Talking Heads: Understanding Inter-layer Communication in Transformer Language Models

Jack Merullo
Department of Computer Science
Brown University
jack_merullo@brown.edu

Carsten Eickhoff
School of Medicine
University of Tübingen
carsten.eickhoff@uni-tuebingen.de
email

Ellie Pavlick
Department of Computer Science
Brown University
ellie_pavlick@brown.edu

Abstract

Although it is known that transformer language models (LMs) pass features from early layers to later layers, it is not well understood how this information is represented and routed by the model. By analyzing particular mechanism LMs use to accomplish this, we find that it is also used to recall items from a list, and show that this mechanism can explain an otherwise arbitrary-seeming sensitivity of the model to the order of items in the prompt. Specifically, we find that models write into low-rank subspaces of the residual stream to represent features which are then read out by specific later layers, forming low-rank **communication channels** between layers. By decomposing attention head weight matrices with the Singular Value Decomposition (SVD), we find that previously described interactions between heads separated by one or more layers can be predicted via analysis of their weight matrices. We show that it is possible to manipulate the internal model representations as well as edit model weights based on the mechanism we discover in order to significantly improve performance on our synthetic Laundry List task, which requires recall from a list, often improving task accuracy by over 20%. Our analysis reveals a surprisingly intricate interpretable structure learned from language model pretraining, and helps us understand why sophisticated LMs sometimes fail in simple domains, facilitating future analysis of more complex behaviors.

1 Introduction

Despite the impressive capabilities of LMs, their practical use is often limited because of seemingly arbitrary sensitivities to prompts. These failure cases are particularly troubling because they are not systematic; it is very difficult to predict when, for example, the order of information seemingly randomly causes a model to fail [Pezeshkpour and Hruschka, 2023, Liu et al., 2024, Li and Gao, 2024, Zheng et al., 2024, Zhou et al., 2023], or the format of a prompt hurts performance [Liu et al., 2023, Sclar et al., 2023, Zhao et al., 2021, Lu et al., 2022, Webson and Pavlick, 2022]. As LLMs become increasingly ubiquitous, we will require more principled ways of anticipating and remedying unstable or unwanted behaviors [Yu et al., 2024, Yong et al., 2023]. Understanding the mechanisms in play within LLMs, and connecting those mechanisms to behavior, could enable such principled approaches.

Preprint. Under review.

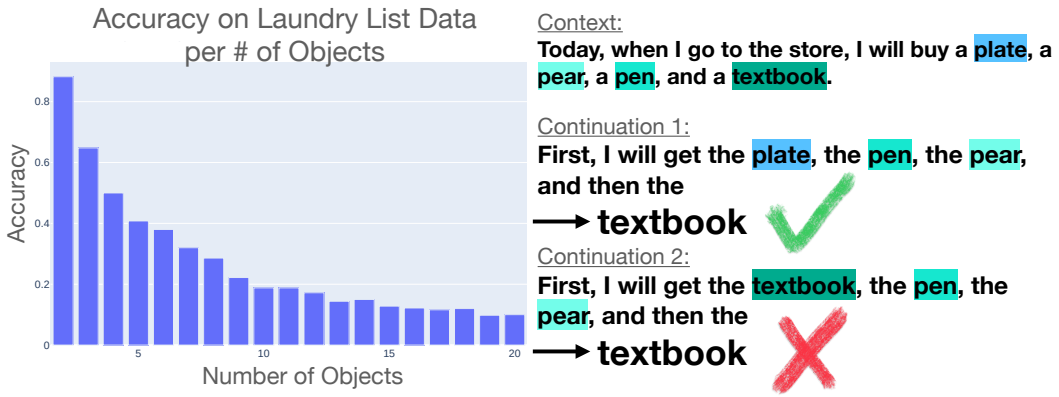

Figure 1: Language models are sensitive to arbitrary changes in a prompt, for example the order in which objects are listed (right). This problem is more pronounced as the number of objects increases (left) even though it is not obvious where the issue stems from in the model. We broadly explore how information is routed through a model and focus on a mechanism that is in part responsible for this (in)ability.

In this paper, we consider a simple *laundry list task* that exhibits one such undesirable instability. Specifically: Transformer language models (LMs) struggle to reliably recall items from a list as the length of the list increases, and performance can vary wildly depending on the position of the item in the list that is being recalled (Figure 1). This instability is not obvious from the model architecture itself–i.e., unlike their predecessors Elman [1991], Transformers [Vaswani et al., 2017] do not have built-in biases for some positions in a sequence over others. We thus use this task as a case study in order to connect the low-level *emergent mechanisms* which are encoded during LM pretraining to observable behavior, and illustrate as a proof of concept that a precise mechanistic understanding of LMs can be used to explicate and, perhaps, remedy model performance in practice.

Specifically, building on recent work in circuit analysis [Elhage et al., 2021, Wang et al., 2022, Goldowsky-Dill et al., 2023, Quirke and Barez, 2024, Merullo et al., 2024, Hanna et al., 2023], we demonstrate how a Transformer LM (GPT2 small) passes information from early laters to later ones using subspaces which we call *communication channels*. We introduce a method using the Singular Value Decomposition to help find these channels by reading off of the model weights directly. We focus on two examples of such channels (inhibition and duplicate detection), and find that they are are very low rank (1 or 2 dimensions), easily interpretable, and causally important for specific model behaviors. Specifically, our contributions are as follows:

- We explore a fundamental question in interpretability on how information passed from layer to layer in a transformer is represented internally. We find “communication channels" that connect attention heads separated by several layers.
- We introduce a method to decompose weight matrices which helps us find such communication channels. This method can be performed directly on the weights of models.
- We show that this mechanism plays a role in prompt sensitivity on an item recall task that otherwise seems idiosyncratic, and that intervention in the mechanism can be used to affect downstream behavior in a way that improves performance.

Our findings indicate more broadly that neural networks are capable of learning intricately structured representations from self-supervised pretraining without inductive biases, which might have important implications for the emergence of abstract and content-independent operations, and for developing methods for steering and understanding these models (see Discussion, §7).

2 Background

Throughout our work, we consider decoder-only transformer language models and primarily use GPT2-Small. The modern attention mechanism used by these models Vaswani et al. [2017] uses

multiheaded attention, where Query and Keys control *which tokens* from earlier in context are attended to and Value and Output matrices control *what information* is moved from these tokens. An important abstraction we use in our work relies on rewriting these matrices as the products QK (Query*Key) and OV (Output*Value)[1]. For an individual head, these matrices are themselves low-rank compared to the embedding dimension of the network. This is a useful property as we are motivated by looking for subspaces that are written into/read from by these matrices and this reduces the search space.

In order to ground our findings about inter-layer communication to real model behaviors, we focus on attention head interactions which we already understand and work backwards to determine how they communicate. Recent works in circuit analysis provide detailed explanations of how different model components interact on controlled datasets. In particular, we make use of the Indirect Object Identification (IOI) circuit discovered in Wang et al. [2022]. We use GPT2-Small, to study three specific types of interactions between heads: cases where heads write information that is used by the keys, queries, or values of later heads.

When we refer to an attention head as 3.0 or 7.9, this means layer 3, head 0 or layer 7 head 9.

Three Types of Composition In attention heads, there are three ways that earlier heads can contribute to the processing done downstream. In all cases, information is written into the residual stream by the OV matrix of an earlier head, and read back out by either the Query, Key, or Value matrices of a later head. These concepts are introduced in Elhage et al. [2021]. We also provide an example of each composition type that we examine further in Section 3. We look for communication channels in one of each type of composition. These are previous token to induction head composition (key) [Olsson et al., 2022, Singh et al., 2024, Reddy, 2023], duplicate token to inhibition head composition (value), and inhibition to mover head composition (query) [Wang et al., 2022]. The variation in the way these heads communicate only changes how we calculate the composition score [Elhage et al., 2021] and individual implementation of the communication, but we do not make claims about how these types of composition differ from each other in more meaningful ways.

The Inhibition-Mover Subcircuit We build on work from IOI [Wang et al., 2022] which documents a circuit that appears in multiple tasks [Merullo et al., 2024]. This circuit includes mover heads, which copy tokens from context to the output, and inhibition heads which (optionally) block the mover heads' attention to certain tokens and thus prevent certain tokens from being copied. Inhibition heads are known to receive signals from duplicate token head value vectors which help inform which tokens to inhibit. In GPT2-Small, the known inhibition heads are 7.3, 7.9, 8,6, and 8.10 and we consider their communication to the mover head 9.9[2]. This is the example we use for query composition experiments. In Section 5 we explore this circuit's role in problems of prompt sensitivity and a learned structure to control the indexing of tokens in the context window.

3 Identifying Communication Channels in Low-Rank Subspaces

In this section, we test the hypothesis that model components like attention heads communicate through signals in low-rank subspaces of the residual stream and that we can find these signals in the weights themselves. We investigate one case of each type of composition outlined in Section 2 and find positive evidence for this hypothesis with query and value composition in 1 and 2 dimensional subspaces, but not key composition (see Appendix A). Because we are able to localize the query and value signals to such small representation spaces, we find that we are able to interpret and control these features with intervention experiments in Section 4.

3.1 Composition Score

The Composition Score (CS) is a weight based metric of how much two weight matrices 'talk' to each other when they are separated by layers. That is, $\mathbf{W_1}$ might write information into a subspace in the residual stream that is read out by $\mathbf{W_2}$. In Query composition for example, $\mathbf{W_1}$ is the OV matrix

[1]Following convention, we refer to this matrix as OV, but the Transformer Lens library implements right-hand matrix multiplication so we actually use VO. This does not effect our results

[2]for simplicity we only consider this mover head, but we do not find the choice matters much.

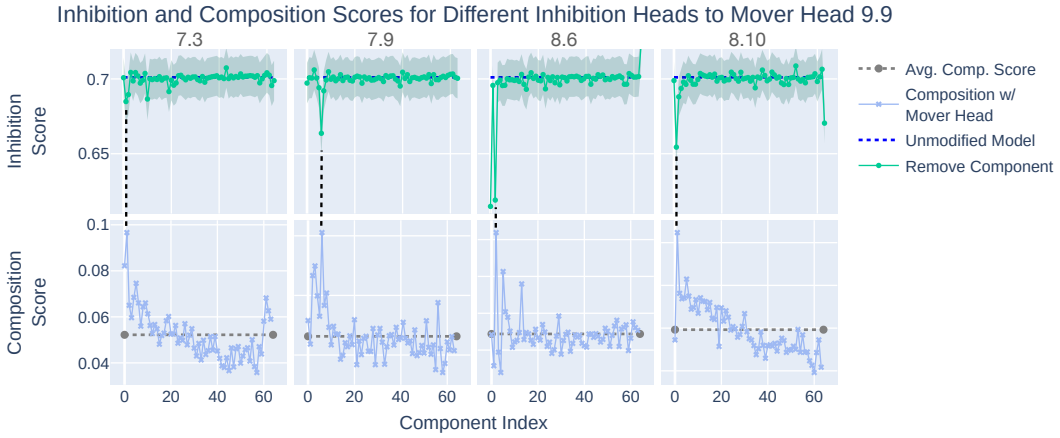

Figure 2: Showing the relationship between the composition score (weight-based, bottom) and inhibition score (data-based, top) between various inhibition head components and mover head 9.9 for the IOI task. The inhibition of each inhibition head is generally highly concentrated in one or two components of the matrix, removing it causes a large drop in the later mover head's ability to downweight one of the names. We therefore show that we can use the composition score when considering decomposed matrices.

of some head, and $\mathbf{W_2}$ is the QK matrix.

$$CS(\mathbf{W_1}, \mathbf{W_2}) = \frac{||\mathbf{W_1}\mathbf{W_2}||_F}{||\mathbf{W_1}||_F * ||\mathbf{W_2}||_F} \tag{1}$$

We take advantage of the fact that circuit analysis in works like Wang et al. [2022] tells us that, for example, head 3.0 (duplicate head) interacts with head 7.9 (inhibition head) through value composition and 7.9 with 9.9 (mover head) in query composition. We initially use the composition score in Equation 1 to try predict these interactions in the weights, but find these results are largely noisy and uninterpretable. This is briefly demonstrated in Appendix G. We find that despite empirically knowing interactions exist between heads, we do not find they reliably have higher composition scores than any random head. Although the composition score has been shown to be useful on small toy models [Elhage et al., 2021], previous work has also shown that on larger models, the signal the composition score conveys is extremely noisy [Singh et al., 2024].

