EUREKA-MOMENTS IN TRANSFORMERS: MULTI-STEP TASKS REVEAL SOFTMAX INDUCED OPTIMIZATION PROBLEMS

David T. Hoffmann[1,2], **Simon Schrodi**[2], **Jelena Bratulić**[2], **Nadine Behrmann**[1]
Volker Fischer[1] **& Thomas Brox**[2]
[1]Bosch Center for Artificial Intelligence, [2]University of Freiburg
hoffmann@cs.uni-freiburg.de

ABSTRACT

In this work, we study rapid improvements of the training loss in transformers when being confronted with multi-step decision tasks. We found that transformers struggle to learn the intermediate task and both, training and validation loss saturate for hundreds of epochs. When transformers finally learn the intermediate task, they do this rapidly and unexpectedly. We call these abrupt improvements *Eureka-moments*, since the transformer appears to suddenly learn a previously incomprehensible concept. We designed synthetic tasks to study the problem in detail, but the leaps in performance can be observed also for language modeling and in-context learning (ICL). We suspect that these abrupt transitions are caused by the multi-step nature of these tasks. Indeed, we find connections and show that ways to improve on multi-step tasks can be used to improve the training of language modeling and ICL. Using the synthetic data we trace the problem back to the Softmax function in the self-attention block of transformers and show ways to alleviate the problem. These fixes reduce the required number of training steps, lead to higher likelihood to learn the intermediate task and to higher final accuracy.

1 INTRODUCTION

A key quality of any intelligent system and foundation models is the ability to deal with complex problems that may consist of multiple sub-problems. They should learn to solve these sub-problems even in the absence of direct feedback. Deep learning has enabled such capabilities to a certain degree. Often, foundation models learn features to solve intermediate tasks to simplify their objective. Language models group tokens to derive their contextual meaning and then predict a new token. In-context learning (ICL) tasks require to first learn similarities and then associate outputs based on positional information by learning induction heads (Olsson et al., 2022). While these examples are promising, designing the training process to learn sub-tasks is often difficult. For instance, reward shaping for reinforcement learning and explicit or implicit intermediate supervision in computer vision, while for language modelling and ICL a good data arrangement plays an important role (Chan et al., 2022). More severely, for foundation models the exact sub-tasks are often unknown, s.t. it commonly remains hidden whether it learned to solve the sub-tasks.

To study implicit multi-step learning, we could study common tasks that might entail multi-step tasks, like BERT (Kenton & Toutanova, 2019) and ICL (Chan et al., 2022). For BERT the network might first learn word frequencies and only later learns to use the context to predict masked tokens. For ICL first similarities of keys might be learned to retrieve the correct label from the context in a second step (i.e. induction heads). For both tasks we observe sudden improvements, similar to those that we study in this work (see Fig. 1c). But real data prohibits a clean study: 1) The sub-tasks and their number are unknown. Hence, they are hard to study and we cannot determine if models learn all tasks. 2) Easy samples do not require multi-step reasoning, overlaying the progress on the multi-step task. 3) The features to solve for the tasks are unknown, i.e. we cannot study what the network fails to learn.

As a remedy, we propose to use synthetic data. Creating two-step tasks in a controlled setting solves these issues all at once. In our dataset, the answer to the first task $p(z|x)$, which is not explicitly

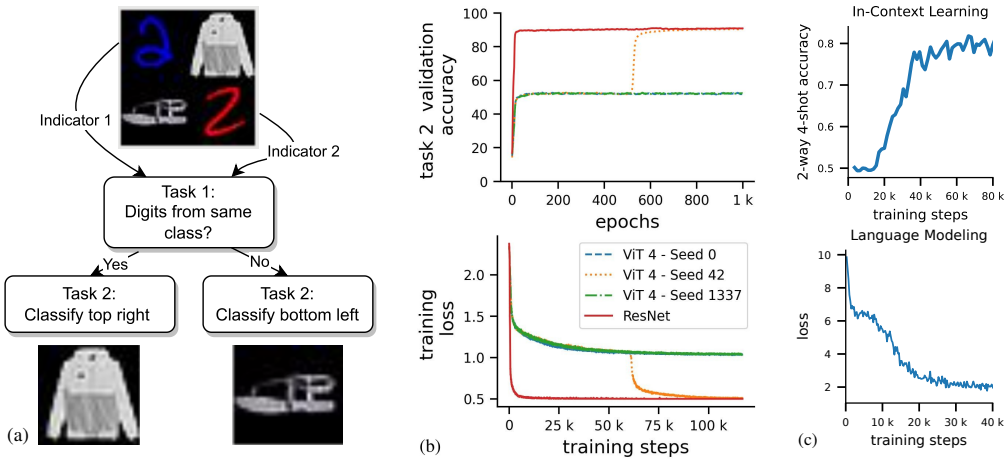


Figure 1: **Transformers can get stuck during optimization for two-stage tasks. (a)** Describes our 2 step decision task used to study Eureka-moments. **Task 1** is to compare the two **indicators** (here digits). If the digits are the same, **task 2** is to classify the top-right image and bottom left else. Thus, task 1 indicates where to look for task 2. Top-right and bottom left are referred to as **targets**. The location of the correct target is referred to as **target location**. **(b) Validation accuracy** and **training loss** for the task in (a). 2 ViTs (blue and green) fail to converge, while one ViT (yellow) has an Eureka-moment. Eureka-moments are characterised by a sudden increase of accuracy and drop of the loss. ResNets are not susceptible to this kind of optimization difficulty. **(c) Eureka-moments on real datasets.** Sharp drops of the training loss after initial plateauing can also be observed for GPT-2 ICL, for the Omniglot task (Chan et al., 2022) and language modeling with RoBERTa on Wikipedia.

modeled in the loss function, must be found by the model in order to correctly solve the second task $p(y|x, z)$. A task description is provided in Fig. 1a. Such multi-step tasks can be described as $p(y|x, z) \cdot p(z|x)$, i.e. the probability of class y given evidence x and the latent variable z.

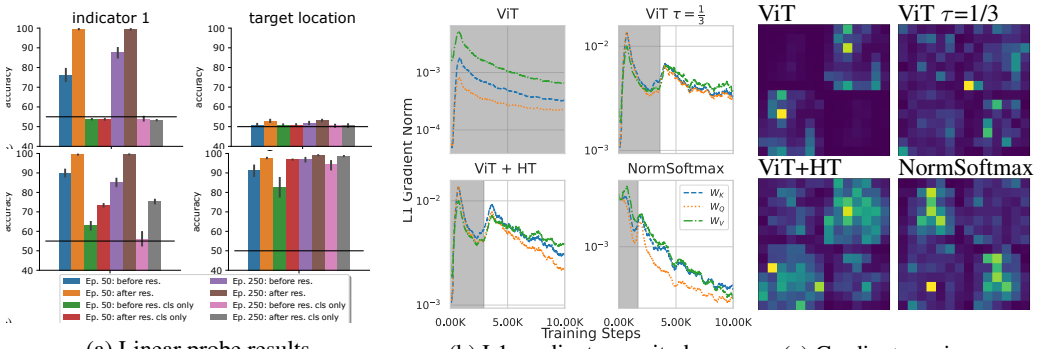
Transformers struggle to learn such two-step tasks (Fig. 1b). They learn only the final classification, followed by a long saturation phase, which can be followed by abrupt improvement, reffered to as Eureka-moment. Here we study this strange phenomenon. In summary, the contributions of this analysis paper are: 1) We study implicit multi-step learning 2) We discover a new failure mode of transformers. 3) We analyse the mechanisms of this failure mode and find that the Softmax function causes (local) small gradients for the key and value weight matrices, thereby hindering learning. 4) To validate this we mitigate the failure mode through targeted interventions. They lead to significantly faster convergence, higher accuracy, higher robustness to suboptimal hyper-parameters, and higher probability of model convergence, affirming our analysis. 5) We find related learning behavior in ICL and language modeling and show that our solution leads to better training behavior, i.e. higher ICL accuracy and earlier Eureka-moments on text completion training (see Fig. 3)

2 Task description & experimental conditions

Recent works found sudden abrupt learning, e.g. sudden improvements of RoBERTa (Liu et al., 2019), rapid emergence of induction heads (Olsson et al., 2022), or the "sudden convergence phenomenon" of control net (Zhang et al., 2023). We hypothesize that all of these constitute multi-step mechanisms $p(y|x, z) \cdot p(z|x)$, where $p(y|x, z)$ (task 2) is dependent on the result of $p(z|x)$ (task 1). But why does it take so long for transformers to learn such mechanisms? We suspect that these training problems are due to the multi-step nature of these tasks. To test this we study multi-step mechanisms on simple two-step tasks. Fig. 1a and it's caption explains our synthetic task. More details are provided in A.3.

Metrics. The **Eureka-ratio** is the ratio of trainings with Eureka-moment across different random seeds. The mean epoch at which the Eureka-moment happened is called **average Eureka-epoch**. Accuracy and Eureka-epoch are averaged only over runs with an Eureka-moment. Thus, they must be jointly considered with the Eureka-ratio, as they ignore failure runs. Details on the training in A.4.

Interventions. To test whether softmax attention leads to small gradients for multi-step tasks we introduce three interventions that change the behavior of the Softmax. **Using lower τ. Heat treatment (HT)**, which starts training with a low τ followed by a schedule to heat it up to the default value of $\sqrt{d_k}$. NormSoftmax Jiang et al. (2022) uses an input adaptive τ. For details see A.1 and A.2.



(a) Linear probe results. (b) L1 gradient magnitude. (c) Gradients on image.

Figure 2: **(a)** Shows linear probe accuracy averaged over heads. Indicator 1 is the top MNIST digit. Differences between ViT and ViT+HT visible for CLS token and target location task. Res. denotes residual layer. Error bars show variance over heads. Results for layer 6 using Z_i. Black line is chance. More information in Fig. 15. **(b)** ViTs, W_K and W_Q receive much smaller gradients than W_V than for others before Eureka-moment (gray regions). **(c)** Gradients on image for W_k (shown ep. 50). For ViT only little gradient for W_K comes from indicators. Details and more plots in Sec. A.6

3 Understanding Eureka-moments & the optimization problems

To investigate why ViT's training fails, we analyze the representations using progress measures. Note that solving task 1 requires ViT to **1)** learn to distinguish the indicators, **2)** carry the information through the layers and **3)** compare the indicator information to obtain the target location. We use linear probes on the output of the attention heads, i.e. $Z_i = \text{Attention}(Q_i, K_i, V_i)$, for all heads i.

Can the transformer distinguish the indicators? The two left bar plots in Fig. 2a show that the indicators can be linearly separated across all layers (Fig. 15). Still, ViT fails to learn task 1 (orange, brown). Thus, the transformer represents the information to distinguish task 1 but does not utilize it.

Does the transformer filter out information required for task 1? Task 1 is not directly supervised, so the ViT might just ignore the indicators. To test this, we probe the representation at multiple locations. The indicator information is available in all layers. Early in training and in deeper layers (Fig. 2a), some indicator information is filtered out in the attention block, but is always recovered by the residual connections (see Fig. 15). Thus, features relevant for task 1 are not filtered out.

Does the transformer fail to combine the information? The target location (solution of task 1) cannot be inferred by the linear probe (Fig. 2a). Therefore, even though the (indicator) information is available, it is not able to use this information to predict the target location of task 2. Thus, the transformer has all the information but fails to combine it to solve the multi-step task.

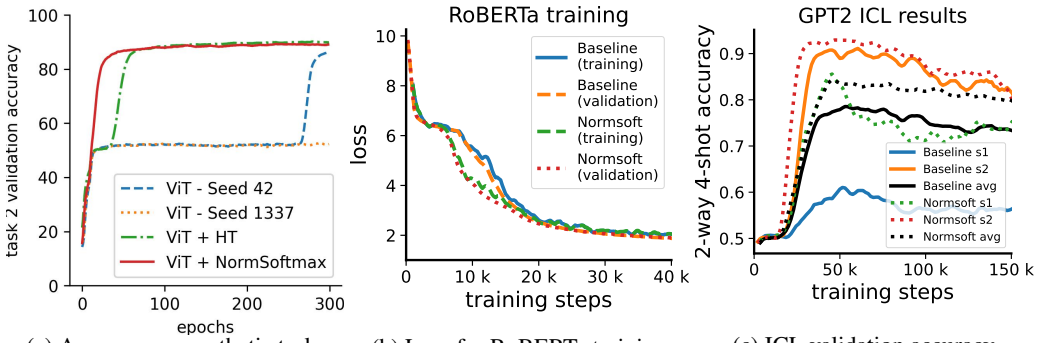
Differences to a transformer that had an Eureka-moment. For a transformer that had an Eureka-moment the target location can be inferred by linear probes (Fig. 2a). Further, indicator information is represented in the CLS token. This is even stronger for early layers (Fig. 15).

Why does the transformer fail to combine the information? The ViT without Eureka-moment does not attend the digits and attends only the targets of task 2 (Fig. 4), whereas the ViT with Eureka-moment attends the indicator digits in early layers. Thus, ViT without Eureka-moment does not pay enough attention to the indicators to match them and fails to learn to attend different regions.

