How Do Transformers Fill in the Blanks? A Case Study on Matrix Completion

Pulkit Gopalani [1] GOPALANI@UMICH.EDU
Ekdeep Singh Lubana [1,2] ESLUBANA@UMICH.EDU
Wei Hu [1] VVH@UMICH.EDU
[1] *University of Michigan, Ann Arbor* [2] *CBS, Harvard University*

Abstract

We formulate the low-rank matrix completion problem as a masked language modeling (MLM) task, and train a BERT model to solve this task. We find that BERT succeeds in matrix completion and outperforms the classical nuclear norm minimization method. Moreover, the mean–squared–error (MSE) loss curve displays an early plateau followed by a sudden drop to near-optimal values, despite no changes in the training procedure or hyper-parameters. To gain interpretability insights, we examine the model's predictions, attention heads, and hidden states before and after this transition. We observe that (i) the model transitions from simply copying the masked input to accurately predicting the masked entries; (ii) the attention heads transition to interpretable patterns relevant to the task; and (iii) the embeddings and hidden states encode information relevant to the problem.

1. Introduction

This paper investigates the behavior of Transformers trained on the classical mathematical task of low-rank matrix completion [5] to gain insights into the mechanisms of Transformers and their training process. In this setup, we assume access to a matrix with some fraction of its entries missing, and would like to complete the missing entries assuming the ground truth matrix is low-rank (Appendix A). By treating a matrix as a sequence of tokens, we find that training a BERT model [11] in an online manner can successfully solve this problem to a small error. Moreover, BERT can outperform the classical nuclear norm minimization algorithm for matrix completion, suggesting that BERT does not simply recover this classical algorithm. Further, the MSE loss curve during training undergoes a sudden decrease (Fig. 3), marking the transition to a model that generalizes well, also observed in [7] for BERT trained in natural language setups. We find that this decrease in loss marks an *algorithmic shift* from the pre-transition model simply copying the input (predicting 0 at masked positions) to the post-transition model accurately predicting missing values at masked positions.

2. BERT Solves Matrix Completion

For BERT model (with parameters θ) and masked matrix $\tilde{X}$ and model output $\hat{X} := \hat{X}(\tilde{X};\theta) \in \mathbb{R}^{n\times n}$, the training objective $L(\theta)$ is the MSE loss at all positions,

$$L(\theta) = \frac{1}{n^2}\sum_{i,j=1}^{n}(X_{ij} - \hat{X}_{ij})^2.$$

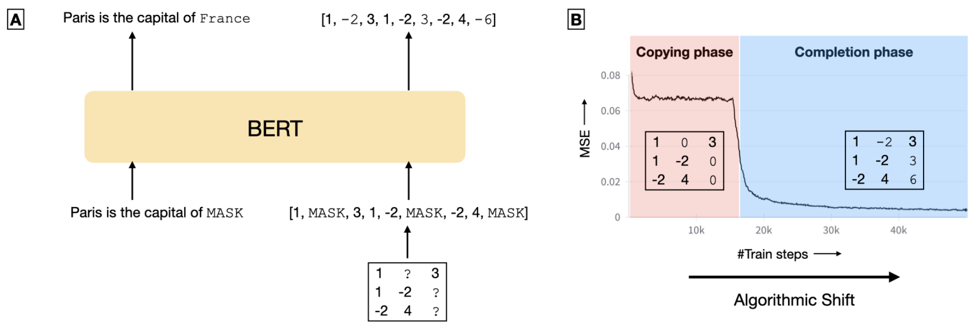


Figure 1: (**A**) Matrix completion using BERT; (**B**) *Algorithmic shift* marked by sharp decrease in loss. The model shifts from simply copying the input (*copying phase*) to computing missing entries accurately (*completion phase*).

Further, we separately track MSE over observed and masked entries, defined as

$$L_{obs} = \frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} (X_{ij} - \hat{X}_{ij})^2 \quad ; \quad L_{mask} = \frac{1}{|\Omega^C|} \sum_{(i,j)\in\Omega^C} (X_{ij} - \hat{X}_{ij})^2$$

for Ω the set of observed entries. Data for matrix completion is generated as

$$X = UV^\top; \quad U, V \in \mathbb{R}^{n\times r}, \;\; U_{ij}, V_{ij} \overset{\text{iid}}{\sim} \text{Unif}[-1, 1] \;\; \forall i, j \in [n] \times [r]$$

so that X has rank at most r. To mask entries at random, we sample binary matrices $M \in \{0, 1\}^{n\times n}$ such that $M_{ij} = 0$ with probability p_{mask}, indicating that the element at position (i, j) is masked in the input matrix, i.e., $\Omega = \{(i, j) \mid M_{ij} = 1\}$. In the subsequent sections, we analyse a 4–layer, 8–head BERT model trained upto MSE $\sim$ 4e$-$3 for analysing training dynamics and interpretability. Pre–shift denotes the model at step 4000, and post–shift denotes model at the end of training (step 50000). Please see Appendix B for further experiment details.

Nuclear Norm Minimization Nuclear norm minimization is one of the most widely studied approaches towards low–rank matrix completion (Appendix A). We use CVXPY to solve matrix completion using nuclear–norm minimization at various levels of p_{mask} comparing it to the output of a BERT model trained on $p_{\text{mask}} = 0.3$. We find that BERT performs better than nuclear norm minimization w.r.t. MSE; at the same time, the nuclear norm of BERT solution is larger (Fig. 2). Further, we also solve the regularized version of the above problem (Eq. 2) to attempt to match the performance of BERT at some $\lambda > 0$. We find that BERT still outperforms regularized MSE minimization (details in Appendix D).

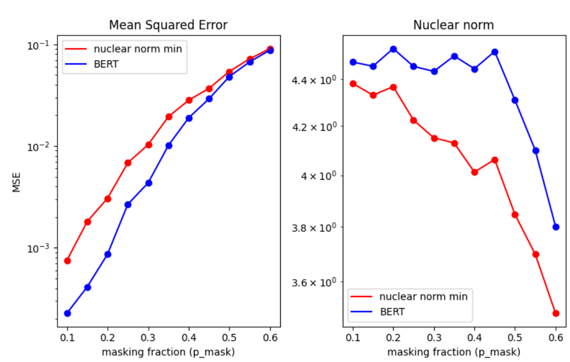


Figure 2: BERT outperforms nuclear norm minimization

3. Matrix Completion Capability Emerges during Training

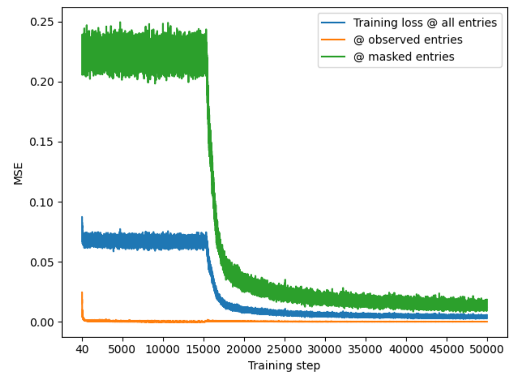


Figure 3: Sharp reduction in training loss shortly after step 15000

We observe a sharp decrease in training loss at approximately step 15000 (Fig. 3). Observe that this decrease in total loss is driven almost exclusively by the corresponding decrease in L_{mask}, since L_{obs} is very close to 0 both before and after the drop. We hypothesize that this sudden drop is caused by an algorithmic shift, i.e., the model switches to a different, more accurate algorithm for prediction at missing entries and hence L_{mask} rapidly decreases following that shift. Moreover, since L_{obs} barely changes during this algorithmic shift, we further argue that (1) the model has two distinct mechanisms for prediction at masked and observed entries after the algorithmic shift, and (2) that the mechanism for prediction at observed positions is not significantly affected by this algorithmic shift. *We note that this sudden drop is in MSE loss, i.e., not in a discontinuous metric like accuracy.* Hence, the idea of *sudden* emergence being an artifact of discontinuous metrics and poorly defined evals [31] is unlikely to explain the full story in our setting.