3.2 Composition with Decomposed Component Matrices

The composition is not useful on its own because otherwise important signals are possibly washed out by the fact that every head reads from/writes into many subspaces to some extent. Therefore, we turn to the Singular Value Decomposition (SVD), defined as $\mathbf{W} = \mathbf{USV^T}$, on the QK and OV matrices to decompose the attention heads into orthogonal components which determine the input and output spaces of the matrix. This allows us to individually view subspaces read from/written into ordered by the amount of variance of the transformation of the matrix they account for. This helps us answer our original hypothesis that model components communicate across layers in low-rank subspaces of the residual stream.

If d is the dimension of the residual stream (768d in GPT2) and h is the dimension of an attention head (64d in GPT2), $\mathbf{OV},\ \mathbf{QK} \in \mathbb{R}^{dxd}$ and are both rank-h. This is because attention heads project down from the residual stream to h (e.g., the job of the $\mathbf{V}$ matrix) and then back up to d (e.g., the job of the $\mathbf{O}$ matrix). Therefore, there are only h non-zero singular values for each matrix.

Equation 2 shows a useful identity of the SVD: we can rewrite some weight matrix $\mathbf{W}$ as the sum of the outer products of the left and right singular vectors, scaled by the corresponding singular value.

$$\mathbf{W} = \sum_{i=0}^{h} s_i * \mathbf{U_i} \otimes \mathbf{V_i} \tag{2}$$

Rewriting the original matrix in this way is useful because we can now use the sum of any subset of component matrices in the composition score (Equation 1). Let the zero-th component of head 3.0 be written as 3.0.0. We can write the composition score between the 3.0.0 OV matrix and the 7.9 OV matrix as CS($\mathbf{OV}^{3.0.0}, \mathbf{OV}^{7.9}$). Since each component matrix is an outer product of two vectors, each matrix has rank-1. This gives us a way to disentangle the full signal of a head into the sum of its component rank-1 matrices, or the subspaces that the head is able to read from/write to.

We find that decomposing weight matrices this way is very effective at cleaning up the composition score signal. We find attention heads that have very high relative composition scores with one component matrix of another head. For example, the second component of head 8.6 (referred to as 8.6.2) composes far more with mover head 9.9 than any of the other 63 components in 8.6 (5 standard deviations higher than the average) or when considering the full matrix as in $CS(\mathbf{OV}^{8.6}, \mathbf{QK}^{9.9})$. The bottom graphs in Figure 2 show these results for the inhibition heads. All of the inhibition head exhibit a similar phenomenon of single component dominance. The duplicate token head 3.0 also value-composes with inhibition heads in a similar way, using two components (3.0.1 and 3.0.2) far more than any other. Results are shown in Appendix C.

We can also use this decomposition to find specific pairs of heads that talk more than others. With the knowledge that two heads talk through a specific component of one head, we can find the other heads that communicate through this pathway. Doing so lets us find the signal encoding almost the full IOI circuit in GPT2-Small directly from the weights, without running the model. We outline these results in Appendix G.

We interpret these as communication channels between heads, but we would still like to establish these channels as directly affecting downstream component's behavior. We verify this is the case through a weight editing in Section 3.3 and through activation interventions in Section 4.

3.3 Model Editing

In the previous section, we found that within a given head, individual component matrices encode a much stronger composition signal than that encoded by the global matrix. This makes the composition score a much more useful tool than when only considering full-rank matrices. In this section, we verify that these identified components are indeed communication channels that carry causally important signals for model behaviors. We first look at the inhibition head channels.

3.3.1 The IOI Dataset and Inhibition Score

Because the behavior of the inhibition heads was initially described on the Indirect Object Identification (IOI) dataset in Wang et al. [2022], we explore the inhibition communication channels on that domain first. An example of the dataset is as follows: *"Then, John[S1] and Mary[IO] went to the store. John[S2] gave a drink to"*. Here, the two name options are possible, but generating "John" does not make sense. The role of the inhibition heads are to tell the mover head (9.9) to attend less to the first John token (and as a result copy the remaining Mary token). We thus define the **Inhibition Score** as the degree to which the mover head prefers attending to the IO token (Mary) over the S1 token (the first John). This is simply the attention score to the IO minus the attention score to the S1. Intuitively, full attention to the IO token would give a score of 1.0, -1.0 would be full attention to S1 (inverse inhibition), and 0.0 would be equal attention to both (no inhibition).

3.3.2 Results

Our editing technique is simply to zero out one component at a time and across a dataset of IOI examples test how this affects. One way to think about this is zeroing out one singular value of e.g., the OV or matrix, or subtracting one of the component matrices from the sum in Equation 2. We must make the edit to the decomposition and then split the matrices back out so that we can run the model. Given $\mathbf{OV} = \mathbf{USV}^T$, after zeroing out some singular value in $\mathbf{S}$ we can set the Output and Value matrices to be $\mathbf{U}\sqrt{\mathbf{S}}$ and $\sqrt{\mathbf{S}}\mathbf{V}^T$, respectively. In the top graphs of Figure 2, we show that removing the speculated communication channel from the inhibition heads almost always results in a significant decrease in the model's ability to pass the inhibition signal, with the exception that changing 7.3 does not have a strong effect on its own. In general removing the single component with the highest composition score reduces the Inhibition score by 7-14%, and it is important to consider that this is only when changing a single component in one head at a time. We perform additional

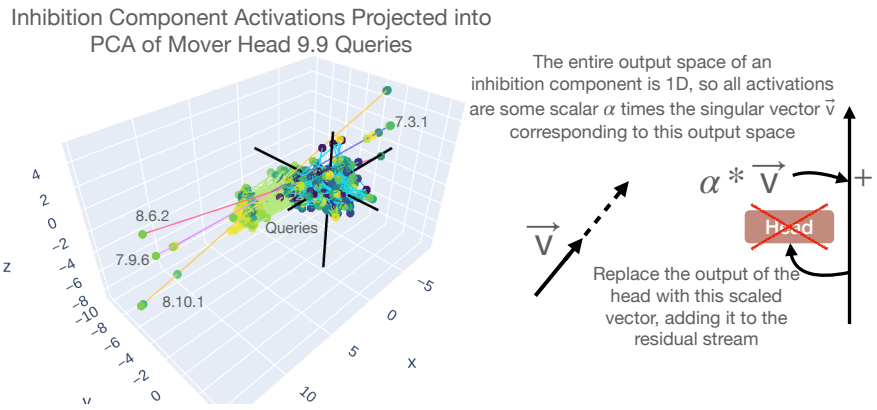

Figure 3: Because component matrices are rank-1, their output spaces are 1D and interpreting them becomes easier. On the left, inhibition component activations go to either side of the origin , and selectively inhiibt the name in either position one or position two in the IOI task. We can scale a vector lying on this line by some scalar alpha and observe how this changes behavior when we add it to the residual stream, or replace the output of an attention head with it (right), which we show in Figure 4.

experiments with removing/modifying multiple of these components in Section 4 and Appendix D. Thus, we have both behavioral and weight based evidence converging on the interpretation of these subspaces as meaningful communicatoin channels.

4 Communication Channels Carry Interpretable Content-Independent Signals

We present further evidence that the communication channels we identify in the model weights carry causally important signals for affecting model behaviors, but also that they carry content-independent signals which are easily controllable and interpretable.

Because the component matrices are rank-1 (Equation 2), their outputs lie entirely on 1D subspaces. This subspace (in our implementation) is the right singular vector corresponding to the index of the component matrix multiplied by some scalar. Since we have shown that these communication channels have a significant impact on downstream performance, a natural question is how information (such as inhibition) is represented along such a simple feature.

4.1 Interventions on Communication Channels

In order to better understand the representations passed through communication channels, we design interventions that add to or replace information from certain heads with vectors that lie on the output space of communication channels. Figure 3 provides an outline of the approach. Since a single component is rank-1, we can set the output of some head to be a point on the output space line and see how information changes along it.

Our dataset contains 200 examples from the IOI task. We have 100 examples where the IO token is the first name ("Mary and John...John gave a drink to") and 100 where the S1 token comes first ("John and Mary... John gave drink to").

On inhibition heads, we find that scaling a single component at a time is enough to switch the attention of the mover head to the other name. The inhibition score is highest when the S1 token is inhibited, but as Figure 4 shows, downscaling 8.10.1 only increases inhibition when S1 is first, and *decreases* it when IO is first. The opposite is true for upscaling the component. This tells us that the component is passing a positional signal: either inhibit name one or name two. This is consistent with what Wang et al. [2022] found, but our intervention shows that we can *completely* divorce the context from this ability. Since we are setting the head output to a scaled singular vector from the weights, we are bypassing the attention mechanism entirely, so non of the information on what to inhibit is coming

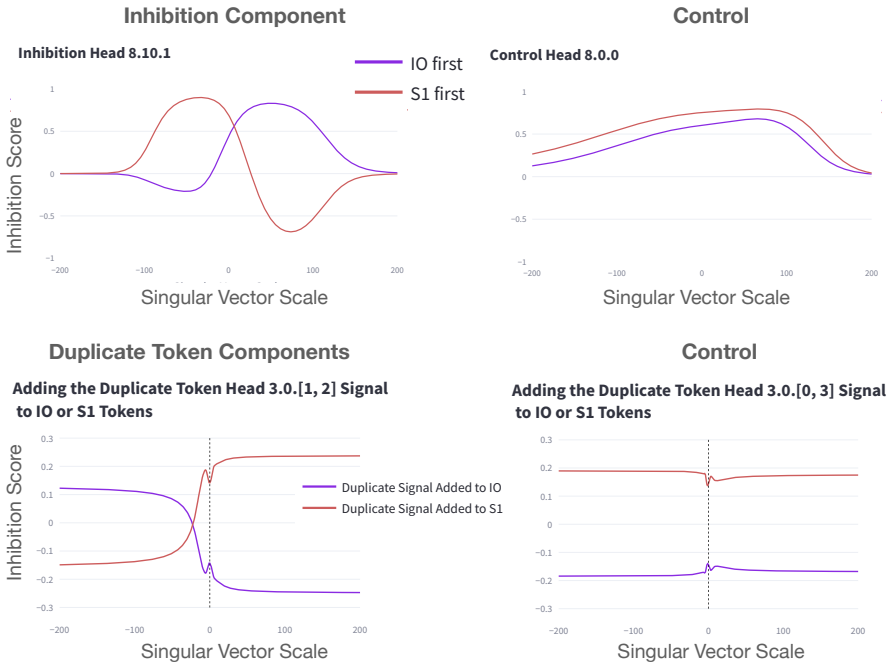

Figure 4: We find that the 1D inhibition components and 2D duplicate token components finely control which name is avoided by the mover head. On the top, we can selectively inhibit either the first or second name depending on how we scale a vector lying on the 8.10.1 output space. This is strictly controlling relative position. On the bottom, we find that adding or removing duplicate token information from the duplicate channel at the IO or S1 tokens also effectively modulates which name is inhibited. Neither random heads, nor non-communication channel components exhibit these same effects (right). See Appendix D for results on other heads.

from the value vectors of the IO or S tokens. This tells us that the model is capable of representing pointers, or storing bindings in the keys of earlier tokens that represent indexed lists, similar to the finding in Feng and Steinhardt [2023]. We explore this further in Section 5. The bottom of Figure 4 shows that scaling diagonally in a 2D subspace of the duplicate token head and adding the resulting vector to the residual stream of the IO or S1 token right before the inhibition heads also allows for modulating the selected name.

Although we get fine control over the attention of the mover head, we have not answered whether this has a real effect on the output behavior. Additionally, Makelov et al. [2024] argue that interventions on subspaces, such as the case here, are prone to a *subspace illusion* in which the effect is not what it seems. To address this, we measure the Fractional Logit Difference Decrease (FLDD) and Interchange Accuracy of patching the subspaces in the inhibition channel between minimally different IOI examples. GPT2-Small achieves an FLDD of 97.5% and and interchange accuracy of 35%. Such strong results from only patching three 1D subspaces supports the view that these subspaces are a primary mechanism controlling name selection. On OpenWebText [Gokaslan and Cohen, 2019], we find that the inhibition heads are primarily active in lists and settings where repetitions should be avoided, such as “serotonin, dopamine, and..." where the heads will attend from the commas to the previous chemicals. A natural followup given the IOI results and these observations is to explore their role as an indexing function, which we do in the Laundry List task.