Why does the transformer fail to learn to attend to the indicators? High attention to some pairs with low attention to all others or uniformly distributed attention can result in vanishing gradients for W_K and W_Q, (Noci et al., 2022) (see A.2 for details). To test this, we visualize the L1-norm of the gradient of the first layer in Fig. 2b. For vanilla ViT the gradients for W_K and W_Q are 0.5-1 orders of magnitude smaller than those for W_V. Thus, only small gradients are backpropagated through the Softmax to W_Q and W_K, and the attention map improves only slowly, which results in the observed slow learning. All interventions solve this issue. Fig. 2c shows the origin of gradients on the image for W_k. Thus, only a small part of the gradient in Fig. 2b helps in attending the indicators more.

Is too small or too large attention-entropy the problem? We visualize the distribution of attention maps over training in Fig. 5: the vast majority of attention scores is very small. This indicates that a too uniform attention is causing small gradients. Larger attentions are rare. Thus, we conclude that **local uniform attention** causes the transformer's learning problems.

Ablations and alternative explanations. We provide a set of ablations and test alternative explanations in A.8. In summary, we do not find any alternative explanations.



(a) Accuracy on synthetic task. (b) Loss for RoBERTa training (c) ICL validation accuracy

Figure 3: **(a)** Interventions help. **(b) + (c)** Sharp drops after initial plateauing can be observed also for RoBERTa pretraining and ICL. Using NormSoftmax leads to an earlier (RoBERTa) and higher (ICL) Eureka-moment. Averages for ICL over 4 seeds, 2 exemplary seeds shown per method.

Can a lower temperature resolves the small gradient and thus mitigate the optimization issue? The previous subsection indicates that a local uniform attention is causing the learning problems of the transformer. To test this hypothesis and show that this causes the observed problems we use targeted interventions modifying the Softmax temperature. HT or using NormSoftmax increases high attention scores (c.f. Fig. 5), the transformer also learns to attend the indicators (Fig. 4b). Further, lower temperature τ, HT and NormSoftmax solve the imbalanced gradient issue (Fig. 2b) and lead to higher gradients in indicator regions (Fig. 2c). Eureka-moments happen much earlier or instantly (see Fig. 3a). Thus, the interventions indeed mitigate the optimization issues. A quantitative evaluation is provided in Tab. 2. Decreasing the temperature or using NormSoftmax increases the Eureka-ratio, accuracy and decreases the Eureka-epoch (i.e. improving the energy landscape). In contrast, increasing the temperature has a negative effect on the Eureka-ratio, showing that the local uniform attention is the main cause for the learning problem.

Table 1: **Quantitative results** on the **main dataset**, as described in Fig. 1a. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

Model	τ	ER ↑	Avg. over EMs Acc. ↑	Avg. EE.
ViT	$\frac{1}{0.025}$	3/10	89.40	174.67
ViT	$\frac{1}{0.075}$	6/10	90.13	181.34
ViT + WD 0.5	$\sqrt{d_k}$	5/10	90.09	177.8
ViT	$\sqrt{d_k}$	7/10	89.48	207.43
ViT + Warmup 20	$\sqrt{d_k}$	8/10	87.65	205.87
W_{QKV} grad scaling	$\sqrt{d_k}$	**10/10**	87.96	119.4
NormSoftmax	$\sqrt{d_k}$	**10/10**	89.56	28.2
NormSoftmax	$\frac{1}{3}$	**10/10**	89.18	23.5
ViT	$\frac{1}{3}$	**10/10**	89.35	66.6
ViT+HT	$\frac{1}{3} \to \sqrt{d_k}$	**10/10**	89.81	74.0
NormSoftmax + HT	$\frac{1}{3} \to \sqrt{d_k}$	**10/10**	**89.83**	17.5

Does the NormSoftmax intervention translate to real data? To validate the results of our analysis in a real setting we train RoBERTa (Liu et al., 2019) with NormSoftmax for language modeling on Wikipedia, where we suspect task 1 to be learning of general word probabilities and task 2 to be modulating these probabilities based on context. Additionally, we train GPT-2 on the Omniglot (Radford et al., 2019) dataset and test its ICL abilities following Chan et al. (2022). Experimental details for RoBERTa and ICL are provided in A.19 and A.20, respectively. Fig. 3 shows that NormSoftmax indeed leads to earlier Eureka-moments for RoBERTa. Furthermore, NormSoftmax leads to higher ICL accuracy and seems to prevent failure cases like no or a small Eureka-moment (Baseline s1 in Fig. 3). Thus, our analysis and results are indeed transferable to real datasets.

4 Limitations and Conclusion

Limitations. Implicit multi-step task learning is a common problem, but it is difficult to study on real datasets. As a result many works follow a trial-and-error approach. In contrast, we try to gain deeper understanding by studying this problem in a controlled, synthetic setting. This comes with the assumption that our analysis transfers to real data, for which we find indications.

Conclusion. In this work, we identified that transformers have difficulties to decompose a task into sub-problems and learn to solve the intermediate sub-tasks. We observed that transformers can learn these tasks suddenly and unexpectedly but usually take a long time to do so. We called these sudden leaps Eureka-moments. We pined the problem down to the Softmax in the attention that leads to small gradients. We proposed simple interventions that specifically target the Softmax and show that they improve the transformers' capabilities to learn sub-tasks and to learn them faster. We identify NormSoftmax as most robust and convenient method, leading to consistently better results. Last, we showed that our observations transfer to real datasets.

REFERENCES

Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34:20014–20027, 2021.

Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*, 2018.

Boaz Barak, Benjamin Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: Sgd learns parities near the computational limit. *Advances in Neural Information Processing Systems*, 2022.

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems*, 35:18878–18891, 2022.

Xiangyu Chen, Qinghao Hu, Kaidong Li, Cuncong Zhong, and Guanghui Wang. Accumulated trivial attention matters in vision transformers on small datasets. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3984–3992, 2023.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. *arXiv preprint arXiv:2302.05442*, 2023.

Ameet Deshpande and Karthik Narasimhan. Guiding attention for self-supervised learning with transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4676–4686, 2020.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning*, pp. 2793–2803. PMLR, 2021.

Philipp Dufter, Martin Schmitt, and Hinrich Schütze. Increasing learning efficiency of self-attention networks through direct position interactions, learnable temperature, and convoluted attention. 2020.

Justin Gilmer, Andrea Schioppa, and Jeremy Cohen. Intriguing Properties of Transformer Training Instabilities, 2023. To appear.

Ankit Gupta and Jonathan Berant. Gmat: Global memory augmentation for transformers. *arXiv preprint arXiv:2006.03274*, 2020.

Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers. *arXiv preprint arXiv:2104.05704*, 2021.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving transformer optimization through better initialization. In *International Conference on Machine Learning*, pp. 4475–4483. PMLR, 2020.

Zixuan Jiang, Jiaqi Gu, and David Z Pan. Normsoftmax: Normalize the input of softmax to accelerate and stabilize training. 2022.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, pp. 2, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ziming Liu, Ouail Kitouni, Niklas Nolte, Eric J Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, 2022a.

Ziming Liu, Eric J Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. *arXiv preprint arXiv:2210.01117*, 2022b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Beren Millidge. Grokking 'grokking', 2022. URL `https://www.beren.io/2022-01-11-Grokking-Grokking/`.

Koichi Nagatsuka, Clifford Broni-Bediako, and Masayasu Atsumi. Pre-training a bert with curriculum learning by increasing block-size of input text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pp. 989–996, 2021.

Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *International Conference on Learning Representations*, 2023.

Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *arXiv preprint arXiv:2206.03126*, 2022.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. *arXiv preprint arXiv:2312.03002*, 2023.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*, 2023.

Kai Shen, Junliang Guo, Xu Tan, Siliang Tang, Rui Wang, and Jiang Bian. A study on relu and softmax in transformer. *arXiv preprint arXiv:2302.06461*, 2023.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Jacob Steinhardt. Future ml systems will be qualitatively different. `https://bounded-regret.ghost.io/future-ml-systems-will-be-qualitatively-different/`, 2022. Accessed: 2024-02-01.

Vimal Thilak, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua M. Susskind. The slingshot mechanism: An empirical study of adaptive optimizers and the \emph{Grokking Phenomenon}. In *Has it Trained Yet? NeurIPS 2022 Workshop*, 2022.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 776–794. Springer, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, volume 139, pp. 10347–10357, July 2021.

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.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

Shulun Wang, Feng Liu, and Bin Liu. Escaping the gradient vanishing: Periodic alternatives of softmax in attention mechanism. *IEEE Access*, 9:168749–168759, 2021.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL `http://arxiv.org/abs/1708.07747`.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Josh Susskind. Stabilizing transformer training by preventing attention entropy collapse. *arXiv preprint arXiv:2303.06296*, 2023.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

A APPENDIX

Here, we provide additional information that supports understanding and helps interpreting the main paper. First, we discuss related work in A.1 and provide some background information in A.2. We provide supplemental experimental results and more detailed analysis. We show the gradient norm for indicators and targets individually, which reveals that most of the already small gradients for W_Q and W_K is attributed to target features for models that do not learn the task and very little to indicator features. We provide the linear probe plots using also Q, K and V representations and the full version of Fig. 2b. Next, we provide an ablation on model scale and the main results from the main paper with standard deviation and training speed up. We explain additional datasets and report results on them. We providing a more complete version of Fig. 2b. Last, we provide details on training and experimental setups and an explanation for vanishing gradients in case of centralized attention maps.

A.1 RELATED WORKS

Emergence and phase transitions. Following Steinhardt (2022) & Wei et al. (2022), *emergence* refers to a qualitative change in a learning system resulting from an increase of model size, training data or training steps, where *phase transitions* are additionally characterized by a sharp change. Eureka-moments are special types of phase transitions. While recent work showed that sharp emergence only may be an artifact of the choice of a discontinuous metric (Schaeffer et al., 2023; Srivastava et al., 2022), we observe rapid changes also for continuous metrics. A connection between phase transitions or emergence and our work may exist, and both may be related to escaping bad energy landscapes as in our work.

Unexplained phase transitions. Previous works reported observations that may be Eureka-moments, without investigating their cause. For instance, rapid improvements happen for in-context-learning (ICL) (Olsson et al., 2022), diffusion model training Zhang et al. (2023) and BERT training (Gupta & Berant, 2020; Deshpande & Narasimhan, 2020; Nagatsuka et al., 2021). Deshpande & Narasimhan (2020) proposed to bias the attention mechanism towards predefined attention patterns and observed speed-ups in BERT training. We also identify the learning of the task-required attention pattern to be the cause of the problem. In concurrent work Reddy (2023) studied phase transitions in ICL. Specifically, they show for a small toy-model that slow initial learning is due to a saddle-point, where one path leads to a sub-optimal minimum (i.e. random context label) and the other path to the ICL solution. They discover a "cliff" in the loss landscape and shallow gradients that lead to the ICL solution. While our setting shows substantial differences, our analysis reveals the same underlying mechanisms, i.e. shallow gradients, which we investigate in more detail.

Grokking. A similar phenomenon has been discovered on synthetic data (Power et al., 2022) and was further studied in (Liu et al., 2022b; Nanda et al., 2023; Thilak et al., 2022; Millidge, 2022; Barak et al., 2022; Liu et al., 2022a). Grokking describes the phenomenon of sudden generalization after overfitting, which can be induced by weight decay. In contrast to Eureka-moments, the training accuracy already saturates at close to 100% (overfitting), a long time before the validation accuracy has a sudden leap from chance level to perfect generalization. For Eureka-moments, validation and training loss saturate (no overfitting) and the sudden leap occurs for *both simultaneously*.

Unstable gradients in transformers. The position of the layer-norm (LN) (Xiong et al., 2020) and instabilities in the Adam optimizer in combination with LN induced vanishing gradients (Huang et al., 2020). Removing the LN (Baevski & Auli, 2018; Child et al., 2019; Wang et al., 2019) or Warmup (Baevski & Auli, 2018; Child et al., 2019; Wang et al., 2019; Huang et al., 2020) resolves this problem, but in our case, Warmup alone does not help. Others identified the Softmax as one of the problems, showing that both extremes, attention entropy collapse, i.e. too centralized attention (Zhai et al., 2023; Shen et al., 2023) and a large number of small attention scores, i.e. close to maximum entropy (Dong et al., 2021; Chen et al., 2023) can lead to small gradients (Noci et al., 2022). As a remedy to vanishing gradients caused by entropy collapse Wang et al. (2021) proposed to replace the Softmax by periodic functions. However, before Eureka-moments, the attention distribution is in neither extreme. Instead the attention is allocated to the wrong tokens.

Temperature in Softmax. A key operation in the transformer is the scaled dot-product attention. Large products can lead to attention entropy collapse (Zhai et al., 2023; Shen et al., 2023), which result in very small gradients. In contrast, Chen et al. (2023) observed close to uniform attention

over tokens. They scaled down very low scores further while amplifying larger scores, but this only amplified the problem when important tokens are already ignored. Instead, Jiang et al. (2022) proposed to normalize the dot product. Their proposed *NormSoftmax* avoids low variance attention weights and thus avoids the small gradient problem. We found it as the most effective intervention on the Softmax function. Others proposed to learn the temperature parameter (Dufter et al., 2020; Ali et al., 2021), but this is difficult to optimize. For very large models the problem becomes more severe. Models with more than 8B parameters show attention entropy collapse (Dehghani et al., 2023). They followed Gilmer et al. (2023) and normalized the QK^T with layer norm before the Softmax.

