used to generate Figure 19 but using the context-free residual state rather than the effective embedding, we find a more faithful

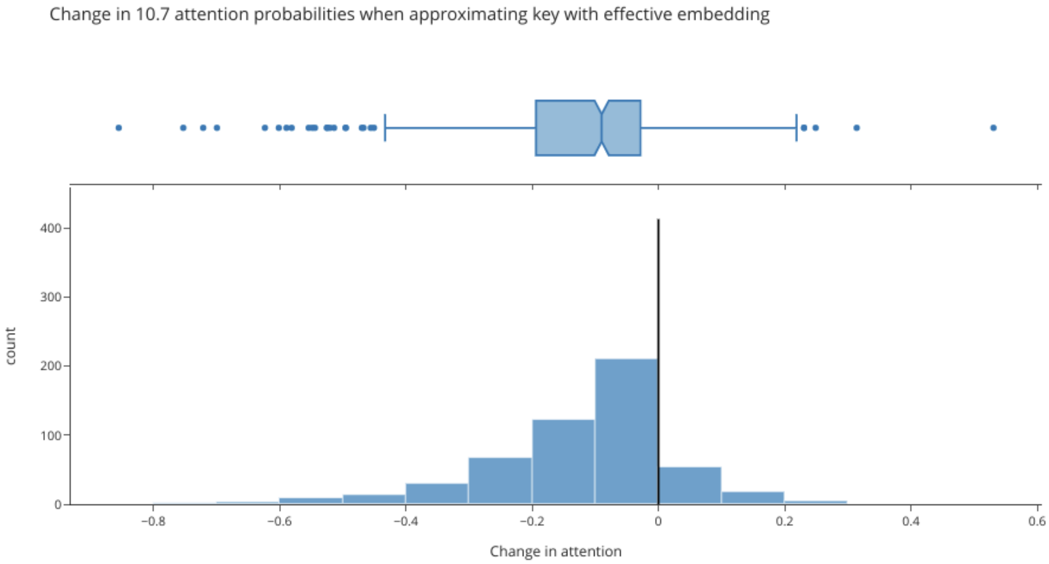


Figure 19: Change in attention on tokens when projecting key vectors onto the effective embedding for tokens.

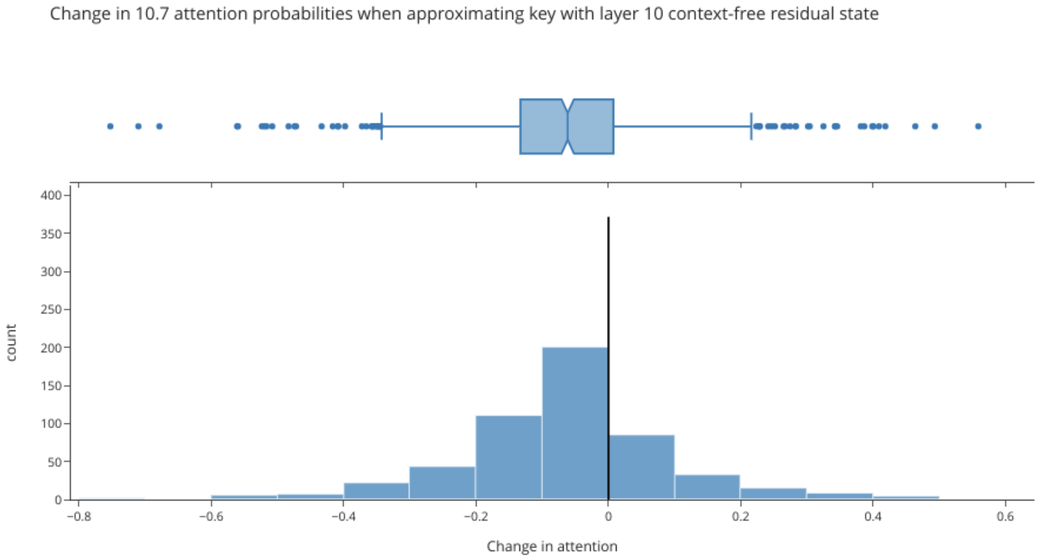


Figure 20: Change in attention on tokens when projecting key vectors onto the context free residual state.

approximation of L10H7's keyside input as Figure 20 shows that the change in L10H7's attention weights is closer to −0.06 which is closer to 0.

K.3 Making a more faithful queryside approximation

We perform a similar intervention to the components on the input to the model's query circuit. We study the top 5% of token completions where L10H7 has most important effect. For the two key tokens with highest attention weight in each of these prompts, we project the query vector onto the unembedding vector for that key token. We then recompute attention probabilities and calculate how much this differs from the unmodified model. We find that again our approximation still causes a lot of attention decrease in many cases (Figure 21).

