Latent Concept Disentanglement in Transformer-based Language Models

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2025

Abstract

When large language models (LLMs) use in-context learning (ICL) to solve a new task, they seem to grasp not only the goal of the task but also core, latent concepts in the demonstration examples. This begs the question of whether transformers represent latent structures as part of their computation or whether they take shortcuts to solve the problem. Prior mechanistic work on ICL does not address this question because it does not sufficiently examine the relationship between the learned representation and the latent concept, and the considered problem settings often involve only single-step reasoning. In this work, we examine how transformers disentangle and use latent concepts. We show that in 2-hop reasoning tasks with a latent, discrete concept, the model successfully identifies the latent concept and does step-by-step concept composition. In tasks parameterized by a continuous latent concept, we find low-dimensional subspaces in the representation space where the geometry mimics the underlying parameterization. Together, these results refine our understanding of ICL and the representation of transformers, and they provide evidence for highly localized structures in the model that disentangle latent concepts in ICL tasks.

1. Introduction

Do the representations of LLMs reflect latent structures present in in-context learning tasks?

LLMs excel at in-context learning (ICL) tasks [11], and there has been much interest in understanding how they learn from in-context demonstrations. This has ranged from work on training models from scratch on various canonical tasks [9, 15], circuit-level mechanistic investigations of pre-trained models [14, 38], and work on understanding how LLMs encode the underlying task [22, 48]. This last line of work has found that LLMs encode in-context learning tasks through specific directions in the weight space. These directions are known as “task vectors” or “concept vectors”, and prior research showed that manipulating (or steering) the representation of the LLM along these directions can modify the expression of the associated task. The study of these representations closely relates to the linear representation hypothesis [7, 29, 33], which says that many (or most) concepts are represented linearly in the representation space of LLMs.

Prior work on task vectors suggests that LLMs can recognize certain concepts during in-context learning, partially answering the leading question at the beginning of this section. However, our understanding of LLMs’ representations is still relatively coarse. While previous works exhibit task vectors for certain in-context learning tasks, they do not answer if the task vectors themselves reflect latent structures and concepts present in the task. If different instances of the task share some underlying latent relationship — such as shared intermediate reasoning steps, or shared parameterization between the tasks — then do task vectors reflect these latent concepts or structures? Our goal is understand if LLMs successfully disentangle such latent concepts via task vectors.

To understand the disentanglement capability of LLMs, we do a mechanistic study of multiple models on a variety of ICL tasks. We choose these tasks to embody various latent structures, such as intermediate reasoning steps or an underlying continuous parameterization. We show that transformer-based models can successfully disentangle latent concepts in these tasks; task vectors in the weight space can capture latent concepts and reflect the underlying parameterization. Therefore, our findings provide evidence that when provided with data with some underlying structure, LLMs can successfully disentangle this latent structure, and moreover, this disentanglement can happen in a highly interpretable fashion via task vectors. Hence, through a few well-chosen problems, we gain visibility into the capabilities and internal workings of transformer-based language models.

2. Disentanglement for Latent Discrete Task in Multi-hop Reasoning

Prior works have mechanistically analyzed how LLMs solve ICL problems that require a single step of reasoning over world knowledge, such as geography puzzles "Country→Capital", "National Park→Country" [41, 50]. However, whether and how LLMs can solve ICL problems with *under-specified reasoning steps*, or essentially those requiring *latent multi-hop reasoning* remains unclear.

The "Source→Target" problem. We create two-hop ICL puzzles by composing two facts linked by a common "Bridge" entity.[1] That is, we sample fact tuples $\{(S_i, r_1, B_i, r_2, T_i)\}_{i=1}^{n}$, where the Source entity S_i is related to the Bridge entity B_i via relation r_2, and B_i is related to the Target entity T_i via r_2. We then create the ICL puzzle in the form $[S_1, T_1. S_2, T_2...S_n,]$ [Answer: T_n]. Note that the bridge entities B_i's are never specified in the prompt. We primarily work with geography puzzles, with input types {City, University, Landmark}, and output types {Capital, Calling code}. An example problem is `Sydney, Canberra. Nantes, Paris. Oshawa,` — here, r_1 is "belongs to the country of", and r_2 is "capital of". Therefore, the prompt's answer is `Ottawa`, the capital of Canada, the country `Oshawa` is in.[2]

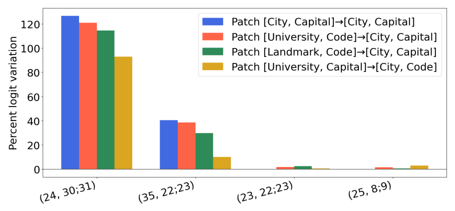
As shown in Figure 6, Gemma-2-27B [16] achieves high accuracy ($> 80\%$) at 20 shots. How is the LLM solving these harder, "source→target" problems that *do not specify the bridge, or even hint at the compositional nature of the problem at all*? In the following, we show a surprising finding that a highly sparse set of attention heads are responsible for "resolving the bridge value".

