A Mechanistic Study of Transformers Training Dynamics

Ambroise Odonnat [1] **Wassim Bouaziz** [2] **Vivien Cabannes** [3]

Abstract

Large-scale pretraining of transformers has been central to the success of foundation models. However, the scale of those models limits our understanding of the mechanisms at play during optimization. In this work, we study the training dynamics of transformers in a controlled and interpretable setting. On the sparse modular addition task, we demonstrate that specialized attention circuits, called "*clustering heads*", can be implemented during gradient descent to solve the problem. Our experiments show that such pathways naturally emerge during training. By monitoring the evolution of tokens via a visual sandbox, we uncover a two-stage learning and the occurrences of loss spikes due to the high curvature of normalization layers. Our findings provide several insights into patterns observed in more practical settings, such as the pretraining of large language models.

facebookresearch/pal

1. Introduction

Transformers (Vaswani et al., 2017) have become the default backbone of state-of-the-art models in a wide range of domains, including natural language processing (NLP, Brown et al., 2020; Touvron et al., 2023a), computer vision (Caron et al., 2021; Dosovitskiy et al., 2021), time series forecasting (Ilbert et al., 2024; Nie et al., 2023), and mathematical reasoning (Comanici et al., 2025; Guo et al., 2025). The models' size and amount of training data have drastically increased as a by-product of the scaling hypothesis (Charton, 2024; Hoffmann et al., 2022; Kaplan et al., 2020). This makes the analysis and ablation studies on such systems challenging. As a result, our understanding of the *training dynamics* and causes of instability in large transformer models remains limited, with much of the knowledge in this area relying on heuristic tricks and techniques.

Mechanistic viewpoint. To address these limitations, we draw inspiration from the field of *mechanistic interpretability*, which aims to understand the internal mechanisms at play in large language models by reverse-engineering their computations (Elhage et al., 2021). A common approach in this framework is to study the pathways, or *circuits*, of a neural network that transform an input into a given output. This viewpoint has been used to study the emerging capabilities of LLMs with in-context learning (ICL) (Edelman et al., 2024; Olsson et al., 2022), chain-of-thought (Cabannes et al., 2024a), and memorization in transformers (Bietti et al., 2023). In line with these studies, we aim to design a controlled setup to decipher the training of transformers without potential confounding factors. Our focus is on obtaining a finer-grained understanding of the evolution of sequence embeddings during the training of a single transformer block. Training the whole architecture could then be considered as the joint training of several blocks, with the inputs of a given block being the output of its preceding block.

Methodology. Motivated by the fact that many NLP and reasoning tasks involve invariances in the data (Robinson & Wingate, 2023; Singh et al., 2025), we consider the *sparse modular addition* problem, which requires a correct handling of invariances to be solved and ensures non-trivial optimization dynamics (Odonnat et al., 2025). Similar mathematical tasks have been used in prior works for their clear structure and ease to observe the pathways implemented (Ameisen et al., 2025). We frame the problem as a supervised question-answering task and train small transformers on it. This enables us to fully characterize how transformers learn to solve the task, which is a step towards opening the black box (Charton, 2024; Koh & Liang, 2017). To analyze the training dynamics on this task, we view sequences of tokens as a system of interacting particles and design a visual sandbox to observe their evolution during training. It allows us to identify and analyze multiple phenomena observed in more practical settings. This showcases the benefits of our setup in studying, at a small scale, the training behavior of bigger models.

[1]Inria, Univ. Rennes 2 [2]Mistral AI; work done while at Meta. [3]FAIR at Meta. Correspondence to: Ambroise Odonnat <ambroise.odonnat@gmail.com>.

Proceedings of the 1st International Conference on Machine Learning Mechanistic Interpretability Workshop, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

Our contributions. The code is open-sourced at https://github.com/facebookresearch/pal and our main contributions can be summarized as follows:

1. **Theoretical analysis:** We frame the *sparse modular addition* problem as a question-answering task and demonstrate that transformers trained with gradient descent implement specific pathways, dubbed *clustering heads*, to solve it by capturing the invariances in the data.
2. **Visual sandbox:** We design a visual sandbox to observe the evolution of tokens, seen as interacting particles, during training and use it to study the implementation of clustering heads.
3. **Training dynamics:** We observe that such circuits emerge after a two-stage learning, akin to the in-context learning settings (Olsson et al., 2022). We observe the occurrence of loss spikes during training and suggest potential strategies for mitigation, relating our findings to the pretraining of large language models.