A.2 Background

Preliminaries. This work investigates the popular dot product attention (Vaswani et al., 2017), defined as

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\tau}\right)V, \tag{1}$$

where the weight matrices W_Q, W_K and W_V map the input X to query Q, key K, value V, and the temperature parameter τ controls the entropy of the output. A low temperature leads to low entropy, i.e. a more "peaky" distribution. Commonly, τ is set to $\sqrt{d_k}$, where d_k is the dimensionality of Q and K. Thus, $\sqrt{d_k}$ is the standard deviation of QK^T under the independence assumption of rows of Q and K with 0 mean and variance of 1 (Vaswani et al., 2017).

Softmax attention can cause vanishing gradients. Attention entropy collapse, i.e. too centralized attention, can cause vanishing gradients (Zhai et al., 2023; Shen et al., 2023), since all entries of the Jacobian of the Softmax will become almost 0 (see . A.22). Similarly, uniform attention can cause vanishing gradients for W_K and W_Q (Noci et al., 2022).

A remedy to both problems is to control the attention temperature τ. A larger τ in the Softmax will dampen differences of QK^T and by that prevent vanishing gradients by low attention entropy. In contrast, a smaller τ will amplify differences of QK^T and prevents vanishing gradients caused by uniform attention. Choosing the right temperature is difficult and can have a strong influence on what the model learns, how fast it converges etc. One of our interventions to test whether the Softmax is indeed the root cause is **Heat Treatment** (HT). We propose to start training with a low temperature and follow a schedule to heat it up to the default value of $\sqrt{d_k}$. This approach has multiple advantages. First, the network gets optimized for "more peaky" attention, but the temperature increases steadily. By that, the network starts with centralized attention but since the next epochs attention will be more uniform than the previous, it does not run into the issue of low attention entropy. Second, the network can focus on most important features early in training and broaden the view over time, attending to other features.

NormSoftmax. An alternative to tame the attention is NormSoftmax (Jiang et al., 2022), which replaces the expected standard deviation $\sqrt{d_k}$ gets by the empirical standard deviation $\sigma(QK^T)$, for each attention block individually. NormSoftmax can be computed by

$$\text{NormSoftmax}(Q, K) = \text{Softmax}\left(\frac{QK^T}{min(\sigma(QK^T), \tau)}\right). \tag{2}$$

If QK^T has low standard deviation differences will be amplified. If $\sigma(QK^T) > \tau$, τ will be used.

A.3 Task description and synthetic dataset details

Task 1 requires the model to indicate where to look at, i.e., top right or bottom left depending on whether the randomly colored MNIST digits (LeCun et al., 2010) in the top left and bottom right match or do not match. Task 2 is a simple classification. Here it is FashionMNIST (Xiao et al., 2017) classification. Note that only task 2 is evaluated and only task 2 gets supervision; akin to, ICL for example, for which supervision is only provided for a the missing token, but not directly for the induction head learning. By design of our datasets, $40 - 55\%$ accuracy can be obtained by only solving task 2 and picking the target at random. The range is due to varying difficulty of task 2. Higher accuracies can only be achieved by learning the multi-step structure.

Vision dataset creation. The visual datasets are based on MNIST (LeCun et al., 2010) and Fashion-MNIST (Xiao et al., 2017). An example and a schematic of the task is shown in Fig. 1a. The samples

are created by sampling 2 random Fashion-MNIST samples and 2 digit samples from the MNIST classes "1" and "2". We apply a random color to the MNIST samples (red or blue). Next, we compose a new image from the 4 images, putting the 2 MNIST samples on top left and bottom right and the Fashion-MNIST samples on the remaining free quadrants. If the 2 MNIST samples are from the same class, the class of the top-right image is the sample label and bottom-left else.

A.4 Details on models and hyper-parameters

Following Hassani et al. (2021), we use a ViT version specifically designed for small datasets. Unless stated otherwise, we train a ViT with 7 layers, 4 heads each with embedding dimension of 64, patch size of 4 and MLP-ratio of 2. Consequently, the default temperature is $\sqrt{d_k} = 8$. The ResNet has a comparable parameter count and consists of 9 layers. For ViT, ViT+HT and NormSoftmax, we tested 5 different temperature parameters in initial experiments. More details on the training setup are provided in A.15. For the reasoning task, we train a transformer on 30% of the entire set of possible inputs (i.e. $11^4 = 14\,641$ input combinations) for 10K epochs over five random seeds. More details on model and training are provided in A.17.

A.5 Attention maps and attention distribution over time.

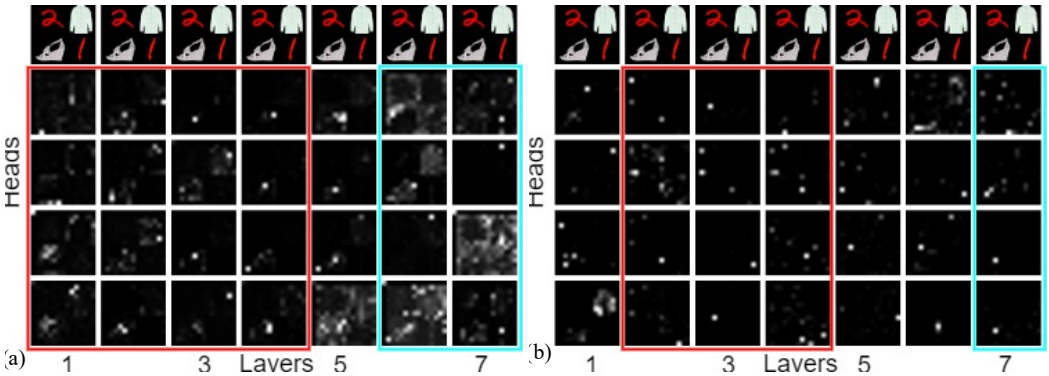


Figure 4: **Attention maps after training for: (a) ViT without Eureka-moment**, i.e. fails to compare the 2 digits. First layers explicitly ignore indicators (digits) (highlighted with red). **(b) ViT+HT with Eureka-moment** attends indicators in first layers (red) and predominantly attends the correct target (ankle boot) in later layers. Black is no and white is high attention. Maps show the average attention of each query, i.e. we average over the key-dimension of the attention map.

A.6 Gradient norm for indicators and targets.

Our particular dataset design allows us to look at the gradients for indicators and targets separately. In particular, we make use of the fact, that indicators and targets are always at the exact same spacial location. More precisely, we use the partial derivatives as proxy for the gradients. We compute $\frac{\partial Z}{\partial Q}$, $\frac{\partial Z}{\partial K}$ and $\frac{\partial Z}{\partial V}$, where Z is the output of the attention function. To analyze the gradient norm for targets and indicators independently we compute $\frac{\partial Z}{\partial Q}$, $\frac{\partial Z}{\partial K}$ and $\frac{\partial Z}{\partial V}$, where Z is the output of the attention function. Since we compute the derivative wrt. the tokens, the spacial dimension remains. By averaging over the batch dimension and heads we can plot the partial derivative for each token. While it's not exactly the same, we will use the term gradient to refer to these partial derivatives in the following.

Since each token corresponds to a region in the image, we can visualize these results as an image. The results are shown in Figs. 2c, 6, 7. It can clearly be seen, that target regions (top-right and bottom-left) receive more gradients than indicator regions for K and V. ViT, ViT+HT and NormSoftmax mitigate this problem, leading to significant gradient for indicator tokens. Indicator regions for Q receive comparatively larger gradients, however, the gradients for Q are much smaller.

Besides that, we compute the mean partial derivative for indicator and target regions of the image i.e. we average the partial derivatives for tokens corresponding to target regions or indicator regions. This allows us to plot the gradient norm for Q, K and V for only target and indicator tokens over the training. We show the results in Fig. 8. We make the following observations:

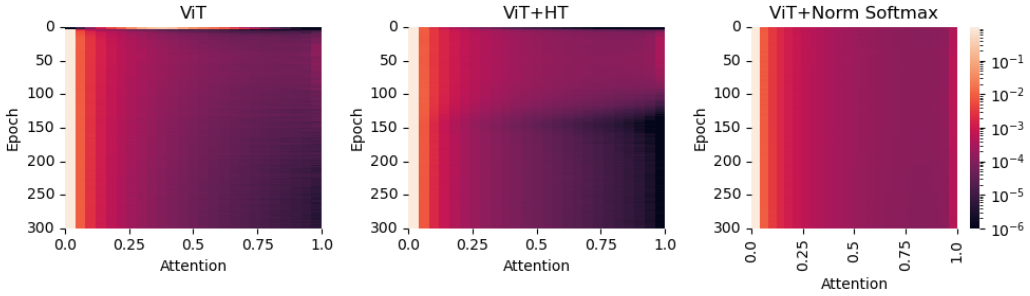


Figure 5: **Attention distribution as a heatmap.** Attention scores are sampled during evaluation after each epoch and binned to 25 bins. The color map is log scaled. For all 3 models, the vast majority of values falls into the first bin. ViT shows very few higher attention scores. ViT+HT and ViT+NormSoftmax lead to a significantly larger number of medium and high attention values. For ViT+HT this is limited to the first 100 epochs.

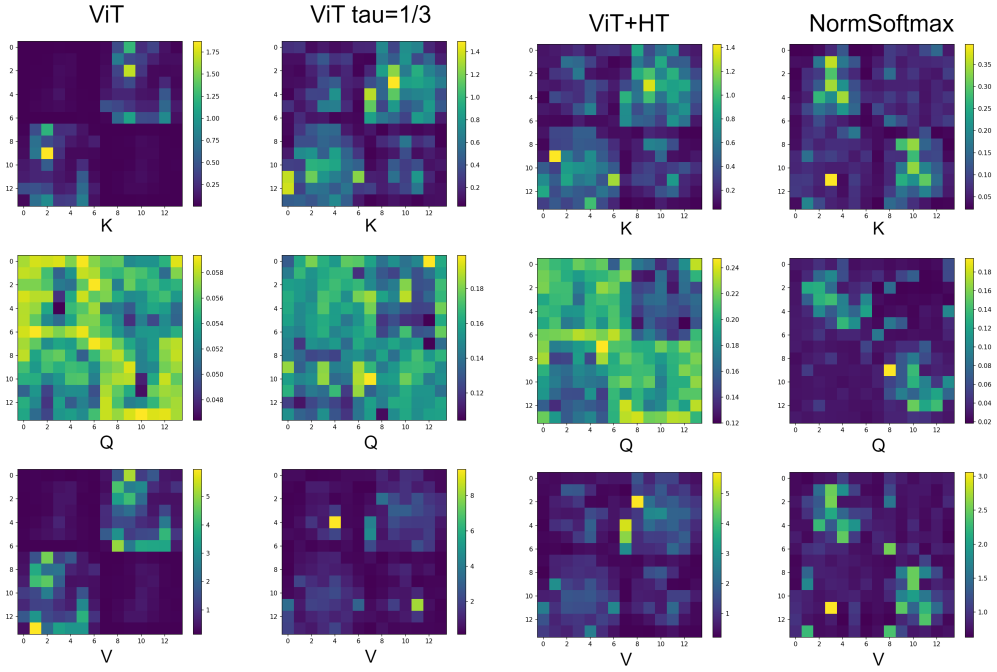


Figure 6: **Gradient norm for different image regions visualized for Q, K, V.** Larger gradients are visible for target regions, i.e, top-right and bottom-left. Indicator receive less gradients for K and V. ViT $\tau = \frac{1}{3}$, ViT+HT and NormSoftmax mitigate this problem well. Indicator regions for Q receive more gradient relative to the target regions, but overall the gradient for Q is very small (see color bar). Plots created at epoch 50.

1. In general, the gradients are not evenly divided between target and indicator K, with usually smaller gradient for the indicator regions. Therefore Fig. 2b even underestimates the difference for the indicator regions, i.e. the regions relevant for an Eureka-moment.
2. This difference can explain why more time is needed to reach an Eureka-moment for ViT in comparison to the other methods.
3. This difference between indicator k and target k can also explain, why the W_{QKV} grad scaling does not solve the problem.
4. For ViT the difference between gradients of "V and K" and "V and Q" is, for most layers, generally much larger compared to the other approaches. This is particularly true before the

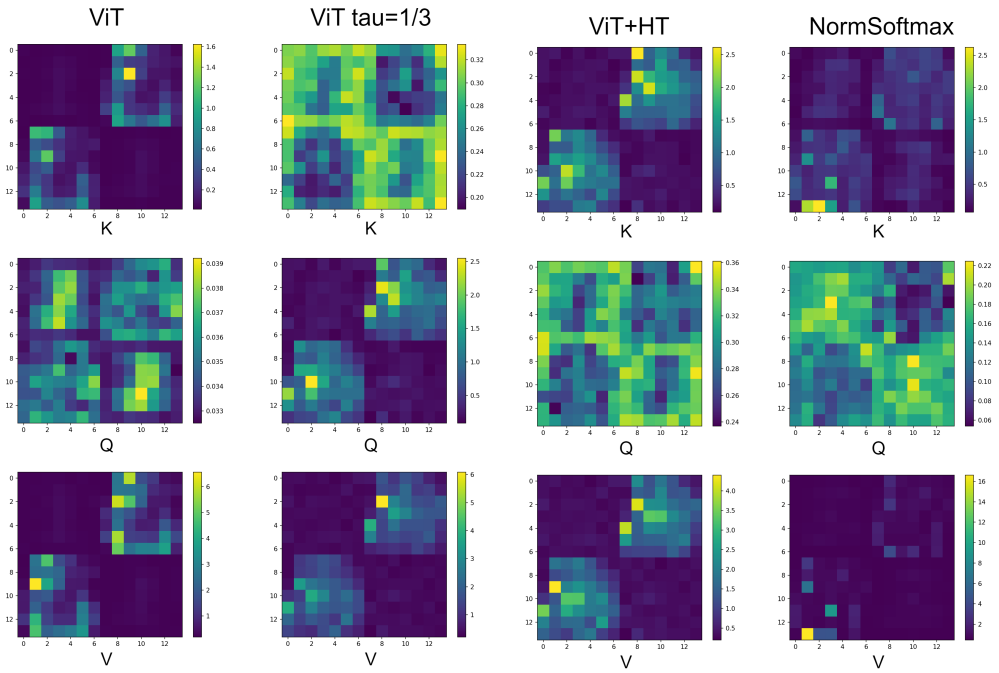


Figure 7: **Gradient norm for different image regions visualized for Q, K, V.** at epoch 13 (Eureka-moment of NormSoftmax).