5 The Inhibition Channel is Crucial in Token Indexing Tasks

Now that we have established the existence of communication channels and their causal role in model behavior, we can revisit our motivating example on prompt sensitivity in the Laundry List task. Figure 1 shows an example of the task and an example of an arbitrary seeming failure to minor variation in the order of presented items. In this section, we explore the hypothesis that the inhibition

communication channel presented so far plays a critical role in how the model selects the right context token to generate next, and reveals a capacity limit that causes the model to fail as the number of objects increases.

5.1 Laundry List Task

We propose a synthetic task that is designed to activate the inhibition heads, but allows us to test their effect on an arbitrary number of candidate tokens. An example is given in Figure 1 and more details on how we generated the data are in Appendix E. Each example first mentions N objects, then N-1, and the model must predict the missing item. We create a dataset of 250 prompts for each value of N from 3 to 20.

To complete this prompt, the model needs to recognize the item missing from the second list and retrieve it from the first list. By shuffling the second list and using a different sentence format, simple mechanisms (like pattern matching with induction heads) can not be relied on solely to solve the task. With this setup, we can very naturally increase the number of objects being considered. This lets us test not only *if* the inhibition heads are used for indexing candidates, but whether this mechanism's behavior changes as it is strained by the number of comparisons that need to be made. The model has a strong bias to predict the last item, regardless of whether it is correct or not, which causes performance to drop, so we are also curious if the mechanism connects to that as well.

We find that the inhibition heads are highly active on this task when predicting the last object, like the analysis in Section 4 would predict. The heads attend to and inhibit the occurrences of all of the repeated object tokens.

5.2 Intervention Experiments

We repeat the inhibition component intervention from Section 4 where we scale the components in the inhibition channel. In the multi-object (>2) case, we find that scaling only one component at a time does not give enough expressive power to change the mover head's attention to reach all of the objects (Figure 5, left). Instead it prefers to attend to either the first or last object, seeming to chunk the remaining objects together as a single point along the line. The bias to ignore information in the middle has also been observed by Liu et al. [2024].

We traverse the 3D space spanned by the top three inhibition head components (7.9.6, 8.6.2, and 8.6.10) and measure how this changes where the mover head attends, and what the model's final prediction is[3]. In Figure 5 (middle), we visualize this traversal with one dot representing a point in the space that we query. We run the entire dataset with the inhibition components set at this point, and the color represents the index of the object that the mover head attends most to. We find that structure emerges as we add items, where areas of this space represent the first and last object, and wedges fill in the space for each item that gets added. Eventually (around 9 or 10 items) these wedges get small and start to fracture (Figure 5, middle bottom). We connect these two phenomena to the performance of the model: the bias to predict later objects, and the inability to handle longer lists. We believe the model is not capable of traversing this space well enough on its own, even though it learned to represent it, and longer lists cause worse performance because the space gets fragmented into smaller and smaller areas that repersent each item.

We design an intervention where we set the model components in a certain area of the 3D space, depending on the index of the correct answer and test how much this improves performance. In Figure 5 (right), we show this causes a sharp increase in the accuracy of the model: 3 object accuracy goes up from 64% to 86%, and 8 objects goes up to about 51%, about the level of 4 objects in the unmodified case. Therefore the inhibition channel we identified seems to form part of a more generic, content-independent subcircuit for indexing items in the previous context.

6 Related Work

There has been significant focus on disentangling features from the representations of language models and vision models [Olah et al., 2020]. Features have been analyzed at the neuron level

[3]We test in increments of 10 from [-100, 100] along each axis, including every combination. it's possible this is not the optimal range

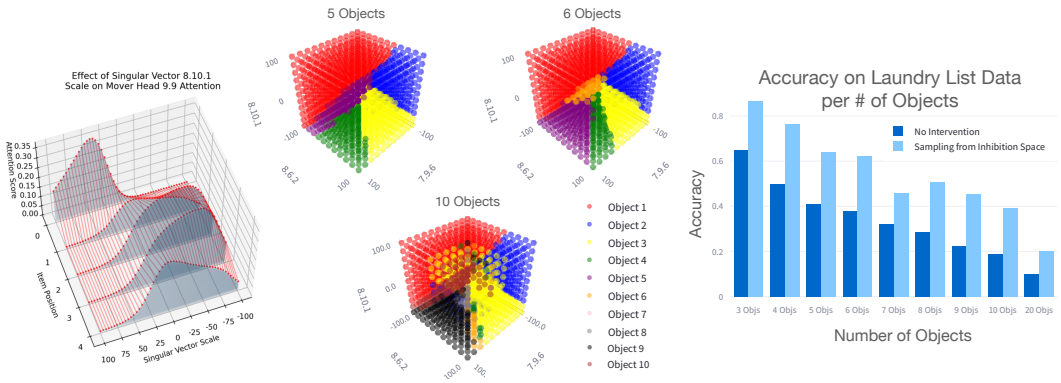

Figure 5: Scaling the inhibition component for a single head (here 8.10, left) is not expressive enough to get the mover head to index between the various objects. Scaling the top three inhibition components (middle) gives us enough expressive power to selectively attend to one of the objects. Here, one dot represents a run on the corresponding dataset and the color represents the index of the object the mover head pays the most attention to on average. A surprising structure emerges that partitions the space according to the index of the objects. However, the neat structure begins to break down as the number of objects grows around 10 or higher, and affects the mover head's ability to attend to the right object, which impacts accuracy. Right: Accuracy improvements as a result of sampling from inhibition space. The model becomes much more capable of handling a bigger number of objects in that the accuracy for N objects after the intervention is about as high as the unaltered model when it sees N/2 objects. However, the representational power of the inhibition channel reaches capacity as the number of objects increases, and performance can not improve as much.

[Gurnee et al., 2023, Mu and Andreas, 2020, Dai et al., 2022, Tang et al., 2024] Sparse Autoencoders and Dictionary Learning [Bricken et al., 2023, Mallat and Zhang, 1993, Cunningham et al., 2023] attempt to deconstruct activations into more primitive features [Rajamanoharan et al., 2024], which is similar in spirit to our decomposition. Park et al. [2023] propose a unification of several perspectives on the linearity of featuers. The Superposition Hypothesis Elhage et al. [2022] posits that linear features are encoded in interfering ways. Our method is similar in flavor of disentangling tangled features to make them easier to read off of the weights.

The SVD has been used for interpretability and weight based analysis in the past [Sule et al., 2023, Praggastis et al., 2022, Martin et al., 2021, beren and Black, 2022].

7 Discussion & Conclusion

Due to the recent and rapid success of language models, there is growing interest in understanding how they are able to use language so flexibly and solve difficult tasks. Our results contribute positive evidence that intricate content-independent structure emerges as a result of self-supervised pretraining. Although similar types of structure were previously thought to be impossible or unlikely to emerge in LMs [Lake et al., 2017], there is emerging evidence that LM pretraining encourages models to organize mechanisms into neat subprocesses [Feng and Steinhardt, 2023]. We use weight decomposition to uncover such structure and contribute to a fundamental understanding on how models route information between layers, a core part of understanding feature representation in models. We also show that low-level mechanisms such as those studied here can make real predictions about prompt sensitivity, a problem that has long plagued the robustness of LMs. The method we employ for weight analysis also holds promise for inference time steering, model editing, and automatic circuit discovery. We hope our work promotes future research on the interpretability of neural networks as well as their responsible deployment, and practical capabilities.

References

beren and Sid Black. The Singular Value Decompositions of Trans-

former Weight Matrices are Highly Interpretable. November 2022. URL https://www.lesswrong.com/posts/mkbGjzxD8d8XqKHzA/the-singular-value-decompositions-of-transformer-weight.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, 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. 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, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/toy_model/index.html.

Jeffrey L. Elman. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7(2):195–225, September 1991. ISSN 1573-0565. doi: 10.1007/BF00114844. URL https://doi.org/10.1007/BF00114844.

Jiahai Feng and Jacob Steinhardt. How do Language Models Bind Entities in Context?, October 2023. URL http://arxiv.org/abs/2310.17191.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*, 2023.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding Neurons in a Haystack: Case Studies with Sparse Probing, June 2023. URL http://arxiv.org/abs/2305.01610.

Michael Hanna, Ollie Liu, and Alexandre Variengien. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=p4PckNQR8k.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017. doi: 10.1017/S0140525X16001837.

Ruizhe Li and Yanjun Gao. Anchored Answers: Unravelling Positional Bias in GPT-2's Multiple-Choice Questions, May 2024. URL http://arxiv.org/abs/2405.03205.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, 2022.

Aleksandar Makelov, Georg Lange, Atticus Geiger, and Neel Nanda. Is this the subspace you are looking for? an interpretability illusion for subspace activation patching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Ebt7JgMHv1.

Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.

Charles H. Martin, Tongsu (Serena) Peng, and Michael W. Mahoney. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. *Nature Communications*, 12(1):4122, July 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-24025-8. URL https://www.nature.com/articles/s41467-021-24025-8.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Circuit component reuse across tasks in transformer language models. In *ICLR*, 2024.

Jesse Mu and Jacob Andreas. Compositional Explanations of Neurons, June 2020. URL https://arxiv.org/abs/2006.14032v2.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

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. *arXiv preprint arXiv:2209.11895*, 2022.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. *arXiv preprint arXiv:2308.11483*, 2023.

Brenda Praggastis, Davis Brown, Carlos Ortiz Marrero, Emilie Purvine, Madelyn Shapiro, and Bei Wang. The SVD of Convolutional Weights: A CNN Interpretability Framework, August 2022. URL http://arxiv.org/abs/2208.06894.

Philip Quirke and Fazl Barez. Understanding addition in transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rIx1YXVWZb.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders, 2024.

Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *The Twelfth International Conference on Learning Representations*, 2023.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*, 2023.

Aaditya K Singh, Ted Moskovitz, Felix Hill, Stephanie CY Chan, and Andrew M Saxe. What needs to go right for an induction head? a mechanistic study of in-context learning circuits and their formation. *arXiv preprint arXiv:2404.07129*, 2024.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

Shashank Sule, Richard G. Spencer, and Wojciech Czaja. Emergence of the SVD as an interpretable factorization in deep learning for inverse problems, August 2023. URL `http://arxiv.org/abs/2301.07820`.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. Language-specific neurons: The key to multilingual capabilities in large language models. *arXiv preprint arXiv:2402.16438*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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*, 2022.

Albert Webson and Ellie Pavlick. Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344, 2022.

Zheng Xin Yong, Cristina Menghini, and Stephen Bach. Low-resource languages jailbreak gpt-4. In *Socially Responsible Language Modelling Research*, 2023.

Zhiyuan Yu, Xiaogeng Liu, Shunning Liang, Zach Cameron, Chaowei Xiao, and Ning Zhang. Don't listen to me: Understanding and exploring jailbreak prompts of large language models. *arXiv preprint arXiv:2403.17336*, 2024.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR, 2021.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large Language Models Are Not Robust Multiple Choice Selectors, February 2024. URL `http://arxiv.org/abs/2309.03882`.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. In *The 3rd Workshop on Mathematical Reasoning and AI at NeurIPS'23*, 2023.

A The Composition Score with and without Weight Decomposition

We include some examples showing outlier components in value and query composition but not with induction head key composition in Figure 6.