We hope our setup can serve as a viable testbed to study training stability on a small scale to gain intuition for larger-scale settings.

2. Synthetic setup

Let $n, m \in \mathbb{N}$ be nonnegative integers. Throughout the paper, we use the notation $[n] = \{1, \dots, n\}$. The Euclidean norm of a vector $x \in \mathbb{R}^n$ is denoted by $\|x\|$. By abuse of notation, we denote by $\|A\|$ the entry-wise ℓ_2-norm of a matrix $A \in \mathbb{R}^{n\times m}$ which amounts to computing the Euclidean norm of A flattened. The probability simplex over $\mathbb{R}^n$ is denoted by Δ_n. The total variation distance between two probability distributions P_1, P_2 is denoted by $d_{\mathrm{TV}}(P_1, P_2)$.

Sparse modular addition. This problem is inspired by the sparse parity problem (Abbe et al., 2023; Barak et al., 2023) and the modular addition problem (Nanda et al., 2023; Power et al., 2022) and is characterized by the following parameters:

- Input length $N \in \mathbb{N}$,
- Vocabulary size $p \in \mathbb{N}$,
- Sparsity index $k \in [N]$, and a set of indices $I \subset [N]$ with cardinality $|I| = k$.

Our default configuration is set to $N = 12$, $k = 5$, and $p \in \{2, 3\}$. Without loss of generality, we assume $I = [k] := \{1, 2, \dots, k\}$. Inputs are sequences of N tokens x_t in the ring $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z} \simeq [p]$, and the corresponding targets are the sum of the first k terms modulo p. Formally, we aim to learn a mapping:

$$f^* : \begin{array}{ccc} (\mathbb{F}_p)^N & \to & \mathbb{F}_p \\ x = (x_1, \dots, x_N) & \mapsto & \sum_{t\in[k]} x_t. \end{array} \qquad \text{(SMA)}$$

This mapping defines a deterministic conditional distribution linking input and output data through the formula $p(y|x) = \mathbf{1}_{\{f^*(x)=y\}}$, and the training samples follow $x, y \sim p(\cdot)$. The (SMA) problem can be framed as a supervised question-answering task with tokens embedded in a vocabulary space $\mathcal{V} = [p]$ of size p. Input questions are represented as sequences of tokens. They are randomly sampled and used to train sequence-to-sequence transformers by minimizing the cross-entropy loss between the model's predictions and the correct answer, that is, the sum of the k first terms of the inputs modulo p. While sparse modular addition might seem simple to solve for a human, this is a hard task to solve for neural networks, which have been shown to struggle on similar arithmetic tasks (Charton, 2024; Dziri et al., 2023; Lee et al., 2024; Saxton et al., 2019), involving non-trivial learning dynamics (Odonnat et al., 2025). In particular, the model needs to capture the invariance in the data and correctly focus on the informative, i.e, *non-spurious*, tokens x_i for $i \in [k]$.

Transformer architecture. To conduct our analysis, we consider a simplified transformer model containing all the key elements of its larger counterparts commonly used in question-answering tasks (Raffel et al., 2023; Vaswani et al., 2017). Notably, it relies on a cross-attention module, a feedforward network, residual connections, normalization layers, and weight tying (Press & Wolf, 2017b). Following modern models (Llama Team, 2024), we use a pre-norm configuration (Xiong et al., 2020) and the RMSNorm (Zhang & Sennrich, 2019), which is similar to LayerNorm (Ba et al., 2016) but computationally more efficient. To ease notations, we incorporate the RMSNorm trainable parameters into the attention and feedforward weights following Castin et al. (2024), which does not change the expressivity or optimization of the model. We recall below how tokens are processed through the model. Since the sparse modular addition problem is inherently discrete, we first embed each token in dimension d with both semantic and positional information before normalizing it on the unit sphere $\mathbb{S}^d$. Formally, given a learnable token embedding $E : \mathbb{F}_p \to \mathbb{R}^d$, and a learnable position embedding $P : [N] \to \mathbb{R}^d$, a sentence in token space is lifted to a sentence in embedding space $z \in \mathbb{R}^{d\times N}$ through the following operation that applies to each x_t for $t \in [N]$

$$z_t := Z(x_t, t) := \frac{E(x_t) + P(t)}{\|E(x_t) + P(t)\|}. \quad \text{(token embedding)}$$

