Eureka-Moments in Transformers: Multi-Step Tasks Reveal Softmax Induced Optimization Problems

David T. Hoffmann [1 2] **Simon Schrodi** [1] **Jelena Bratulić** [1] **Nadine Behrmann** [3] **Volker Fischer** [2] **Thomas Brox** [1]

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 the synthetic 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, to higher final accuracy and training becomes more robust to hyper-parameters.

1. Introduction

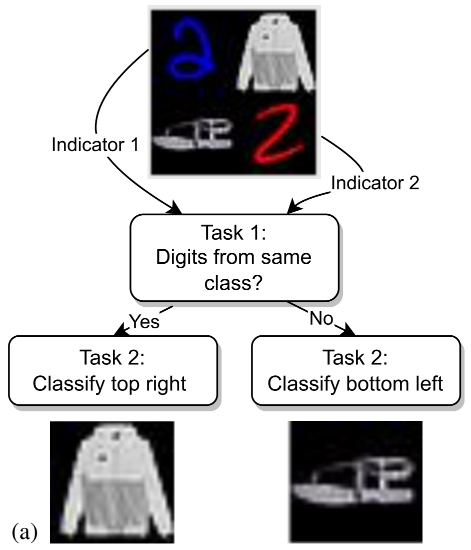
A key quality of any intelligent system is its ability to deal with complex problems that may consist of multiple subproblems. It should learn to solve these sub-problems even in the absence of direct feedback. Deep learning has enabled such capabilities to a certain degree. For example, deep classifiers learn the hierarchical feature representations necessary to build a good classifier. Reinforcement learning learns object representations required to predict how to receive a sparse reward. 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 tokens based on positional information by learning induction heads (Olsson et al., 2022). While aforementioned examples show great promise, researchers spend a large effort on designing the training process to learn sub-tasks. For instance, reward shaping is common in reinforcement learning, many computer vision works use explicit or implicit intermediate supervision, while for language modelling and ICL a good data arrangement plays an important role (Chan et al., 2022). For some of these problems, a saturation of the learning process followed by a sudden improvements can be observed, as shown in Fig. 1c. However, the relation of implicit multi-step learning to saturation of the learning process followed by abrupt improvements in training loss has not been investigated.

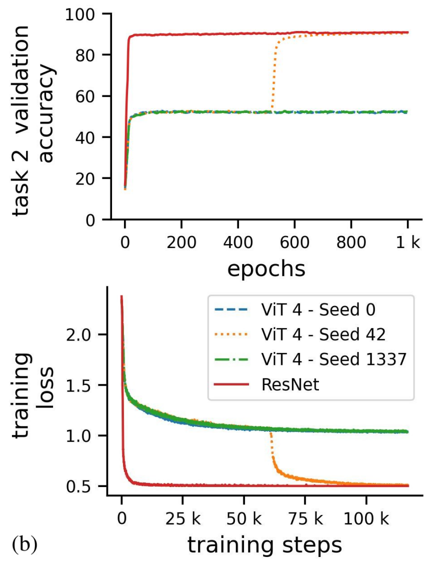
But how can we study multi-step learning? One may tend to study popular tasks in detail, for which many benchmarks and results already exist. BERT pretraining (Devlin et al., 2019) and ICL (Chan et al., 2022) are candidates that are likely to entail a multi-step task. For BERT the network might first learn word frequencies and might learn to use the context to predict masked tokens in a second step. Similarly ICL can be understood as multi-step task, where first similarities of tokens have to be learned, followed by learning where to look up the correct label, i.e., learning an induction head. For both tasks we observe sudden improvements, similar to those that we study in this work (see Fig. 1c). Unfortunately, real data prohibits a clean study due to multiple factors: 1) The exact sub-tasks are typically unknown and, hence, hard to study. 2) There are many easy samples that do no require multi-step reasoning, overlaying the progress on the multi-step task. 3) The features necessary for the tasks are unknown, i.e., we cannot study what the network fails to learn, let alone the reason for it. 4) Even the number of steps is unknown, thus, we cannot determine if models learn only a subset of the tasks.

As a remedy, we propose to analyze multi-step learning by

[1]University of Freiburg [2]Bosch Center for AI [3]Amazon (work done while at Bosch). Correspondence to: David T. Hoffmann <hoffmann@cs.uni-freiburg.de>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).


Indicator 1
Indicator 2
Task 1: Digits from same class?
Yes
No
Task 2: Classify top right
Task 2: Classify bottom left
(a)



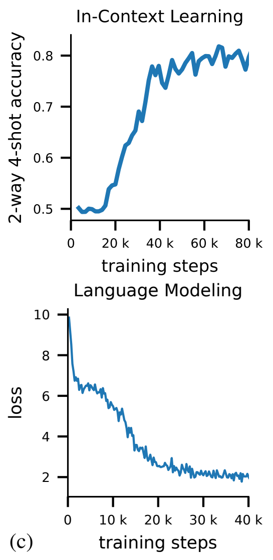


Figure 1: **Transformers can get stuck during optimization for two-stage tasks.** **(a)** Describes our 2 step decision task used to study Eureka-moments. **Task 1** is to compare the two **indicators** (here digits). If the digits are the same, **task 2** is to classify the top-right image and bottom left else. 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 a Eureka-moment. Eureka-moments are characterised by a sudden increase of accuracy and drop of the loss (in contrast to Grokking (Power et al., 2022)). ResNets are not susceptible to this kind of optimization difficulty. **(c) Eureka-moments on real datasets.** Sharp improvements after initial plateauing can also be observed for GPT-2 ICL, here in the Omniglot ICL task (Chan et al., 2022) and language modeling with RoBERTa on Wikipedia. We will show later that our analysis transfers to these tasks (see Fig. 9).

controlling the data-generating process with synthetic data. This allows us to create clean two-step tasks in a controlled setting that facilitates a detailed study. Specifically, we remove confounding variables, know the number of tasks, and know for each task the relevant features, their location and the total number of steps. Thus, it solves the issues above all at once. This comes with the assumption that our findings on synthetic data transfer to related observations on real data. Indeed, our understanding how Eureka-moments appear on synthetic tasks and the corresponding way to improve training leads to better training behavior on non synthetic tasks, i.e., higher ICL accuracy on Omniglot and earlier Eureka-moments on masked language modeling on Wikipedia (see Fig. 9).

In each of our synthetic datasets, the answer to the first task $p(z|x)$, which is not explicitly modeled in the loss function, must be found by the model in order to correctly solve the final second task depending on it $p(y|x, z)$. For example, in Fig. 1a, the model must first classify the two digits to find out if they are of the same class, which determines where to look for the subsequent FashionMNIST classification task. The loss only provides a training signal for the latter task. Thus, the model must figure this out by itself during training. Formally, 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.

Our study reveals that transformers have difficulties in learning such two-step tasks (Fig. 1b). After they learned to classify a randomly selected FashionMNIST image $p(y|x, z)$, they saturate and only after a long time suddenly learn $p(z|x)$, i.e., the task to select the right fashion image by comparing the digits. We call this phenomenon a Eureka-moment. Intriguingly, we find that they never learn task 1 within 1000 epochs and stick with the prior $p(z)$ for some random seeds. We later find that the probability of Eureka-moments depends on the difficulty of the task. In contrast to the transformer, a ResNet learns both tasks immediately.

Our goal is not to add to the old transformer vs. CNN discussion, but we want to investigate this particular problem. What is its cause? Is it due to a too small capacity? Is it the number of heads or the learning rate? Is it the spatial arrangement of task 1 and 2? We found that these factors play only a minor role and finally traced the problem back to the Softmax function in the transformer's attention blocks. We found parts of the gradient's components become very small depending on the Softmax output. This cannot be fixed by using a larger learning rate, since other components of the gradient are large, but it can be simply fixed by a normalization of the Softmax function.

In summary, the contributions of this analysis paper are: 1) We study multi-step learning without intermediate supervision via a fully controlled data-generating process on synthetic tasks. 2) We discover a new failure mode of transformers. 3) We analyse the mechanisms underlying this failure mode and find that the Softmax function leads to (local) small gradients for the key and value weight matrices, thereby hindering learning. 4) To validate the role of Softmax, we mitigate the failure mode through targeted interventions. We show that these interventions 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 transfers to these settings. The code to reproduce the results and create the datasets is available[1].

2. 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 may just be an artifact of the choice of a discontinuous metric (Schaeffer et al., 2024; Srivastava et al., 2023), we observe rapid changes also for continuous metrics in our setting. 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; ano, 2023; Chen et al., 2024). Deshpande & Narasimhan (2020) proposed to bias the attention mechanism towards predefined attention patterns and observed speed-ups in BERT training, while Chen et al. (2024) connect the sharp drop to sudden learning of syntactic attention structures. 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.

[1] `https://github.com/boschresearch/eurekaMoments`

Grokking. A similar phenomenon has been discovered on synthetic data (Power et al., 2022) and was further studied in (Liu et al., 2023; Nanda et al., 2023; Thilak et al., 2022; Millidge, 2022; Barak et al., 2022; Liu et al., 2022). 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, 2019; Child et al., 2019; Wang et al., 2019) or Warmup (Baevski & Auli, 2019; 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 results 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. (2023) 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.

3. Background

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

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

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.11). 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. As a remedy, we propose to start training with a low temperature and follow a schedule to heat it up to the default value of $\sqrt{d_k}$. We refer to this approach as **Heat Treatment** (HT) and it is one of our interventions to test whether the Softmax is indeed the root cause of the optimization difficulties. This approach has multiple advantages. First, it removes the difficulty of choosing the exact temperature. Second, 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 (due to increasing the temperature), it does not run into the issue of low attention entropy. Last, 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., 2023), 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

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

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

4. Task Description & Experimental Conditions

Recent works across diverse fields found sudden abrupt learning behaviors, 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), where only the last task is supervised. But why does it take so long for transformers to learn such mechanisms and why is the improvement so sudden?

Task description. We suspect that these training problems are due to the multi-step nature of these tasks. To test this we study such multi-step mechanisms on a simple two-step tasks. Note that many-step tasks would also be possible but are a more complicated study object. Fig. 1a provides a schematic overview for one of our vision tasks: Task 1 requires the model to indicate where to look at, i.e., top right or bottom left depending on whether the 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.

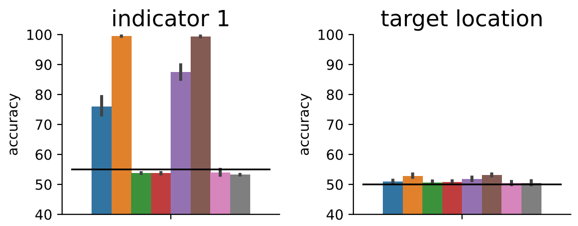
Vision dataset creation. The visual datasets are based on MNIST (LeCun et al., 2010) and Fashion-MNIST (Xiao et al., 2017). An example and a schematic of the task is shown in Fig. 1a. The samples are created by sampling 2 random Fashion-MNIST samples and 2 digit samples from the MNIST classes "1" and "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.