B Limitations

A limitation of our approach is that we are relying heavily on previous knowledge of the language model that we are using (GPT2-Small), which has been extensively studied. However, the insights that we are able to glean by building on this foundation of knowledge we view as more reason to approach interpretability work as building directly on model-specific knowledge. Additionally, our

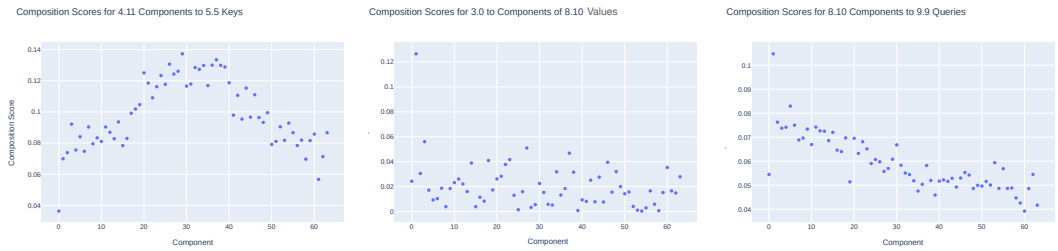

Figure 6: Examples of composition scores of individual components with other heads. 4.11 is a previous token head, 5.5 is an induction head, 3.0 is a duplicate token head, 8.10 is an inhibition head, and 9.9 is a mover head. We find that there are large outlier components in value and query composition, but not in the induction head, thus motivating our focus on those heads in the main text.

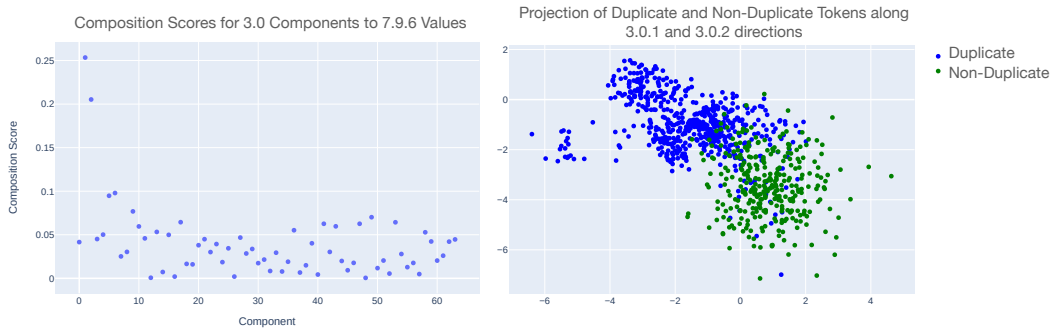

Figure 7: Left: Composition scores between each component of duplicate token head 3.0 and inhibition component 7.9.6. Components 1 and 2 are clearly outliers. Right: on long contexts of random tokens with inserted duplicates, we find that these directions separate duplicates from non-duplicates quite well. This leads us to believe that these two components form a duplicate communication channel. Our results in Section 4 support this interpretation.

findings may be able to feed back into automating interpretability of new models. Another limitation of our approach is the inability to calculate query and key composition scores with models that implement relative positional embeddings like RoPE [Su et al., 2023] because of the non-linearities between the Query and Key products preventing QK to be calculated cleanly. It may be possible to simply take the composition between the Q and K matrices individually, but we do not experiment with that extension here.

C Duplicate Token Heads

We focus on the inhibition communication channel in the main paper and do not show the where the duplicate token channel comes from. In Figure 7, we show that two component matrices in duplicate token head 3.0 (3.0.1 and 3.0.2) compose strongly with inhibition heads (7.9.6 shown here). On long sequences of random tokens, we show that these direction encode whether or not a token has been duplicated.

D More IOI Interventions

Additional interventions testing the efficacy of inhibition heads to change behavior of a downstream mover head are provided in Figures 8 and 9. We include the remaining inhibition heads and additional control heads for the component scaling experiment from Section 4 in Figure 10

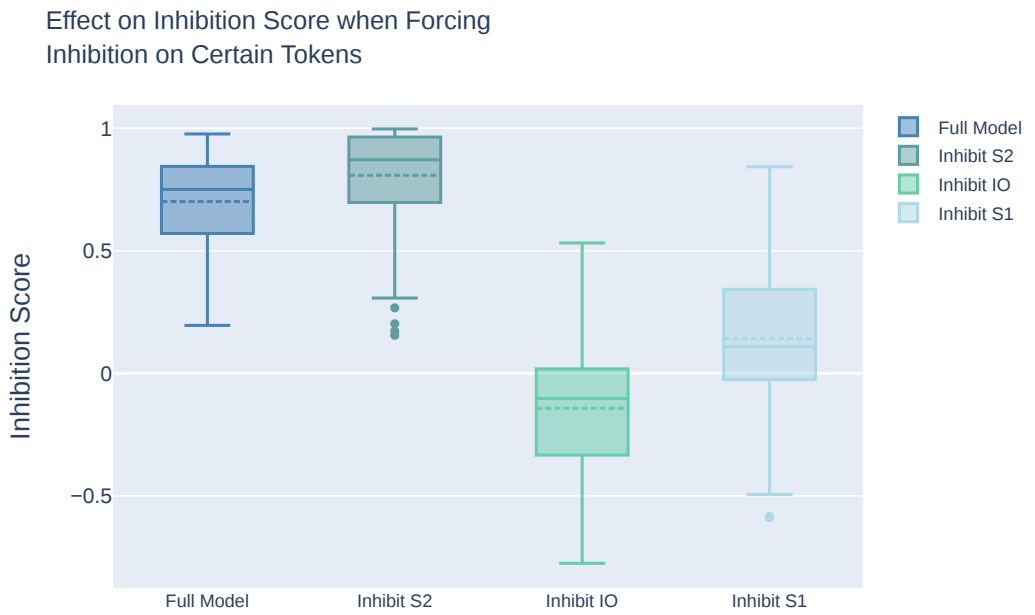

Figure 8: Effect of applying the top component interventions at the same time to some token. We can control the inhibition by selecting which token these components attend to. Higher score means more inhibition on S2, and lower score means more inhibition on IO.

E Laundry List Data Generation

The Laundry List task is a leave-one-out task where the model must identify the object that was not mentioned. Each input is two sentences (see Figure 1 for an example). The first sentence lists objects that need to be purchased and the second describes the order that they are to be bought in, with the next token prediction being the item from the first list that is to be bought last. This setup allows us to freely shuffle the order of the information provided to the model as well as vary the number of objects presented in each example. There are 22 objects that can be sampled, given below: ‘‘pencil", ‘‘notebook", ‘‘pen", ‘‘cup", ‘‘plate", ‘‘jug", ‘‘mug", ‘‘puzzle", ‘‘textbook", ‘‘leash", ‘‘necklace", ‘‘bracelet", ‘‘bottle", ‘‘ball", ‘‘envelope", ‘‘lighter", ‘‘bowl", ‘‘apple", ‘‘pear", ‘‘banana", ‘‘orange", ‘‘steak" .

The first sentence can start a few ways, chosen randomly: ‘ Today,’, ‘ Tonight,’, ‘ Tomorrow,’, ‘’. And the second sentence start can be chosen randomly: ‘ First,’, ‘’, ‘ When I go,’, ‘ I think’

F Inhibition Mechanism in Pythia and Training Progression of Inhibition

We verify both that communication channels appear in other models, and that inhibition is a more general mechanism that that just appearing in GPT2. To show this we analyze Pythia-160m [Biderman et al., 2023]. Because Pythia provides training checkpoints, we are also able to analyze the formation of the inhibition component we find to some extent.

F.1 Path Patching on IOI

We perform path patching [Wang et al., 2022, Goldowsky-Dill et al., 2023] on Pythia-160m on the IOI task to see if the model also implements mover and inhibition heads. We find evidence for one

Figure 9: Effect on the inhibition score when removing components from the inhibition heads. If we only take the top-1 composing heads that affect the inhibition score (circled in Figure 2) we retain close to have of the average inhibition score (0.7 to 0.3). If we only use the component matrices that correspond to the 0th singular value of the inhibition heads, which represents the subspace most strongly written to by the head, the average inhibition score is only 0.04. Recall that a negative inhibition score means placing more attention on the subject rather than IO token.

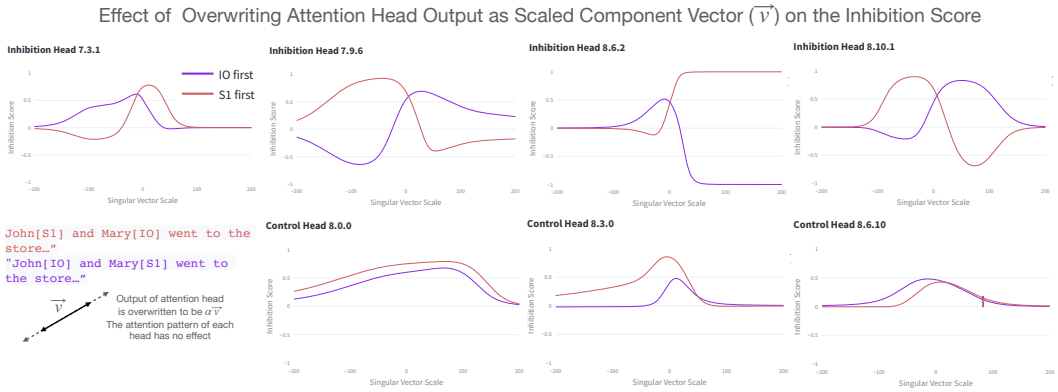

Figure 10: We intervene on the forward pass of the model by replacing the output of some attention head as the vector obtained by scaling a component vector by some scalar α. By doing so, the actual attention head pattern has no effect on the downstream performance. We show the inhibition component vectors have the unique effect of controlling the position of the name being attended to by the downstream mover head (9.9). Random control head components do not have this effect.

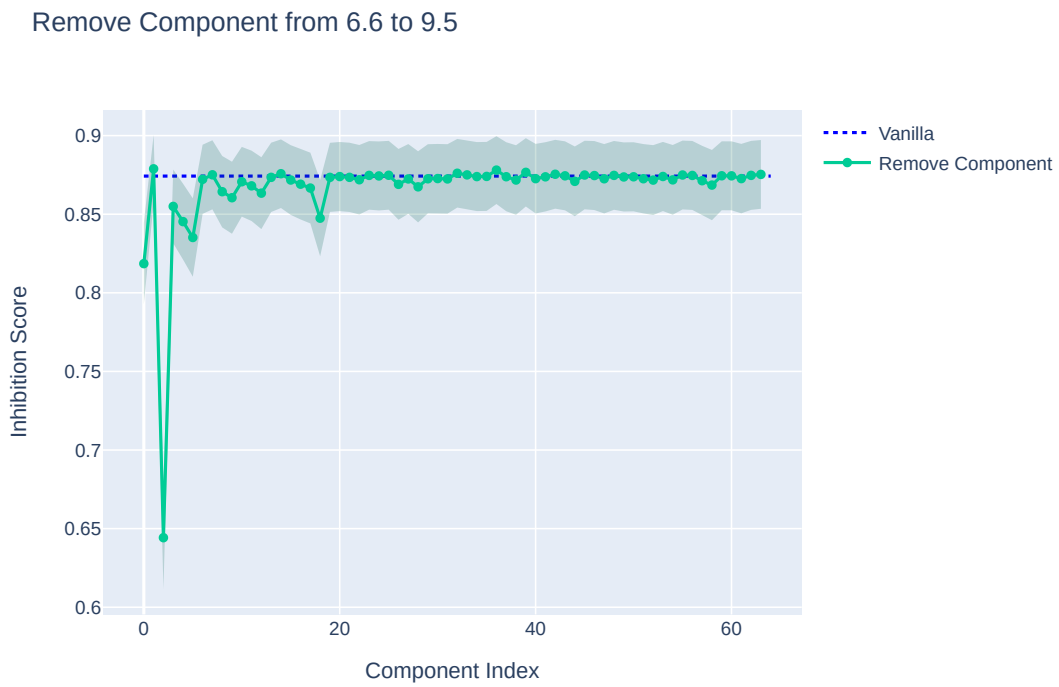

Figure 11: Pythia-160m also has a single component (6.6.2) in an inhibition head that dominates the inhibition signal.

inhibition head (6.6) in the model talking to a mover head (9.5). We find that like GPT2, the inhibition head communicates primarily through a single component, as is shown in Figure 11.

Additionally, an induction head (4.11) strongly value composes with the inhibition component 6.6.2 as shown in Figure 12.

F.2 Training Progression of Inhibition Components

Because Pythia releases 144 intermediate checkpoints (per 1000 steps), we can track the emergence of the inhibition head during training. We saw that the inhibition component vector clusters Name1 on one side and Name2 on the other side, representing which name is being inhibited. Everything else ends up around the origin. Since we have minimal pairs of examples that differ only in the position of the name that should be inhibited, we can measure the Separability of the inhibition component vector by making sure that if one name is in one cluster, the minimal pair example is in the other cluster. This is a measure of how well the inhibition head is structuring according to this idealized separation of the two names.