3.1. Before the Algorithmic Shift – Copying

In this section, we demonstrate that before the algorithmic shift, the model simply copies the input at all positions in the matrix, through the following approach – replace the masked elements in the 7×7, rank-2 input by the token corresponding to a real value m. For such input, we would like to see whether the model implements copying and outputs m at the masked positions. For model output $\hat{X}$ on this input, MSE at observed positions is L_{obs}, and for masked positions the MSE is defined as

$$L'_{mask} = \frac{1}{|\Omega^C|} \sum_{(i,j)\in\Omega^C} (\hat{X}_{ij} - m)^2.$$

L_{obs} and L'_{mask} for this experiment averaged over 512 samples are compiled in Row 1, Table 1 (Appendix C). The small loss values verify the copying hypothesis – model output matches the ground truth at observed positions, while at masked positions it outputs a value nearly equal to m, the replacement mask value. Note that when the mask token is MASK (i.e., no replacement), we set $m = 0$, indicating that the model is outputting 0 at the masked locations. This hypothesis is also confirmed for random 7×7 input matrices, that is, all entries i.i.d. uniformly in $[-1, 1]$ (not necessarily low–rank); results in Row 2, Table 1.

Role of Attention Heads Attention heads at this stage (Fig. 13(a)) do not appear to attend to any specific tokens in an interpretable manner. In fact, the final structure of the attention heads does not start to appear unless just after the algorithmic shift (Figure 13). Since the model is simply copying the masked input, we hypothesize that attention heads (that combine different tokens) are inconsequential to the model output. To quantitatively verify this hypothesis we use *uniform ablation*: simply replace the softmax probabilities in the attention head by $1/n^2$ for all elements i.e. equally

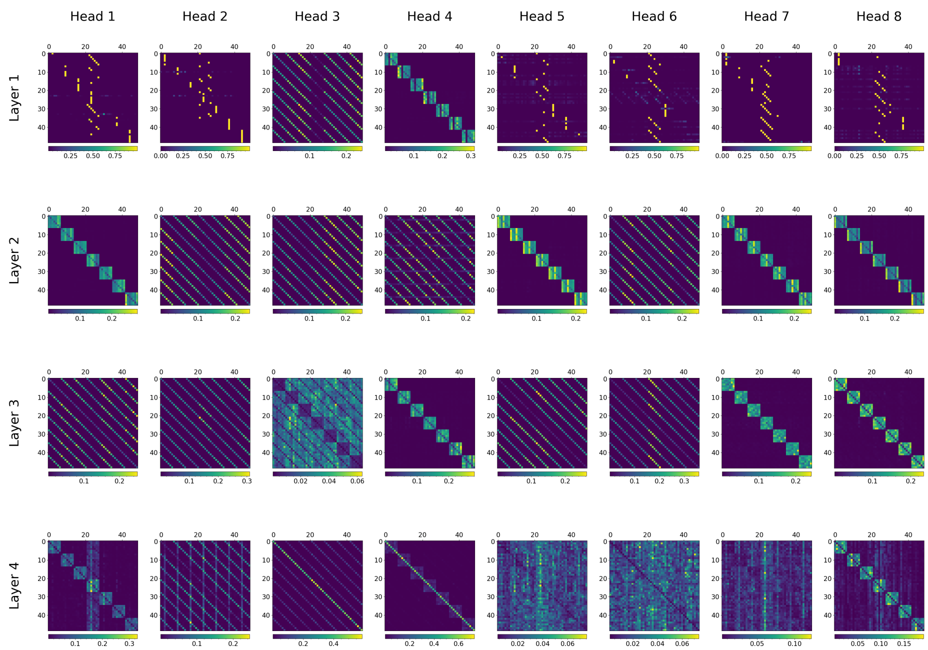


Figure 4: Attention heads in post–shift model show distinct regions in the input they attend to.

attend to all tokens (Sec. 4.6, [18]). With these ablations, there is negligible change in model performance at both observed and masked positions. Averaged over 256 samples, $L_{obs} = 3.4\text{e}{-4}$ and $L_{mask} = 0.2236$ when using all attention heads; whereas, on ablating all heads, these values are $3.2\text{e}{-4}$ and 0.2236 respectively. Clearly in this case, attention heads do not substantially affect the model prediction.

As further confirmation, we replace the key, query and value weights in the pre–shift model by those from the post-shift model. Averaged over 256 samples, L_{obs} is 5e–3, that is similar to the optimal total MSE obtained at the end of training, while $L_{mask} = 0.2246$, similar to that obtained without replacing the weights.

3.2. After the Algorithmic Shift – Matrix Completion

We analyze the post-shift model separately for missing and observed entries, with a focus on the role of attention heads given the apparent interpretable patterns in Fig. 4. We find that for observed entries, the model output is still not substantially affected by the attention heads, whereas, for masked entries this effect is substantial.

3.2.1. OBSERVED ENTRIES

To check the effect of attention heads, we uniformly ablate *all* attention heads in the post-shift model. Averaged over 256 samples, this leads to $L_{obs} = 9.2\text{e}{-5}$ when using all attention heads, compared to $3.7\text{e}{-3}$ with ablation (close to the total MSE at model convergence). However, L_{mask} increases from 0.0128 to 0.2183, essentially the value in the pre-shift model. Further, we replace attention (key, query, value) weights in the post–shift model by weights from a pre–shift model. Indeed, averaged over 256 samples, $L_{obs} = 9.5\text{e}{-4}$ in this case, supporting our claim.

Finally, since attention crucially depends on the position of elements, we randomly permute the positional embeddings in the post–shift model. That is, the embedding originally encoding position i in the input now represents position $\pi(i)$ for some random permutation $\pi : [n^2] \to [n^2]$. Averaged over 256 samples, $L_{obs} = 2.4\text{e}{-4}$, whereas $L_{mask} = 0.5687$, implying that the observed positions are negligibly affected compared to masked positions due to this intervention. Intuitively, positional information is not required for copying, and this result supports our 'sub–algorithm' hypothesis.

3.2.2. MISSING ENTRIES

To confirm that attention heads causally affect the model output for missing entries, in addition to uniform ablations, we perform *causal interventions* (activation patching) [37] on the hidden states just after the attention heads. This involves replacing the hidden state after an attention head for input A with the hidden state obtained at the same attention head, but for a different input A'. Ideally, if that head is causally relevant to the output, then such an intervention should steer the model towards the output for A', instead of A. We find in our case that for $A = X$ and $A' = -X$, such an intervention simultaneously on all attention heads steers the model output at missing entries towards $-X$ for input X (more details in Appendix I).

Denote attention head H in layer L by the tuple (L, H). We can group the attention heads depending on the specific regions of the input matrix they attend to: (a) the same row as the query element (the 'block–diagonal' patterns, e.g. (2, 1)); (b) the same column as the query element (the 'parallel–off–diagonal' patterns, e.g. (2, 2)); (c) the query element itself (the 'diagonal' patterns, specifically in the last layer, e.g. (4, 3)). There are also some other attention heads that do not neatly fit into either of these 3 categories – for example, all heads in layer 1 except (1,2), (1,3); (3,3); (4,2), (4,5–7). In this context, uniformly ablating heads (3,3), (4,2), (4,5–7) gives $L_{obs} = 9.36\text{e}{-5}$, $L_{mask} = 0.01575$ compared to $L_{obs} = 9.44\text{e}{-5}$, $L_{mask} = 0.01428$ without ablation, i.e. these uninterpretable heads do not significantly affect the output.