Reasoning task. Complementary to above 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 $g(a,\ b) =$

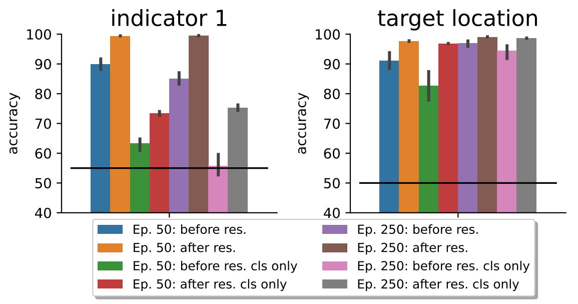
$\mathbb{1}[(\text{a is even} \wedge \text{b is odd}) \vee (\text{a is odd} \wedge \text{b is even})]$ and task 2 is a simple copying task of either c or d. More details are provided in A.16.

Metrics. We define the **Eureka-ratio** as the proportion of training runs with Eureka-moment across the different random seeds. To automatically detect Eureka-moments, we set a conservative threshold at a validation accuracy of 70%, as this threshold can not be crossed without solving task 1. **Accuracy after Eureka-moment**, in the following referred to as accuracy, is computed over all runs that had a Eureka-moment. Note that this metric must be jointly considered with the Eureka-ratio, since high accuracy with low Eureka-ratio indicates that optimization typically failed (no Eureka-moment) but possibly just one "lucky" training run learned to solve the multi-step mechanism. Finally, **average Eureka-epoch** provides the average epoch at which the Eureka-moment happened. It is computed only for runs with Eureka-moment and again must be interpreted jointly with the Eureka-ratio.

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.14. 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.16 .



(a) ViT (without Eureka-moment)



(b) ViT $\tau = \frac{1}{3}$ (with Eureka-moment)

Figure 2: **What is represented in different parts of the attention block.** Bar plots show linear probe accuracy averaged over heads. Indicator 1 is the top MNIST digit. Both ViT and ViT $\tau = \frac{1}{3}$ extract the indicator class information from the images and it is available in each layer. Information is available before and after the residual connection, therefore it is not entirely ignored by the attention. Differences between ViT and ViT $\tau = \frac{1}{3}$ are 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 indicates chance. Indicator 2 plots are similar. More layers and indicator 2 plots are shown in Fig. 19. Q, K, V linear probes in Figs. 20 to 22. A similar analysis using more sensitive information-theoretic probes (Voita & Titov, 2020) can be found in Appendix A.12

5. Understanding Eureka-moments & Optimization Problems of Transformers

Here, we analyze the problem on the dataset described in Fig. 1a. In Sec. 5.3 we provide supporting experiments and finally show indications, that the results can be transferred to real datasets, i.e., Wikipedia text completion and ICL.

5.1. Why Do Transformers Fail to Learn Two-step Tasks?

To investigate why ViT's training fails, we analyze the learned 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. However, note that linear separation becomes very likely as feature dimensionality increases. Thus, a high accuracy on the linear probe classification does not imply that the transformer is using that information. It only shows that the information is represented and linearly separable.

Does the transformer fail to distinguish the indicators? The left bar plots in Fig. 2a show that the indicators can be linearly separated well (orange, brown) across all layers (Fig. 19). Nonetheless, ViT fails to learn task 1 (Fig. 2a right plot). Thus, the transformer represents the information to solve task 1 but does not utilize it.

Does the transformer filter out information required for task 1? Since the loss provides a training signal only for task 2, the transformer may learn to just ignore the indicators. Task 1 information could be ignored in the attention blocks, i.e., the attention function does not attend to the indicators. To test this, we probe the representation after the attention operation at two locations, before and after

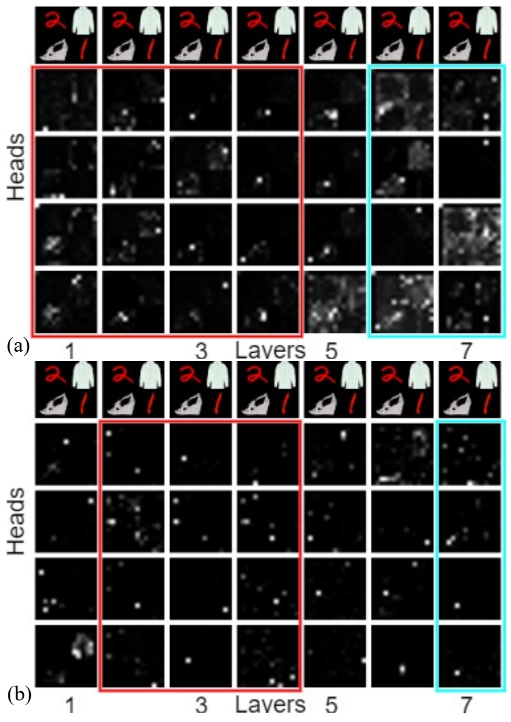


Figure 3: **Attention maps after training for: (a) ViT without Eureka-moment**. It fails to compare the 2 digits. First layers explicitly ignore indicators (digits) (highlighted with red). **(b) ViT $\tau = \frac{1}{3}$ with Eureka-moment** attends indicators in first layers (red) and predominantly attends 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.

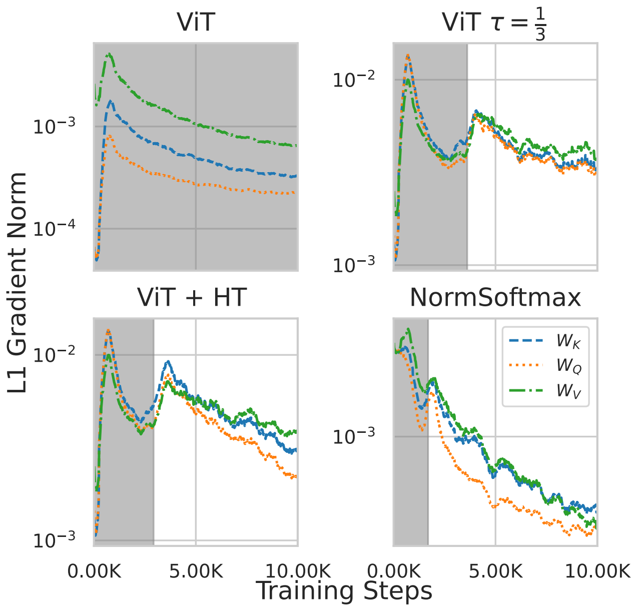


Figure 4: **L1 gradient norm during training** for W_K, W_Q and W_V for the first layer. For ViT, W_K and W_Q receive much smaller gradients than W_V. Before Eureka-moment (gray regions), the differences between gradient magnitudes are much smaller for smaller temperatures or NormSoftmax. The y-axis is log scaled. All layers shown in Fig. 14.

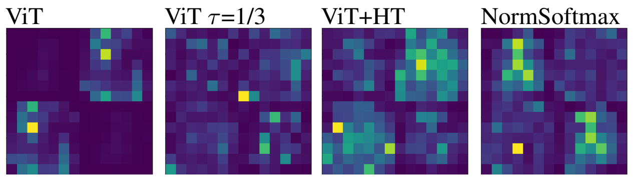


Figure 5: **Gradients on image for** W_k at Epoch 50. For ViT the gradient for W_K comes mostly from target regions, while for the other approaches indicator regions provide substantial gradient. A detailed explanation of this plot and plots for Q and V can be found in Sec. A.2

the residual connection. Fig. 2a reveals that the indicator information is available in all layers. Early in training, some indicator information is filtered out in the attention block (blue). Indicator information is partially filtered out in deeper layers, but is always recovered by the residual connections (see Fig. 19). Thus, features required for task 1 are not filtered out.

Does the transformer fail to combine the information? We observe that 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 utilize 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 a Eureka-moment. Fig. 2b shows the linear probe results for a transformer that had a Eureka-moment. Interestingly, the target location can be inferred by linear probes from all layers and all tested representations with high accuracy. This is in stark contrast to transformers that had no Eureka-moment. The second striking difference is that indicator information is represented in the CLS token. This is even stronger for early layers (Fig. 19). We suspect, that the transformer uses the CLS token to match the classes of the indicators.

Why does the transformer fail to combine the information? We visualize the attention maps of two fully trained ViTs in Fig. 3. We find that the transformer without Eureka-moment does not attend the indicator digits and attends only the targets of task 2, whereas a transformer with Eureka-moment attends the indicator digits in early layers. This suggest that a ViT without Eureka-moment does not pay enough attention to the indicators to match them and struggles to learn to attend different regions.

Why does the transformer fail to learn to attend to the indicators? Based on the discussion in Sec. 3, we hypothesize that ill-distributed attention scores lead to small gradi-

Table 1: **Quantitative results** on the **main dataset**, as described in Fig. 1a. and the **No position task** (Fig. 7). τ not optimized for *No position task*. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch.

			Avg. over EMs	
Model	τ	ER ↑	Acc. ↑	Avg. EE.
Main Dataset				
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
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	$\frac{1}{3} \to \sqrt{d_k}$	1/4	88.36	242
NormSoftmax + HT	$\frac{1}{3} \to \sqrt{d_k}$	1/4	90.63	**19**

Table 2: **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 $\frac{1}{3} \to \sqrt{d_k}$	25/36

ents for W_K and W_Q, which in turn inhibit learning and in particular learning to shift attention towards the indicators. Note that high attention to some pairs with low attention to all others or uniformly distributed attention can result in vanishing gradients (Noci et al., 2022). To test this, we visualize the L1-norm of the gradient of the first layer in Fig. 4. 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. Differences between the gradients are much smaller for ViT $\tau = \frac{1}{3}$, ViT+HT and NormSoftmax, in particular before the Eureka-moment. Fig. 5 shows the origin of gradients on the image plane for W_k. For ViT, the gradients mostly originate from the target regions, which further explains why many steps are needed to move attention to the indicators, as these gradients only lead to improved target recognition.

Is too small or too large attention-entropy the problem? We visualize the distribution of attention maps over training in Fig. 10: the vast majority of attention scores is very small. This indicates that a too uniform attention is causing small gradients. Larger attentions are rare, but not absent, as can be seen in Fig. 3a, but indicator regions have small and uniform attention. Thus, we conclude that **local uniform attention** causes the transformer's learning problems.

5.2. Can Enforcing Lower Entropy Attention Maps Resolve the Small Gradients?

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. Particularly, we modify Softmax's temperature τ in the attention block. Large temperatures increase entropy, while small temperatures decrease it. We apply following interventions: training with lower/higher temperature; HT, where the temperature increases from a low value to default temperature during the first half of training; and NormSoftmax, which adaptively changes the temperature for each sample, head and layer.

Does a lower temperature solve the small gradient issue and thereby mitigate the optimization issues? Increasing the temperature from low to default or using NormSoftmax increases high attention scores (c.f. Fig. 10). Importantly, the transformer learns to also attend to the indicators (Fig. 3b). Furthermore, all approaches (lower temperature τ, HT and NormSoftmax) solve the imbalanced gradient issue for W_V, W_Q and W_K (Fig. 4) and lead to higher gradients in indicator regions (Fig. 5). Eureka-moments happen much earlier or instantly (see Fig. 6). Thus, the interventions indeed mitigate the optimization issues. A comprehensive comparison between a vanilla ViT and other versions is provided in Tab. 1. 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.