2.1. Bridge-value resolution for answering the query

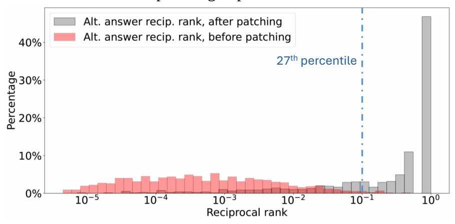
We analyze how the LLM infers the answer (target entity) from the query (source entity), given sufficiently many in-context examples. We will show both causal and correlational evidence that the LLM latently infers the bridge entity as an abstract concept representation, and compose it with output-concept representations to produce the answer. To achieve this, there are two main steps of our experiments, which we discuss below.

We perform activation patching [42, 53] on normal and altered problem pairs with different bridge entities (different countries), across different source and target types, at the last token position. If our hypothesis of certain attention heads resolving the bridge value from S_n for prediction is correct, then this bridge representation must be *transferable across source and target types*: for instance, patching a [University→Calling Code] prompt's activation (i.e. the "altered activation") onto that of a normal prompt [City→Capital], should cause the model to favor the alternative prompt's answer, but in the form of Capital (instead of Calling Code).

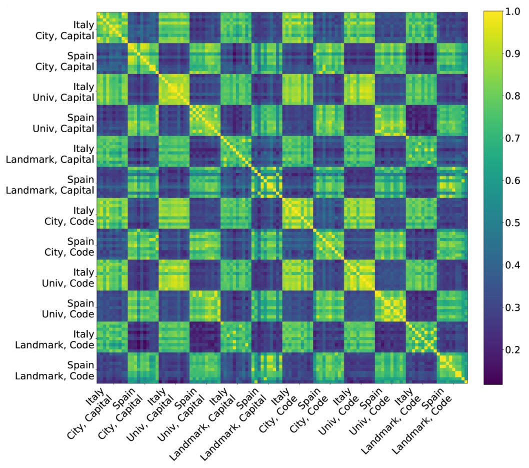
1. We think of this as a systematic ICL version of TWOHOPFACT [49]. Here, the model must figure out relations between the *input-bridge* and *bridge-output* facts from the ICL examples.
2. We discuss dataset details in the appendix.



(a) Logit difference variations from select activation patching experiments



(b) Reciprocal rank of the altered answer before and after patching head group (24,30;31)



(c) Cosine similarity of "Italy" and "Spain" representations from head group (24,30;31)

Figure 1: Evidence of the bridge-resolving attention heads in Gemma-2-27B. We present causal evidence in (a) and (b), and correlational evidence in (b). (a) and (b) provide causal evidence and (c) provides correlational evidence for the bridge-resolving heads, which we elaborate in the main text.

This suggests a slightly unorthodox intervention experiment. Instead of taking the normal-alternative answer pairs to be the ground truths, we convert the alternative answer into the same output type as the normal one, i.e., set $\hat{T}_n^{(alt)} = \text{Type}_{norm}(T_n^{(alt)})$. The logit difference and (reciprocal) rank of the alternative answer would use $\hat{T}_n^{(alt)}$ instead of the raw target $T_n^{(alt)}$. For instance, if the target type of the normal prompt is Capital, and the alternative prompt ends with "Sydney, 61", then $\hat{T}_n^{(alt)} = $ Canberra.

We report the results in Figure 1. In Figure 1(a), we report results on a select set of patching experiments: on *both* tasks with and without overlap in source and target types, we observe that a sparse set of attention heads consistently exhibits very strong causal effects; the head group (24,30;31) is especially dominant.[3] To further understand whether (24,30;31) is really boosting the alternative answer (instead of only decreasing model's confidence on the normal answer, which logit difference might not tell), in Figure 1(b), we show an example patching experiment result of [University, Code]→[City, Capital]. Surprisingly, at least 73% of the time, patching this single head group can boost the model's rank of the alternative prompt's answer into top 10 (and directly become the top-1 answer more than 40% of the time!), when its original rank, obtained on the normal prompt without intervention, is typically in the hundreds to thousands.

Finally, to understand the nature of (24,30;31)'s output embeddings better, in Figure 1(c), we visualize an example cosine similarity matrix of this attention head, with either "Italy" or "Spain" as the bridge values for S_n (the query source entity) in the prompts, across a total of 12 different combinations of bridge, source and target types. Specifically, for each combination of the bridge and source-target type shown in the grid, we sample 10 prompts which obey such requirement[4], giving us a total of 120 prompts. We then obtain head group (24,30;31)'s embedding of these prompts at the last token position, and compute the pairwise cosine similarities. Observe that the embedding

3. In our CMA experiments, we account for grouped-query attention by patching heads in groups of 2 on Gemma-2-27B. We noticed that this tends to produce stronger causal effects than with individual heads.
4. We only specify the bridge entity for S_n in the prompt; the bridge for S_i for all $i < n$ are randomly chosen.

consistently exhibits strong *disentanglement* with respect to the bridge value in the prompt, *regardless of source and target types*. We delay further analysis of the circuit and more general statistics of disentanglement strength of the bridge concepts to the appendix due to space limitations.