The component that we use is the 6.6.2 matrix from the fully trained inhibition head. We test parity by projecting the model's activations onto this matrix.

In addition we can measure how well the fully trained component matrix activates (or removes) the inhibition signal by adding or subtracting it from earlier checkpoints. These results are in

G Static Weight Analysis for Circuit Discovery

To show the effectiveness of the decomposition at finding heads that communicate to a significant degree, we use the composition score only to find a large chunk of the IOI circuit from Wang et al. [2022] encoded directly in the weights. Figure 14 shows these results. The composition score after decomposing the inhibition head 7.9 more clearly reveals the communication between the in-circuit heads (circled in red) than if the composition score is used without decomposition. It is possible this approach can be built upon to find circuits in models without requiring the model to be run. We leave this for future work.

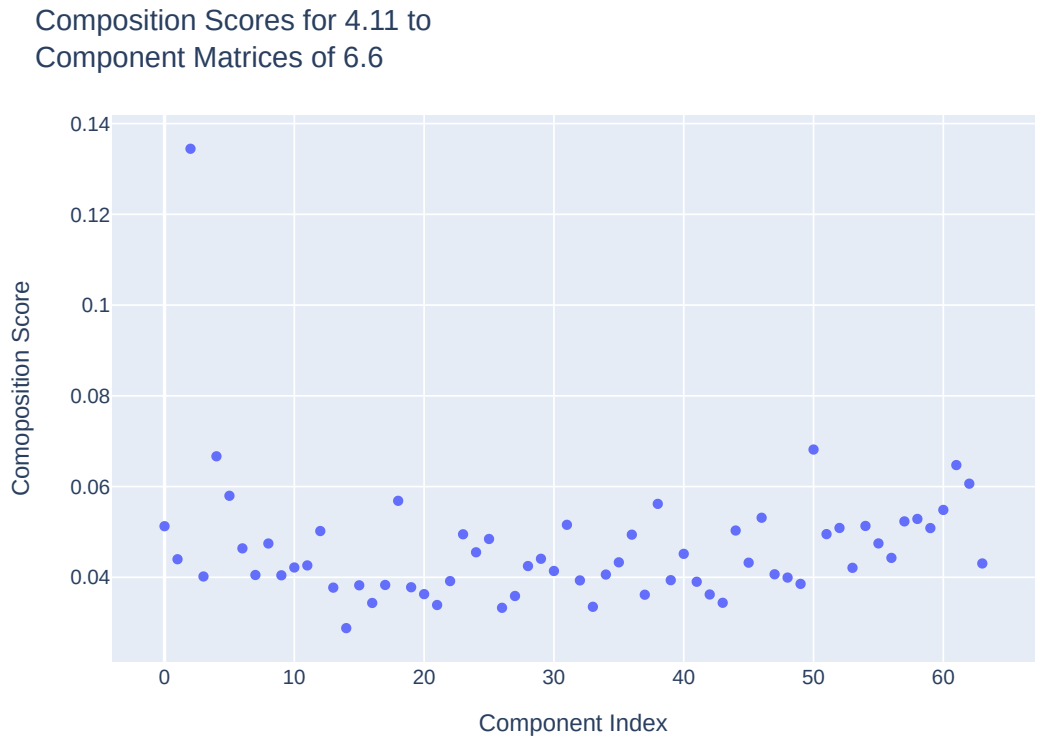

Figure 12: An example of an induction head (4.11) value composing strongly with the single inhibition component 6.6.2 in Pythia-160m, suggesting a circuit for controlling attention through mover head 9.5. We leave analysis of this for future work.

H More Information on Composition

H.1 Value Composition: Duplicate Token Heads

Value composition dictates that the value vectors of an earlier head write information that affect the values of later heads. Duplicate token heads are a well established type of attention head that specialize in attending to duplicates in the previously seen context. That is, given the text “A B A", the duplicate token head will attend from the second “A" token to the first. The IOI circuit finds value composition between heads 3.0, a duplicate token head, and 7.9, an inhibition head.

H.2 Query Composition: Inhibition Heads

Value vectors of earlier heads affect the query vectors of later heads, thus changing what they attend to. A canonical example originating in the IOI paper is with *inhibition heads*. These are a key part in a token copying circuit in which the value vectors of such heads prevent later query vectors from attending to the duplicated name in the IOI task. Mover heads (such as 9.9) We study these heads in greater detail in Section 5. Whatever a mover head attends to will be promoted as the next token prediction. An inhibition head tells a mover head to avoid attending to certain tokens, which is helpful when there are multiple options to generate.

H.3 Key Composition: Induction Heads

An induction head is a pattern completing attention head. For example, seeing the pattern “A B A B A" will cause the model to attend from the last A to the last B, since a pattern is present where B must follow A. The mechanism that typically implements induction heads requires a previous token head (which will always attend from the current token to the one right before it) to affect the key of a later induction head. At a later timestep the query of the induction head will notice the signal left in

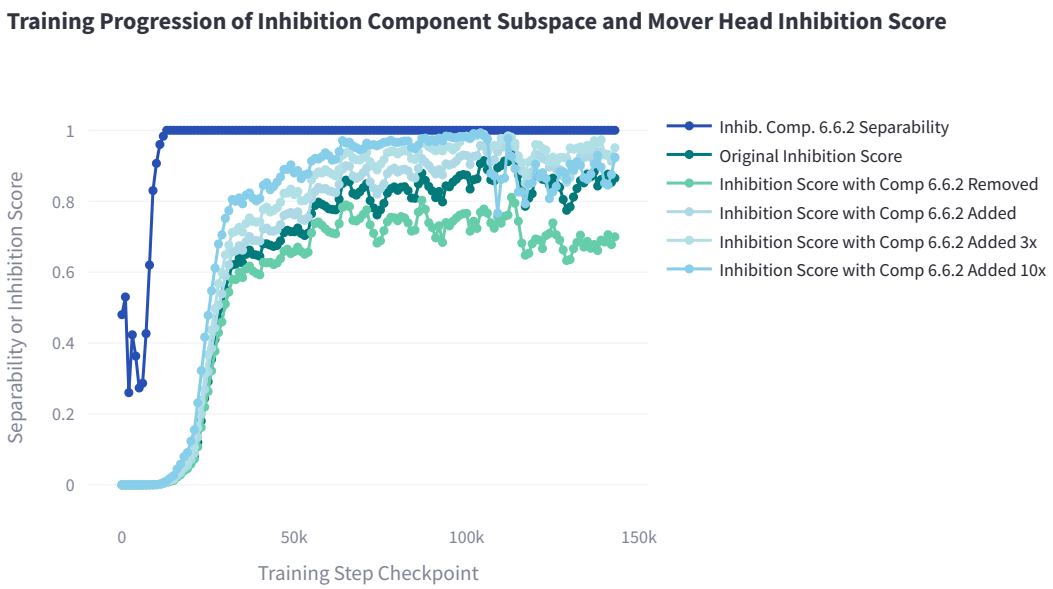

Figure 13: Pythia training progression of inhibition component (6.6.2) and effect of model editing. Adding the component matrix to the inhbition head strengthens the inhibition channel and improves the ability to use inhibition in earlier checkpoints, subtracting it makes inhibition weaker. Separability is simply the extent to which activations for IOI minimal pairs are split into clusters based on the order of names (IO, S1 or S1, IO).

the earlier key, and choose to attend to it. We consider the key composition between the previous token head 4.11 to induction head 5.5 from the IOI circuit.

I Extra Laundry List Interventions

The results for scaling individual components across Laundry List datasets (varying number of objects) are in Figures 16, 17, 18, 19, 20, 21, 22, 23, and 24

We also include results from traversing the 3D inhibition subspace used in Figure 5 for a greater range of settings for the number of objects in Laundry List dataset examples. We test datasets set to have 3-10 objects and one dataset set to have 20 objects. The results are shown in Figure 15.

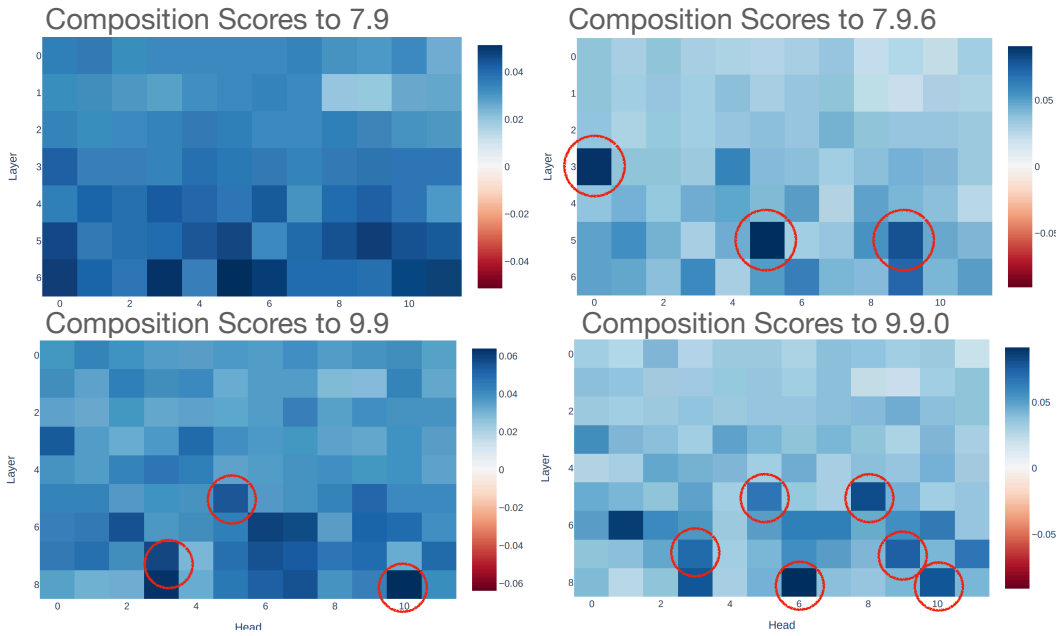

Figure 14: Decomposing weight matrices cleans up the composition score enough that we can start to read off components that belong to the IOI circuit without running the model. By starting with a known inhibition head component (7.9.6) we can find the heads that compose into that component and the heads for which the inhibition component composes into that belong to the IOI circuit from Wang et al. [2022]. Left graphs show the composition score without any decomposition, which is noisy. On the right, we find in-circuit heads (circled) qualitatively to stand out more. See Wang et al. [2022] for more details.