5.3. 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

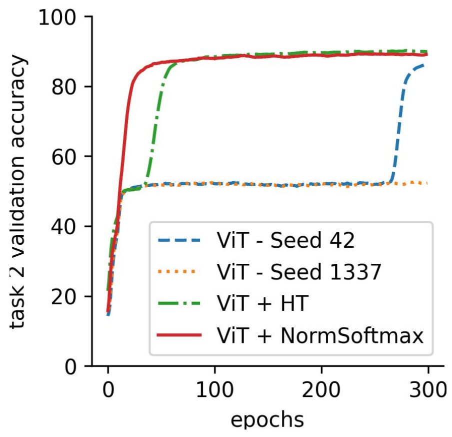


Figure 6: **Validation accuracy curves on *main dataset*.** Both interventions, ViT+HT and NormSoftmax drastically reduce the saturation period and can even lead to a complete disappearance of the saturation period.

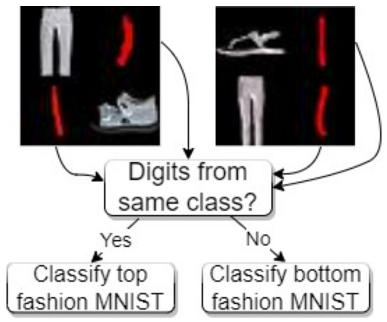


Figure 7: ***No position task* description.** This task is identical to the *main task*, but removes more information by swapping indicator (digit) and target (fashion) in each row with a probability of 0.5, i.e., task 2 is to classify either the top or bottom fashion sample. Two samples are shown to highlight differences from the task described in Fig. 1a.

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. 7. We observe that removing the fixed position for indicators and targets makes the task even more difficult (Tab. 1 bottom) 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. 2). See A.5 for a more detailed answer to the posed question.

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.6 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.10.

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 work for the *Main task* but not for the *No position task*, (see Tab. 1) and is very sensitive to the learning rate (see Tab. 2). We attribute this to the differences in gradient magnitudes for indicators and targets and discuss it further in Sec. A.4.

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. 4, which indicates that gradients become small, but not 0. Indeed we observe that training for 3000 epochs results in a 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 are hard to predict.

Is this truly a transformer optimization problem?

To show that Eureka-moments are 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. 4). Here, features (numbers) are directly provided as tokens and need not be extracted. Fig. 8 reveals that 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.

Does the NormSoftmax intervention translate to real data? Jiang et al. (2023) reported improved performance and faster convergence on ImageNet and machine translation tasks using NormSoftmax. Both tasks likely contain some innate multi-step tasks, e.g., identifying a common discriminative feature and then discriminating between the difficult classes for ImageNet. These improvements may be due to easier multi-step learning with NormSoftmax.

To further 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

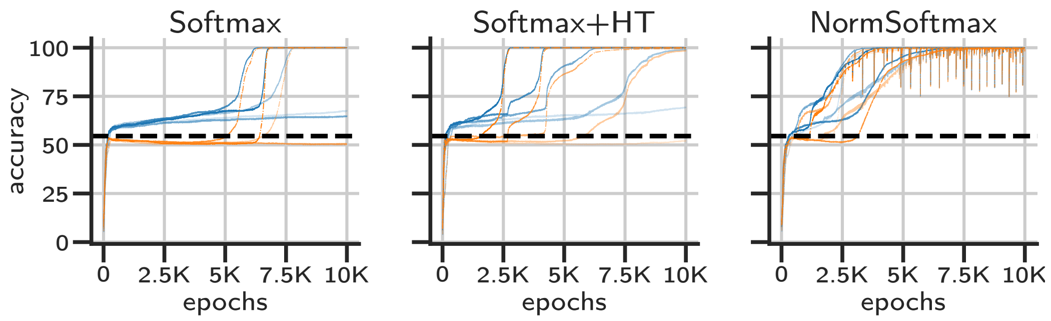
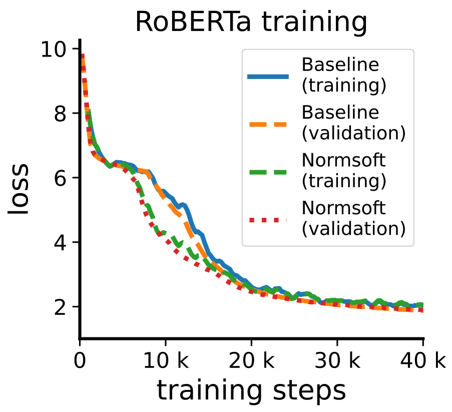
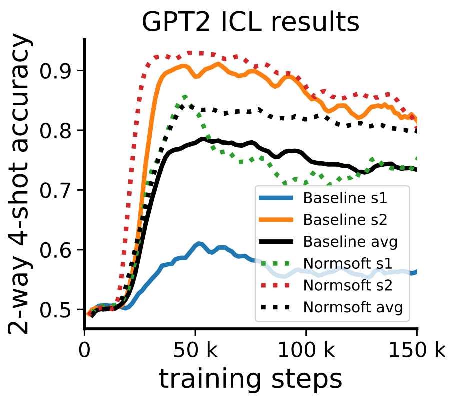


Figure 8: **Eureka-moments for single-layer transformers on a simple reasoning task.** We show the train (blue) and test (orange) accuracies for attention with Softmax, Softmax+HT, or NormSoftmax, over 5 random seeds (transparencies). Chance probability is $6/11 \approx 54\%$ (black).



(a) Language modeling



(b) In-Context learning

Figure 9: **Eureka-moments on real datasets.** Sharp improvements after plateauing can be observed 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.

1 to be learning of general word probabilities and task 2 to be modulating these probabilities based on context, or modulating them based on syntactic relations (Chen et al., 2024). 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.18 and A.19, respectively. Fig. 9 shows that NormSoftmax indeed leads to earlier Eureka-moments for RoBERTa. Furthermore, NormSoftmax also improves ICL, i.e., GPT-2 with NormSoftmax trained on the Omniglot ICL task results in higher ICL accuracy and seems to prevent failure cases like no or a small Eureka-moment (Baseline s1 in Fig. 9). Thus, our analysis and results are indeed transferable to real datasets and also to transformers with causal attention,i.e., GPT-2.

6. Limitations and Conclusion

Limitations. The ability to decompose tasks into sub-problems and learn to solve those sub-tasks is a common problem, but it is difficult to study on real datasets, due to many confounding factors. 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. We find evidence for that, as the intervention (NormSoftmax) leads to improvements on real data and can prevent getting stuck (Fig. 9).

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.

Impact Statement

We do not see any particular societal impact other than the overall impact of advancing the field of Machine Learning.

Acknowledgements

We thank Philipp Schröppel and the entire Computer Vision Group Freiburg for inspiring discussions. Sudhanshu Mittal for feedback on the manuscript and Matteo Farina for reporting observations of potential Eureka-moments on vision and language tasks.

This research was funded by the Bosch Center for Artificial Intelligence, the Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz (BMUV, German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection) based on a resolution of the German Bundestag (67KI2029A), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1597 – 499552394 and 417962828.

References

Escaping the plateau: Dynamic context length adaptation for efficient bert pretraining. 2023. URL `https://openreview.net/forum?id=urzQrPofiMM`. Anonymous review.

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. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=ByxZX20qFQ`.

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.

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`.

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 Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pp. 7480–7512. PMLR, 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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

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. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 3630–3636, December 2020. URL `https://aclanthology.org/2020.coling-main.324`.

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: Normalizing the input of softmax to accelerate and stabilize training. In *2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, pp. 1–6. IEEE, 2023.

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

Thomas Kurbiel. Derivative of the softmax function and the categorical cross-entropy loss. https://towardsdatascience.com/derivative-of-the-softmax-function-and-the-categorical-cross-entropy-loss-ffceefc081d1, 2021.

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 S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 34651–34663, 2022.

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`.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

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. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27198–27211, 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. In *The Twelfth International Conference on Learning Representations*, 2023.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In *Advances in Neural Information Processing Systems*, volume 36, 2024.

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. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=uyTL5Bvosj`.

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 grokking phenomenon. In *Has it Trained Yet? NeurIPS 2022 Workshop*, 2022.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European Conference on Computer Vision*, pp. 776–794, 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.

Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 183–196, 2020. URL `https://aclanthology.org/2020.emnlp-main.14`.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. Learning deep transformer models for machine translation. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1810–1822, 2019. URL `https://aclanthology.org/P19-1176`.

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, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL `https://openreview.net/forum?id=yzkSU5zdwD`. Survey Certification.

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 Joshua M Susskind. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pp. 40770–40803. PMLR, 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. 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 a more complete version of Fig. 4. Provide more insights into why gradient magnitude scaling does not work and provide more details on our analysis of competing hypotheses. 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 repeat the linear probe analysis with more sensitive information probes. We provide the linear probe plots using also Q, K and V representations and the full version of Fig. 4. Last, we provide details on training and experimental setups and an explanation for vanishing gradients in case of centralized attention maps.

A.1. Attention Distribution Over Time

We show the change of the attention distribution during the course of training in Fig. 11.

A.2. 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. 5, 11, 12. 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. 13. We make the following observations:

1. In general, the gradients are not evenly divided between target and indicator K, with usually smaller gradient for the indicator regions. Therefore Fig. 4 even underestimates the difference for the indicator regions, i.e., the regions relevant for a Eureka-moment.
2. This difference can explain why more time is needed to reach a 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 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.3. Gradient Norm

Fig. 14 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.4. Why Gradient Magnitude Scaling Does Not Work.

Following the observation of Fig. 4, 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. 11. Simply scaling up the gradients would not solve the imbalance between indicator and target gradients.

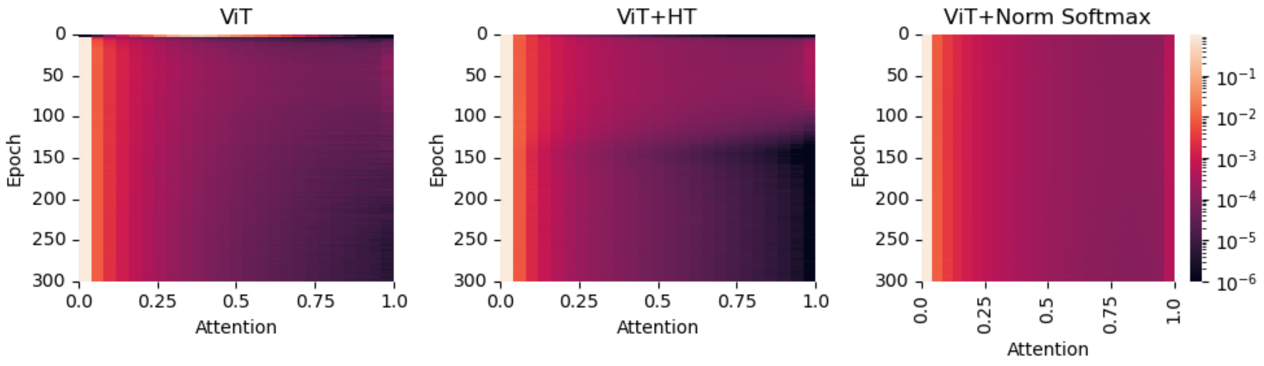


Figure 10: **Attention distribution as a heatmap.** Attention scores are sampled during evaluation after each epoch and binned to 25 bins. The color map is log scaled. For all 3 models, the vast majority of values falls into the first bin. ViT shows very few higher attention scores. ViT+HT and ViT+NormSoftmax lead to a significantly larger number of medium and high attention values. For ViT+HT this is limited to the first 100 epochs.