3.3. Additional Experiments

We investigate attention heads when the observed entries in the input are arranged in a structured manner to analyse the function of individual attention heads (Appendix E). Further, we also probe hidden states of intermediate layers to understand the working of the model (Appendix F). Moreover, positional and token embeddings of the model also exhibit structure relevant to the problem, in some cases *before* the algorithmic shift occurs (Appendix G).

4. Discussion

In a toy task of matrix completion with masked language modeling, we have shown that a sudden drop in training loss marks an emergent, algorithmic shift in the model from a phase of merely copying the input to actually solving the task. We have also demonstrated that components in the post–shift model display clear evidence of learning useful abstractions relevant to the task. The question of *why* this shift occurs suddenly, rather than gradually, is an important avenue for future work. A concrete characterization of the algorithm used by the model for computation is also an interesting direction for future research, but not the primary focus of this work.

Acknowledgements We thank Yu Bai, Andrew Lee, and anonymous reviewers for their helpful suggestions. WH acknowledges support from the Google Research Scholar Program.

References

[1] Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/8ed3d610ea4b68e7afb30ea7d01422c6-Abstract-Conference.html.

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

[3] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2017. URL https://openreview.net/forum?id=ryF7rTqgl.

[4] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=liMSqUuVg9.

[5] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009. doi: 10.1007/S10208-009-9045-5. URL https://doi.org/10.1007/s10208-009-9045-5.

[6] Francois Charton. Linear algebra with transformers. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=Hp4g7FAXXG.

[7] Angelica Chen, Ravid Shwartz-Ziv, Kyunghyun Cho, Matthew L Leavitt, and Naomi Saphra. Sudden drops in the loss: Syntax acquisition, phase transitions, and simplicity bias in MLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MO5PiKHELW.

[8] Xiang Cheng, Yuxin Chen, and Suvrit Sra. Transformers implement functional gradient descent to learn non-linear functions in context, 2024.

[9] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes, editors, *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4828. URL https://aclanthology.org/W19-4828.

[10] Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko,

Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/34e1dbe95d34d7ebaf99b9bcaeb5b2be-Abstract-Conference.html.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[12] 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.

[13] Deqing Fu, Tian-Qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn higher-order optimization methods for in-context learning: A study with linear models, 2023.

[14] 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.

[15] 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.

[16] Yifan Hou, Jiaoda Li, Yu Fei, Alessandro Stolfo, Wangchunshu Zhou, Guangtao Zeng, Antoine Bosselut, and Mrinmaya Sachan. Towards a mechanistic interpretation of multi-step reasoning capabilities of language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.299. URL https://aclanthology.org/2023.emnlp-main.299.

[17] Adam Jermyn and Buck Shlegeris. Multi-component learning and s-curves - ai alignment forum. URL https://www.alignmentforum.org/posts/RKDQCB6smLWgs2Mhr/multi-component-learning-and-s-curves.

[18] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of BERT. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4365–4374, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1445. URL https://aclanthology.org/D19-1445.

[19] János Kramár, Tom Lieberum, Rohin Shah, and Neel Nanda. Atp*: An efficient and scalable method for localizing llm behaviour to components, 2024.

[20] Nayoung Lee, Kartik Sreenivasan, Jason D. Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers, 2023.

[21] Michael A. Lepori, Thomas Serre, and Ellie Pavlick. Uncovering intermediate variables in transformers using circuit probing, 2023.

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

[23] Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 34651–34663. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/dfc310e81992d2e4cedc09ac47eff13e-Paper-Conference.pdf.

[24] Ziming Liu, Eric J Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zDiHoIWa0q1.

[25] Arvind V. Mahankali, Tatsunori Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8p3fu56lKc.

[26] Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Circuit component reuse across tasks in transformer language models, 2024.

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

[28] Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In Yonatan Belinkov, Sophie Hao, Jaap Jumelet, Najoung Kim, Arya McCarthy, and Hosein Mohebbi, editors, *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.blackboxnlp-1.2. URL https://aclanthology.org/2023.blackboxnlp-1.2.

[29] 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.

[30] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8, 2020. doi: 10.1162/tacl_a_00349. URL https://aclanthology.org/2020.tacl-1.54.

[31] Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ITw9edRDlD.

[32] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1580. URL https://aclanthology.org/P19-1580.

[33] 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.

[34] 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.

[35] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.

[36] Zhiwei Xu, Yutong Wang, Spencer Frei, Gal Vardi, and Wei Hu. Benign overfitting and grokking in reLU networks for XOR cluster data. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=BxHgpC6FNv.

[37] 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.

Appendix A. Low Rank Matrix Completion

Low-rank matrix completion is a well-studied problem in machine learning and statistics. This problem finds applications in recommender systems, where given an incomplete matrix of user ratings on some items, the goal is to recover the missing entries *assuming* the ground truth matrix is low-rank. For a matrix $X \in \mathbb{R}^{n\times n}$, denote its observed (visible) entries by the set $\Omega \subset [n] \times [n]$, and the set of missing entries by $\Omega^C = [n] \times [n] \setminus \Omega$. Formally, the problem is

$$\min_U \ \operatorname{rank}(U) \qquad \text{s.t. } U_{ij} = X_{ij} \ \ \forall (i,j) \in \Omega.$$

Nuclear norm minimization Since rank is not a convex function of the matrix entries, nuclear norm minimization [5] is a widely used convex optimization approach to low-rank matrix completion. The modified optimization problem is,

$$\min_U \|U\|_* \qquad \text{s.t. } U_{ij} = X_{ij} \ \ \forall (i,j) \in \Omega \tag{1}$$

where $\|U\|_*$ denotes the nuclear norm (sum of singular values) of matrix U. A regularized version of this problem for $\lambda > 0$ is

$$\min_U \left[\frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} (U_{ij} - X_{ij})^2 + \lambda \|U\|_* \right]. \tag{2}$$

Appendix B. Experimental details

Data Preprocessing We tokenize real values as follows: discretize the range $[-10, 10]$ (all matrix entries in our experiments are in this range) in steps of size $\epsilon = 0.01$, and assign token IDs to these values with IDs starting from 1; the mask token (MASK) is assigned token ID 0. Input to the transformer is the tokenized masked sequence $X_{mask} = \text{TOK}\left(\text{Vec}(X \odot M)\right)$, where TOK denotes tokenization, Vec denotes vectorizing the $n \times n$ matrix to a n^2-dimensional vector and $\odot$ denotes the element-wise product. Due to this discretization, in experiments, MSE will be with the rounded-off version of X to 2 decimals.

Training We use the BERT model implementation from the HuggingFace library [35], with 'absolute' positional embeddings and no dropout. Since the model maps a sequence of discrete token IDs to a sequence of real values, we compute the MSE loss between the real valued model output, and the discretized real values in the ground truth matrix. For example, for input sequence $[0.12, 0.45, 0.87] \in \mathbb{R}^3$, corresponding masked token sequence ["0.12", "MASK", "0.87"], and output $[x_1, x_2, x_3] \in \mathbb{R}^3$, the MSE loss is $\frac{1}{3}\left[(x_1 - 0.12)^2 + (x_2 - 0.45)^2 + (x_3 - 0.87)^2\right]$.

We use the MSE loss on *all* elements of the input and output matrices for training. We additionally fix the masking probability $p_{\text{mask}} = 0.3$ in all cases. Using a 12–layer, 12–heads per layer BERT model with a linear read–out layer, the train loss is optimized using Adam with constant step size 5e$-$5 for 50000 epochs (without weight decay or warmup). Data at each step is obtained by sampling 256 train and 64 test matrices. Since the training is 'online', train and test losses are nearly identical at all points in training, and thus we will not separately analyze them.