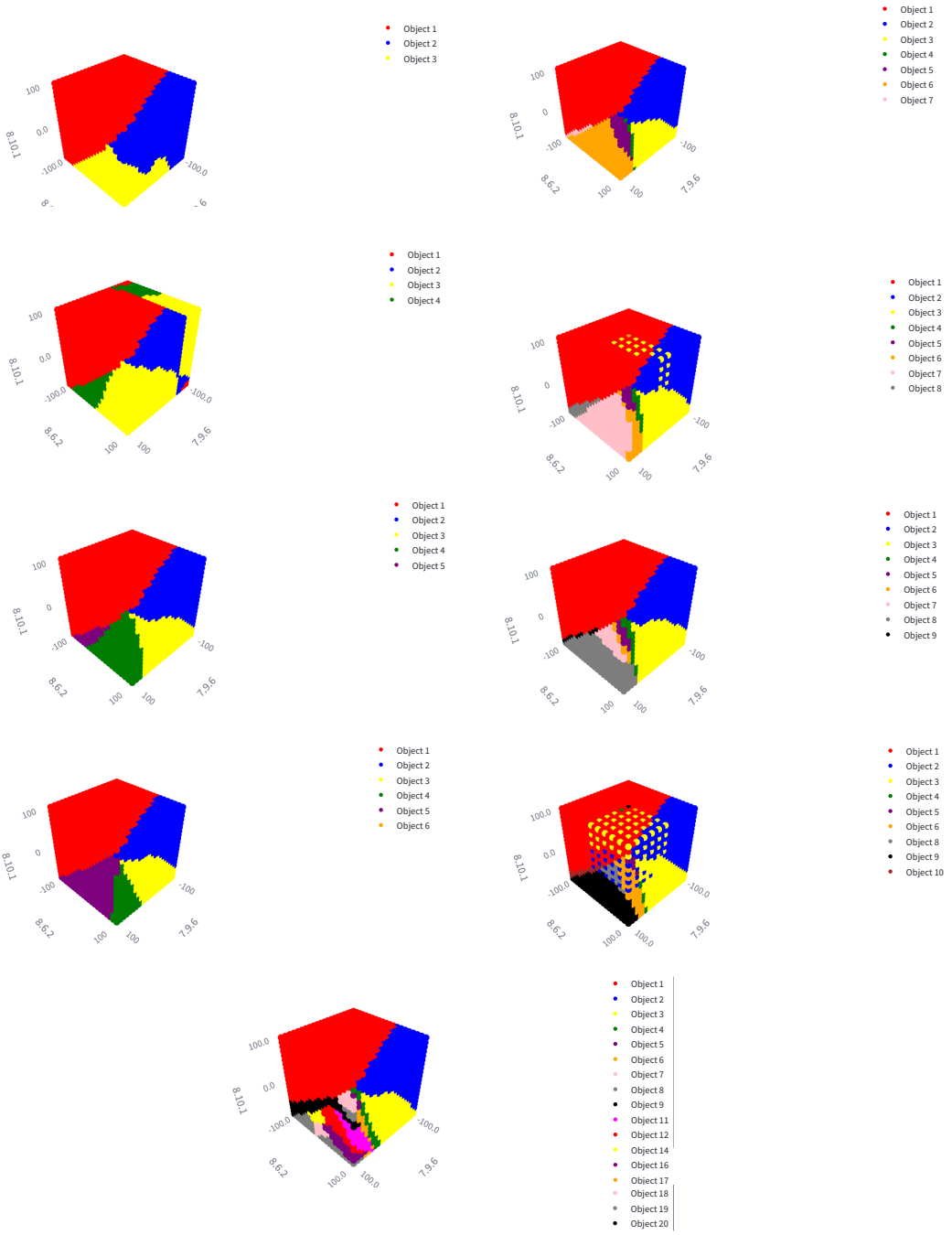

Figure 15: How the 3D inhibition subspace responds to a different number of objects in laundry list prompts. As we add objects, a new 'slice' of the space is allocated (not always visible) for attention to that object until the middle set of objects is squeezed into a small neighborhood of the space. The space is very well structured, except for two cases where artifacts form in the 8 and 10 object settings.