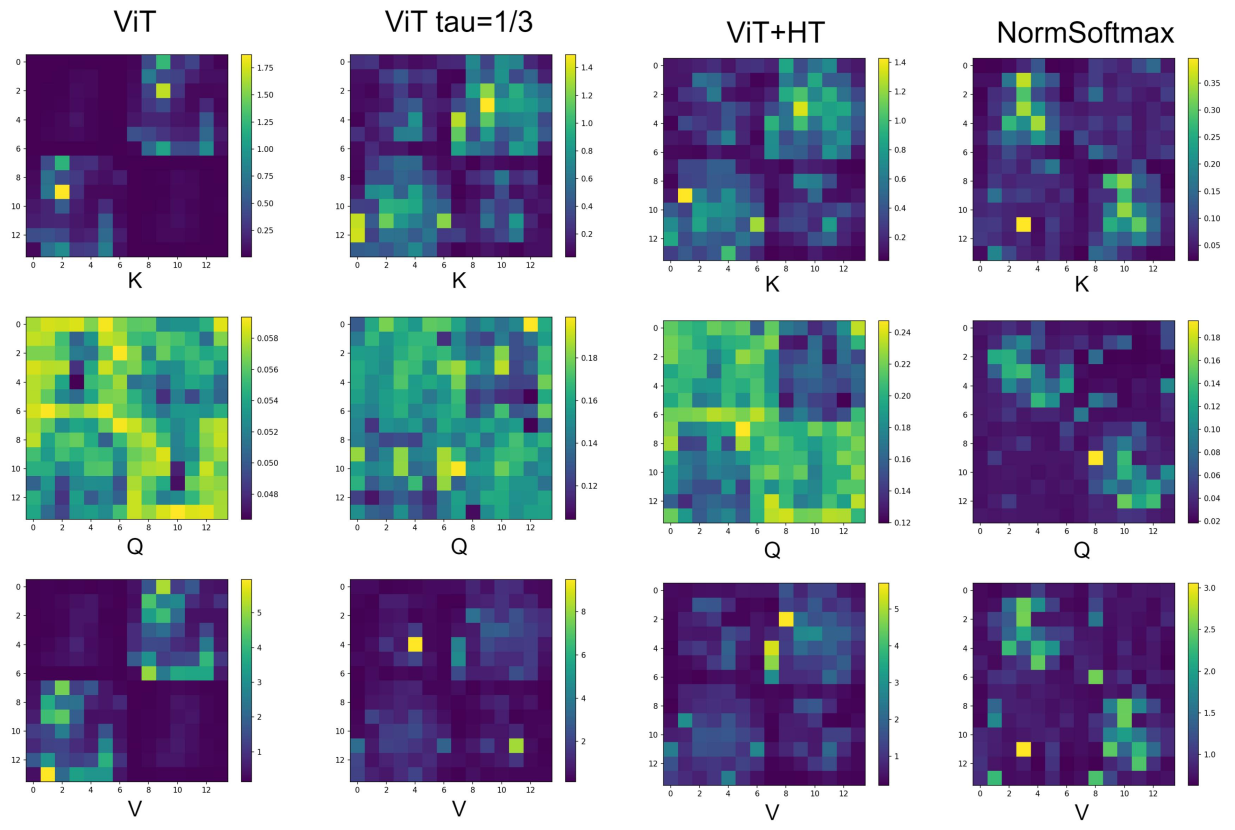


Figure 11: **Gradient norm for different image regions visualized for Q, K, V.** Larger gradients are visible for target regions, i.e, top-right and bottom-left. Indicator receive less gradients for K and V. ViT $\tau = \frac{1}{3}$, ViT+HT and NormSoftmax mitigate this problem well. Indicator regions 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.

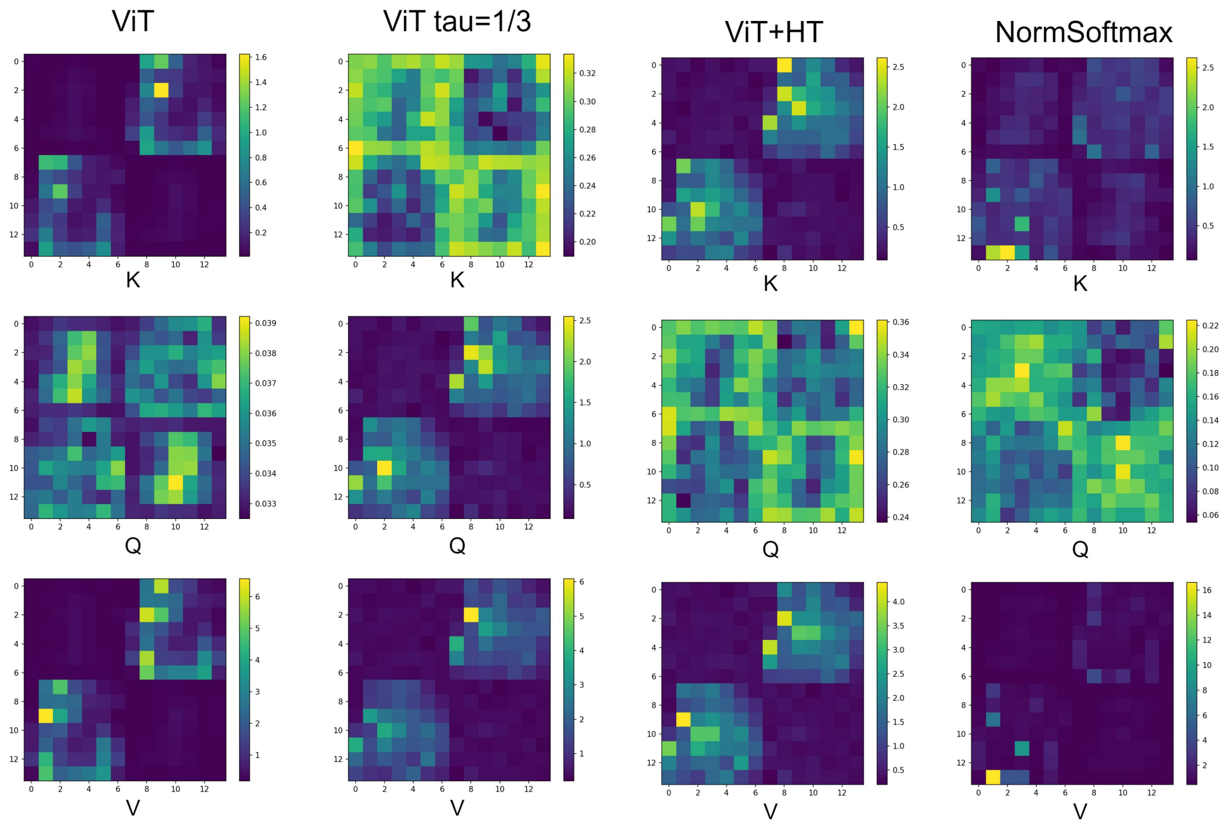


Figure 12: **Gradient norm for different image regions visualized for Q, K, V** at epoch 13 (Eureka-moment of NormSoftmax).

We can see in Tab. 1, that W_{QKV} grad scaling helps on the main dataset, but is very sensitive to the learning rate (Tab. 2). 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.

A.5. 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. 2) is high. The average Eureka-epoch is very late (Tab. 1 left) and we found more Warmup epochs lead to worse results on harder tasks (Tab. 1 right). The Eureka-ratio is sensitive to the learning rate schedule. We test 9 learning rate schedules for each method, (Tab. 11). Lower temperatures are less sensitive to the learning rate schedule (see Tab. 2).

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. 1) and does not help in solving transformer's learning issue.

A.6. 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. 3. 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 a Eureka-ratio of 4/4, but, as can be seen in Tab. 5, this architecture does not generalize to other datasets.

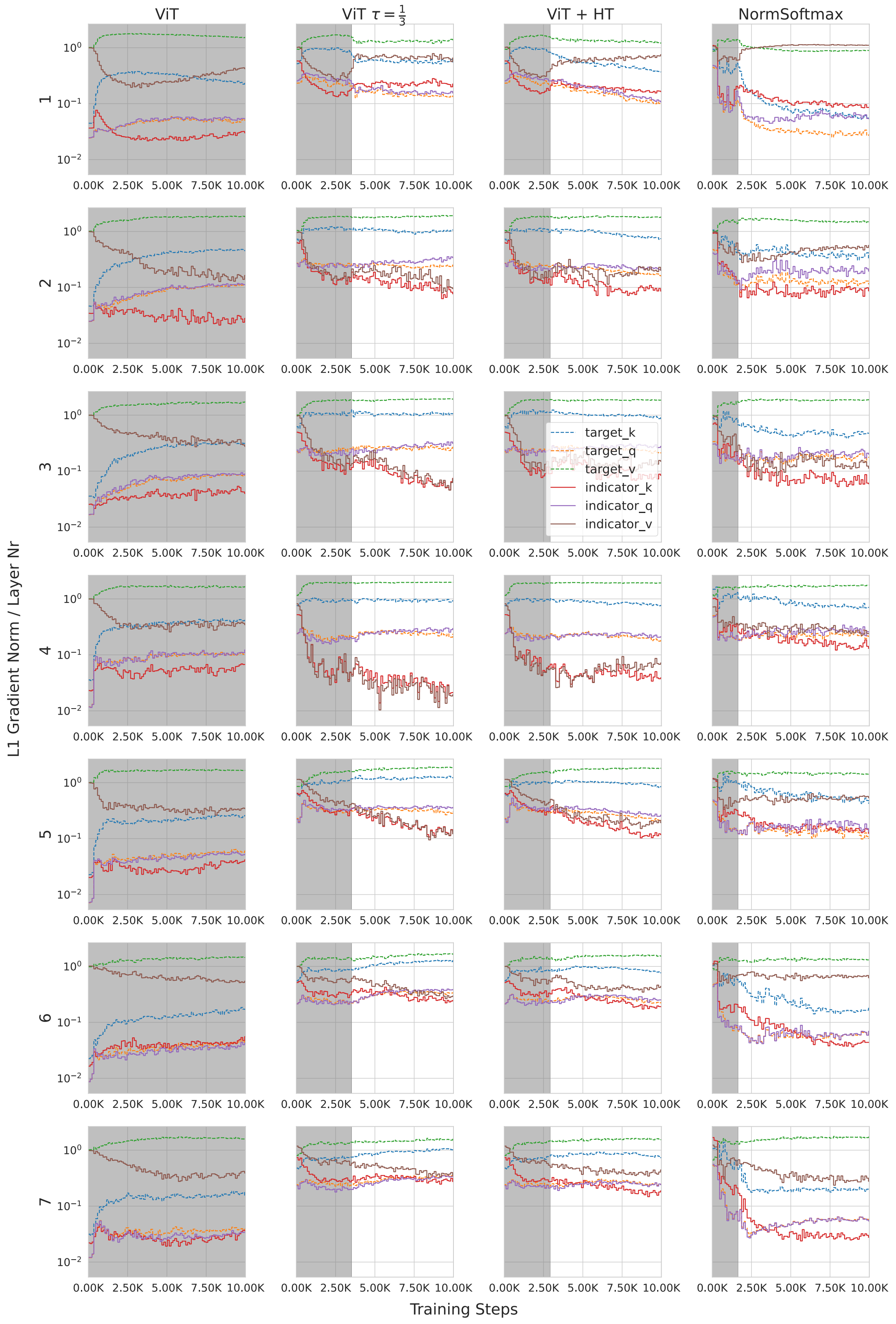


Figure 13: **L1 gradient norm separately for indicator and target tokens.** Indicator regions/features receive much less gradient than target regions. Gray region indicates steps before Eureka-moment.