3. Disentanglement for Latent Continuous Parameterization

In this section, we consider two problems with numerical or continuous parameterization. For these experiments we study a very small transformer, with a similar architecture to GPT-2 [35]. We use a 2-layer 1-head transformer, with embedding dimension 128, trained with the AdamW optimizer. Additional details about training and hyperparameter choices are in the Appendix.

add-k Problem. Each task is a sequence consisting of pairwise examples $\{(x_i, y_i)\}_{i=1}^{n+1}$, where $y_i = x_i + k$, for a given offset k. Here, we use integer inputs and offsets; all values are in $\{0, \dots, V-1\}$, each treated as a distinct token. We consider a collection of K tasks parameterized by different offset values in $\{k_i\}_{i=1}^{K}$, where $k_1 = 1$ and we fix $k_{i+1} - k_i = 3$. The model is trained autoregressively to predict the label for each example in the sequence. At test time, the model observes the first n examples and should predict $y_{n+1} = x_{n+1} + k$ for the last example.

Circular-Trajectory Problem. Here a task consists of a sequence $\{\mathbf{x}_i\}_{i=1}^{n+1}$ of points on a circle centered at the origin. Each task is parameterized by the circle's radius r; for K tasks, the set of radii $\{r_i\}_{i=1}^{K}$ is sampled uniformly from $[1, 4]$. A task sequence is generated as follows. We first sample θ_0 uniformly at random in $[0, \frac{\pi}{2}]$, so $\mathbf{x}_1 = r[\cos\theta_0, \sin\theta_0]^T$. Then, we select the *period* p randomly from $\{2, 3, 4\}$, which determines the number of equal consecutive step-sizes. Specifically, we first sample a sequence of $\lfloor \frac{n}{p} \rfloor + 1$ unique step-sizes uniformly between $[0, 1]$, and then get the full sequence of steps $\{a_i\}_{i=1}^{n}$, where $a_j = a_{j+1} = \cdots = a_{j+p-1}$ for $j \in \{0, p, 2p, \dots\}$. Here, context length $n = 12m + 1$ for integer m. We also sample $c \in \{\pm 1\}$, which denotes if the trajectory is clockwise or anticlockwise. Next, we generate a sequence of angles $\{\theta_i\}_{i=1}^{n}$, where $\theta_i = \theta_0 + c\frac{2\pi}{n}\sum_{j \le i} a_j$. Using the sequence of angles, we generate the sequence, $\mathbf{x}_{i+1} = rR(\theta_i)\mathbf{x}_i$, where $R(\theta)$ is the 2D rotation matrix for θ. Figure 9 shows an example. As in the previous problem, we train the transformer autoregressively on these types of sequences.

Existence of Task Vectors. We first outline the process to identify the task vectors for the add-k problem. We set $V = 100$, $n = 4$, and $K = 2$. Figure 2 shows the cosine similarities between the layer-2 attention embeddings at the last position for 200 input sequences from each of the two tasks. We observe strong clustering between intra-task embeddings. This shows that the model disentangles the concept of different offset values in its representation.

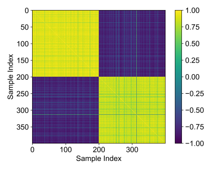


Figure 2: Cosine similarities between the layer-2 attention embeddings for 200 input sequences from two tasks/offsets for the *add-k* problem. Strong clustering between intra-task embeddings shows that the model disentangles the concept of different offsets.

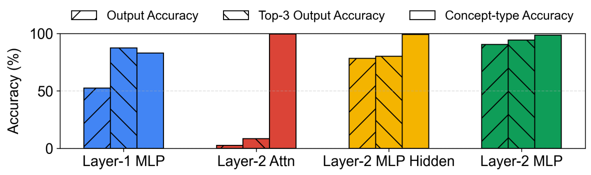


Figure 3: Results for linear probing the embeddings of the trained model at various locations to predict the final output and the task type for the *add-k* problem. The task type becomes disentangled at layer-2 attention, and the output is computed in layer-2 MLP.

To provide causal evidence for disentanglement, and locate where the task vectors emerge in the model, we linear probe the embeddings of the trained model at various locations to predict the final output and the offset/task type. We probe embeddings from the output of the MLP at the first layer, the attention block at the second layer, and the hidden and output layers of the MLP at the second layer. The results are shown in Figure 3. We observe that task type becomes disentangled at layer-2 attention, and the output is computed at layer-2 MLP. For each task, we treat the layer-2 attention embeddings averaged across 200 input sequences from that task as the task vector.