Eureka-moment, where larger gradients for indicator K and indicator Q are crucial to get an early Eureka-moment. Most prominent is the difference between target V and indicator K, showing a large mismatch. This explains, why the attention maps change so slowly and why simply increasing the learning rate does not solve the problem.

A.7 Gradient Norm

Fig. 9 shows the L1 gradient norm for all layers and all methods. It can be seen, that throughout all layers and the entire training ViT has larger gradients for W_V in comparison to W_K and W_Q. For the other methods differences are much smaller and less consistent.

A.8 Is this an artificial problem caused by other factors?

Does the transformer simply ignore specific indicator locations? The task and dataset used in the previous subsections showed indicators and targets always at the same location, i.e. indicators on top-left and bottom right. Such a dataset design might result in two undesired effects: 1) The transformer might learn to ignore features based on the associated positional embeddings. 2) The task might be easier, since positional embeddings can be used as shortcut to find indicators without the need to rely on the actual features. To disprove both cases we create another dataset, explained in Fig. 10. We observe that removing the fixed position for indicators and targets makes the task even more difficult (Tab. 2 right) and differences between methods are even more apparent. Thus, ViTs without Eureka-moment do not simply learn to ignore regions of the image.

Is this a mere artifact of a bad choice of hyper-parameters? We test various competing explanations that could lead to the observed phenomenon, like training instabilities due to inadequate Warmup, bad choice of learning rate schedules and Weight Decay to force circuit formation as observed for grokking (Nanda et al., 2023). Weight Decay and Warmup do not improve the results. We find that Eureka-moments are sensitive to the learning rate. NormSoftmax and HT reduce the sensitivity to sub-optimal learning rates drastically (see Tab. 3). See A.10 for a more detailed answer to the posed question.

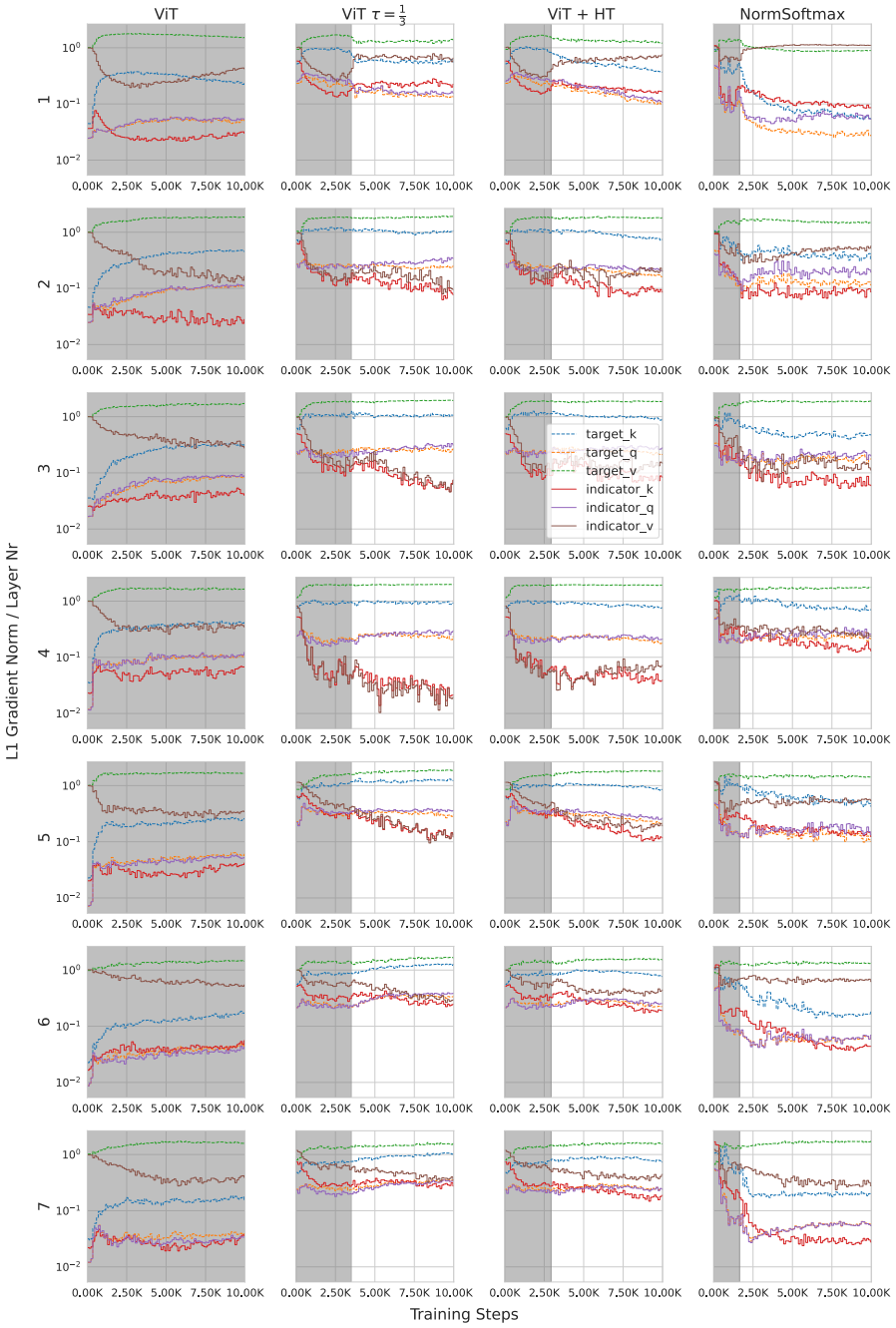


Figure 8: **L1 gradient norm separately for indicator and target tokens.** Indicator regions/features receive much less gradient than target regions. Gray region indicates steps before Eureka-moment.

Influence of model scale, dataset size and image resolution on Eureka-moments. We found no consistent influence of model scale on Eureka-ratio. See A.11 for more details. Further, we can rule out that this observation is an artifact of image resolution or dataset size by showing the phenomenon on a ImageNet-100 based dataset in A.21.

Can the problem be fixed by rescaling of the gradient magnitude for W_V, W_Q and W_K? The observation that lower gradient imbalance leads to higher Eureka-ratio suggests, that simply rescaling of the gradients may solve the problem. We find that this does not work consistently (see Tab. 2)

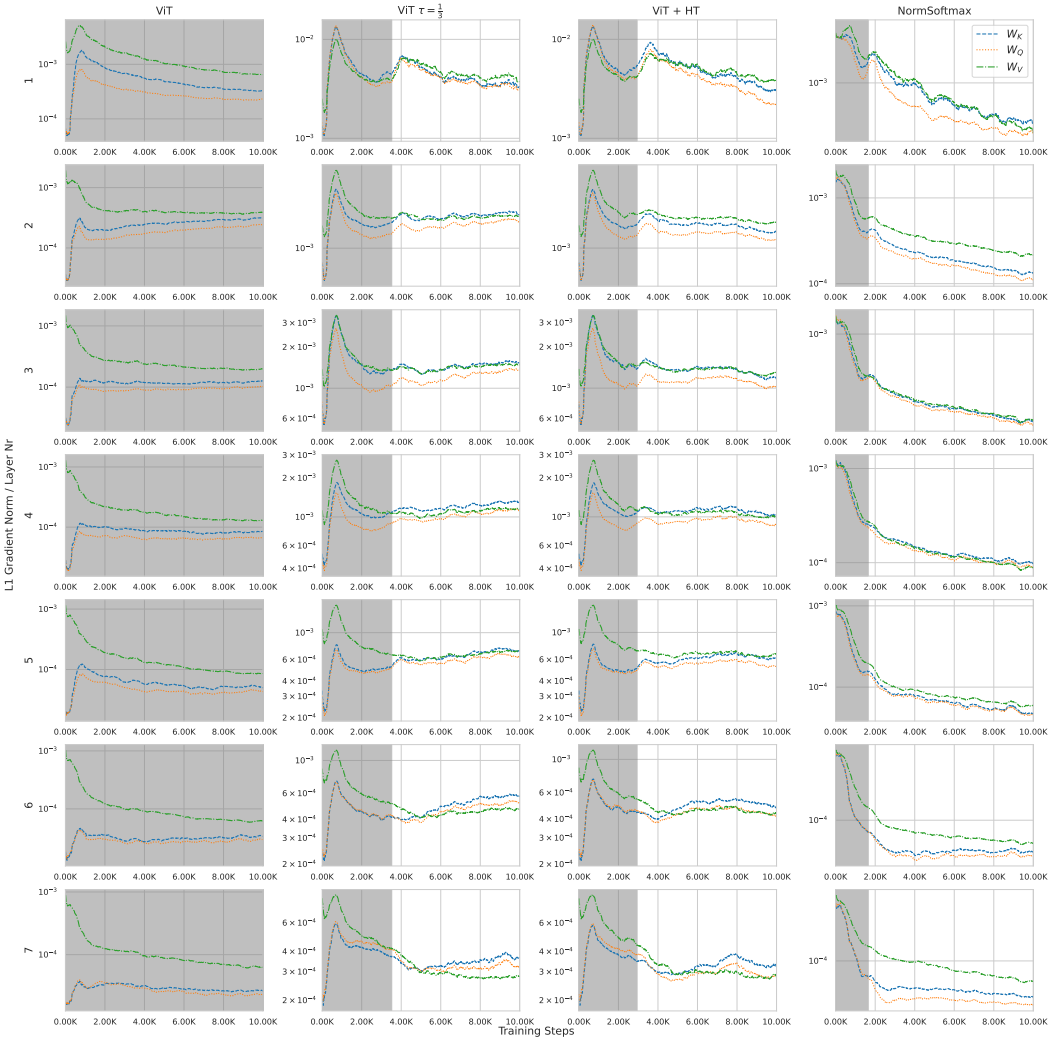


Figure 9: **L1 gradient norm plot for all layers, all models and entire training.** This is the complete version of Fig. 2b. For ViT it can be seen that for all layers the gradient for W_V is significantly larger than for W_Q and W_K. For ViT+HT, ViT with $\tau = \frac{1}{3}$ and NormSoftmax the gradient norm is very similar for the weight matrices (note that the y-axis is not shared). NormSoftmax achives this also for deeper layers. Often after Eureka-moment W_V starts to get larger attention than W_K and W_Q. We conjecture that this is because the attention is already optimized, while task 2 can still improve by modifying the feature representation. Gray region indicates steps before Eureka-moment.

and is very sensitive to the learning rate (see Tab. 3). We attribute this to the differences in gradient magnitudes for indicators and targets and discuss it further in Sec. A.9.

Do gradients vanish completely and can transformers recover? Fig. 1b already suggests, that one potential solution to reliably get Eureka-moments is very long training. This observation is supported by Fig. 2b, which indicates that gradients become small, but not 0. Indeed we observe that training for 3000 epochs results in an Eureka-ratio of 4/4 for all the learning rate schedules. In practice, this is of little help because the number of sub-tasks is unknown and Eureka-moments hard to predict.

Is this truly a transformer optimization problem? To show that Eureka-moments general to all transformers and not just artifacts of vision data, we also show their occurrence in single-layer 4 head transformers on simplistic algorithmic tasks, referred to as the **reasoning task** (Sec. 2). Here, features (numbers) are directly provided as tokens and need not be extracted. Fig. 11 reveals that

Table 2: **Quantitative results** on the **main dataset**, as described in Fig. 1a. and the **No position task** (Fig. 10). τ not optimized for No position task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

Main Dataset				
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\frac{1}{0.025}$	3/10	89.40	174.67
ViT	$\frac{1}{0.075}$	6/10	90.13	181.34
ViT + WD 0.5	$\sqrt{d_k}$	5/10	90.09	177.8
ViT	$\sqrt{d_k}$	7/10	89.48	207.43
ViT + Warmup 20	$\sqrt{d_k}$	8/10	87.65	205.87
W_{QKV} grad scaling	$\sqrt{d_k}$	**10/10**	87.96	119.4
NormSoftmax	$\sqrt{d_k}$	**10/10**	89.56	28.2
NormSoftmax	$\frac{1}{3}$	**10/10**	89.18	23.5
ViT	$\frac{1}{3}$	**10/10**	89.35	66.6
ViT+HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	**10/10**	89.81	74.0
NormSoftmax + HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	**10/10**	**89.83**	17.5
No Position Task				
ViT	$\sqrt{d_k}$	0/4	-	-
ViT + Warmup 20	$\sqrt{d_k}$	1/4	89.55	117
W_{QKV} grad scaling	$\sqrt{d_k}$	0/4	-	-
NormSoftmax	$\sqrt{d_k}$	**3/4**	**88.98**	228
NormSoftmax	$\frac{1}{3}$	1/4	89.77	20
ViT	$\frac{1}{3}$	1/4	89.68	191
ViT+HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	1/4	88.36	242
NormSoftmax + HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	1/4	90.63	**19**

Table 3: **Sensitivity to learning rate**. Lower temperatures and NormSoftmax drastically increase robustness to imperfect learning rate schedules. Eureka-ratio computed over seeds and schedules.