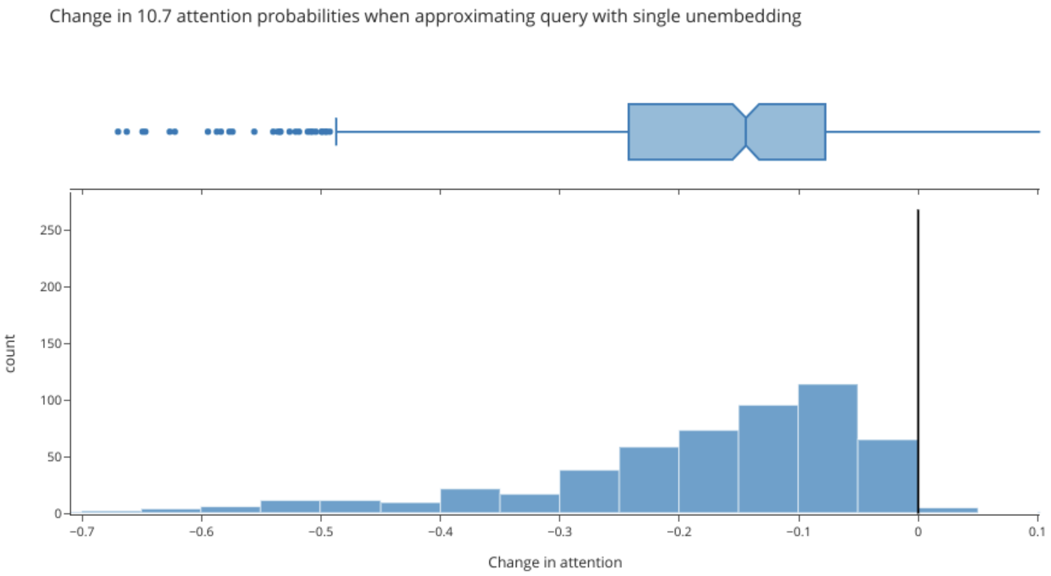


Figure 21: Change in attention on tokens when projecting query vectors onto the unembedding vectors for particular tokens.

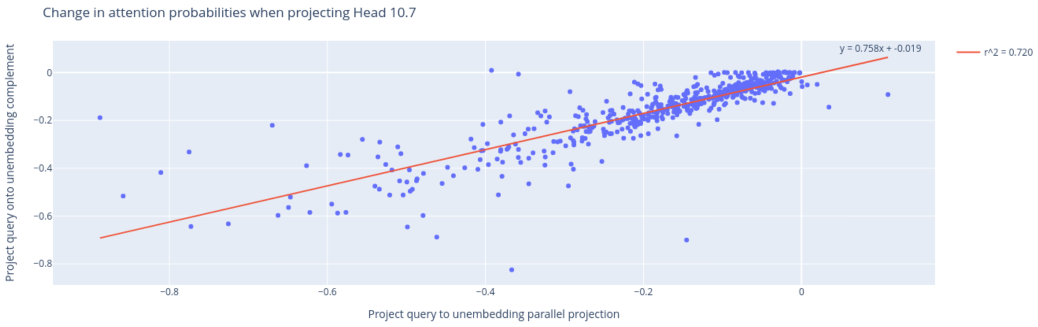


Figure 22: Correlation between change in attention on tokens when projecting onto the component parallel to the unembedding and (x-axis) and also projecting onto the component perpendicular to the unembedding (y-axis).

There is a component of the queryside input perpendicular to the unembedding direction that is important for L10H7's attention. This component seems more important for L10H7s attention when the unembedding direction is more important, by performing an identical experiment to the experiment that produced Figure 21 except projecting onto the perpendicular direction, and then measuring the correlation between the attention change for both of these interventions on each prompt, shown in Figure 22. The correlation shows that it's unlikely that there's a fundamentally different reason why L10H7 attends to tokens other than copy suppression, as if this was the case it would be likely that some points caused little attention change on projection to the unembedding direction, and large attention change on projection to the perpendicular direction, which doesn't occur often.
We're not sure what this perpendicular component represents. Section 5.2 dives deeper into this perpendicular component in the IOI case study, and Appendix J further shows that the model parts that output large unembedding vectors (the Name Mover heads) are also the parts that output the important perpendicular component.