Geometry of Task Vectors. We analyze the geometry of the task vectors for the two problems. We visualize the task vectors by performing PCA and projecting them onto the first two PCs.

Figure 4 presents the 2D PCA projection of the task vectors for the add-k problem, for $K = 4, 8, 16$ tasks/offsets. We observe that in all three settings, the task vectors lie on a 1D linear manifold. In each of the three cases, more than 99.9% of the variance is explained by the first PC. Notably, the model compresses the concept of offsets into a line with the ordering of the offsets (lower to higher) preserved on the manifold (left to right). To corroborate these results, we study the effect of steering using the task vectors, and include the results in the Appendix.

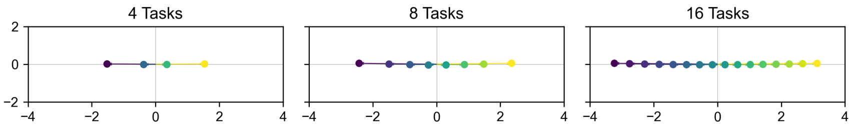


Figure 4: 2D PCA projection of the task vectors for the *add-k* problem. The task vectors lie on a 1D linear manifold. Here the number of tasks refers to the number of values of k.

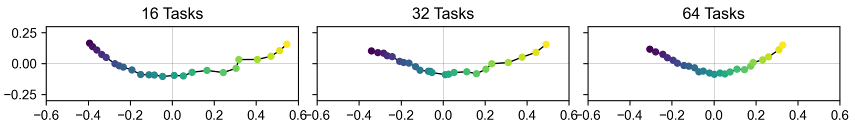


Figure 5: 2D PCA projection of the task vectors for the Circular-Trajectory problem. The task vectors lie on a smooth low-dimensional manifold. Here the number of tasks refers to the number of radius values used for training.

Figure 5 presents the 2D PCA projection of the task vectors for the Circular-Trajectory problem, for $K = 16, 32, 64$ (training) tasks/radii. In this setting, we consider $K = 24$ radii, spaced evenly between $[1, 4]$ to visualize the task vectors, since this task is continuous. We observe that in all three settings, the task vectors lie on a low-dimensional manifold. The variance explained by the first two PCs in the three cases is $97.05\%, 96.44\%, 93.68\%$, respectively. Similar to the previous setting, the order of the radii (lower to higher) is preserved in the compressed representation.

4. Discussion and Conclusion

We showed that transformer-based models latently *disentangle* core concepts in the provided in-context examples, and *manipulate* them well. For 2-hop tasks, we found that models contain sparse sets of attention heads responsible for first inferring the bridge entity and then resolving the output. For tasks with a continuous parameterization, we found that the model uses task vectors which closely capture the underlying parameterization. It would be interesting to understand the extent of this disentanglement across more diverse ICL tasks with various types of latent structures.

References

[1] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. In *Int. Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

[2] Badr AlKhamissi, Greta Tuckute, Antoine Bosselut, and Martin Schrimpf. The llm language network: A neuroscientific approach for identifying causally task-relevant units, 2025. URL https://arxiv.org/abs/2411.02280.

[3] Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction, 2024. URL https://arxiv.org/abs/2406.11717.

[4] David D. Baek and Max Tegmark. Towards understanding distilled reasoning models: A representational approach, 2025. URL https://arxiv.org/abs/2503.03730.

[5] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection, 2023. URL https://arxiv.org/abs/2306.04637.

[6] Aleksandra Bakalova, Yana Veitsman, Xinting Huang, and Michael Hahn. Contextualize-then-aggregate: Circuits for in-context learning in gemma-2 2b. *arXiv preprint arXiv:2504.00132*, 2025.

[7] Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Aggregate and conquer: detecting and steering llm concepts by combining nonlinear predictors over multiple layers. *arXiv preprint arXiv:2502.03708*, 2025.

[8] Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Aggregate and conquer: detecting and steering llm concepts by combining nonlinear predictors over multiple layers, 2025. URL https://arxiv.org/abs/2502.03708.

[9] Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and LLMs by learning to learn discrete functions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ekeyCgeRfC.

[10] Jannik Brinkmann, Abhay Sheshadri, Victor Levoso, Paul Swoboda, and Christian Bartelt. A mechanistic analysis of a transformer trained on a symbolic multi-step reasoning task. *arXiv preprint arXiv:2402.11917*, 2024.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In

H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[12] Clément Dumas, Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. Separating tongue from thought: Activation patching reveals language-agnostic concept representations in transformers, 2025. URL https://arxiv.org/abs/2411.08745.

[13] Ezra Edelman, Nikolaos Tsilivis, Benjamin L. Edelman, Eran Malach, and Surbhi Goel. The evolution of statistical induction heads: In-context learning markov chains. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=qaRT6QTIqJ.

[14] 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.

[15] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=flNZJ2eOet.