Model	Eureka-ratio ↑
ViT	04/36
VIT + Warmup 20	14/36
W_{QKV} grad scaling	5/36
NormSoftmax	**36/36**
ViT $\tau = \frac{1}{3}$	20/36
ViT+HT $\frac{1}{3} \rightarrow \sqrt{d_k}$	25/36

Eureka-moments appear even in this minimal setting. Both HT and NormSoftmax reduce the training steps required for Eureka-moments to occur and increase the Eureka-ratio from 3/5 to 4/5 or 5/5, respectively.

A.9 Why gradient magnitude scaling does not work.

Following the observation of Fig. 2b, it stands to reason to simply scale the gradients for W_V, W_Q and W_K to the same value. To this end, we compute the gradient norm for W_V, W_Q and W_K for each layer and the overall mean. We scale the gradients for W_V, W_Q and W_K, such that their norm is equal to the mean norm. This removes the imbalanced gradient issue and results in identical effective learning rate.

While this approach might work, it solves only part of the problem. Different features might receive differently large gradients. For instance the indicator features (here digits) receive little gradient, while target (here fashion) receive large gradients, as can be seen in Fig. 6. Simply scaling up the gradients would not solve the imbalance between indicator and target gradients.

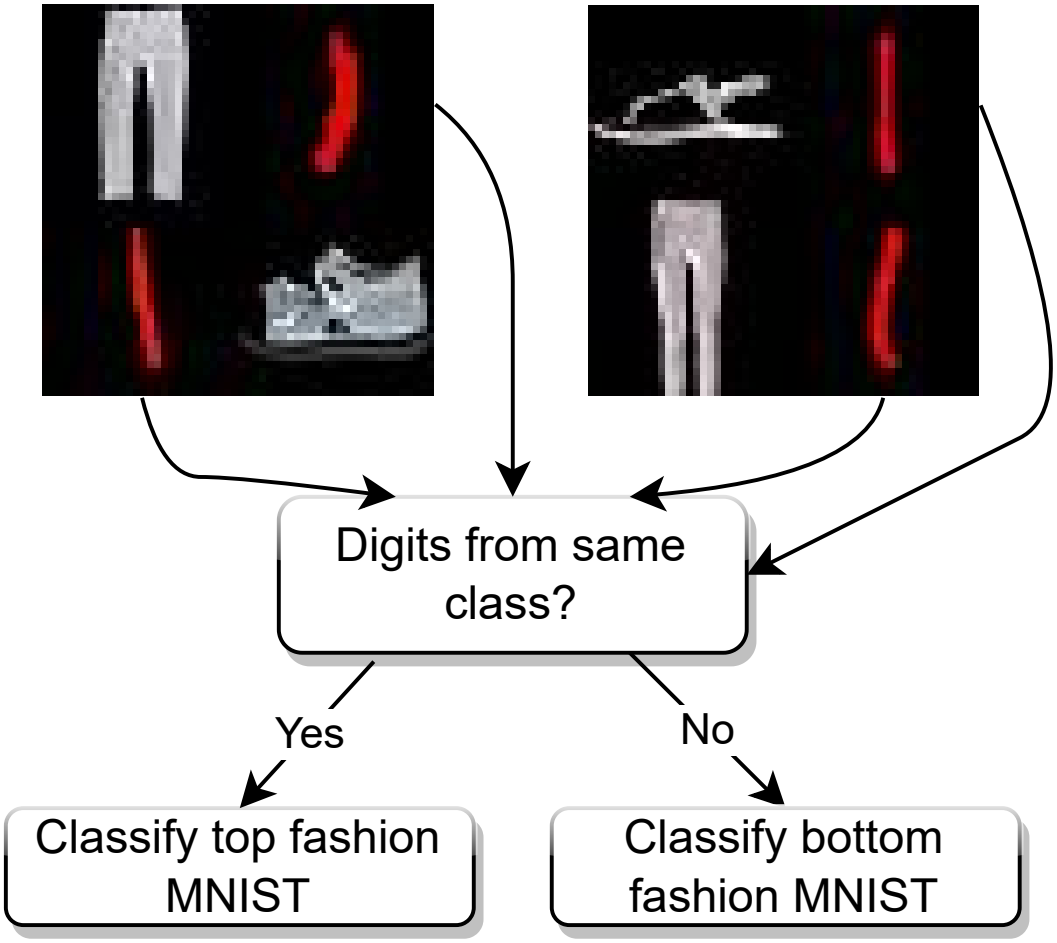


Figure 10: **No position task description.** This task is identical to our main task, but removes more information by swapping indicator (digit) and target (fashion) in each row with a probability of 0.5, i.e. the task is now to classify either the top or bottom fashion sample. Two samples are shown to highlight differences from the task described in Fig. 1a.

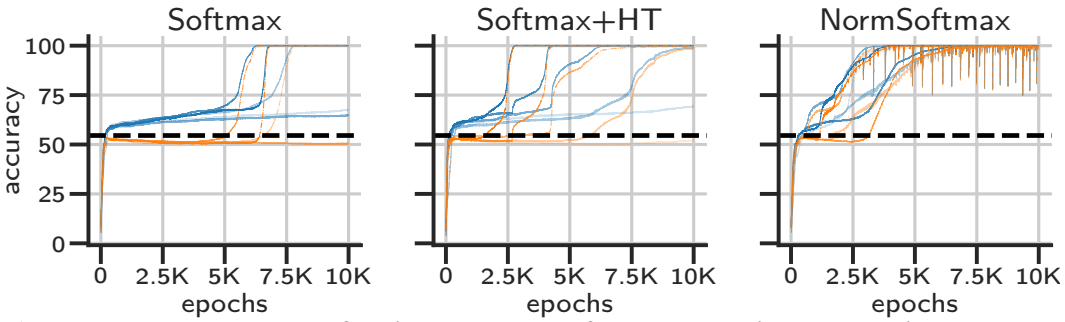


Figure 11: **Eureka-moments for single-layer transformers on a simple reasoning task.** We show the train (blue) and test (orange) accuracies for attention with Softmax, Softmax+HT, or NormSoftmax, over 5 random seeds (transparencies). Chance probability is $6/11 \approx 54\%$ (black).

We can see in Tab. 2, that W_{QKV} grad scaling helps on the main dataset, but is very sensitive to the learning rate (Tab. 3). However, it completely fails on the harder task. The learning rate sensitivity and the failure on the harder task are most likely due to the gradient imbalanced discussed above.

Table 4: **Influence of model scale on Eureka-ratio**. Eureka-ratio is only partially influenced by the architecture. More shallow models and more heads both improve the results. The combination leads to 4/4 Eureka-ratio, but as can be seen in Tab. 6 this architecture fails at other tasks, while our solutions lead to improvements even on the "no position task".

					Avg. over subset with Eureka-moment	
Heads	Emb. Dim.	Depth	MLP	Eureka-ratio ↑	Accuracy ↑	Avg. Eureka-epoch
4	64	7	2	2/4	88.61 ± 1.64	232.50 ± 35.50
4	64	7	4	1/4	90.16 ± 0.00	162.00 ± 0.00
4	64	4	2	3/4	88.77 ± 1.20	216.33 ± 46.64
4	64	10	2	2/4	89.93 ± 0.26	221.00 ± 18.00
4	48	7	2	1/4	90.18 ± 0.00	137.00 ± 0.00
4	96	7	2	2/4	89.76 ± 0.03	140.50 ± 70.5
2	64	7	2	0/4	-	-
6	64	7	2	3/4	89.67 ±0.25	139.67 ± 55.16
6	64	4	2	**4/4**	**89.75 ± 0.24**	152.25 ± 44.49

A.10 Are the learning problems a mere artifact of a bad choice of hyper-parameters?

We always use the default of 5 Warmup epochs to avoid training instabilities during early stages of training. We found that 20 Warmup epochs were most effective in mitigating the problem. However, sensitivity to the learning rate schedule (Tab. 3) is high. The average Eureka-epoch is very late (Tab. 2 left) and found more Warmup epochs lead to worse results on harder tasks (Tab. 2 right). The Eureka-ratio is sensitive to the learning rate schedule. We test 9 learning rate schedules for each method, (Tab. 12). Lower temperatures are less sensitive to the learning rate schedule (see Tab. 3).

Weight Decay (WD) can facilitate grokking (Power et al., 2022) by forcing the network to learn general mechanisms (Power et al., 2022; Nanda et al., 2023). In our setting, we only found mild improvements for higher WD. However, more random seeds revealed, that higher WD rather reduces the Eureka-ratio (Tab. 2) and does not help in solving transformer's learning issue.

A.11 Influence of model scale on Eureka-ratio.

The low Eureka-ratio could also be due to a too large or too small architecture. Also the number of heads might play an important role, since different features can be attended in different heads. More heads might increase the likelihood of one head specializing in indicators. Also the embedding dimension per head might just be too small or large for the task at hand. Maybe, even the hidden dimension of the MLP at the end of the attention block is the bottleneck. Many parameters of the transformer itself could explain why it fails to solve our tasks. We test these hypotheses in Tab. 4. While most changes lead to a lower Eureka-ratio, reducing the depth and increasing the number of heads leads to mild improvements. Combining both leads to an Eureka-ratio of 4/4, but, as can be seen in Tab. 6, this architecture does not generalize to other datasets.

A.12 Main results with standard deviation, ResNet and convergence speed improvements

Due to space and readability constraints we report in the main paper only the mean over all seeds. In Tabs. 5 & 6 we show the same tables including the standard deviation.

Additionally, Tabs. 5 & 6 also provide a comparison to a ResNet9.

Lastly, for Tab. 5 we report the improved convergence speed as a percentage of the number of training steps to reach 95% of ViT accuracy (averaged only over seeds with Eureka-moments), denoted as "% of steps". This value is computed only over the fraction of seeds, that actually lead to a higher accuracy than 95% of ViTs accuracy. In the last column, we also report this fraction. Note, that the Eureka-ratio is the maximum possible value for the "95%-ratio", i.e. for "ViT +Warmup 20" 8/10

seeds have an Eureka-moment. Out of these 8 only 5 reach an accuracy higher than 95% of the ViT accuracy.

Table 5: **Main dataset – Comparison of proposed solutions and baselines.** This is a complete version of Tab. 2 including standard deviation, and speed improvements. For the **main dataset**, as described in Fig. 1a. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch. **% of steps** indicates the % of steps needed to reach 95% of ViTs accuracy. **95%-ratio** indicates the ratio of models that actually reached 95% of ViTs accuracy.

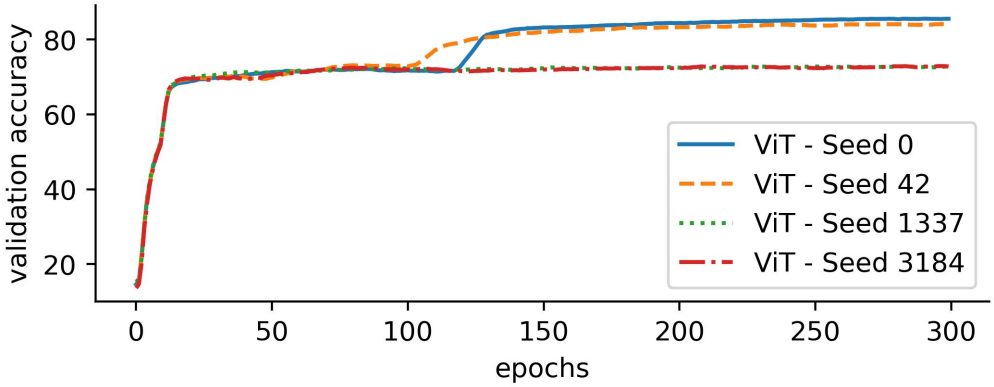
		Main Dataset				
			Avg. over EMs		Avg. over ViT 95% Acc.	
Model	τ	ER ↑	Acc. ↑	Avg. EE.	% of steps	95%-ratio
ResNet		10/10	99.40 ± 0.10	3.00 ± 00.00	-	-
ViT	$\frac{1}{0.025}$	3/10	89.40 ± 0.08	174.67 ± 37.82	84.26	3/10
ViT	$\frac{1}{0.075}$	6/10	90.13 ± 0.40	181.34 ± 24.94	86.79	2/10
ViT + WD 0.5	$\sqrt{d_k}$	5/10	90.09 ± 0.27	177.80 ± 52.30	84.70	5/10
ViT	$\sqrt{d_k}$	7/10	89.48 ± 1.10	207.43 ± 46.65	100.00	4/10
ViT + Warmup 20	$\sqrt{d_k}$	8/10	87.65 ± 6.48	205.87 ± 57.05	91.87	5/10
W_{QKV} grad scaling	$\sqrt{d_k}$	**10/10**	87.96 ± 2.45	119.4 ± 62.70	73.79	5/10
NormSoftmax	$\sqrt{d_k}$	**10/10**	89.56 ± 0.65	28.20 ± 34.85	19.87	10/10
NormSoftmax	$\frac{1}{3}$	**10/10**	89.18 ± 0.36	23.50 ± 08.15	19.25	10/10
ViT	$\frac{1}{3}$	**10/10**	89.35 ± 0.28	66.60 ± 58.55	36.71	10/10
ViT+HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	**10/10**	**89.81** ± 0.29	74.00 ± 61.29	39.88	10/10
NormSoftmax + HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	**10/10**	89.83 ± 0.41	17.50 ± 04.84	16.41	10/10