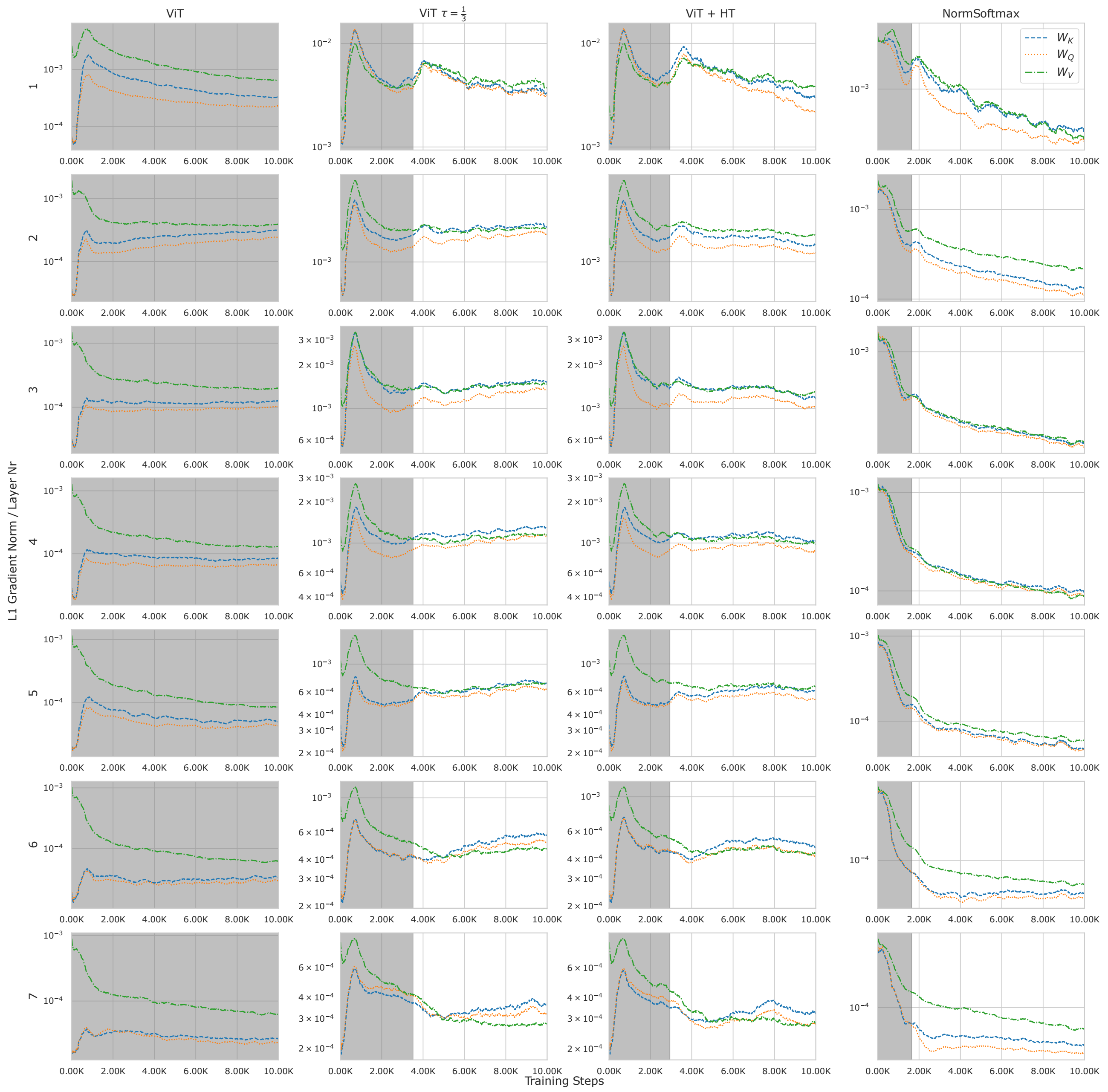


Figure 14: **L1 gradient norm plot for all layers, all models and entire training.** This is the complete version of Fig. 4. For ViT it can be seen that for all layers the gradient for W_V is significantly larger than for W_Q and W_K. For ViT+HT, ViT with $\tau = \frac{1}{3}$ and NormSoftmax the gradient 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.

A.7. 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

Table 3: **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. 5 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

probability of the top-right target to be the target location to 0.65. As can be seen in Fig. 15, the transformer initially learns the shortcut of always picking the more likely target.

A.8. 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. 4 & 5 we show the same tables including the standard deviation.

Additionally, Tabs. 4 & 5 also provide a comparison to a ResNet9.

Lastly, for Tab. 4 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 a Eureka-moment. Out of these 8 only 5 reach an accuracy higher than 95% of the ViT accuracy.

A.9. Description of and Results on More Datasets

In the following we report results on 5 more datasets. The datasets are depicted in Fig. 16.

Cifar task 1. A schematic for this task is shown in Fig. 16a. 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 MNIST digit. If not, the tasks is to classify the fashion MNIST sample. Results are reported in Tab. 6. 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.

Top if above. The task description is summarized in Fig. 16b. 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, only the 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 16b).

Same color decision task. The task is explained in Fig. 16b. 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. 8, this task is again very easy. Color seems to be easily accessible for ViT and ViT has little trouble to compare the indicator colors.

Color or fashion classification. This task is shown in Fig. 16d. 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.

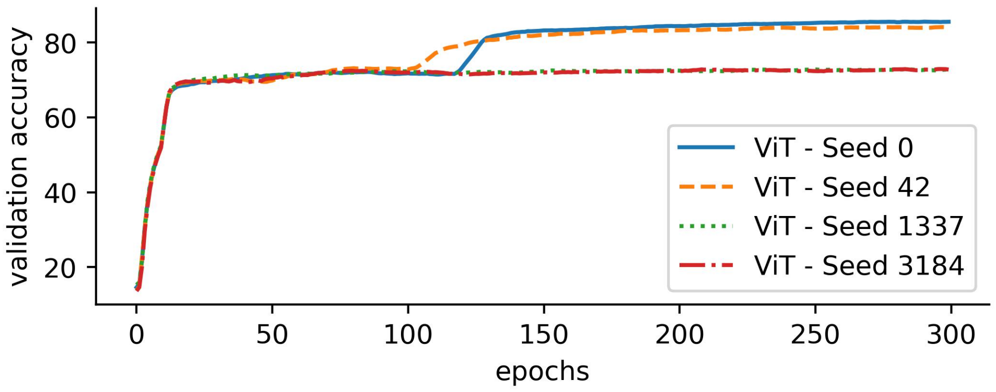


Figure 15: **Validation accuracy curves for "main dataset" with changed target probabilities.** ViT learns that one target is more likely than the other and learn to always pick this target, as can be seen by the higher plateau accuracy.

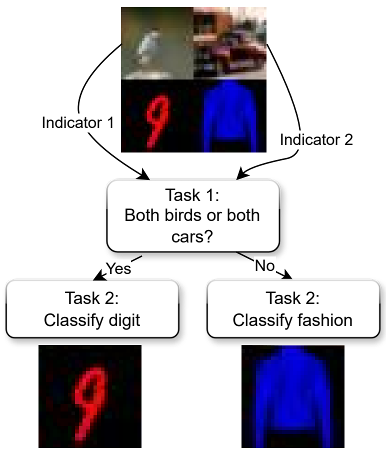
Table 4: **Main dataset – Comparison of proposed solutions and baselines.** This is a complete version of Tab. 1 including standard deviation, and speed improvements. For the **main dataset**, as described in Fig. 1a. **ER**: Eureka-ratio, **Acc.**: Accuracy, **Avg. EE.**: average Eureka-epoch. **% of 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

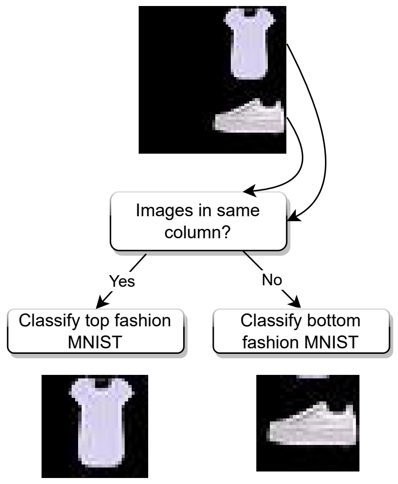
Digit grouping. Finally, we make the indicator task more difficult. We follow the same setting as for the *Main dataset*, as described in Fig. 1a. However, indicators are not sampled from digits 1 and 2, but from 1, 2, 3 and 4. Task 1 is to find out whether both indicators are smaller or both indicators 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. 10, 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.9.1. What makes a multi-step task hard to learn?