[16] Gemma Team. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

[17] Tianyu Guo, Hanlin Zhu, Ruiqi Zhang, Jiantao Jiao, Song Mei, Michael I Jordan, and Stuart Russell. How do llms perform two-hop reasoning in context? *arXiv preprint arXiv:2502.13913*, 2025.

[18] Lovis Heindrich, Philip Torr, Fazl Barez, and Veronika Thost. Do sparse autoencoders generalize? a case study of answerability, 2025. URL https://arxiv.org/abs/2502.19964.

[19] Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=QYvFUlF19n.

[20] Guan Zhe Hong, Nishanth Dikkala, Enming Luo, Cyrus Rashtchian, Xin Wang, and Rina Panigrahy. How transformers solve propositional logic problems: A mechanistic analysis. *arXiv preprint arXiv:2411.04105*, 2024.

[21] Xinyan Hu, Kayo Yin, Michael I Jordan, Jacob Steinhardt, and Lijie Chen. Understanding in-context learning of addition via activation subspaces. *arXiv preprint arXiv:2505.05145*, 2025.

[22] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic, 2023. URL https://arxiv.org/abs/2212.04089.

[23] Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition. *arXiv preprint arXiv:2502.00873*, 2025.

[24] Yuxiao Li, Eric J. Michaud, David D. Baek, Joshua Engels, Xiaoqing Sun, and Max Tegmark. The geometry of concepts: Sparse autoencoder feature structure. *Entropy*, 27(4), 2025. ISSN 1099-4300. doi: 10.3390/e27040344. URL https://www.mdpi.com/1099-4300/27/4/344.

[25] Zhuowei Li, Zihao Xu, Ligong Han, Yunhe Gao, Song Wen, Di Liu, Hao Wang, and Dimitris N. Metaxas. Implicit in-context learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=G7u4ue6ncT.

[26] Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[27] Sheng Liu, Haotian Ye, Lei Xing, and James Zou. In-context vectors: making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[28] Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets, 2024. URL https://arxiv.org/abs/2310.06824.

[29] Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Language models implement simple Word2Vec-style vector arithmetic. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5030–5047, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.281. URL https://aclanthology.org/2024.naacl-long.281/.

[30] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.759. URL https://aclanthology.org/2022.emnlp-main.759/.

[31] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer*

Circuits Thread, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[32] Core Francisco Park, Ekdeep Singh Lubana, and Hidenori Tanaka. Competition dynamics shape algorithmic phases of in-context learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XgH1wfHSX8.

[33] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *International Conference on Machine Learning*, pages 39643–39666. PMLR, 2024.

[34] Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models, 2025. URL https://arxiv.org/abs/2406.01506.

[35] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. In *OpenAI blog*, 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

[36] Nived Rajaraman, Marco Bondaschi, Ashok Vardhan Makkuva, Kannan Ramchandran, and Michael Gastpar. Transformers on markov data: Constant depth suffices. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=5uG9tp3v2q.

[37] Jacob Russin, Ellie Pavlick, and Michael J. Frank. The dynamic interplay between in-context and in-weight learning in humans and neural networks, 2025. URL https://arxiv.org/abs/2402.08674.

[38] 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. In *International Conference on Machine Learning*, pages 45637–45662. PMLR, 2024.

[39] Aaditya K. Singh, Ted Moskovitz, Sara Dragutinovic, Felix Hill, Stephanie C. Y. Chan, and Andrew M. Saxe. Strategy coopetition explains the emergence and transience of in-context learning, 2025. URL https://arxiv.org/abs/2503.05631.

[40] Curt Tigges, Oskar John Hollinsworth, Neel Nanda, and Atticus Geiger. Language models linearly represent sentiment, 2024. URL https://openreview.net/forum?id=iGDWZFc7Ya.

[41] Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

[42] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural*

Information Processing Systems, volume 33, pages 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

[43] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35151–35174. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/von-oswald23a.html.

[44] Dimitri von Rütte, Sotiris Anagnostidis, Gregor Bachmann, and Thomas Hofmann. A language model's guide through latent space, 2024. URL https://arxiv.org/abs/2402.14433.

[45] 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.

[46] Zijian Wang and Chang Xu. Functional abstraction of knowledge recall in large language models, 2025. URL https://arxiv.org/abs/2504.14496.

[47] Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah Goodman. Interpretability at scale: Identifying causal mechanisms in alpaca. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=nRfClnMhVX.

[48] Liu Yang, Ziqian Lin, Kangwook Lee, Dimitris Papailiopoulos, and Robert Nowak. Task vectors in in-context learning: Emergence, formation, and benefit, 2025. URL https://arxiv.org/abs/2501.09240.

[49] Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10210–10229, 2024.

[50] Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning? *arXiv preprint arXiv:2502.14010*, 2025.

[51] Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning?, 2025. URL https://arxiv.org/abs/2502.14010.

[52] Zeping Yu and Sophia Ananiadou. Interpreting arithmetic mechanism in large language models through comparative neuron analysis. *arXiv preprint arXiv:2409.14144*, 2024.