We additionally train a smaller BERT model with 4 layers and 8 heads per layer, with step-size 1e$-$4 on square matrices of order 7 and rank$-$2. We use this smaller model primarily to keep our interpretability analyses tractable; in any case the attention heads are similar to those in larger models (Appendix J). The model converges to a total MSE of the order $1e{-3}$ (i.e. solves matrix completion well) for all runs – square matrices of order $7, 10, 12, 15$ and rank $2, 3, 3, 4$ respectively. For the $4-$layer $8-$heads case, we obtain comparable performance (final total MSE $\sim 4e{-3}$) to the 12–layer, 12–head model.

Appendix C. Pre–shift copying

	Mask = "MASK"		Mask = "0.44"		Mask = "-0.24"	
Input Samples ↓	L'_{mask}	L_{obs}	L'_{mask}	L_{obs}	L'_{mask}	L_{obs}
Rank-2 matrices	3e-4	3.3e-4	4e-4	2.8e-4	3.7e-4	2.7e-4
Random matrices	2.8e-4	3.5e-4	3.7e-4	3e-4	3.6e-4	2.8e-4

Table 1: Pre–shift model implements copying, predicting the value for mask token at missing entries.

Appendix D. Nuclear Norm Minimization

We use the regularized version of the nuclear norm minimization problem as detailed in Sec. 2, and obtain the following L, L_{obs}, L_{mask} for various values of λ. We average our results over 256 samples generated in the same way as the training data for BERT (including rounding off to 2 decimal places) for the sake of comparison.

λ	L_{obs}	L_{mask}	L
0.0005	$1.015e{-5}$	0.040728	0.012173
0.001	$3.686e{-5}$	0.040456	0.01211
0.0015	$7.959e{-5}$	0.040505	0.012155
0.002	0.00013769	0.040734	0.012264
0.005	0.00078591	0.043402	0.013516

Appendix E. Attention Heads with Structured Mask

Since the maps in Fig. 4 are averaged over multiple random masks and input matrices, it is difficult to extract more specific details about the algorithm, apart from the coarse–grained insights as in Section 3.2.2. To remedy this, we generate inputs with specific mask structure, see for example Fig. 5. This implies that for different input matrices, the mask i.e. Ω^C remains the same. This step helps us highlight how an attention head attends to input elements based on the element being masked or observed. From the results in Fig. 5, it is evident that different attention heads focus on specific parts of the input. For instance,

1. (2, 1), (3,4) and (4,8) are significantly active only at the masked rows, and in those cases has maximal attention at the only observed positions in those rows. In other words, this head acts as a 'masked–only' head.

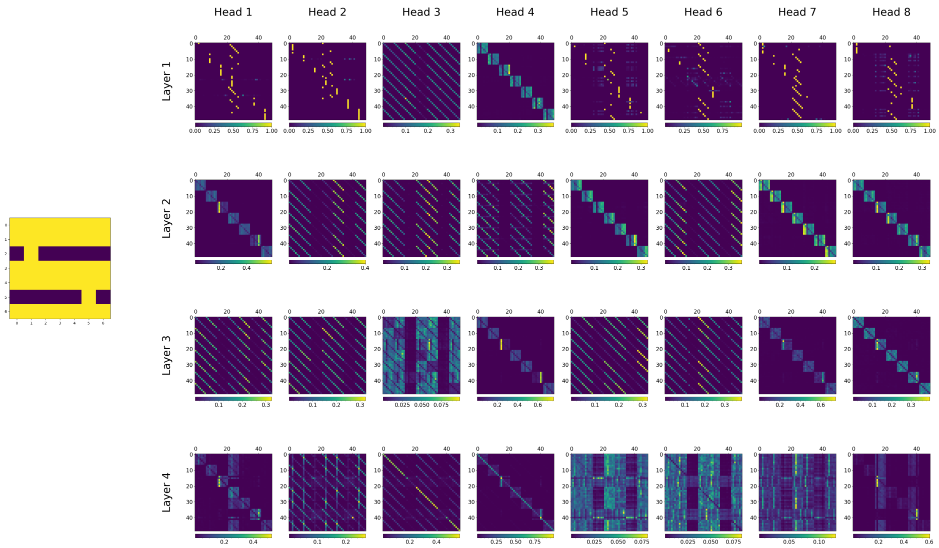


Figure 5: Attention heads for a structured mask attend to specific entries in the input. Left: structured mask (blue denotes missing entries)

2. (4,3) and (4,4) correspond roughly to an identity map, slightly deviating in the masked rows. In these cases, again the maximal attention score corresponds to the only observed position in these rows. That is, this head acts as an 'identity–map' head.

3. Further, there are multiple 'parallel off-diagonal' heads that completely ignore the masked rows for their computation. These heads include (2,2–4), (2,6); (3,2), (3,3), (3,5). Additionally, there are also attention heads like (3,1), (3,6) that attend to only the observed element of each masked row. Collectively these heads act as 'observed-only' heads, attending to only observed entries, and using this information to compute missing entries.

4. There also exist attention heads that respond systematically to changes in the mask. For example, consider attention heads (2, 5), (2, 7), (2, 8) in Fig. 10. For each row, these heads attend to the element in the 6th and 2nd column respectively for part (a) and (b). On a closer look, the connecting link between these two mask patterns is that, the longest contiguous unmasked column is exactly the column that these heads attend to. We hypothesize that this information is somehow used by the model in its inner computation for masked entries.

5. Finally, Heads (1,1–2), (1, 5–8) do not fall in any of the categories above . These heads are mostly static across different mask / input variations (for example, comparing Fig 4 and 5), and the patterns suggest that these heads almost exclusively focus on the middle row of the input matrix and some other elements. A possible function of these heads is to process positional and token embeddings (input to the first layer) so that this information can be used appropriately in the subsequent layers.

Appendix F. Probing

We probe for properties of the input matrix in the hidden states of the model, to concretely determine how the model computes the output. We use our 12–layer model in this case, for enhancing contrast between probing in different layers.

Specifically, for every element in the input, we use a linear probe [3] on its hidden state after a given layer, mapping the hidden state to the $n-$dimensional masked row that this element belongs to. Missing entries are replaced by 0, and the linear probe is fit using least squares. The results for this experiment in Fig. 6 demonstrate that, layer 3 and 4 in the model correspond quite strongly to the probe target, compared to other layers. This suggests that the model tracks input information in its intermediate layers and uses it for computation.

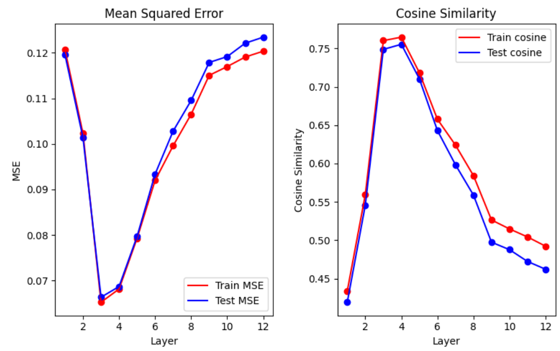
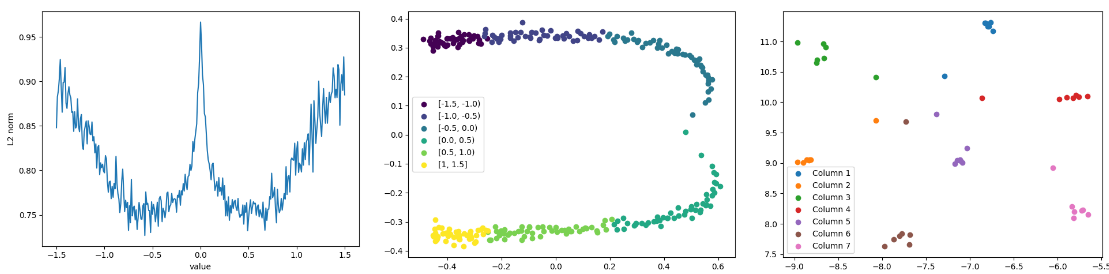


Figure 6: Layer 3 and 4 store information about the rows of the masked input matrix.