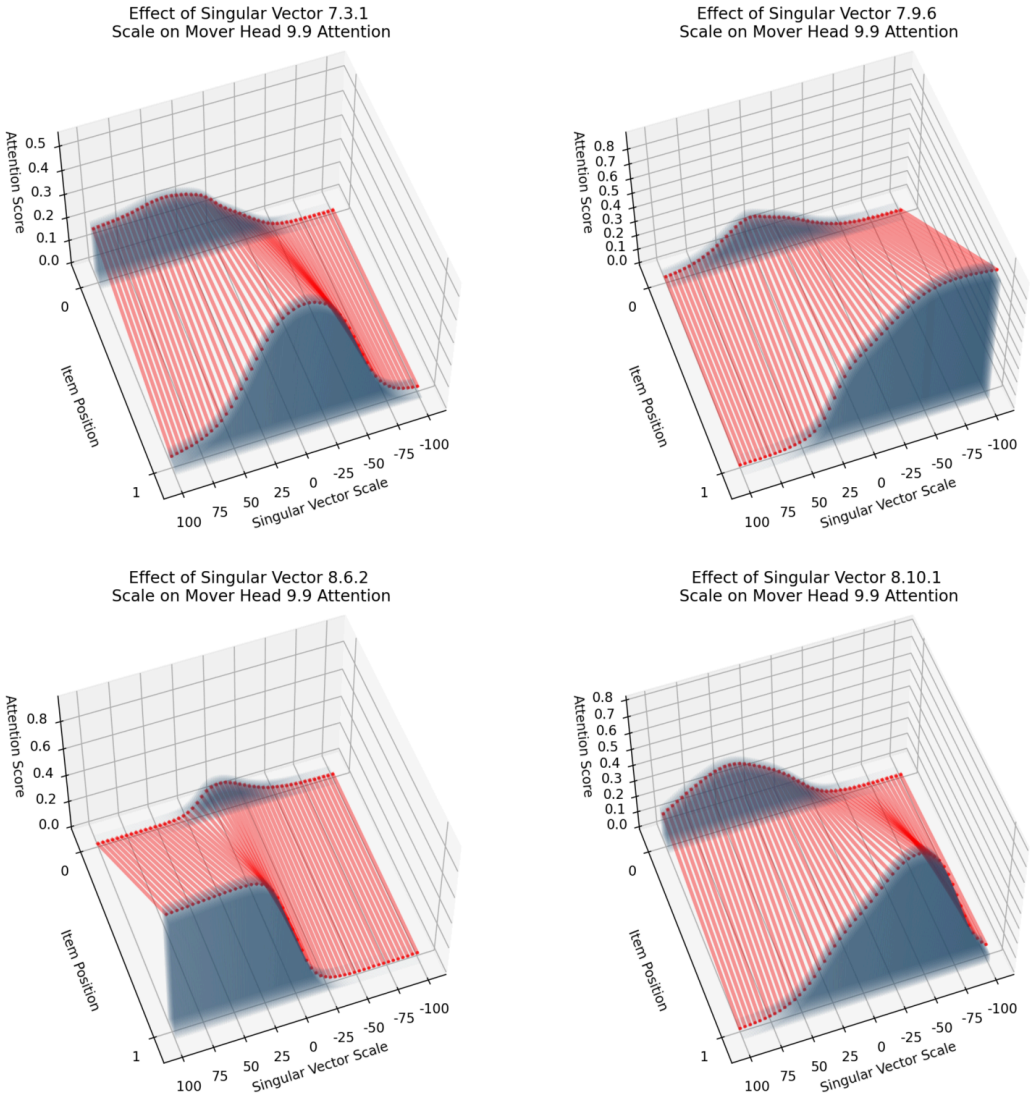

Figure 16: 2 Objects

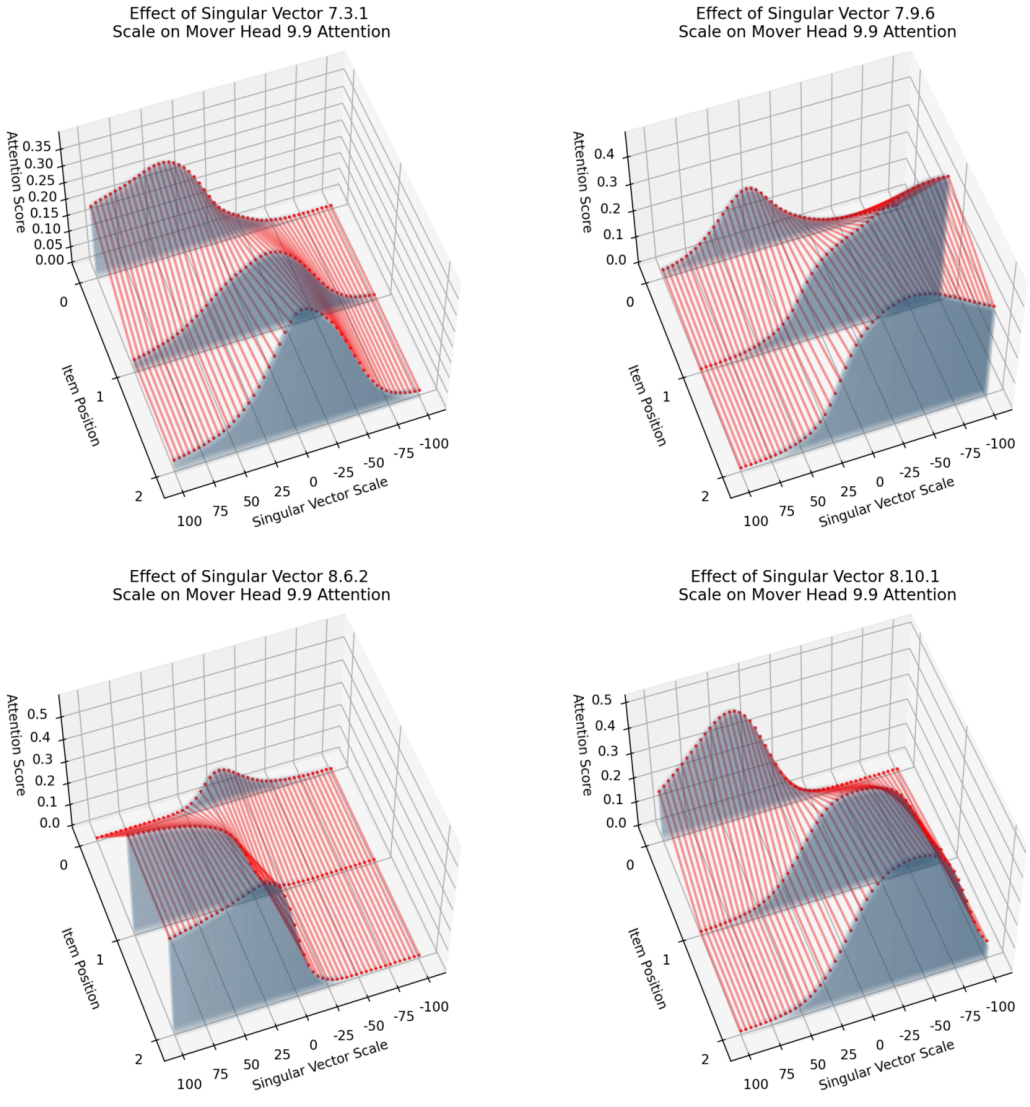

Figure 17: 3 Objects

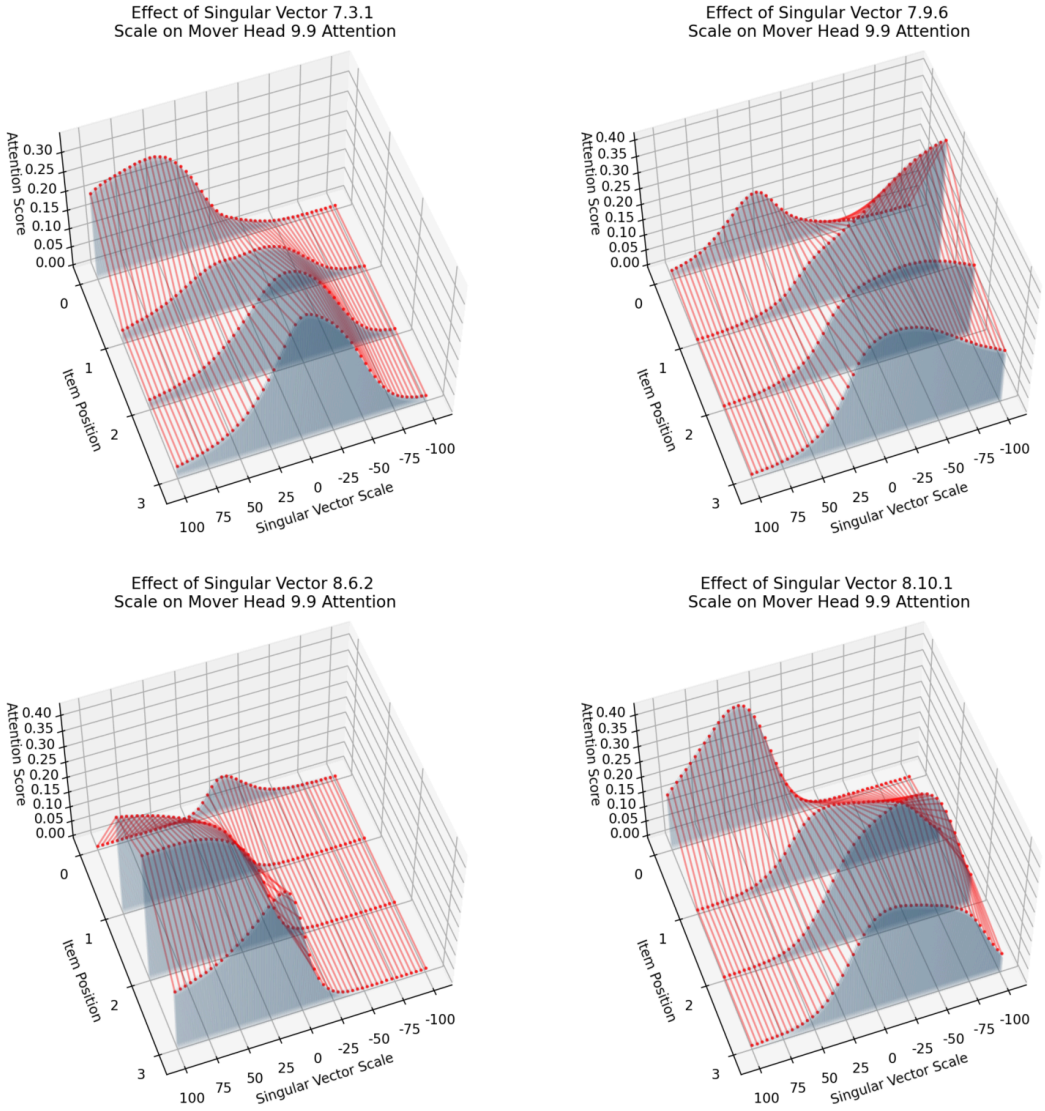

Figure 18: 4 Objects

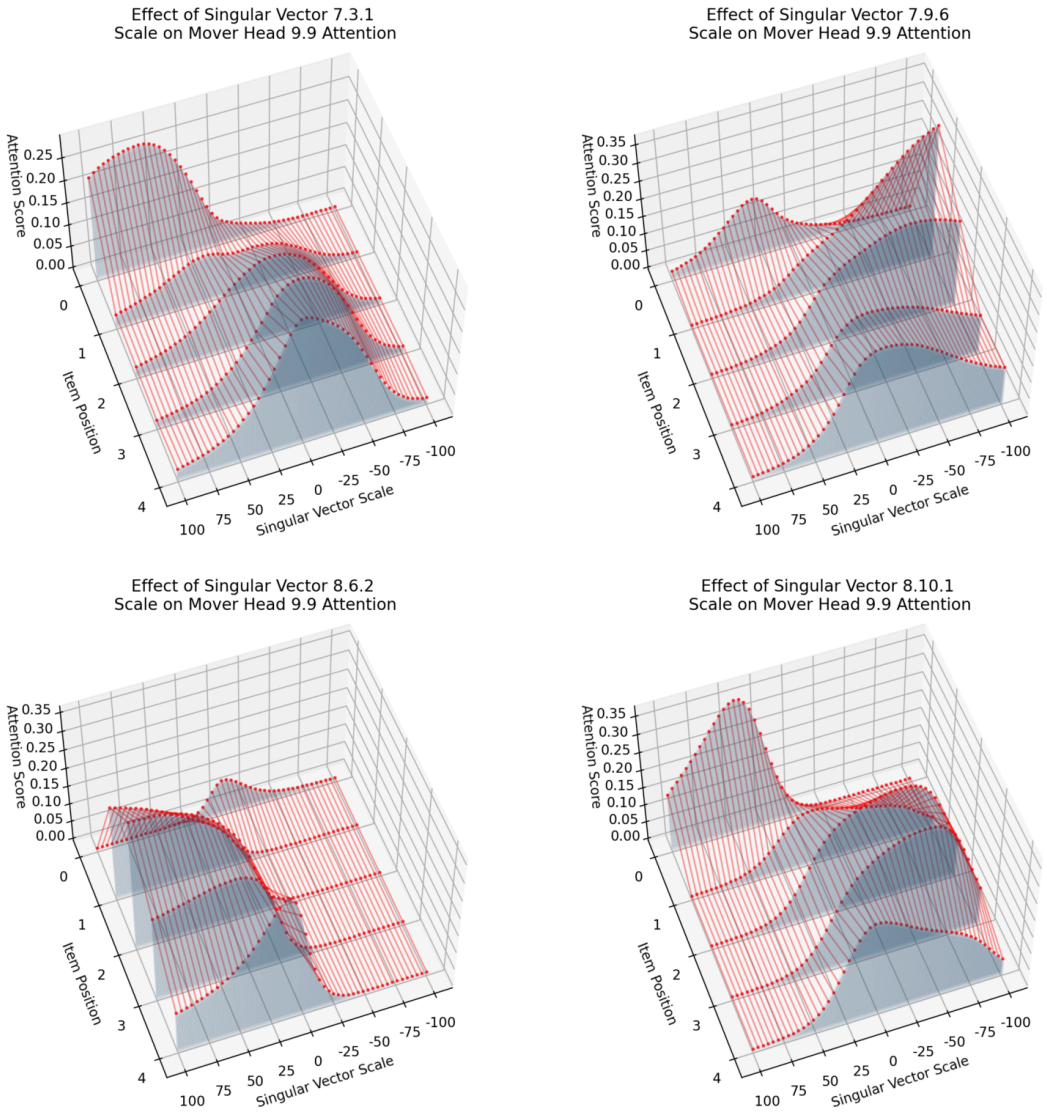

Figure 19: 5 Objects

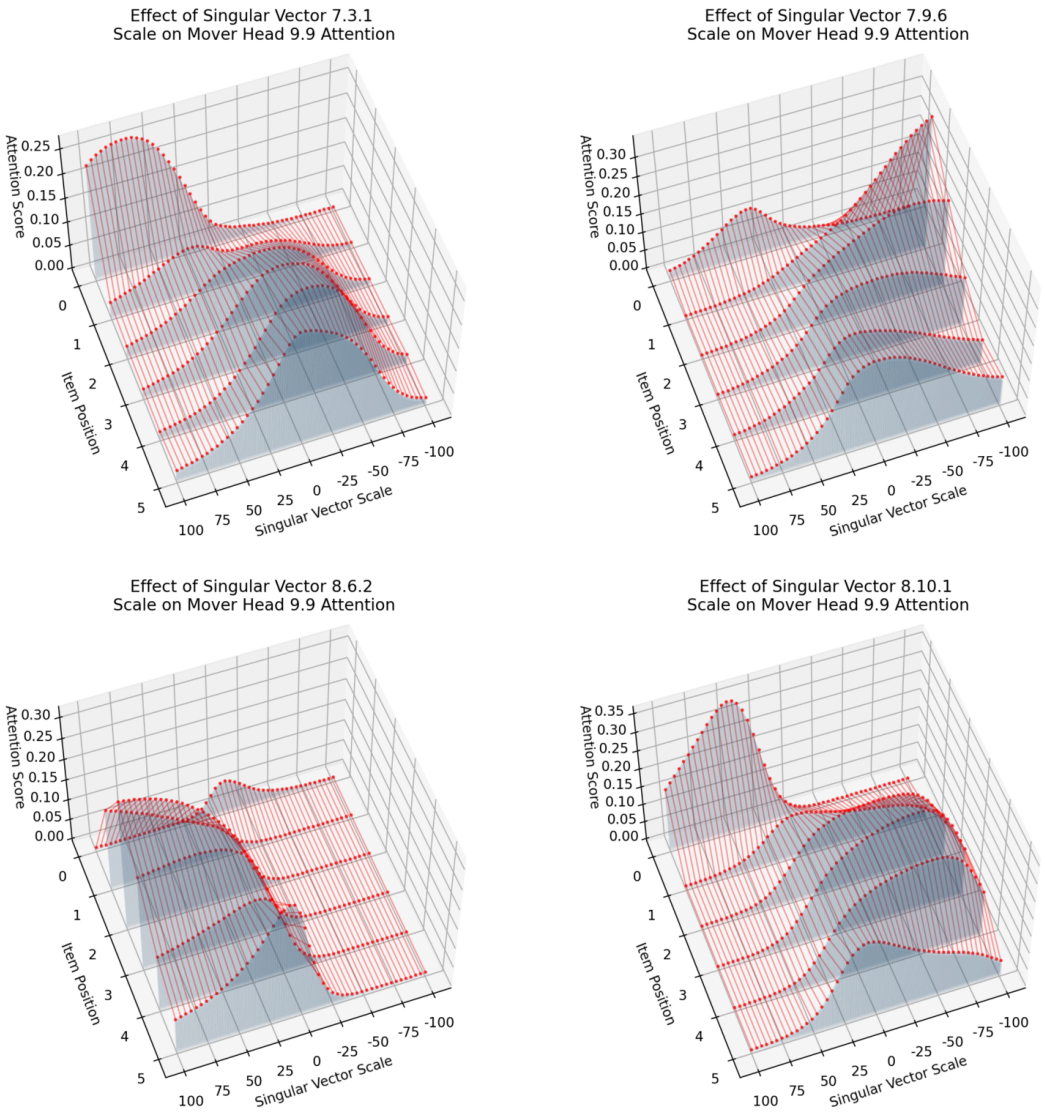

Figure 20: 6 Objects

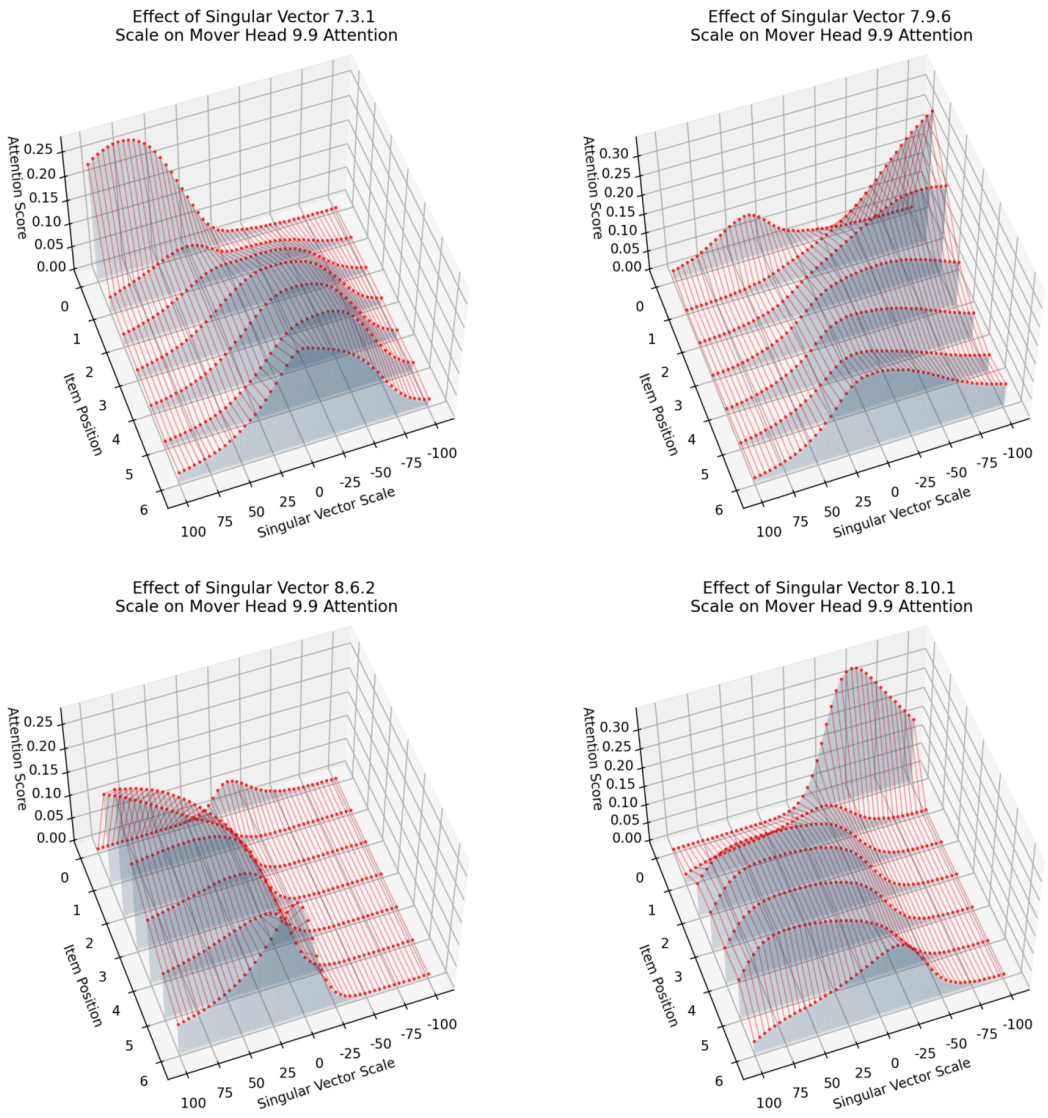

Figure 21: 7 Objects

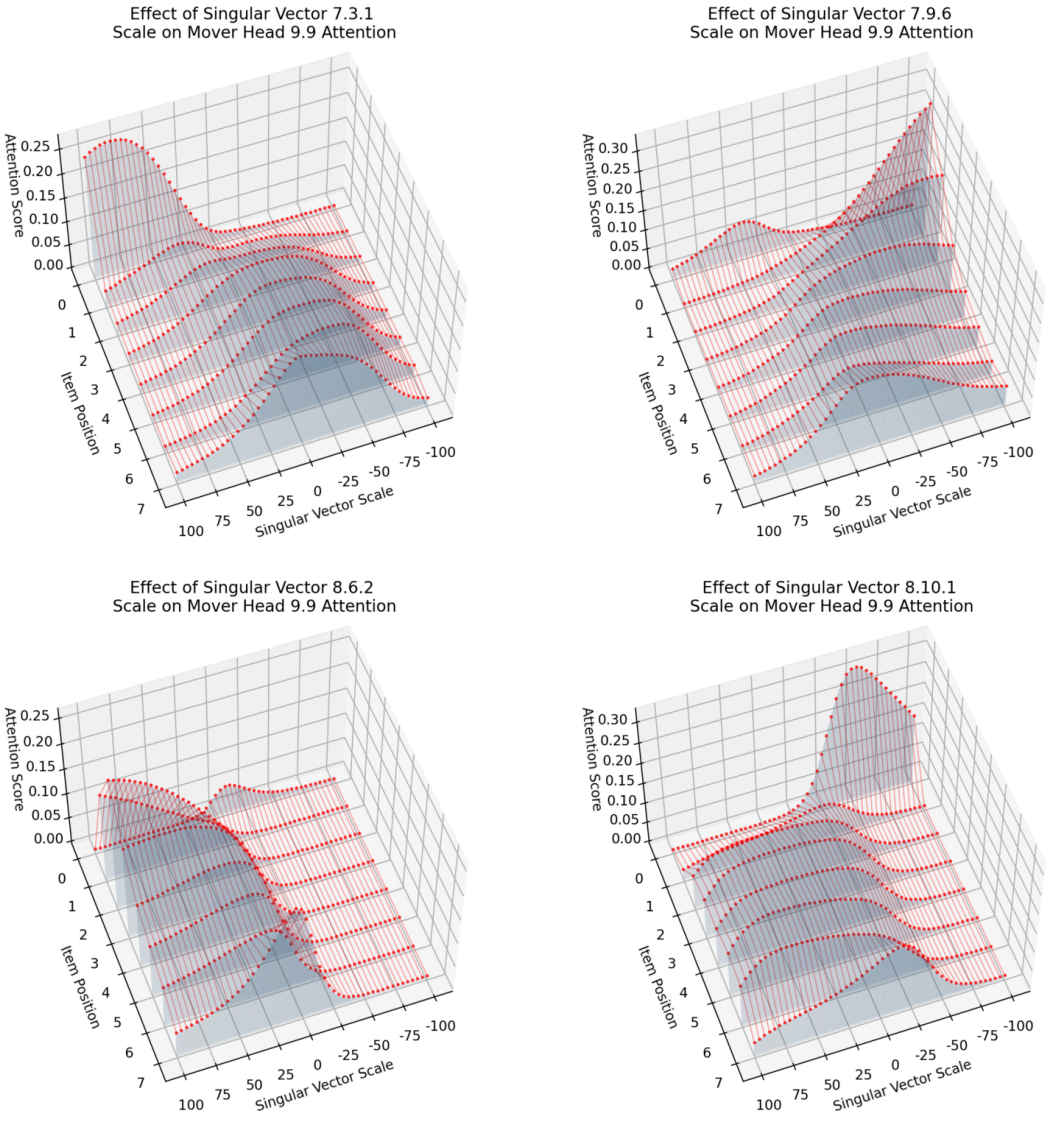

Figure 22: 8 Objects

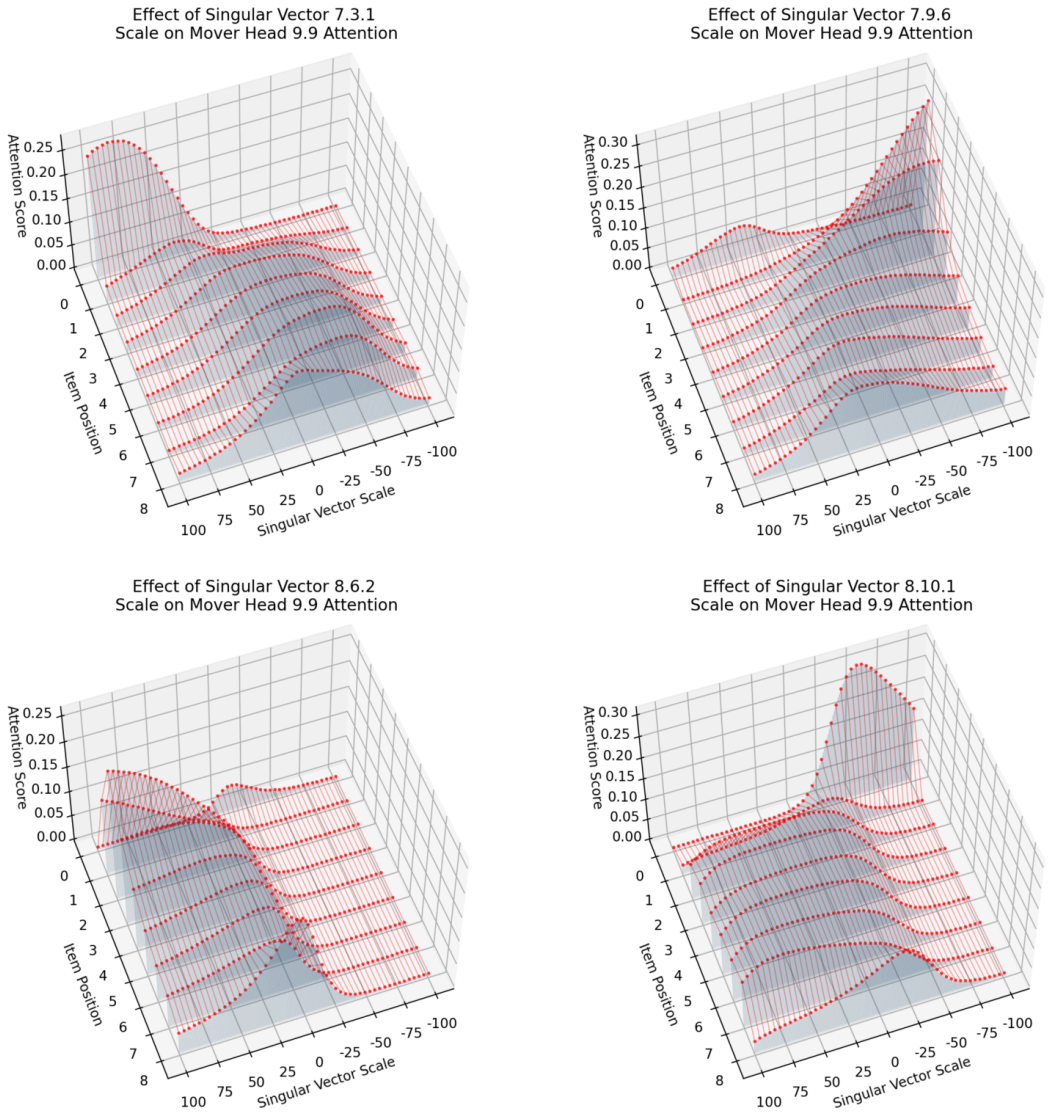

Figure 23: 9 Objects