Table 6: **No Position Task – Comparison of proposed solutions and baselines.** For the **No position task**, as described in Fig. 10. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

		No Position Task		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ResNet		4/4	91.27 ± 0.30	4.25 ± 00.43
ViT	$\sqrt{d_k}$	0/4	-	-
ViT + Warmup 20	$\sqrt{d_k}$	1/4	89.55 ± 0.00	117 ± 00.00
W_{QKV} grad scaling	$\sqrt{d_k}$	0/4	-	-
NormSoftmax	$\sqrt{d_k}$	**3/4**	**88.98 ± 0.55**	**228.67** ± 08.22
NormSoftmax	$\frac{1}{3}$	1/4	89.77 ± 0.00	20.00 ± 00.00
ViT	$\frac{1}{3}$	1/4	89.68 ± 0.00	191.00 ± 00.00
ViT+HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	1/4	88.36 ± 0.00	242.00 ± 00.00
NormSoftmax + HT	$\frac{1}{3} \rightarrow \sqrt{d_k}$	1/4	90.63 ± 0.00	19.00 ± 00.00
ViT 6 heads, depth 4		0/0	-	-

A.13 TRANSFORMERS LEARN THE PRIOR $p(z)$

Given a task like $p(y|x, z) \cdot p(z|x)$, i.e. the probability of class y given evidence x and the latent variable z, we argue, that transformers first learn a prior $p(z)$, ignoring the evidence. Sometimes they fail to unlearn this and pay attention to the evidence. In all previous experiments, the probability of target 1 or target 2 being the target to classify was 0.5. In a setting without 0.5 probability, the transformer should pick the target which is more frequently correct, in case it actually learns the prior $p(z)$. We test this by changing the probability of the top-right target to be the target location to 0.65. As can be seen in Fig. 12, the transformer initially learns the shortcut of always picking the more likely target.



[]

Figure 12: **Validation accuracy curves for "main dataset" with changed target probabilities.** ViT learns that one target is more likely than the other and learn to always pick this target, as can be seen by the higher plateau accuracy.

A.14 Description of and results on more datasets

In the following we report results on 5 more datasets. The datasets are depicted in Fig. 13.

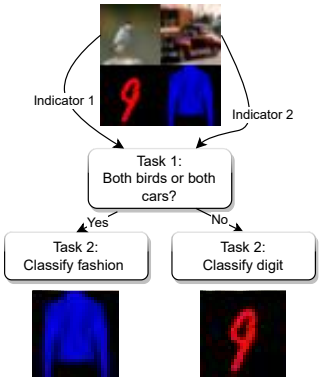
Cifar task 1. A schematic for this task is shown in Fig. 13a. The "Cifar task 1" dataset uses Cifar-10 Krizhevsky et al. (2009) images of classes "automobile" and "bird" as indicators. Targets are sampled from fashion MNIST and MNIST. All 4 images are randomly placed on a 4x4 canvas and we apply random colors (red or blue) to the MNIST and fashion MNIST samples. Task 1 is to compare the Cifar-10 classes. If they come from the same class, task 2 is to classify the fashion MNIST sample. If not, the tasks is to classify the MNIST digit. Results are reported in Tab. 7. normal ViT fails in 1/4 cases and Eureka-epoch is usually late. Note, that this task may seem difficult, but differences in color distribution of "bird" and "automobile" simplify the task.

Table 7: **Results "Cifar task 1" dataset**. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

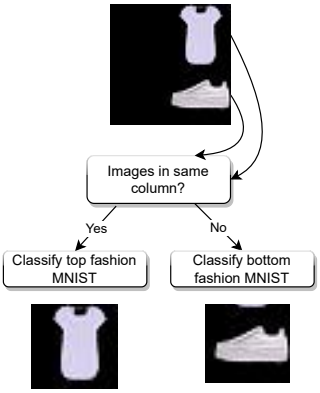
		Cifar task 1		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\sqrt{d_k}$	3/4	83.43 ± 1.83	187.67 ± 42.46
ViT	$\frac{1}{3}$	**4/4**	$\mathbf{86.86 \pm 0.75}$	$\mathbf{76.50 \pm 08.90}$
NormSoftmax	$\sqrt{d_k}$	**4/4**	82.89 ± 0.31	100.0 ± 27.89

Top if above. The task description is summarized in Fig. 13b. For data creation we sample 2 images from fashion MNIST and place one in the top row of a 4x4 canvas and the other in the bottom row. The column is selected randomly for both. Task 1 is to check whether the 2 samples are in the same column. If they are task 2 is to classify the top image. If not, the image in the bottom row must be classified. This task is relatively simple, as it removes additional indicators. Instead, relative location of the images is the relevant information to solve task 1. This task is very simple and leads to a low Eureka-epoch for all methods (see 13b).

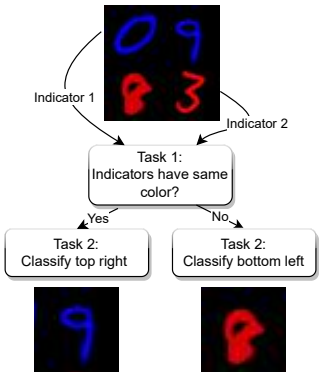
Same color decision task. The task is explained in Fig. 13b. For data creation we sample only MNIST digits and apply random colors (red or blue) to all digits. If color of the indicators is identical, the top right must be classified and bottom left if not. As can be seen in Tab. 9, this task is again very easy. Color seems to be easily accessible for ViT and ViT has little trouble to compare the indicator colors.



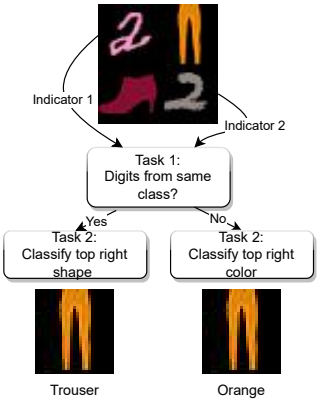
(a) **"Cifar task 1"**



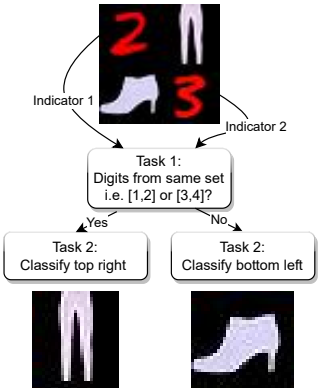
(b) **"Top if above"**



(c) **"Same color decision"**



(d) **"'Color or fashion class"**



(e) **"Digit grouping"**

Figure 13: **Schematics for additional datasets.**

Table 8: **Results "Top if above" dataset**. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

		Top if above		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\sqrt{d_k}$	**4/4**	**91.47** $\pm$ **0.13**	13.75 $\pm$ 2.19
ViT	$\frac{1}{3}$	**4/4**	90.38 $\pm$ 0.13	9.5 $\pm$ 1.25
NormSoftmax	$\sqrt{d_k}$	**4/4**	90.84 $\pm$ 0.27	**9.25** $\pm$ **1.09**

Table 9: **Results "Same color decision task" dataset**. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

		Same color decision task		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\sqrt{d_k}$	**4/4**	98.35 $\pm$ 0.58	91.5 $\pm$ 60.04
ViT	$\frac{1}{3}$	**4/4**	**98.95** $\pm$ **0.10**	8.25 $\pm$ 1.48
NormSoftmax	$\sqrt{d_k}$	**4/4**	98.93 $\pm$ 0.03	**7.75** $\pm$ **0.43**

Color or fashion classification. This task is shown in Fig. 13d. For the creation of the dataset we define 10 random colors, i.e. (brown, blue, yellow, orange, red, green, purple, gray, pink, turquoise) and apply a random color to each target and each indicator sample. For targets we use fashion MNIST samples and indicators are MNIST classes 1 and 2. Task 1 is to compare digits. If they are the same class, the top right fashion sample must be classified. If not, the color of the top right sample must be classified.

Table 10: **Results "Color or fashion class" dataset**. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

		Color or fashion class		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\sqrt{d_k}$	4/4	92.85 $\pm$ 0.25	12.00 $\pm$ 3.32
ViT	$\frac{1}{3}$	4/4	92.53 $\pm$ 0.40	20.75 $\pm$ 2.49
NormSoftmax	$\sqrt{d_k}$	4/4	**92.75** $\pm$ **0.71**	**11.25** $\pm$ **0.83**

Digit grouping. Finally, we make the indicator task more difficult. We follow the same setting as for the "main dataset", as described in Fig. 1a. However, indicators are not sampled from digits 1 and 2, but from 1, 2, 3 and 4. Task 1 is to find out whether both indicators are smaller or both indicators are larger or equal to 3. I.e. we build indicator sets $[1, 2]$ and $[3, 4]$ if both indicators are from the same group the top-right image should be classified. As can be seen in Tab. 11, increasing the difficulty of the indicator task quickly makes the dataset too hard. Further optimization of hyper-parameters and architecture are likely to solve the tasks.

A.15 Experimental setup — Vision Task

We mostly follow the DeiT Touvron et al. (2021) training recipe without distillation. For optimization we use AdamW Loshchilov & Hutter (2019) with default values, i.e. $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Unless otherwise stated we Warmup the learning rate for 5 epochs from 10^{-6} to the maximum learning rate, use a weight decay of 0.05 and train for 300 epochs. We train all models

Table 11: **Results "Digit grouping" dataset**. τ not optimized for this task. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

		Digit grouping		
			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
ViT	$\sqrt{d_k}$	0/4	-	-
ViT	$\frac{1}{3}$	0/4	-	-
NormSoftmax	$\sqrt{d_k}$	0/4	-	-

with a batch size of 512, which fits on a single V100, for all the architectures that we considered. Since position and color can be important cues in our datasets we train without data augmentation. We only sample color-noise from a standard normal Gaussian with standard deviation 0.05 for each color channel independently. Color-noise is sampled for each sub-sample (i.e. each indicator and each target) independently and added to the RGB value.

Learning rate schedules for all models. To compare the tested methods and models fairly, we run a search over 9 learning rate schedules with 4 random seeds, each. We anneal the learning rate from a maximum to a minimum using a cosine scheduler. We also use Warmup, as described in the section on training details. The different schedules can be seen in Tab. 12. We pick the schedule that leads to highest Eureka-ratio for each model. In case of a tie we pick the schedule with higher accuracy.

Table 12: **Learning rate schedules.** We use cosine annealing from "max learning rate" to "min learning rate".

max learning rate	min learning rate
10^{-3}	10^{-5}
10^{-3}	$5 * 10^{-6}$
10^{-4}	10^{-5}
$5 * 10^{-4}$	$5 * 10^{-6}$
$5 * 10^{-4}$	10^{-6}
10^{-4}	10^{-6}
$5 * 10^{-5}$	10^{-6}
10^{-5}	10^{-6}
10^{-5}	10^{-7}

Info on taus: In initial experiments we tested for ViT 5 τ, $\tau = \frac{2}{3}$, $\tau = \frac{1}{2}$, $\tau = \frac{1}{3}$, $\tau = \frac{1}{4}$ and $\tau = \frac{1}{5}$. We found $\tau = \frac{1}{3}$, $\tau = \frac{1}{4}$ to work well and did not further optimize them for the different methods. For HT we set the goal temperature to the default $\sqrt{d_k}$ and tried also $\frac{1}{2*\sqrt{d_k}}$. Further optimizing these parameters for each model and dataset will most likely lead to improvements, but would add very little to a deeper understanding of the Eureka-moments.

A.16 Implementation details NormSoftmax

In practice, $\sigma(\cdot)$ can be defined by arbitrary functions. As highlighted in the background section, the standard deviation is a theoretically motivated choice. Alternatives are discussed by Jiang et al. (2022). In this work, we find the variance to work better for ViT and RoBERTa, while we stick to the standard deviation for the reasoning task.

A.17 Experimental setup – reasoning task

Reasoning task. Complementary to the synthetic vision tasks, we further simplified the multi-step task to an algorithmic task of the form $f(a,\ b,\ c,\ d) = \begin{cases} c, & g(a,\ b) \\ d, & \text{otherwise} \end{cases}$, where $a, b, c, d \in$

$\{0,\ 1,\ ...,\ n\}$, task 1 is $g(a,\ b) = 1[(\text{a is even} \wedge \text{b is odd}) \vee (\text{a is odd} \wedge \text{b is even})]$ and task 2 is a simple copying task of either c or d.

The input of the model is of the form "a b c d =", where, where $a, b, c, d \in \{0, 1, \ldots, n\}$. In our experiments, we set $n = 11$. We train the transformer on 30% of the entire set of possible inputs (i.e., $11^4 = 14\,641$ input combinations), that is with a batch size of 4392. The rest is used as test set. We train for 10 000 epochs over five random seeds. We use token embeddings of size of $d = 2^{\lceil \log_2 n \rceil} = 16$, four attention heads of dimension of $d/4 = 4$, $4d = 64$ hidden units in the MLP, and learned positional embeddings.