[53] Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Hf17y6u9BC.

[54] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models, 2023. URL https://arxiv.org/abs/2205.10625.

[55] Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. Pre-trained large language models use fourier features to compute addition. In *NeurIPS*, 2024.

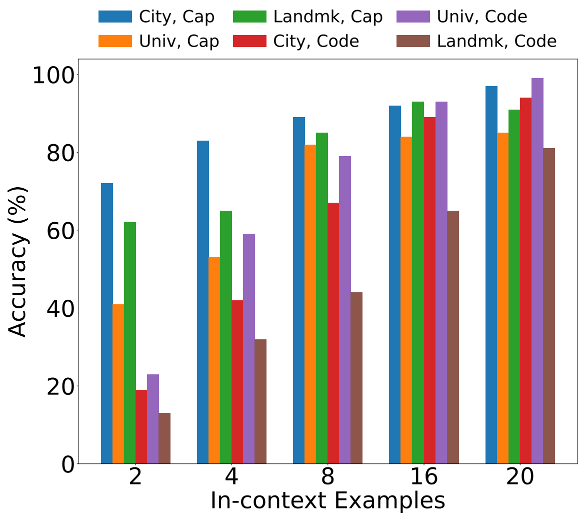


Figure 6: Accuracy of Gemma-2-27B on the two-hop "Source→Target" ICL problems.

Appendix A. Additional Details/Results for Section 2

A.1. Further details on problem setup

We collect the puzzles' data for over 40 countries in the world. For each source type in {City, University, Famous Landmark}, we collect at least 10 entities per country, leading to over 400 possible source entities per source type in the puzzles.

Furthermore, we show the accuracy of Gemma-2-27B on the problems with different number of in-context examples in Figure 6.

A.2. Output-concept attention heads

We discussed in the main text the evidence for the bridge-resolving heads. Here, we elaborate on how the "second step" of the ICL problem is solved, that is, how the model specializes the abstract "bridge value" concept representation into a concrete output type (i.e. "Capital" vs. "Calling Code"). To support the hypothesis that the LLM refines the bridge value via certain output-concept representations, we need to perform CMA on normal-alternative prompt pairs which differ only in output type, and examine their disentanglement strength with respect to the output type.

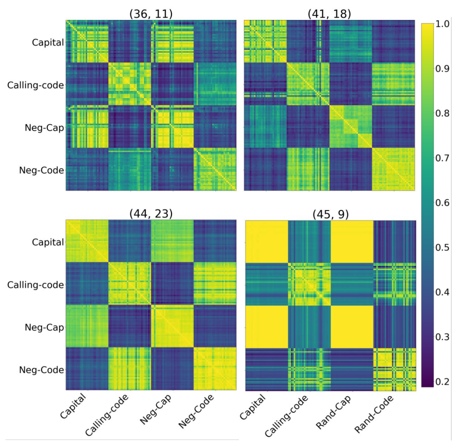


Figure 7: Cosine similarity maps of "output-concept" heads, the majority of which are in the deeper layers.

In particular, to construct an altered prompt, we keep everything in it the same as the normal demonstrations, besides the output type: for instance, if the normal output type is Capital, then the altered output type is Calling Code. This helps us isolate components which are *sensitive to the output concept*, *without*

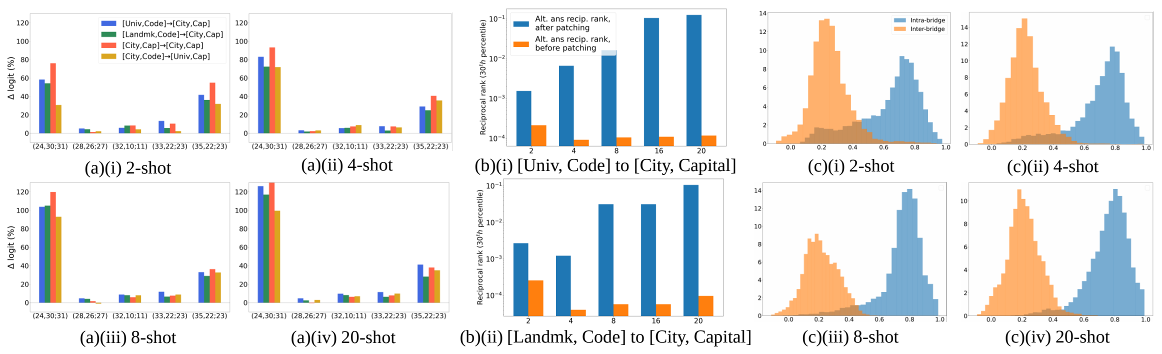


Figure 8: This figure illustrates how the transferability and disentanglement of the bridge representation increases as we increase the number of in-context examples. Figure series (a) and (b) present the transferability result, obtained by performing cross-problem-type patching, and measuring the causal influence of the patched representation. In (a)(i) to (iv), we plot the percentage logit variation of the attention heads found to output "bridge" values, measured on several intervention experiments. For (b)(i) and (ii), we zoom in on head group (24,30;31), and show its causal effects on two patching experiments. The x-axis is the number of in-context examples, and the y-axis is the 30^{th} percentile of the reciprocal rank of the alternative prompt's answer. For (c)(i) to (iv), we plot the disentanglement strength of the representations of head group (24,30;31).

surfacing components which are *bridge-value dependent*. We visualize the cosine similarity matrix of the top-scoring heads, shown in Figure 7.