Appendix G. The Curious Case of Embeddings



(*a*) ℓ_2 norm of token embeddings is symmetric around 0

(*b*) PCA of token embeddings shows distinct components for sign and magnitude of real value

(*c*) Positional embeddings in the same column cluster together (t-SNE)

Figure 7: Embeddings in the post–shift model display interpretable behavior.

Interpretable Embeddings In the post–shift model, positional and token embeddings also exhibit interesting properties related to the input elements and structure. For instance, the ℓ_2 norm of token embeddings corresponding to values from -1.5 to 1.5 is symmetric w.r.t. 0 as seen in Fig. 7(a). Further, the PCA of token embeddings in Fig. 7(b) shows that the embeddings have a separable structure based on the sign of the real–valued input (y–axis), and continuous variation w.r.t. the absolute value of the real–valued input (x–axis).

The t-SNE projection of positional embeddings also show an interesting clustering pattern; positions in the same column tend to cluster together as seen in Fig. 7(c). This is especially important because we have not used any marker tokens to mark the end of a row or column. Additionally, the ℓ_2 norm of positional embeddings (Fig. 8) is nearly constant across positions, except for a drop at positions around $21 - 26$; that is, most of the middle row of the 7×7 input. This can be understood as the model marking the 'origin' of the position range from 1 to 49, and use it in subsequent computation.

From these observations about embeddings, it is clear that the model utilizes the actual real–value corresponding to the discretized tokens, and also has non–trivial positional information about the input that take into account the matrix structure relevant to the task.

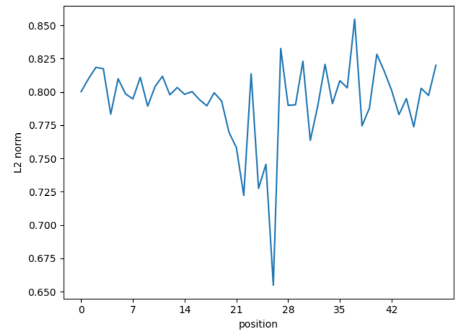
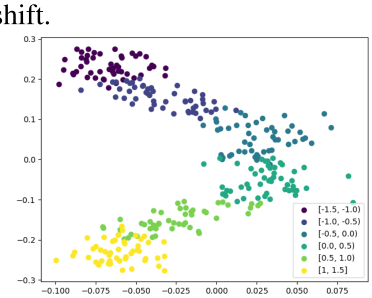


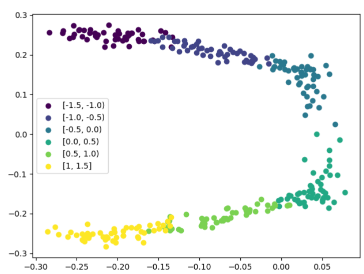
Figure 8: ℓ_2 norm of positional embeddings in post–shift model.

Do embeddings change abruptly? Unlike attention heads (Fig. 13), embeddings might not abruptly change with the algorithmic shift. Motivated by the experiments in [23], we compute the top–2 principal components of the token embeddings at the final step (50000), and project the token embeddings at intermediate training steps on these components. The results (Fig. 9) show that the embeddings align very closely to the final arrangement before the actual drop in loss.

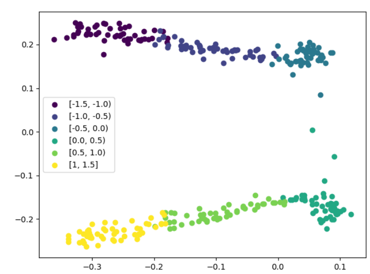
These results hint towards a conjecture that even though the model might undergo a sudden algorithmic shift, some components evolve beforehand and possibly are a driving force behind the shift.



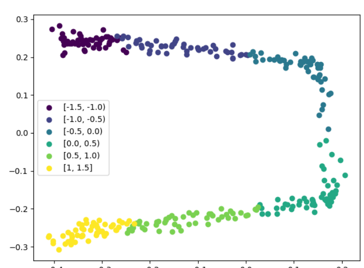
(*a*) Step 4000



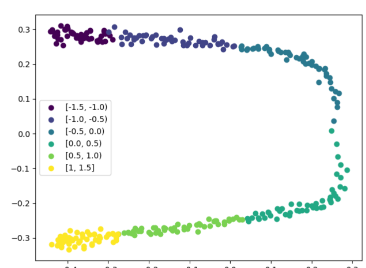
(*b*) Step 10000



(*c*) Step 14000



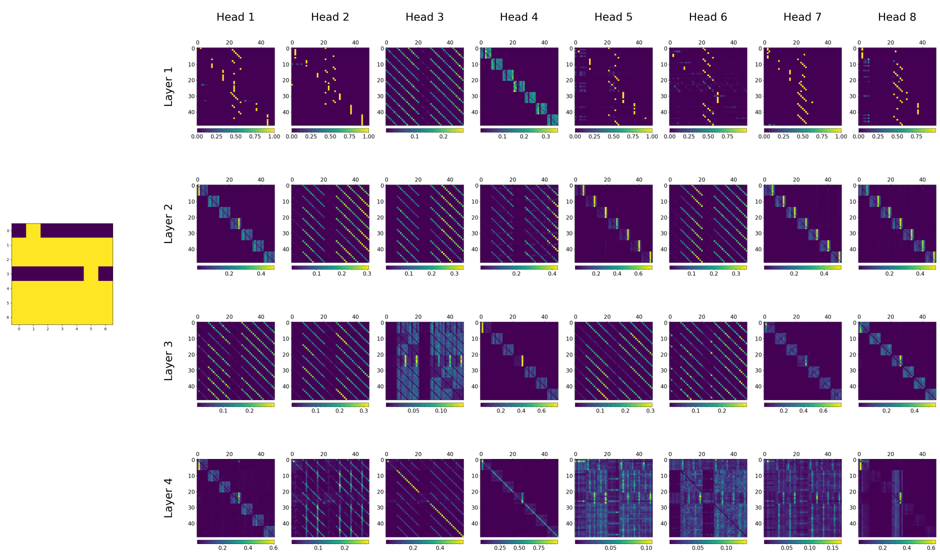
(*d*) Step 16000



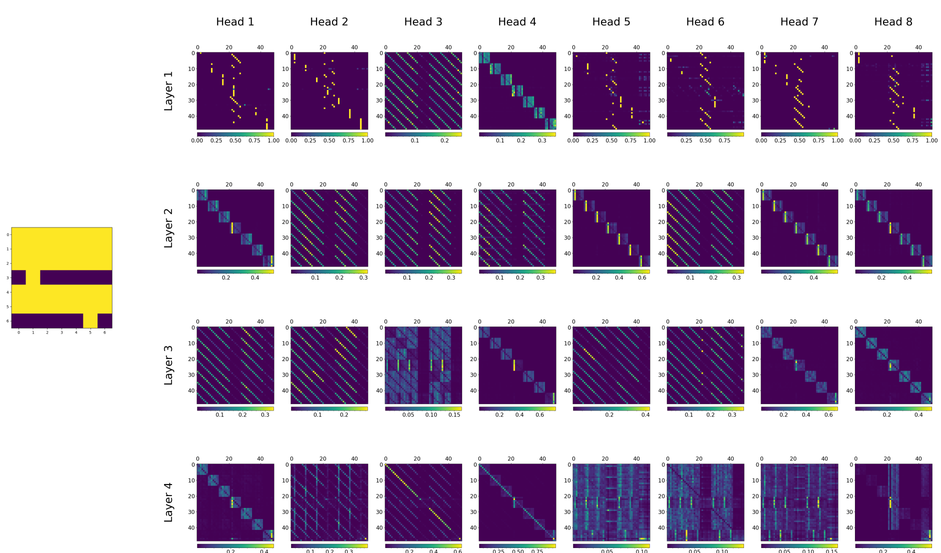
(*e*) Step 20000

Figure 9: Projection of token embeddings along principal components of embeddings at step 50000.