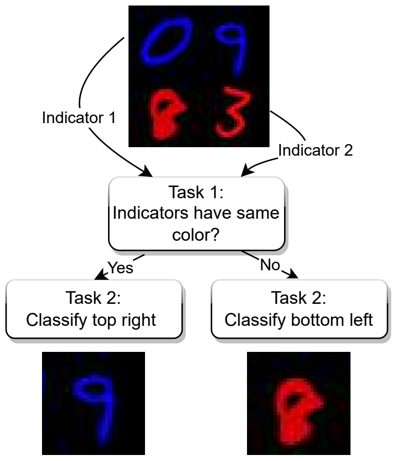
Based on the additional datasets described above we observe, that task 1 difficulty plays a major role in what makes a multi-step task difficult. For example the *Same color decision task* (Fig. 16c and Table 8) is significantly easier than our *Main task* (compare to Avg. Eureka-epoch in Table 4). We attribute this to the readily available feature (color) used to solve task 1. Similarly, the *Top if above* task (Fig. 16b and Table 7) is very easy to learn, as it removes one task 1 feature completely and relies solely on the position features. Making task 1 harder leads to much rarer and later Eureka-moments, as can be seen for the *No position task* (Fig. 7



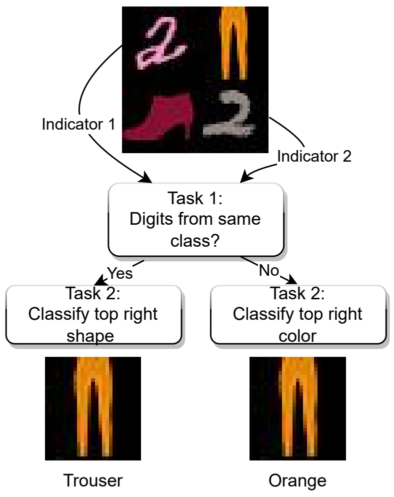
(a) **"Cifar task 1"**



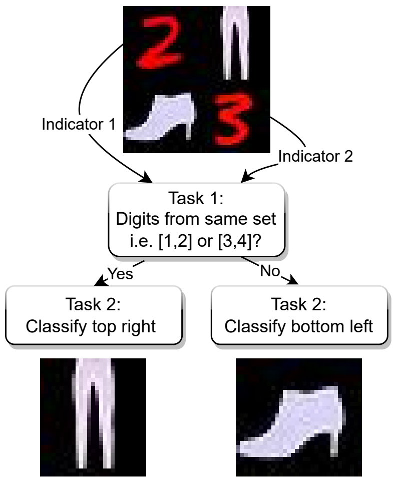
(b) **"Top if above"**



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



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



(e) **"Digit grouping"**

Figure 16: **Schematics for additional datasets.**

Table 5: **No Position Task – Comparison of proposed solutions and baselines.** For the **No position task**, as described in Fig. 7. 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**	$\mathbf{88.98 \pm 0.55}$	$\mathbf{228.67} \pm 08.22$
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	$\frac{1}{3} \to \sqrt{d_k}$	1/4	88.36 ± 0.00	242.00 ± 00.00
NormSoftmax + HT	$\frac{1}{3} \to \sqrt{d_k}$	1/4	90.63 ± 0.00	19.00 ± 00.00
ViT 6 heads, depth 4		0/0	-	-

Table 6: **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

Table 7: **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**	$\mathbf{91.47 \pm 0.13}$	13.75 ± 2.19
ViT	$\frac{1}{3}$	**4/4**	90.38 ± 0.13	9.5 ± 1.25
NormSoftmax	$\sqrt{d_k}$	**4/4**	90.84 ± 0.27	$\mathbf{9.25 \pm 1.09}$

Table 8: **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 ± 0.58	91.5 ± 60.04
ViT	$\frac{1}{3}$	**4/4**	$\mathbf{98.95 \pm 0.10}$	8.25 ± 1.48
NormSoftmax	$\sqrt{d_k}$	**4/4**	98.93 ± 0.03	$\mathbf{7.75 \pm 0.43}$

Table 9: **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 ± 0.25	12.00 ± 3.32
ViT	$\frac{1}{3}$	4/4	92.53 ± 0.40	20.75 ± 2.49
NormSoftmax	$\sqrt{d_k}$	4/4	$\mathbf{92.75 \pm 0.71}$	$\mathbf{11.25 \pm 0.83}$

and Table 5) and the *Digit grouping task* (Fig. 16e and Table 10). They make task 1 harder by removing the position information to locate and match indicators or introducing more variance to the appearance of indicators, respectively. We conjecture, that task 1 difficulty plays a major role in determining whether and when a Eureka-moment will happen. Task difficulty here seems to depend on the availability of the features required to solve task 1.

A.10. 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 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.

Table 10: **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	-	-

For ViT-S with NormSoftmax we observe a Eureka-Moment at epoch 131. The training and validation curves are shown in Fig. 17.

A.11. 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 $S = (s_1, \ldots, s_n)$. 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

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

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. Please see (Kurbiel, 2021) for a more detailed analysis.

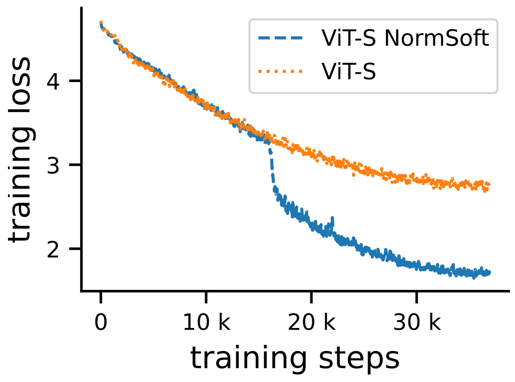
A.12. Information-Theoretic Probes

While our main analysis of accessibility of information in the main paper is entirely sufficient to support our claims, a more detailed look into accessibility of the information can complement our analysis. Our analysis tests for linear decodability of particular features from the learned representations. However, it does not measure how easily information is accessible. Voita & Titov (2020) propose to measure this "effort" by changing the amount of data needed to learn to extract the feature in question, referred to as online codelength. The codelength is measured in bits. The idea of online codelength is to measure the "availability" of a feature by limiting the data to learn the probe. Intuitively, if the representation has a high degree of order and the feature in question can be easily separated from other features, less data is required to learn a good representation. Thus, a short codelength corresponds to a feature being easily accessible. Following Voita & Titov (2020), we use 0.1%, 0.2%, 0.4%, $0,8\%$, 1.6%, 3.2%, 6.25%, $12,5\%$, 25% and 50% of the data to learn the probes. We train each probe for 70 epochs, e.g., for the first subset (0.1%) the linear probe sees the same 0.1% of the dataset 70 times.

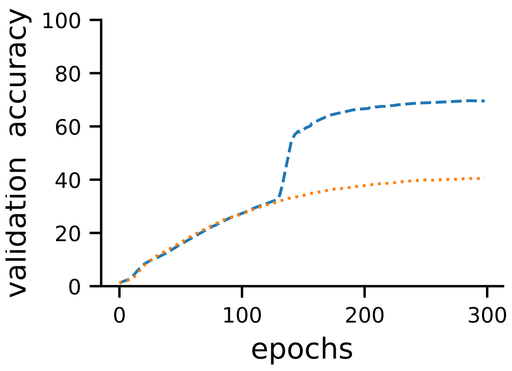
As can be seen in Fig. 18, the codelength necessary to extract the indicators does not differ much. Similarly as in Fig. 2, differences can be observed for the target location. For ViT (Fig. 18a) we see that the codelength for the target location is large for all types of representations across all layers. For ViT+HT (Fig. 18b) we can see that late in training the feature becomes easily accessible, in particular in deeper layers and in the full representation. In contrast, ViT $\tau = \frac{1}{3}$ (Fig. 18c) represents the feature already early in training and also the CLS token contains a better representation of the target location compared to ViT+HT.

A.13. 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.



(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 17: 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.

Linear probe results for Z for all layers. Fig. 19 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, 4 and 6 for the ViT with Eureka-moment (Fig. 19b) 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. 20 & 21 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, 4 and 6 for Q and K contain significantly more information about the indicators and target location for the ViT with Eureka-moment (compare Figs. 19b, 20b, 21b).

The linear probe results for V look very similar to those for Q and K (see Fig. 22).

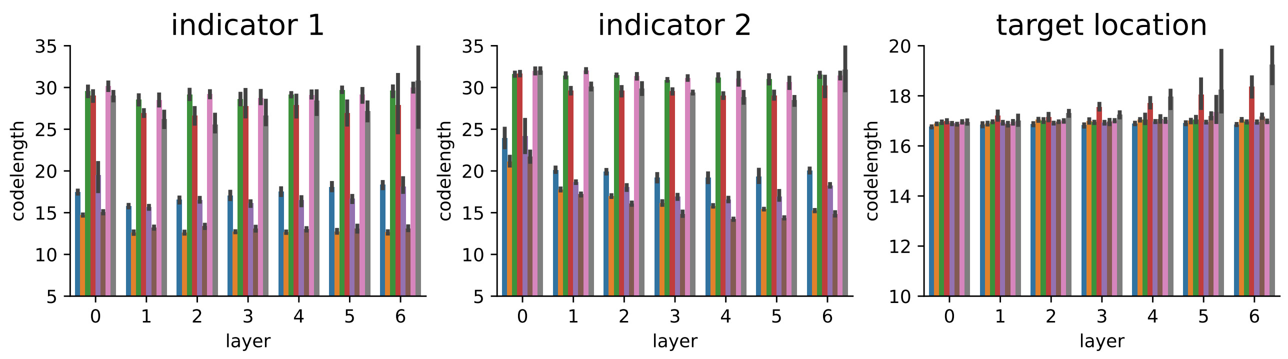
Linear probes from Z to targets. Last, we show the linear probe results when predicting the targets from Z. As can be seen, from the entire representation, for both ViT with Eureka-moment and ViT without Eureka-moment, target classes can be predicted with high accuracy. Differences can be observed when 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. 23).

A.14. Experimental setup — Vision Task

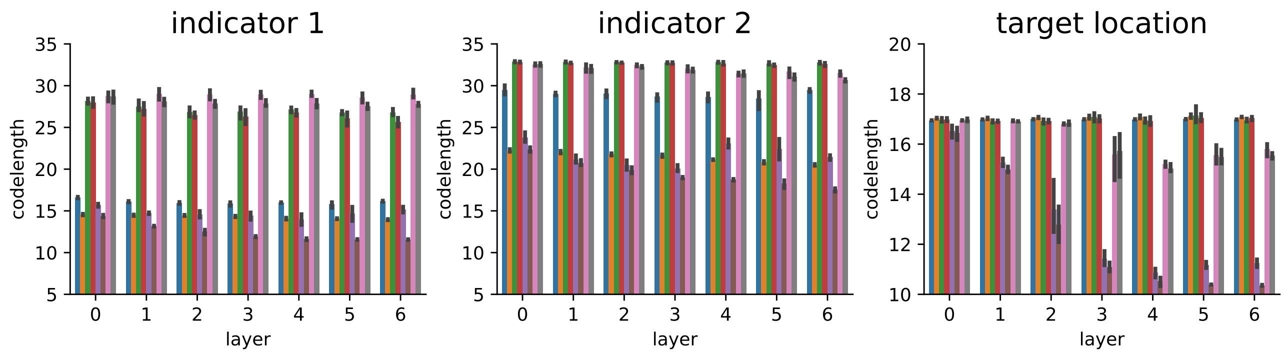
We mostly follow the DeiT (Touvron et al., 2021) training recipe without distillation. For optimization we use AdamW (Loshchilov & Hutter, 2019) with default values, i.e., $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Unless otherwise stated we Warmup the learning rate for 5 epochs from 10^{-6} to the maximum learning rate, use a weight decay of 0.05 and train for 300 epochs. We train all models 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 subsample (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 paragraph above. The different schedules can be seen in Tab. 11. 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.