In addition to measuring cosine similarity on our multi-hop prompts with Capital or Calling Code type output, we also design *negative-control* prompts, where we randomly shuffle the output. This leaves the prompts without structure, except asking for the model to output tokens belonging to either the "Capital" or "Calling Code" concept. We note that, as long as the output concept agrees, there is high similarity of these heads' embeddings on the multi-hop and out-concept-only prompts.

A.3. Multi-hop circuit formation and the number of in-context examples

This sub-section focuses on illustrating the relation between the number of in-context examples versus (1) how strong a role the multi-hop mechanism plays in the LLM's inference (via causal interventions), (2) disentanglement strength of key bridge-resolving attention heads. As we will show below, there is a general positive correlation between the number of shots and the two factors.

More demonstrations $\Longrightarrow$ stronger causal score. Figure 8 visualizes the experimental results. From Figure 8(a) and (b) and sub-figures, we observe a correlation between the number of shots (ICL examples) and the bridge-resolving heads' "causal importance" in the model's inference. When the number of shots is low, we find that they tend to exhibit weak causal influence on the model's inference. For instance, as (b)(i) shows, at 2 shots, the 30^{th} percentile of the alternative answer's rank *after* patching at (24,30;31) is on the order of 10^3. This is in stark contrast to how strong this head group's causal influence is at 20 shots as we saw before.

More demonstrations $\Longrightarrow$ stronger disentanglement, with a catch. In Figure 8(c)(i) to (iv), we observe that the intra-bridge cosine similarity tends to cluster better as the number of shots increase, while the inter-bridge cosine similarities decay toward 0.2, with the two distributions overlapping less and less. Interestingly, the bridge-disentanglement strength is still non-trivial

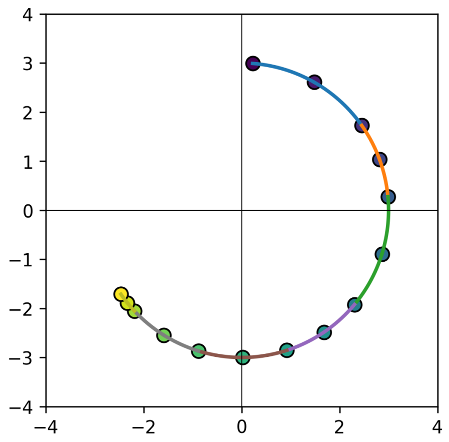


Figure 9: Illustration of an input sequence for the circle trajectory problem. Here, radius $r = 3$, period $p = 2$, sequence length $n = 13$. Every p consecutive steps on the trajectory are equal. We first sample $\lfloor \frac{n}{p} \rfloor + 1$ unique step-sizes in $[0, 1]$, and get the full sequence $\{a_1, a_2, a_3, a_4, \ldots\}$, where same colors denote equal step-sizes. Then, we generate the trajectory by rotating point $\mathbf{x}_i$ clockwise by angle $a_i \cdot \frac{2\pi}{n}$ (see text for formal description).

with very few shots, mirroring the causal-intervention results: regardless of how disentangled the representations are in the very-few-shot regime, the LLM does not "realize" how it should utilize the multi-hop sub-circuit.

Appendix B. Additional Details/Results for Section 3

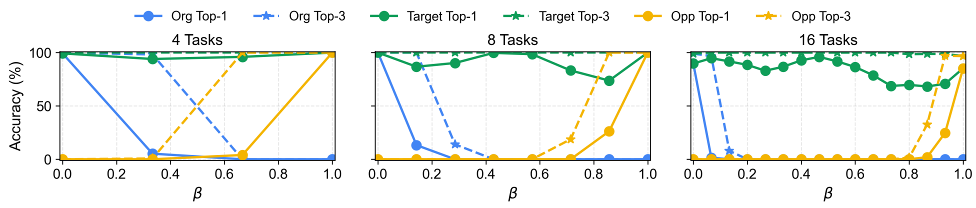


Figure 10: Steering with the task vectors for tasks k_1 and k_K for the *add-k* problem (see text for details). We plot the top-1 and top-3 accuracies for predicting the output based on the original offset k_1 (k_K), the 'opposite' offset k_K (k_1), or the target offset $(1-\beta)k_1 + \beta k_K$ ($(1-\beta)k_K + \beta k_1$), where $\beta \in [0, 1]$, The result shows that the model output can be steered toward the target.