Appendix H. Effect of changing mask structure



(a)



(b)

Figure 10: Attention heads and corresponding masks; blue denotes masked position in the input matrix.

Appendix I. Causal effect of Attention heads

To verify whether attention heads actually contribute towards the model output, or are simply a side–effect of some other latent factor in the model, we employ 2 methods used earlier to quantify the contribution of attention heads in transformers.

1. **Uniform Ablation** Following the methodology in (Sec 4.6, [18]), for a square matrix input of order n, we set each element of the $n^2 \times n^2$ softmax attention matrix to $1/n^2$. That is, attend equally to all tokens in the input sequence, and remove any learned information about attending to specific positions in the input.

2. **Causal Interventions** In the uniform ablation setup, it is possible that setting the softmax probabilities to a given value might change the distribution of resultant hidden states, and consequently degrade model performance. A more principled technique to analyse the effect of a specific component is to replace the hidden state just after that component by hidden states on a different input, and analyse how this affects the final output [37]. In our case, we intervene on attention heads by replacing the hidden state after an attention head for input matrix X by the hidden state for input $(-X)$. Importantly, this change does not affect properties like rank of the input, and hence the hidden states obtained are from the same distribution as those for input X.

Pre–shift In the pre–shift model, we want to demonstrate that removing attention heads does not affect the model predictions significantly. For this, we uniformly ablate all attention heads in the pre–shift model, and measure the effect averaging over 256 samples. We get that $L_{obs} = 3.4\text{e}{-4}$ and $L_{mask} = 0.2236$ when using all attention heads; whereas, on ablating all heads, these values are $3.2\text{e}{-4}$ and 0.2236 respectively. Clearly, in the pre–shift model, attention heads do not substantially affect the model prediction.

Post–shift In the post–shift model, we want to demonstrate that the attention heads causally affect the output. Using uniform ablation, we get that $L_{obs} = 9.2\text{e}{-5}$ and $L_{mask} = 0.0128$ when using all attention heads; whereas, on ablating all heads, these values are $3.7\text{e}{-3}$ and 0.2183 respectively.

From these observations, we could claim causal effect of attention heads for prediction at missing entries. A stronger test however is through causal interventions,

Step 1 Extract the hidden states for all attention layers from the model on some input matrix X; call these h_+. Concretely, these hidden states are obtained just after the matrix product of the softmax attention probabilties and the value matrix and hence before the output matrix product.

Step 2 Change the input to the model to $-X$, however, also replace the hidden states *just after* the attention layers with h_+ obtained in Step 1. Call the output of the model in this setup as $f_p(-X, X)$.

We observe that, the MSE between $f_p(-X, X)$ and X, averaged over 256 samples at masked positions is approximately 0.014 (this is comparable to optimal L_{mask}), compared to the MSE between $f_p(-X, X)$ and $-X$ being 0.8066. This demonstrates that the attention heads are causally relevant to the model output for missing entries.