Then, a cross-attention mechanism is applied to the sequence of tokens z, leading to a single sentence embedding $\xi \in \mathbb{R}^d$. The sentence embedding can be written in matrix form as

$$\xi := (Vz)\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \in \mathbb{R}^d, \quad \text{(sentence embedding)}$$

where we follow Zhang et al. (2024a) and merge the query and value into a vector $q \in \mathbb{R}^d$ to avoid cumbersome no-

tations. We omit the key matrix, which would act as extra parameters that do not increase the expressivity of our model since z can be set to anything thanks to E and P. The sentence embeddings are then passed through a feedforward network with hidden dimension h (typically taken as $h = 4d$) implemented as an MLP with two layers $U \in \mathbb{R}^{d\times h}$ and $W \in \mathbb{R}^{h\times d}$, followed by a residual connection. It implements the following transformation:

$$\psi := \xi + U\sigma\left(\frac{W\xi}{\|\xi\|}\right) \in \mathbb{R}^d \qquad \text{(sentence transform)}$$

with $\sigma\colon x \mapsto x\varphi(x)$ the GeLU activation (Hendrycks & Gimpel, 2023). W receives the sentence embeddings before projecting them into a higher dimension, while U reassembles the embeddings, projecting them back to the original dimension. As such, the feedforward can be equivalently seen as a combination of $h \in \mathbb{N}$ "receptors" weights $w_i \in \mathbb{R}^d$ and h "assemblers" vectors $u_i \in \mathbb{R}^d$ for $i \in [h]$, with $(u_i)_{i=1}^h$ the columns of U and $(w_i^\top)_{i=1}^h$ the rows of W. The final sentence vector ψ is decoded back to token space based on how it aligns with the respective token embeddings, which amounts to using weight-tying in the implementation (Press & Wolf, 2017a). More specifically, the model's logits write

$$\zeta = (E(v)^\top \psi)_{v\in\mathbb{F}_p} \in \mathbb{R}^p. \qquad \text{(logits)}$$

Abstracting all the learnable weights into a single vector θ, the probability distribution over answers in $[p]$ writes $\hat{p}_\theta(\cdot|x) \in \Delta_n$ and is obtained by applying a softmax layer on top of the logits. For any $j \in [p]$, $\hat{p}_\theta(y = j|x)$ is simply the j-th entry of the vector $\hat{p}_\theta(\cdot|x)$. The model is optimized by minimizing the cross-entropy loss between its prediction and the correct answer. The loss writes

$$\mathcal{L}(\theta) := \mathbb{E}_{x,y}\left[-\log(\hat{p}_\theta(y|x))\right], \qquad \text{(loss)}$$

where the expectation is taken over the distribution of training samples (x, y).

3. Learning clustering heads by gradient descent

In this section, we study how transformers learn to solve the (SMA) problem with gradient descent. We describe a natural circuit, dubbed *clustering head*, that, if correctly implemented, enables the transformer to solve the task, and we demonstrate that such a circuit can be efficiently learned by gradient descent.

Circuit perspective. In the sparse modular addition problem, the output y is invariant to two sets of transformations of the inputs $x = (x_1, \dots, x_N)$.

- "Permutation invariance:" y does not depend on permutation of non-spurious tokens, i.e. for any permutation σ of $\{1, \dots, k\}$,

$$f^*(x) = f^*((x_{\sigma(1)}, \dots, x_{\sigma(k)}, x_{k+1}, \dots, x_N)).$$

- "Suffix invariance:" y does not depend on the suffix $(x_{k+1}, \dots, x_N)$.