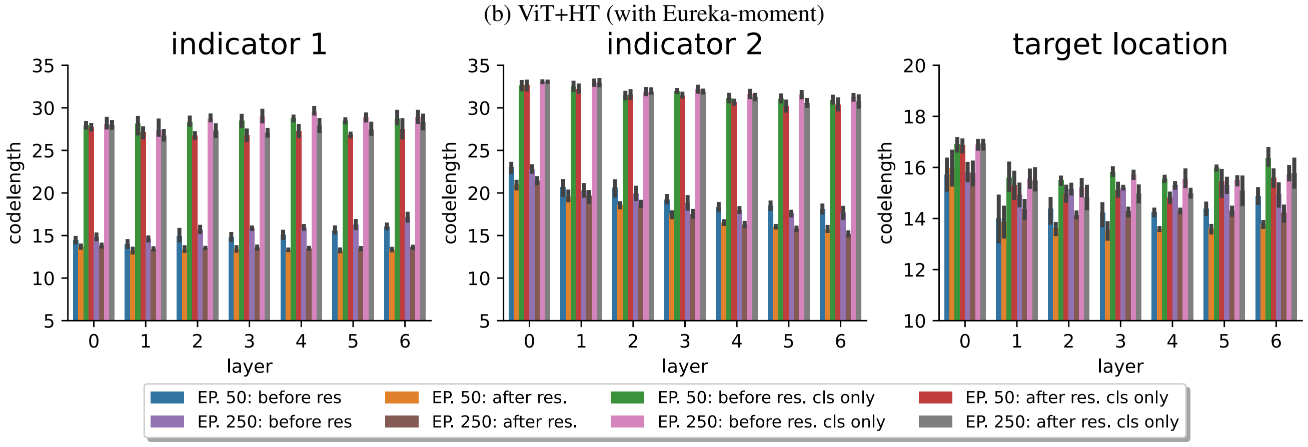
Info on taus: In initial experiments we tested for ViT 5 values for τ, $\tau = \frac{2}{3}$, $\tau = \frac{1}{2}$, $\tau = \frac{1}{3}$, $\tau = \frac{1}{4}$ and $\tau = \frac{1}{5}$. We found $\tau = \frac{1}{3}$, $\tau = \frac{1}{4}$ to work well and did not further optimize them for the different 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



(a) ViT (without Eureka-moment)



(b) ViT+HT (with Eureka-moment)



(c) ViT $\tau = \frac{1}{3}$ (with Eureka-moment)

Figure 18: **How easily is information accessible in different parts of the network for (a)** ViT without Eureka-moment, **(b)** ViT+HT (with Eureka-moment) and **(c)** ViT $\tau = \frac{1}{3}$ (with Eureka-moment). To measure how easily information can be extraceted we follow the online codelength approach of Voita & Titov (2020). In contrast to Fig. 2, lower is better for codelength.

Table 11: **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}

add very little to a deeper understanding of the Eureka-moments.

A.15. 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. (2023). 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.16. Experimental Setup – reasoning task

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.17. 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.18. 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 the 2nd of August 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.

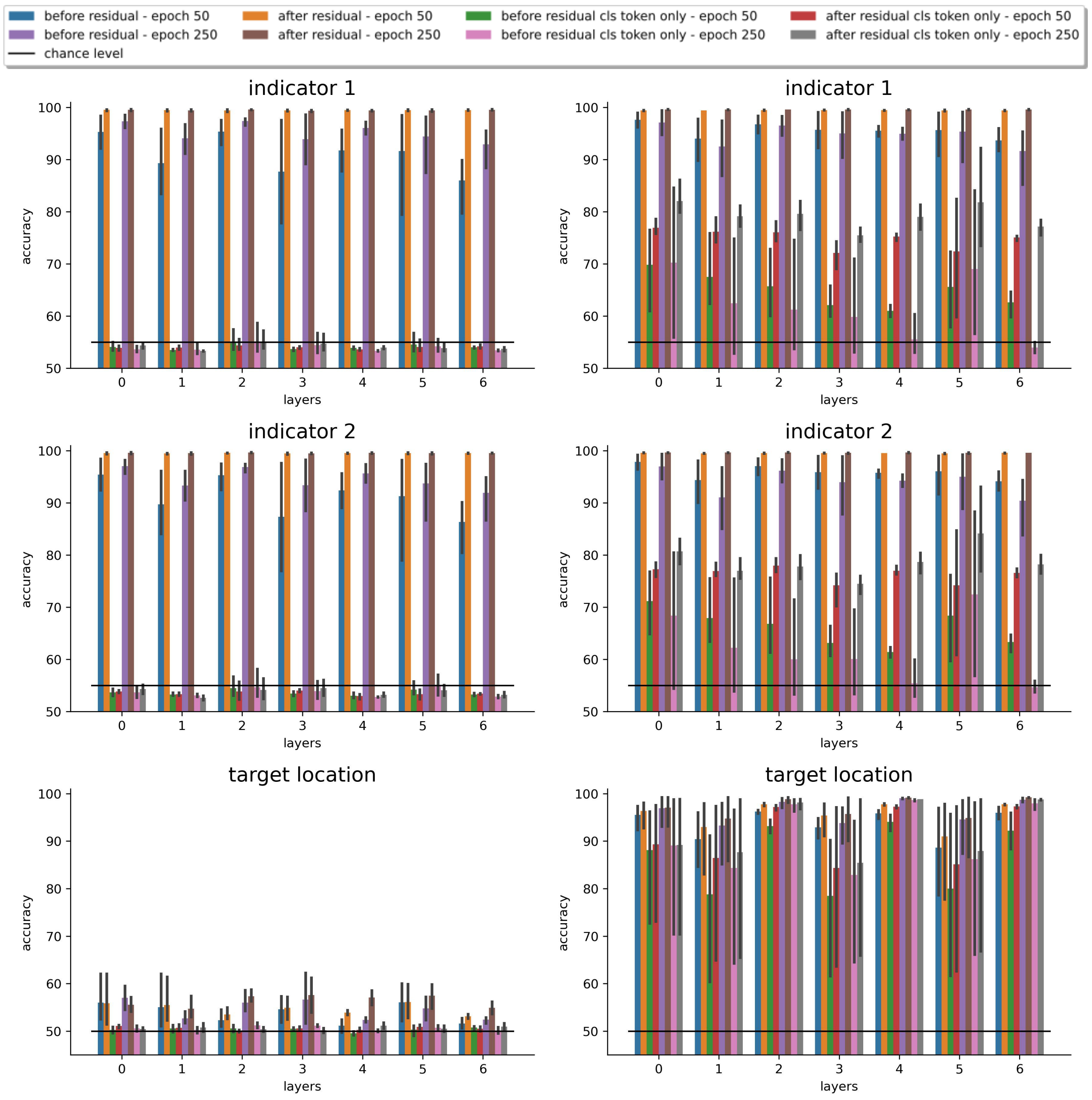
A.19. Experimental Setup – In-Context Learning

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 the experiments reported here. We ran each experiment with 4 different seeds - 0, 42, 1337 and 80085. We report the averaged results and show 2 instructive examples for each method.

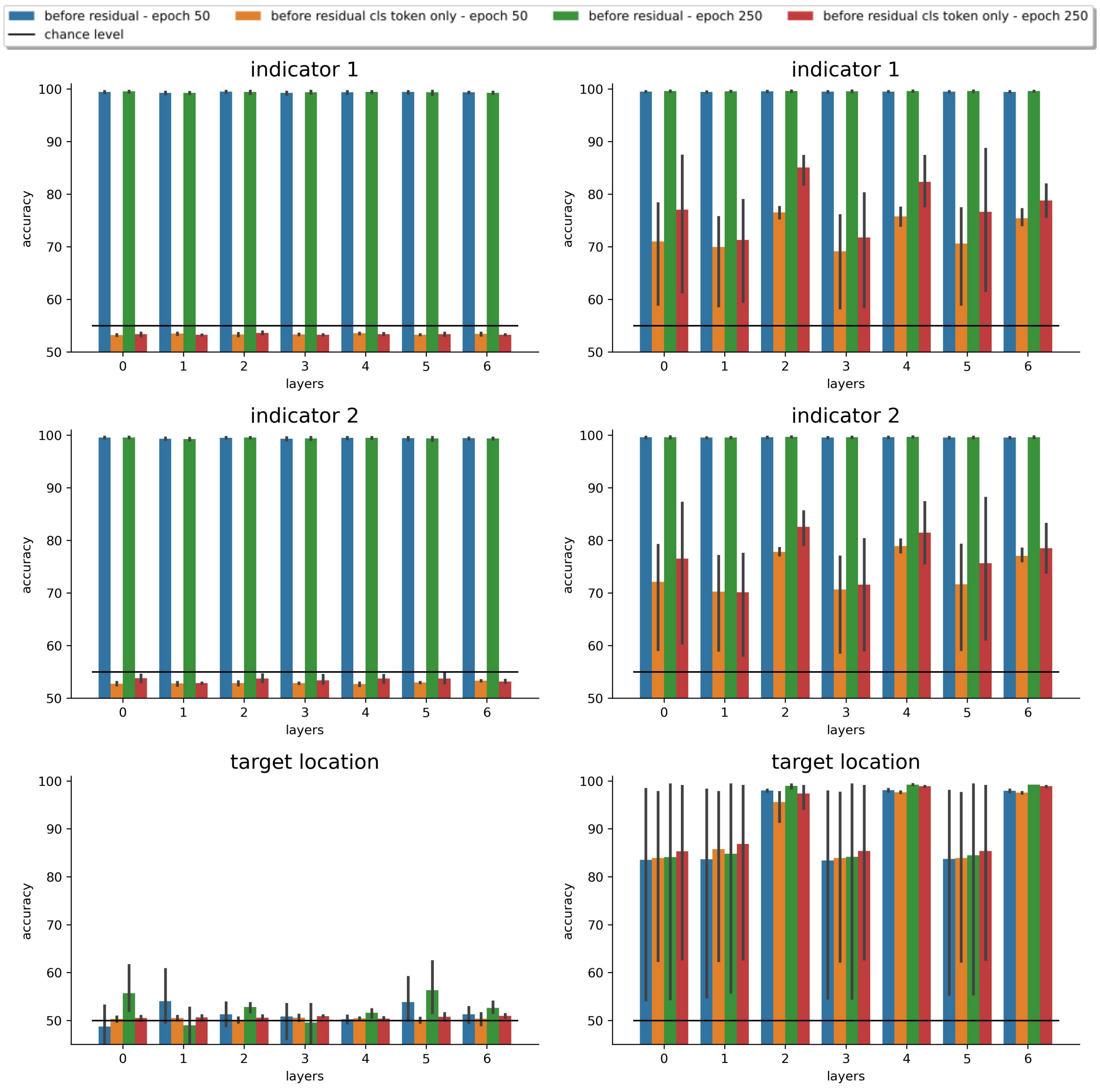
The in-context performance is reported on the 10 holdout classes not seen during training in the 2-way 4-shot few-shot evaluation setting. Here, the sequence consists only of 2 labels (0/1) with 4 images from each and we consider zero in-context performance for random chance level of 50 %. For all experiments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