Appendix J. Attention Heads for larger inputs

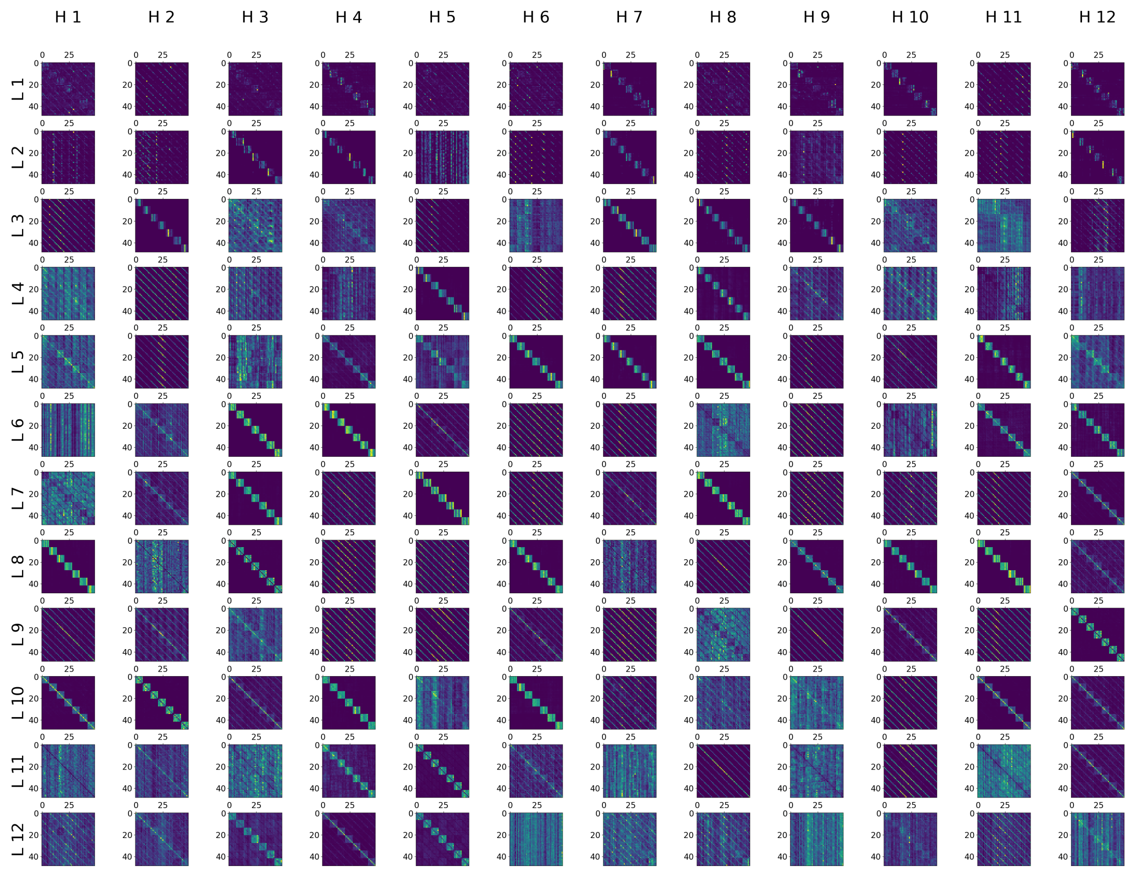


Figure 11: Attention heads in 12 layers, 12–heads model on 7×7 rank–2 input

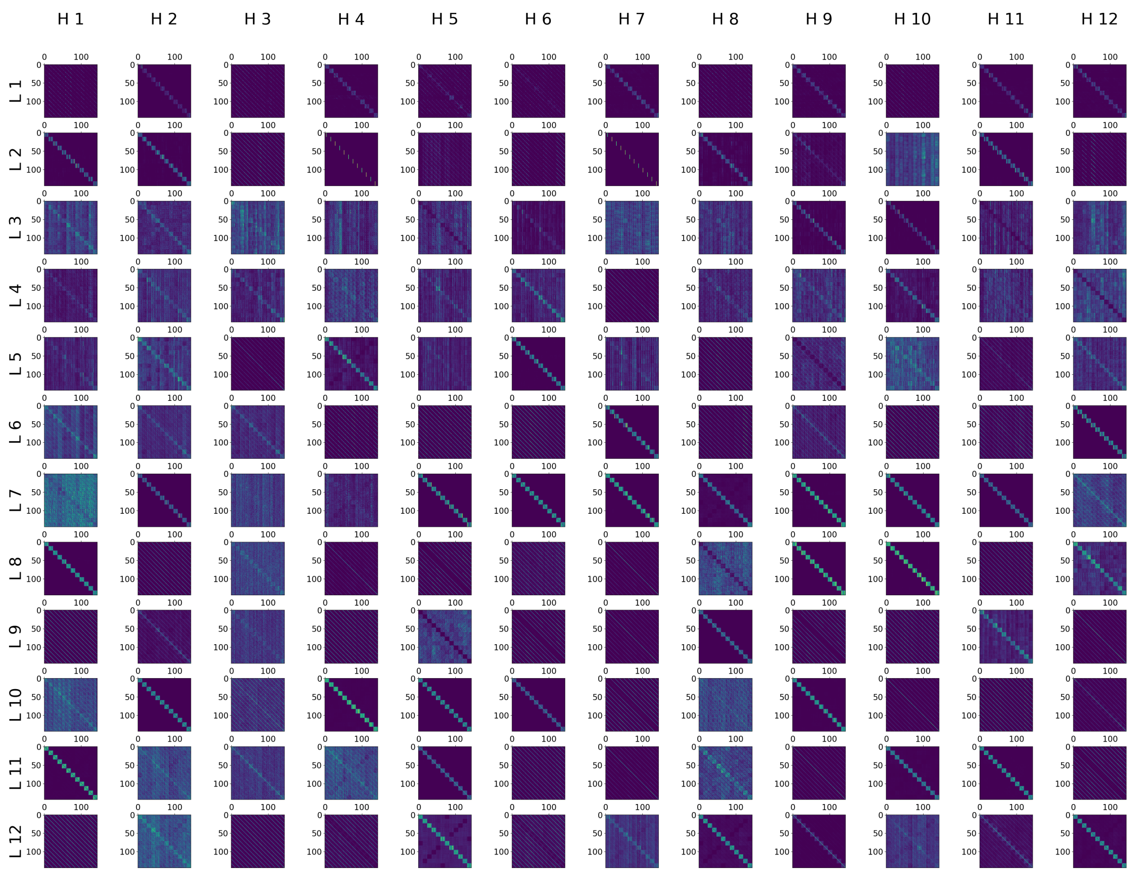


Figure 12: Attention heads in 12 layers, 12–heads model on 12×12 rank–3 input

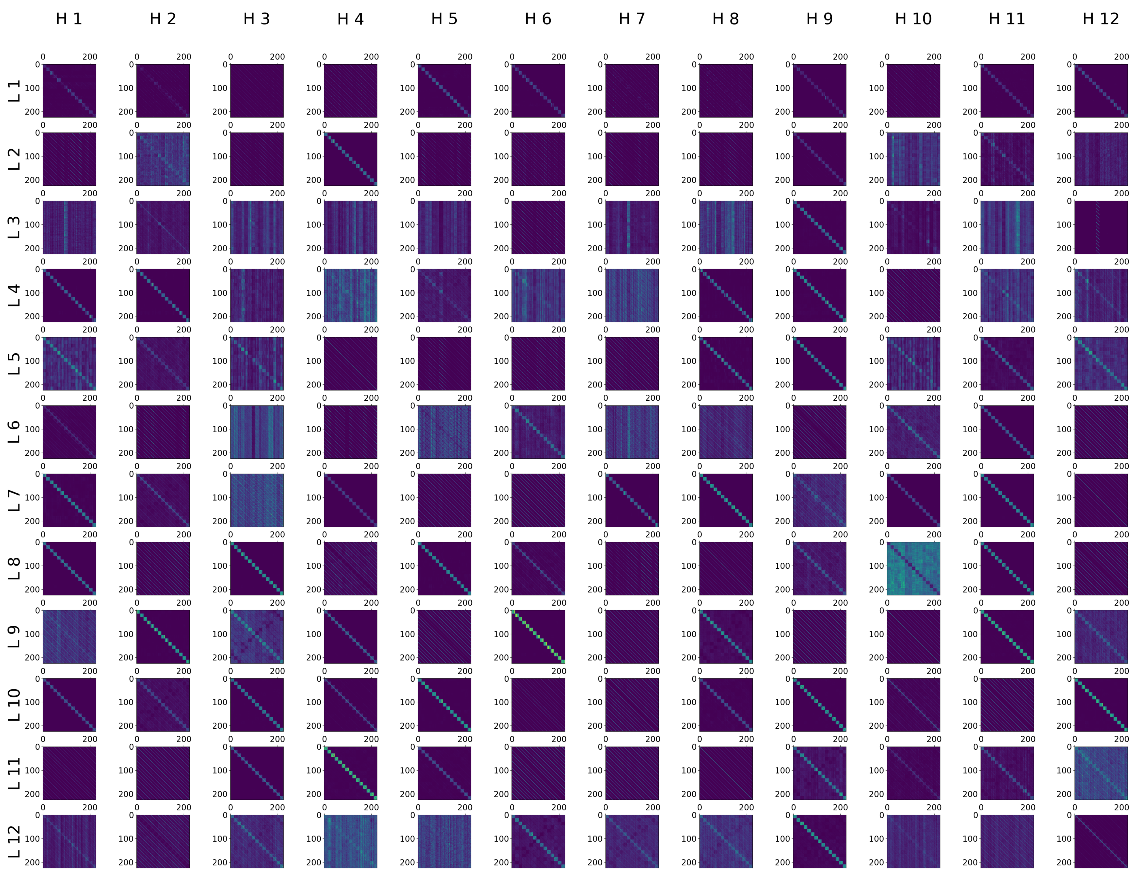
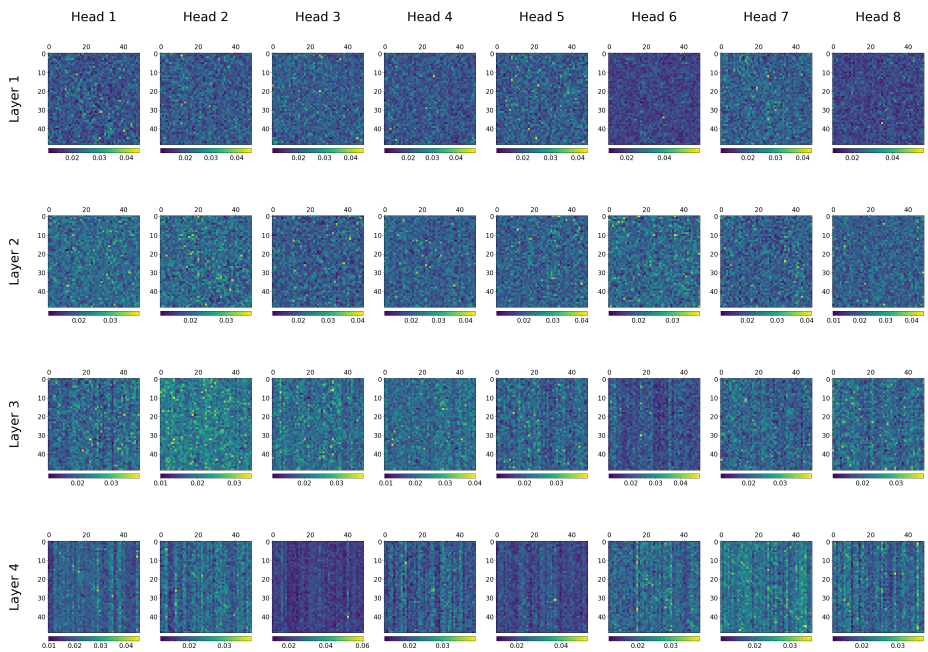
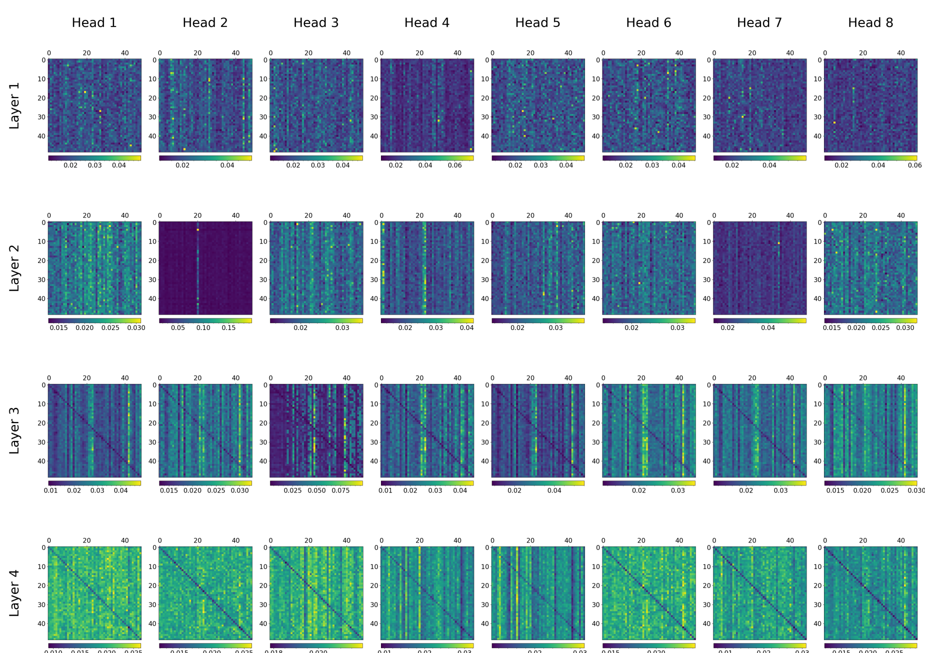