We trained with full batch gradient descent using AdamW (Loshchilov & Hutter, 2019) with a cross-entropy loss. We optimized learning rates via grid search over $[10^{-4},\ 10^{-2}]$ on seed 0. Following Nanda et al. (2023) we use a weight decay of 1.

A.18 Slingshot effects on reasoning task

We observed that NormSoftmax caused slingshot effects (Thilak et al., 2022) during the convergence phase of some of the training runs but believe this may be due to the interaction of gradients at different scales with adaptive optimizers (Nanda et al., 2023). Since slingshot effects only occur after Eureka-moments, they cannot be the cause for their occurrence. We did not further investigate this observation.

A.19 Experimental Setup – RoBERTa

The RoBERTa experiments are based on the Code provided by (Deshpande & Narasimhan, 2020). We follow the data acquisition and preparation strategy of Shoeybi et al. (2019). Thus, we train on the latest Wikipedia dump (downloaded on 08.02.2023). We train a 12 layer RoBERTa model with 12 heads. We use a batch size of 84 and a learning rate of 5e-5.

A.20 Experimental Setup – In-Context Learning

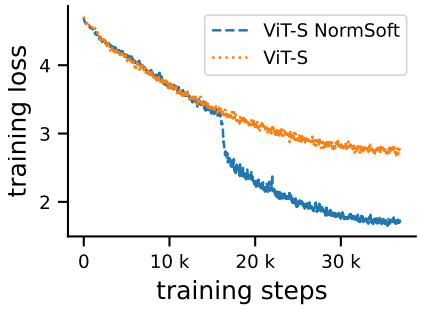
The in-context learning experiments are based on the experimental setup of Chan et al. (2022) using the Omniglot dataset (Lake et al., 2015). During training the network is presented with sequences of image label pairs, where each image is followed by it's true label. For the last image the label is missing and must be predicted by the transformer. The sequences are usually constrained in such a way, that the target label is often present at least once in the sequence (burstiness). The task can be partially solved by simply learning to associate image to a label, called in-weights learning (IWL). To generalize to unseen or rare samples, a better strategy is to exploit the solution given by the example in the context, referred to as in-context learning (ICL).

Experiments are based on the code of (Chan et al., 2022). We follow the original setup and use GPT-2 (Radford et al., 2019) with 12 layers, embedding dimension of 64 and 8 heads. The images were embedded using a non pretrained ResNet-18 architecture with (16, 32, 32, 64) channels per group while the labels were embedded using a standard embedding layer. Following the original training procedure, we ran the experiments for 500k iterations on a single GPU with batch size 32 using the Adam optimizer. We use a learning rate scheduler with a linear Warmup over 4000 iterations to maximum learning rate followed by square root decay.

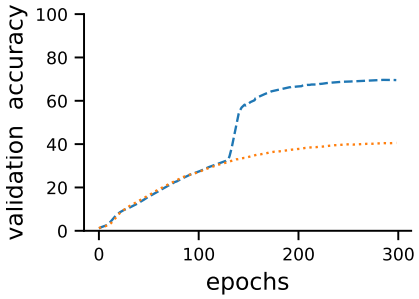
We optimized the learning rate for each method independently by testing 5 learning rates in the interval [3e-5, 3e-3]. For each learning rate and method we trained two models with different random seed. For both, the NormSoftmax variant of GPT-2 and GPT-2 a learning rate of 9e-4 lead to best results and was used for all experiments. We ran each experiment with 4 different seeds - 0, 42, 1337 and 80085. We report the averaged results and show instructive examples.

The in-context performance is reported on the 10 holdout classes not seen during training in the 2-way 4-shot few-shot evaluation setting. Here, the sequence consists only of 2 labels (0/1) with 4 images from each and we consider zero in-context performance for random chance level of 50 %.

For all experiments we used 50% of burstiness in data and uniform sampling.



(a) Training loss on ImageNet-100 dog decision for ViT-S trained with NormSoftmax and vanilla ViT-s



(b) Validation accuracy on ImageNet-100 dog decision for ViT-S trained with NormSoftmax and vanilla ViT-s

Figure 14: Results on ImageNet-100 based task. Also for a larger ViT, realistic data, high-resolution images we observe Eureka-moments. In particular, NormSoftmax leads to a Eureka-moment, while the vanilla ViT fails to learn task 1. Task description: If both indicators show the image of an identical dog the top-right image is the target and bottom left otherwise. Dog samples taken from 2 dog classes. Probability of top-right being the target is set to 0.5.

A.21 Eureka-moments on realistic, large scale high-resolution images (ImageNet-100 based)

To show that Eureka-moments can also be observed on large scale high-resolution datasets s we create a dataset with high-resolution natural images using ImageNet-100 Tian et al. (2020). For simplicity, we follow the same dataset design as for the MNIST-like datasets, i.e. we place the targets in the top-right and bottom left, while the other two quadrants show indicator images. Targets are simply images from ImageNet-100. The indicators are sampled from 2 of the ImageNet dog classes. If both indicators show the exact same sample, the top-right image needs to be classified and bottom left otherwise. The probability of top-right location being the target is 0.5. We train "ViT-S" and "ViT-S with NormSoftmax " following the standard ImageNet training setting using strong augmentations from Deit training Touvron et al. (2021). Vanilla ViT obtains only 41% accuracy and we don't observe a Eureka-Moment. For ViT-S with NormSoftmax we observe a Eureka-Moment at epoch 131. The training and validation curves are shown in Fig. 14.

A.22 Vanishing gradient in the Softmax.

Softmax attention can cause vanishing gradients for W_q and W_K. To see that softmax attention can result in vanishing gradients it helps to take a look at the gradients of the attention function. Let

$$A(W_q, W_k, W_v, X) = Z \tag{3}$$

$$A(W_q, W_k, W_v, X) = S\Big(D\big(Q(W_q, X), K(W_k, X)\big)\Big)(W_v, X), \tag{4}$$

$$S(D) = \text{softmax}(D), \tag{5}$$

$$D(Q, K) = \frac{QK^T}{\tau}, \tag{6}$$

$$Q(W_q, X) = W_q X, \tag{7}$$

$$K(W_k, X) = W_k X, \tag{8}$$

$$V(W_v, X) = W_v X. \tag{9}$$

be the attention function, where W_k, W_q and W_v are weight matrices, X is the input.

Using the chain rule we get

$$\frac{\partial A}{\partial W_q} = \frac{\partial A}{\partial D}\frac{\partial D}{\partial Q}\frac{\partial Q}{\partial W_q} \tag{10}$$

$$\frac{\partial A}{\partial W_k} = \frac{\partial A}{\partial D}\frac{\partial D}{\partial K}\frac{\partial K}{\partial W_k}. \tag{11}$$

Since $\frac{\partial D}{\partial Q}, \frac{\partial Q}{\partial W_q}, \frac{\partial D}{\partial K}, \frac{\partial K}{\partial W_k}$ are constants we only need to look more closely into $\frac{\partial A}{\partial D}$.

$\frac{\partial A}{\partial D}$ is given by $\frac{\partial A}{\partial D} = \frac{\partial S}{\partial D}V$, where $S(D)$ takes the values $S = (s_1, \dots, s_n)$. Therefore, to analyze how the gradients $\frac{\partial A}{\partial W_q}$ and $\frac{\partial A}{\partial W_k}$ behave, we need to analyze the $\frac{\partial S}{\partial D}$, i.e. the Jacobian of the Softmax $S(D)$. It is given by

$$\frac{\partial S}{\partial D} = \begin{pmatrix} s_1(1-s_1) & -s_1 s_2 & \dots & -s_1 s_n \\ -s_2 s_1 & s_2(1-s_2) & \dots & -s_2 s_n \\ \vdots & \vdots & \ddots & \vdots \\ -s_n s_1 & -s_n s_2 & \dots & s_n(1-s_n) \end{pmatrix}. \tag{12}$$

It can be easily seen, that almost all entries in the Jacobian are close to 0 whenever a single s_i is close to 1 and all others are almost 0.

A.23 Linear Probe results for Q, K and V and for target classification.

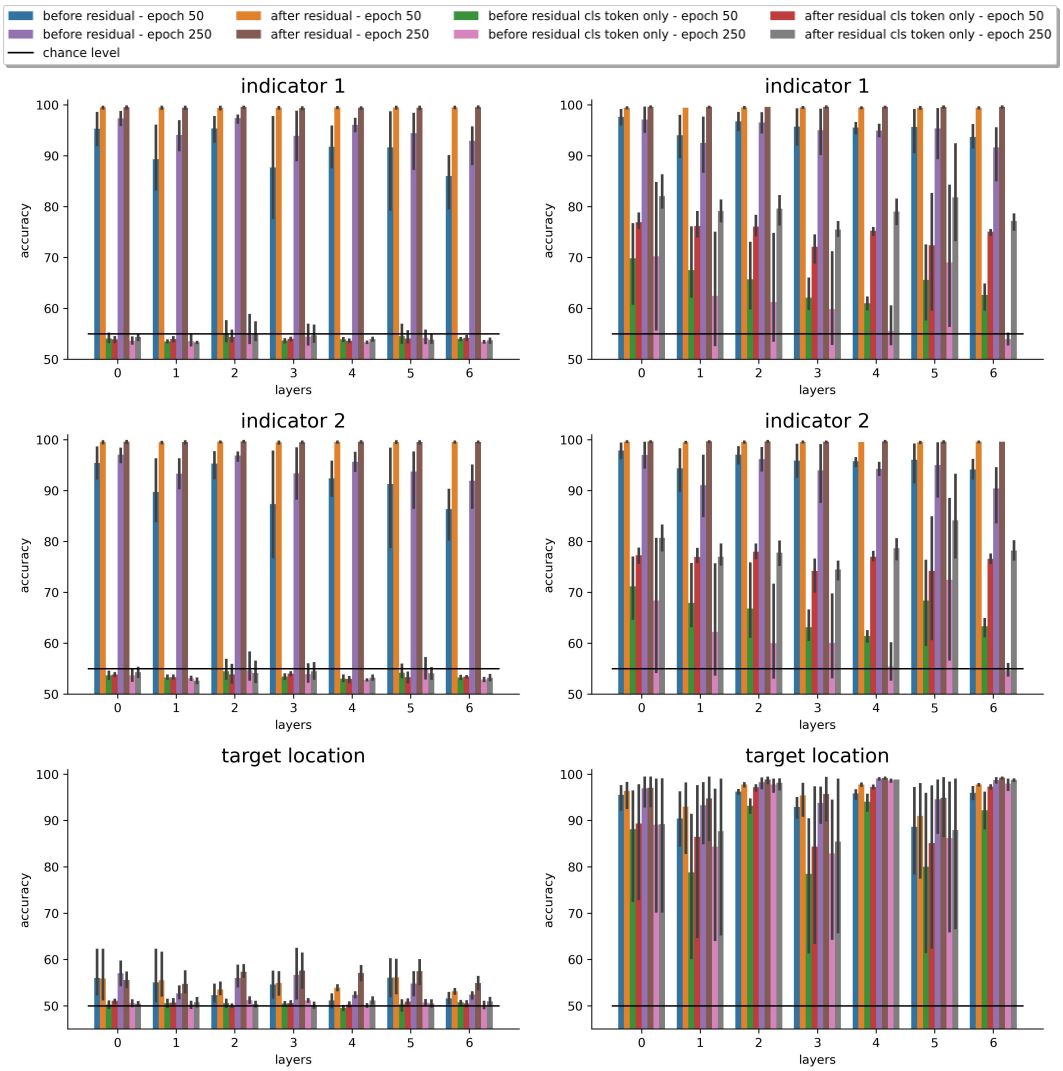
In the following we will show the linear probe results for Q, K, V and Z for all layers.

Linear probe results for Z for all layers. Fig. 15 shows the same plot as in the main paper, but for all layers. In addition to the observations made for the main paper, we can see that layer 2, 3 and 6 for the ViT with Eureka-moment (Fig. 15b) represent significantly more information about the target locations than other layers in the CLS token. Indicating, that this information is extracted in these layers and written on the CLS token.

Linear probes for Q and K and V. Figs. 16 & 17 show the results when using the Q and K as input for the linear probes. Note, that Q and K are not updated by the residual connection of the attention block, therefore, no bars for "after residual" are plotted. The linear probe classification accuracies for Q and K are very similar. Again, we can see that the ViT without Eureka-moment does not represent indicator information in the CLS token and target location can not be linearly separated from other information. Similarly, as for linear probe results with Z, layer 2, 3 and 6 for Q and K contain significantly more information about the indicators and target location for the ViT with Eureka-moment (compare Figs. 15b, 16b, 17b).

The linear probe results for V look very similar to those for Q and K (see Fig. 18).