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

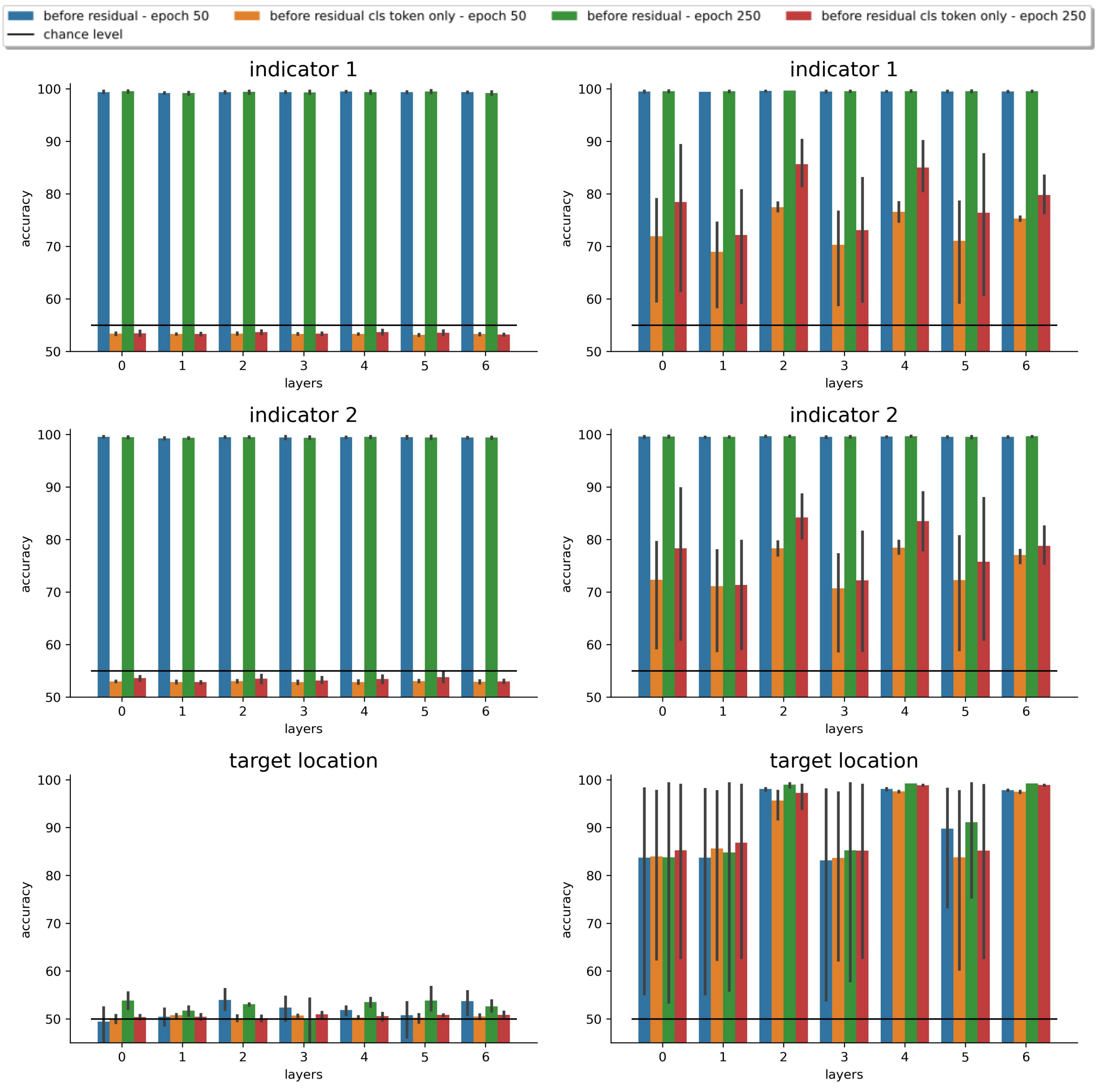
Figure 19: **Linear probe results for with Z as input (all layers).** This the complete version of Fig. 2. Additionally, we can observe in (b) that layers 2, 3 and 6 contain more target location information than other layers, indicating, that this information can be processed in these layers.



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

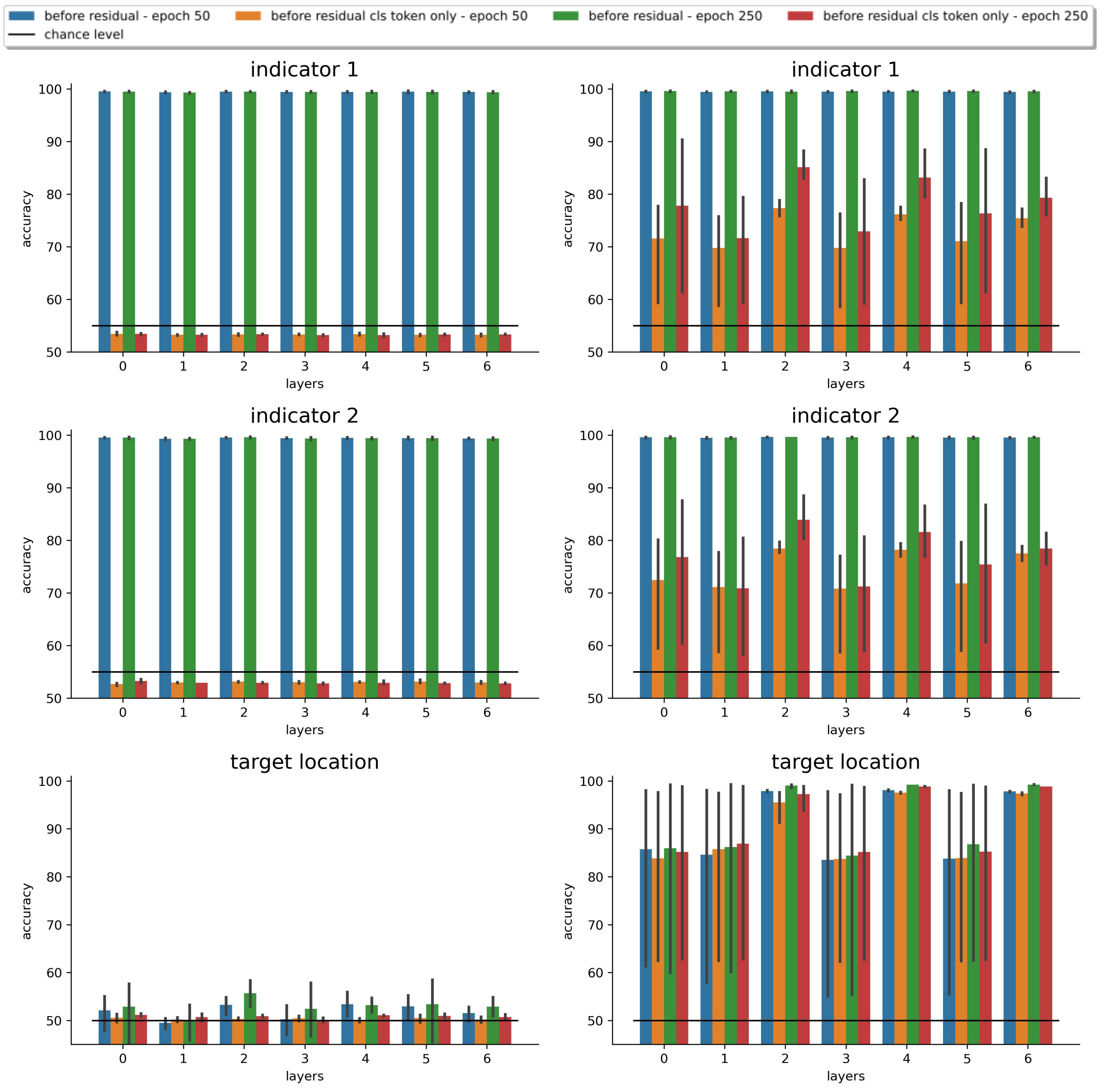
Figure 20: **Linear probes for Q.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

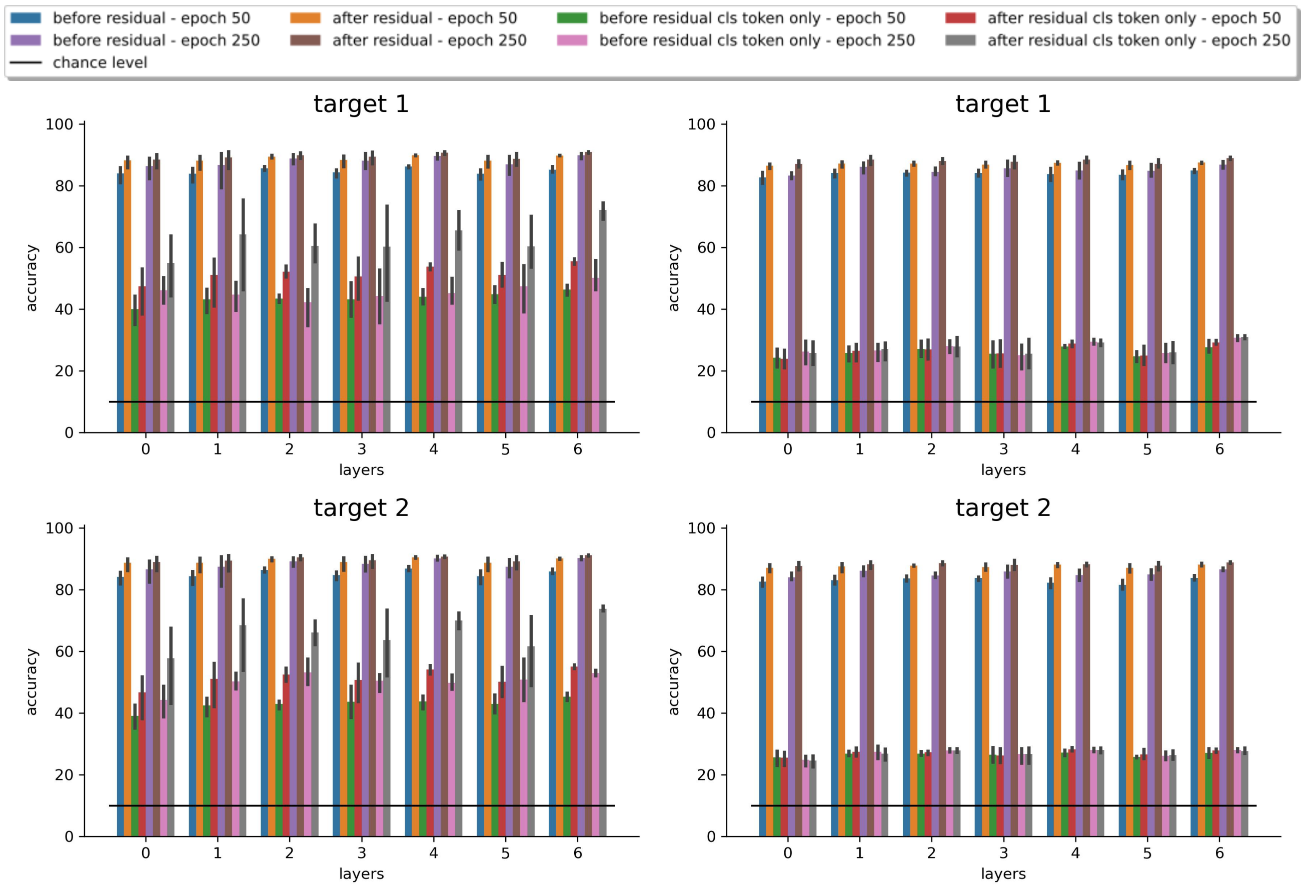
Figure 21: **Linear probes for K.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

Figure 22: **Linear probes for V.**



(a) ViT without Eureka-moment

(b) ViT with Eureka-moment

Figure 23: **Linear probes for Z with target classification.**