Figure 13: Attention heads in 12 layers, 12–heads model on 15×15 rank–4 input

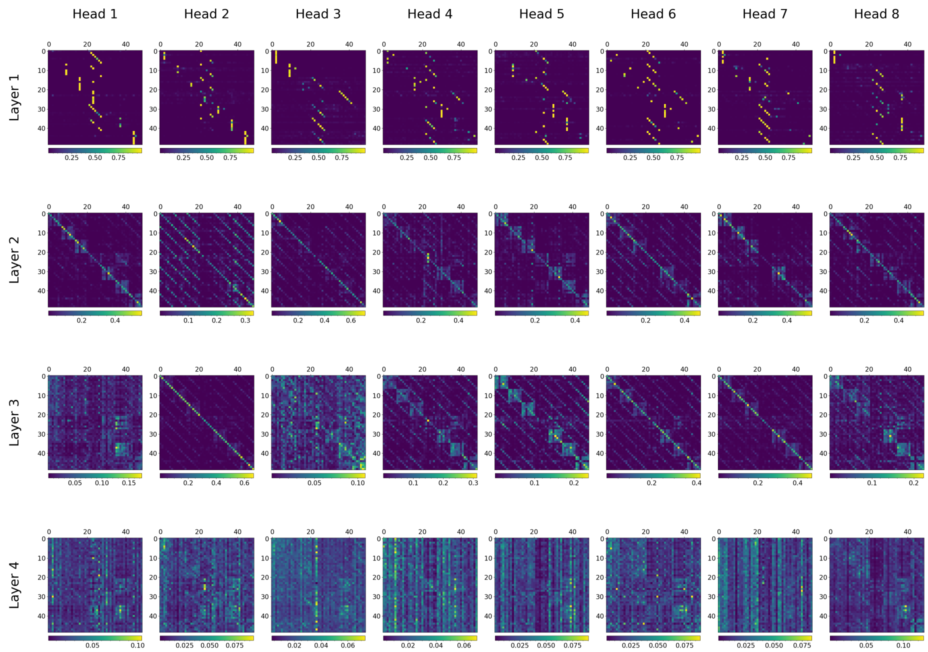
Appendix K. Attention Heads variation along training



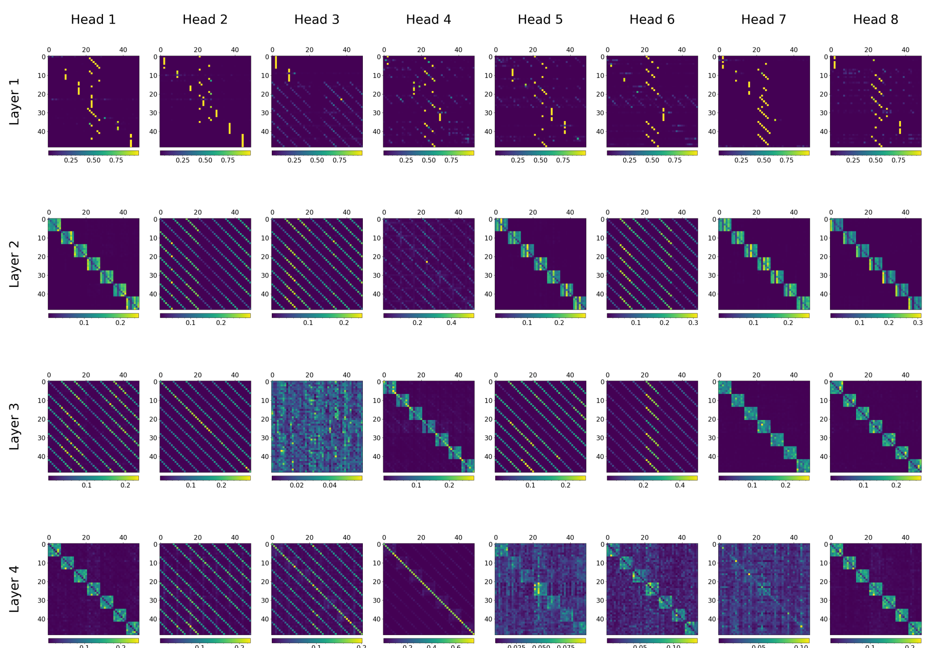
(*a*) Step 4000



(*b*) Step 14000



(*c*) Step 16000



(*d*) Step 20000

Figure 13: Attention heads across various training steps.

Appendix L. Related Work

Mathematical problem solving capabilities of Transformers have been a topic of interest lately [4, 6, 20]. In fact, [20] show that learning addition from samples is equivalent to low–rank matrix completion. Further, [6] show that it is possible to train a transformer based model to solve various linear algebraic tasks e.g. eigendecomposition, matrix inversion, etc.; however, to the best of our knowledge, interpretability studies for such tasks have not been conducted before. For interpretability in simpler math tasks, [15] mechanistically analyse GPT-2 small on predicting whether a number is 'greater-than' a given number, by formulating the problem as a natural language task. [9, 30, 32] analyse BERT from an interpretability perspective. More recently, there has been a line of research works analysing decoder based models to reverse–engineer the mechanisms employed by these models, termed as 'mechanistic interpretability' [10, 12, 16, 19, 21, 22, 26–29, 34]. We note that our setting is distinct from the recent work on solving mathematical tasks like linear regression through 'in–context' learning in transformers [1, 2, 4, 8, 13, 14, 25, 33]. Whether our model learns to implicitly 'implement' an optimization procedure as shown in some of these works is an open question.

Further, [17, 23, 24, 27, 36] analyse 'grokking', the sudden emergence of generalization during model training. In the context of training dynamics of MLM, [7] analyses 'breakthroughs' (sudden drop in loss and associated improvement in generalization capabilities of the model), specifically for BERT. They show that the breakthrough marks the transition of the model to a generalizing one. Their work however is focused on language tasks, distinct from our setting which is more mathematical in nature. We also note that their work is not in the online training setting; our setup is online in the sense of sampling new data at every step of training.