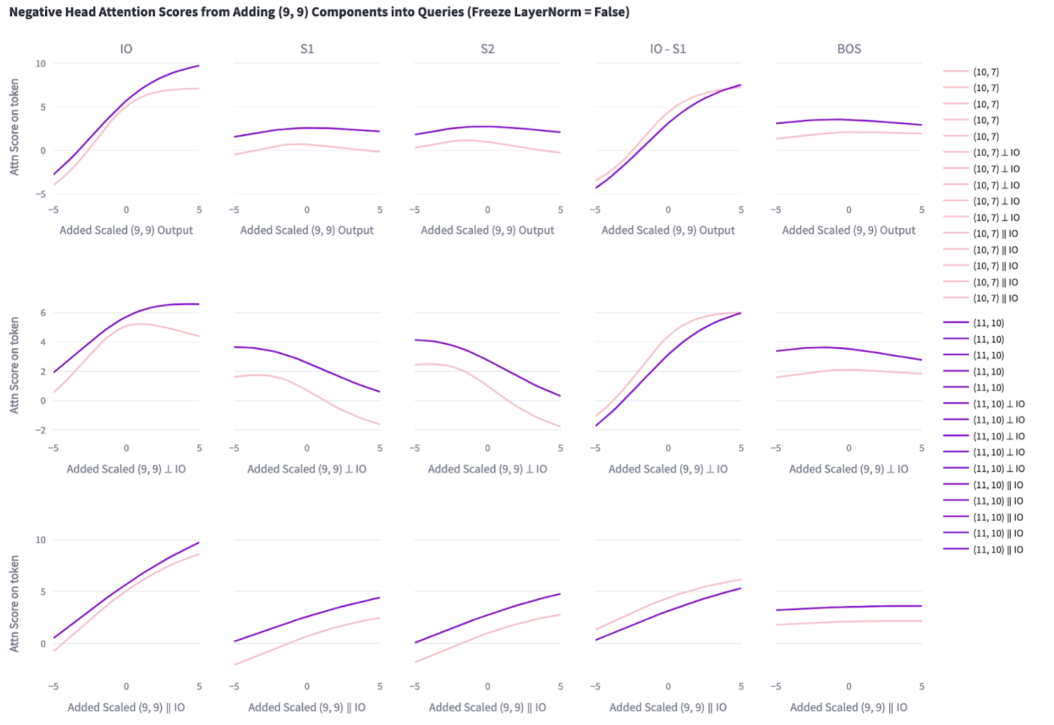


Figure 23: Observing the change in attention scores of Negative Heads upon scaling the presence of the IO unembedding in the query

L Universality of IOI Self-Repair

Since Negative Heads exist across distributions and models, we also expect that IOI self-repair potentially exists universally as well. Initial investigations across other models about self-repair via Copy Suppression highlight similarities to the phenomena we observe here but with some subtleties in the specifics. For instance, one head in Stanford GPT-2 Small E performed self-repair via a similar “” behavior, however, on a clean run, it attended to both the IO and S2 tokens equally (thus not decreasing the logit difference on a clean run, like the Negative Heads in GPT-2 Small).

M Amplifying Query Signals into Backup Heads

TO further investigate the claim that the unembedding isn’t the full story for self-repair in the Negative Heads, we intervened on the query-side component of the Negative Heads by adding in vectors of scaled lengths at various places and measuring the new attention scores or patterns of the Negative Heads. Figure X shows a specific instance in which we take the output of head 9.9, and add scaled versions of it into the residual streams of the negative heads which produce their queries (before LayerNorm). Additionally, we do an analogous operation with the projection of 9.9 onto the IO unembeddings, as well as the projection of 9.9 away from the IO unembeddings.

We see in figure 23 that there is a positive slope on the IO subgraph even when projecting the output of 9.9 onto only the IO unembedding. This implies that the greater the presence of the IO unembedding in the query of the negative name mover head, the greater the attention paid to the IO token. The result still holds whether or not we add the vector before or after LayerNorm, or whether or not we freeze LayerNorm. Using the output itself yields the same results, while using the perpendicular component is slightly more nuanced (but mostly similar) dig out pdf version of this

N Most Heads are not Monosemantic

From Neel: “Idea - it could be cool to have an appendix with Cody’s results on name movers doing other coherent tasks on other distributions. IMO the point that “most heads are not monosemantic” is non-obvious to people (it’s not obvious to me, though the arguments make sense, let alone to non-interp people) and I think this is a core part of what makes our “we found a monosemantic head” results exciting”

,

TODO: make hyperlinks for all instances of glossary words in the main text?

Glossary

Block Following Alammar (2019) and Nanda & Bloom (2022), a transformer block consists of an attention layer and the next MLP layer.

Copy suppression is a mechanism in a language models determined by the three steps **naive copying**, **attention** and **suppression**, as described in Section 1.

Copy suppression-preserving ablation (CSPA) refers to our ablation that deletes all functionality of attention head 10.7 except the copy suppression mechanism (Section 4.3).

Destination token refers to the token where we’re measuring the model’s next-token prediction..

Direct Logit Attribution is defined in `https://www.neelnanda.io/mechanistic-interpretability/glossary`.

Effective embedding is described and defined at the start of Section 4.

IOI The IOI task is the prediction that ‘ John’ completes the sentence ‘When John and Mary went to the store, Mary gave a bottle of milk to’ (Wang et al., 2023).

Logit Lens is defined in Section 3.

Name Mover Heads are heads that attend to IO rather than S in the IOI task.

Self-repair TODO.

The embedding matrix in transformer language models is a $\mathbb{R}^{d_{\text{vocab}} \times d_{\text{model}}}$ dimension matrices that maps tokens to embeddings in a transformer model’s residual stream.