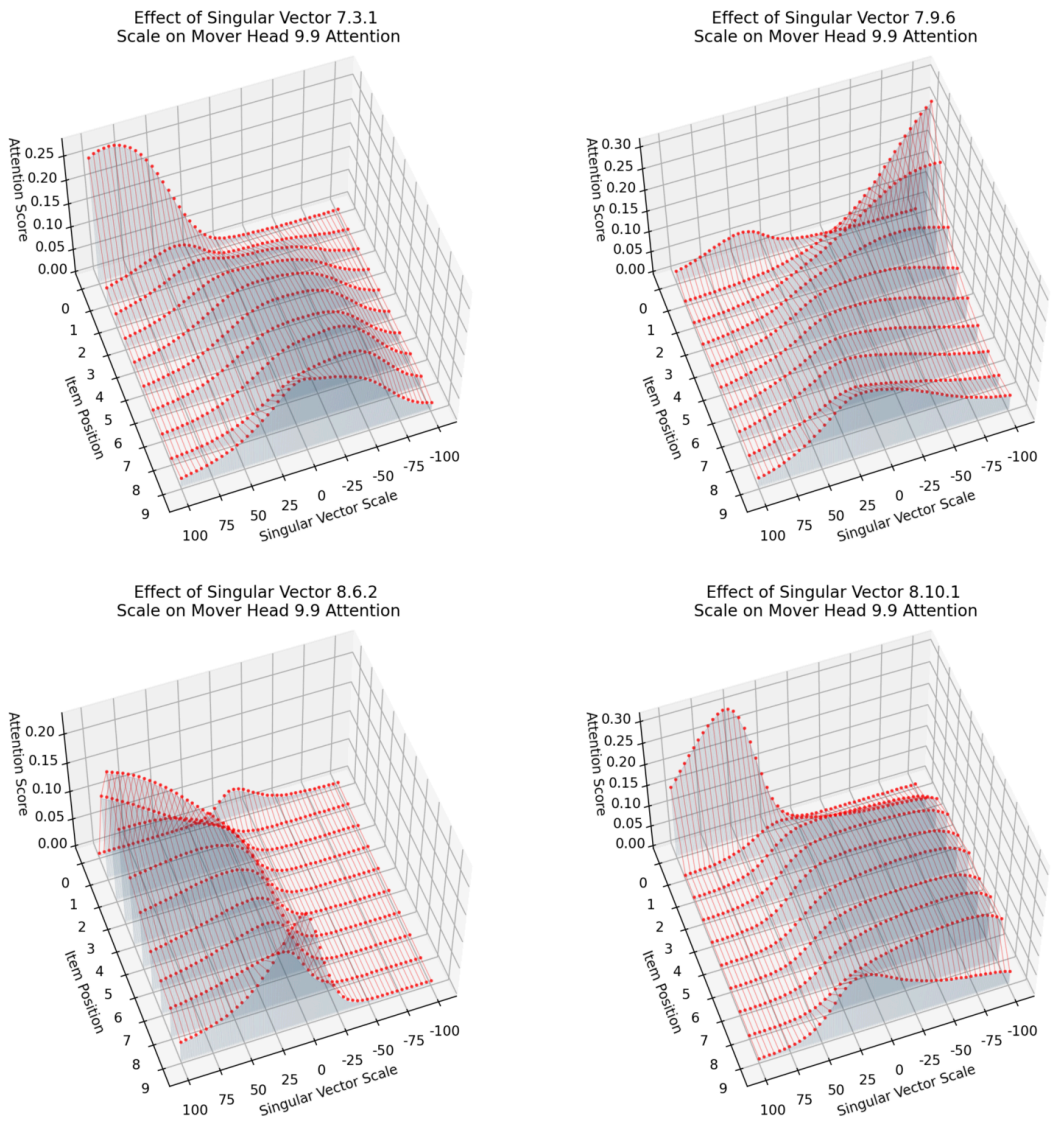

Figure 24: 10 Objects

J Compute

The models we use in this paper are small, in the range of 100M parameter tranfsormer models. The compute required to reproduce the results is therefore relatively small, but does require access to modern GPUs. We primarily used Nvidia 3090 GPUs for this work. Running the linear combinations of inhibition components in Section 5 was the most expensive experiment. Each dataset took about 12 hours on either a RTX 3090 or Quadro RTX gpu.