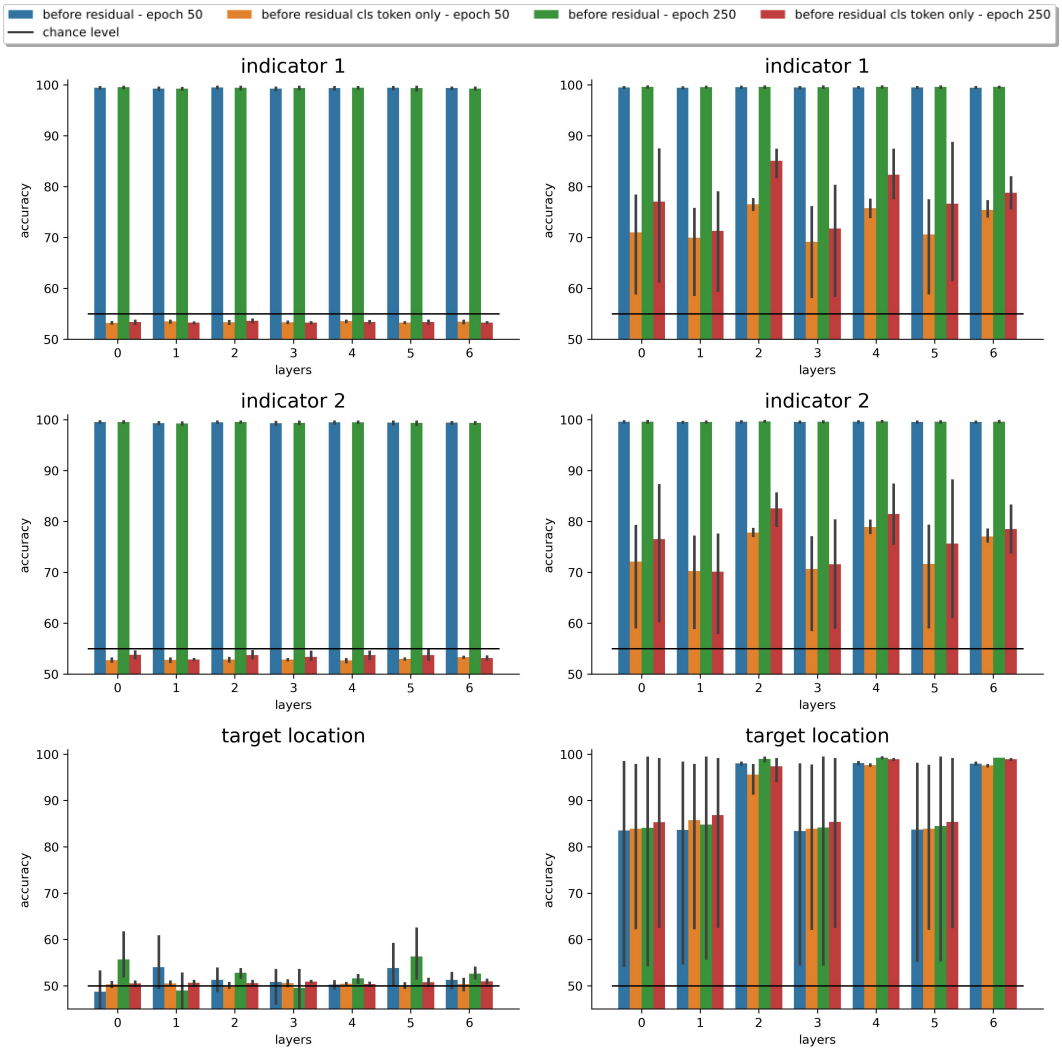
Linear probes from Z to targets. Last, we show the linear probe results when predicting the targets from Z. As can be seen, from the entire representation, for both ViT with Eureka-moment and Vit without Eureka-moment, target classes can be predicted with high accuracy. Differences can be observed when using only the CLS token. Here, we observe that more target information is in the CLS token of the model without Eureka-moment (see Fig. 19).



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

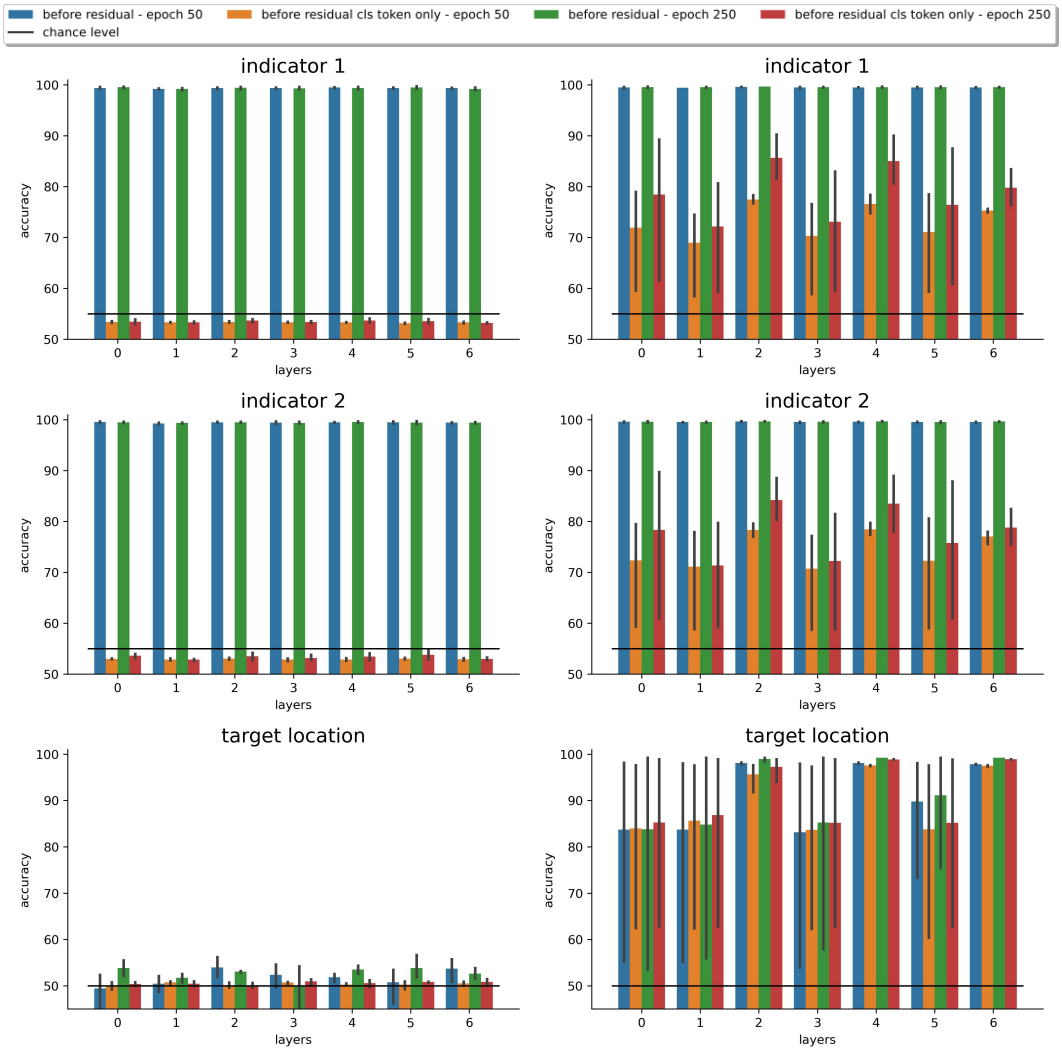
Figure 15: **Linear probe results for with Z as input (all layers).** This the complete version of Fig. 2a. Additionally, we can observe in (b) that layers 2, 3 and 6 contain more target location information than other layers, indicating, that this information can be processed in these layers.



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

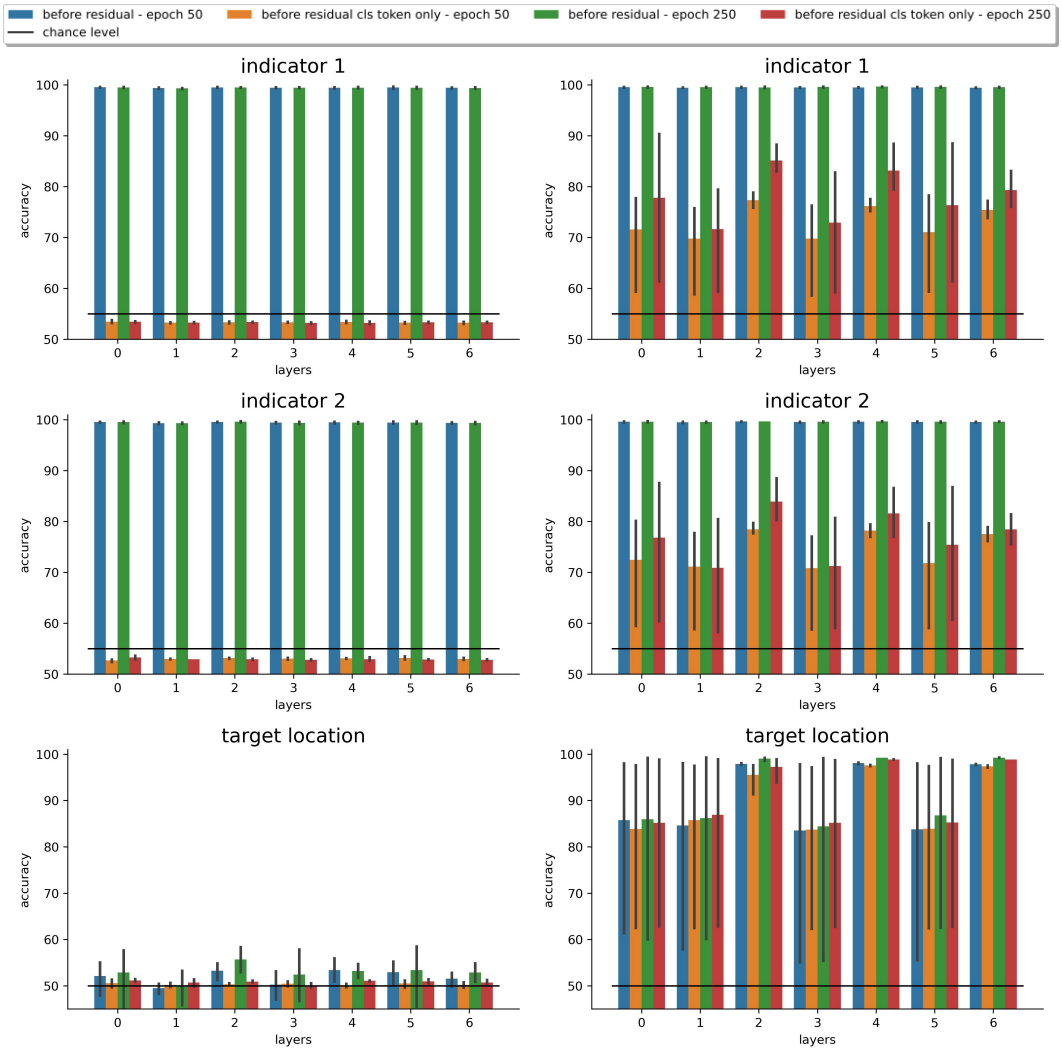
Figure 16: **Linear probe for Q.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

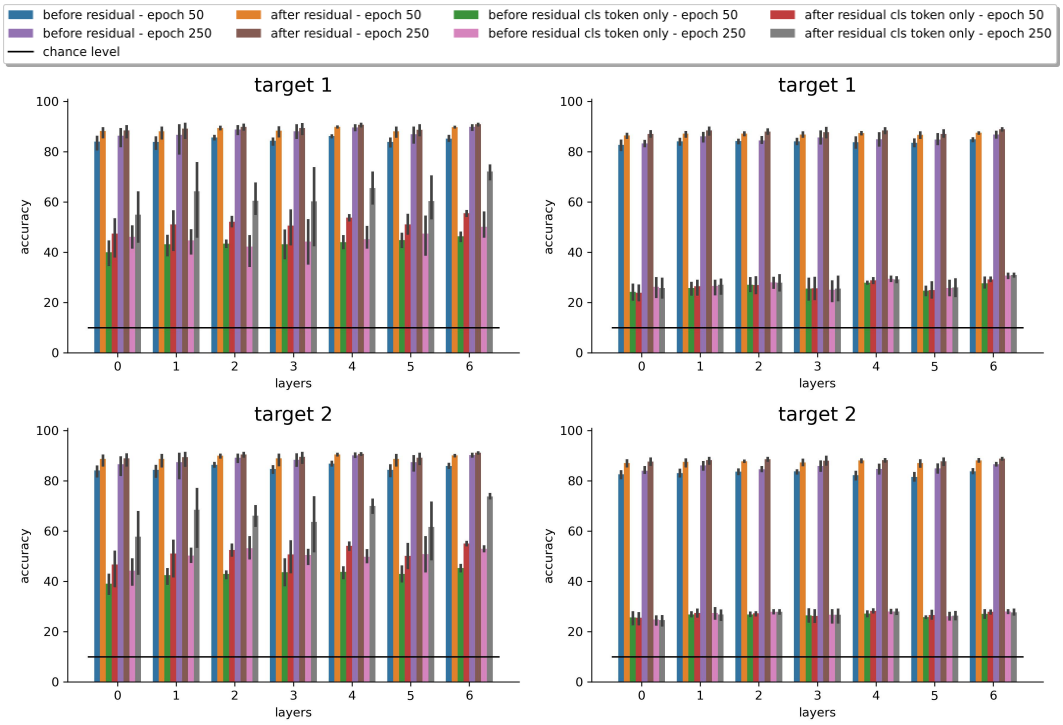
Figure 17: **Linear probe for K.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

Figure 18: **Linear probe for V.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

Figure 19: **Linear probe for Z with target classification.**