Results for Steering with the Task Vectors. Let $\mathbf{t}_1$ and $\mathbf{t}_K$ denote the task vectors for tasks k_1 and k_K, respectively. Then, for task k_1 (k_K), we consider steering with $(1-\beta)\mathbf{t}_1 + \beta\mathbf{t}_K$ ($(1-\beta)\mathbf{t}_K + \beta\mathbf{t}_1$) and evaluate the accuracy for predicting the output based on the original offset k_1 (k_K), the 'opposite' offset k_K (k_1), or the target offset $(1-\beta)k_1 + \beta k_K$ ($(1-\beta)k_K + \beta k_1$), where $\beta \in [0, 1]$. Figure 10 presents the top-1 and top-3 accuracies for each case. High top-1 accuracies and $\approx 100\%$ top-3 accuracies for the target for all considered values of β indicate that the model output is steered toward the target. This shows that interpolating along the top principal direction

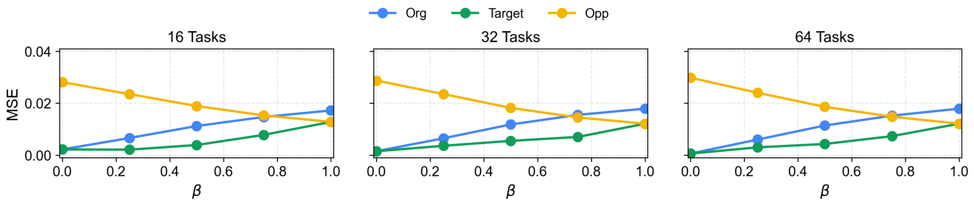


Figure 11: Steering with the task vectors for tasks r_1 and r_K for the Circular-Trajectory problem (see text for details). The MSE between the radius inferred from the model output and the original radius r_1 (r_K), the 'opposite' radius r_K (r_1), or the target radius $(1-\beta)r_1 + \beta r_K$ ($(1-\beta)r_K + \beta r_1$), where $\beta \in [0, 1]$, indicates that the model output can be steered toward the target.

is successful at interpolating values of k in the task space, showing that the model is somewhat strikingly successful at capturing the latent concept.

Figure 11 presents the results for steering the model output using the task vectors for radii r_1 and r_K. We follow the same procedure as in the add-k problem, with a different evaluation metric. We compute the norm of the generated output after steering as the model's radius (since the center of the circles is fixed at the origin), and consider the MSE between these radii and the original radius r_1 (r_K), the 'opposite' radius r_K (r_1), or the target radius $(1-\beta)r_1 + \beta r_K$ ($(1-\beta)r_K + \beta r_1$), where $\beta \in [0, 1]$, averaged over 200 sequences from each task. We observe that the MSE with the target radius is the lowest, which indicated the task vector can steer the model's output toward the target.

Appendix C. Related Work

Task and Function Vectors. Task vectors [22] refer to directions within the latent space of a model associated with a task. Other works have identified related concepts such as in-context vectors [27], function vectors [41, 51], and context vectors [25]. Work on addition tasks finds Fourier or trigonometric interpretations of LLM computations [23, 52, 55]. [48] examines the emergence of task vectors in transformers trained from scratch on various ICL tasks. In contrast, our focus is understanding if the structure of the task is captured by the representations through task vectors. Our work also complements the function vector analysis from contemporaneous work [21], providing add-k results for smaller models where we have full control over training and can hence conclude that the geometry of the task vector only arises from the latent task structure. We also compare results from add-k with other ICL tasks, giving additional insights.

Linear Representation Hypothesis (LRH). Our results are also connected to the LRH, which essentially speculates that LLMs represent high-level concepts in (almost) linear latent directions [8, 12, 22, 24, 29, 33, 34]. Many papers motivated by the LRH then find "concept" vectors that can capture directions of truthfulness [3, 28], sentiment [40], humor [44], etc. We deepen this study, asking how LLMs' representations capture/disentangle latent input concepts. In addition, the LRH is rooted in the field of mechanistic interpretability, which aims to reverse engineer mechanisms in transformer-based LMs [2, 4, 6, 10, 14, 18, 20, 31, 39, 42, 45–47].

In-context Learning Interpretation. ICL abilities of transformer-based models were first observed by Brown et al. [11], which sparked work in analyzing this ability. This includes analyzing how pretrained LLMs solve ICL tasks requiring abilities such as copying, single-step reasoning, basic linguistics [19, 30, 31, 41, 54], and smaller models trained on synthetic tasks like regression [1, 5, 15, 17, 43], discrete tasks [9], and mixture of Markov chains [13, 32, 36]. These setups enable discovery of relations between in-context and in-weight learning [26, 37, 39], and internal algorithms that models implement [13, 31, 32, 50]. We contribute to this line of work, by shedding light on how transformers solve ICL problems which have more intricate latent structures.