These two sets of invariants can be easily enforced by the embedding layer. Any architecture where the position embeddings satisfy $P(t) = P(1)$ for $t \in [k]$, will be permutation invariant, meaning that its output will be invariant to permutations of the non-spurious tokens. Similarly, suffix invariance can be enforced by ensuring that the query vector primarily aligns with the token embeddings z_t for $t \in [k]$, allowing the sequence embedding ξ to be invariant to the sequence suffix. Such a construction would yield $\binom{k+p-1}{k}$ clusters of sequence embeddings,[1] which the feedforward layer could scatter into as many decision regions to map each sequence embedding ξ to the correct output class $y \in [p]$. The pathway that realizes such operations is called a *clustering head*. Clustering heads define one way for the transformer to handle invariances: the model first attends to the non-spurious tokens $(x_i)_{i=1}^k$, then regroups them by value to determine the parity of each subgroup and, finally, of the whole sequence. Although this is not the only possible circuit, it resonates with recent studies (Lindsey et al., 2025; Odonnat et al., 2025), which showed that transformers and notably large language models split hard problems into simpler sub-problems, for instance, to compute a complex arithmetic task.

Learning clustering heads. A natural question is whether *clustering heads* can be implemented in practice by a transformer trained via gradient descent, which aims at finding local minimizers of the loss $\mathcal{L}$. Such minimizers are stationary points, i.e., solutions of $\nabla\mathcal{L} = 0$. Akin to Nichani et al. (2024), we focus on the attention and feedforward dynamics, assuming fixed embedding weights. In the following lemma, we provide a closed-form expression of the gradients (the full statement is deferred to Lemma 2).

Lemma 1 (Closed-form gradients)**.** *Let $m = d(1+d+2h)$. The trainable parameters of the transformer write $\theta \in \mathbb{R}^m$, encompassing the attention and feedforward matrices. Denoting by $\hat{p}_\theta(\cdot|x)$ and $p(\cdot|x)$ the model's prediction and ground-truth for an input sequence $x = (x_1, \dots, x_N)$, we have*

$$\nabla\mathcal{L}(\theta) = \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y = j|x) - p(y = j|x)) \cdot \nabla_\theta \zeta_j].$$

[1] This number corresponds to the number of ways to split k into p buckets, which is also the number of stars and bars configurations with k stars and $p - 1$ bars.

Computing the loss gradient involves reweighting each logit's gradient by the corresponding discrepancy between the model's prediction $\hat{p}_\theta$ and the ground-truth p. Lemma 1 allows us to characterize the stationary points of the loss and show that the parameters of transformers that implement clustering heads are stationary points (the full statement is deferred to Proposition 3).

Proposition 1 (Learning clustering heads, informal)**.** *Let θ be the parameters of a transformer model that implements a clustering head. Then, θ is a stationary point of the loss, that is $\nabla\mathcal{L}(\theta) = 0$.*

Proposition 1 implies that clustering heads can be learned by gradient descent. We note that it does not provide guarantees on the convergence to such a circuit (we will see in the next section that, in practice, clustering heads are implemented by transformers). Further theoretical treatment to characterize the stationary points as local, global minima, or saddle points is an interesting direction left for future work. In the following, we focus on the training dynamics during gradient descent.

4. Visualizing the training dynamics

In this section, we see tokens as interacting particles, and design a visual sandbox to observe their evolution during training. It allows us to study the practical implementation of clustering heads, noticing a two-stage learning and the occurrence of loss spikes during training, and further relate our findings to the pretraining of large language models. The implementation details are given in Section A.

4.1. Visual sandbox

Tokens processed by transformers interact through the attention module, which allows us to study their evolution as a dynamic system of interacting particles controlled by an ordinary differential equation (Geshkovski et al., 2023b, 2025; Lu et al., 2019). We adopt this macroscopic viewpoint and follow the mechanistic viewpoint (Olah, 2023) by embedding all computations in the plane by taking $d = 2$ and tracking several key aspects during training, such as:

Position embeddings. We visualize them as a point cloud with the "spurious" embeddings $P(t)$ for $t \notin [k]$ represented by squares, and the "non-spurious" ones, the $P(t)$ for $t \in [k]$, by circles. Ideally, the transformer would collapse all spurious (resp. non-spurious) position embeddings into a single point, learning invariance of y to sentence suffixes (resp. to permutation of non-spurious tokens positions).

Token embeddings. We visualize the token embeddings $Z(x,t) \propto E(x) + P(t)$ for $(x,t) \in \mathbb{F}_p \times [N]$. We maintain the same circle and square distinction and use the same color for both $E(x) + P(t)$ and $E(x') + P(t)$, for $x' \in \mathbb{F}_p \setminus \{x\}$. On the normalized plot, we also plot the query q as an arrow in $\mathbb{R}^2$, helping us understand where the attention module learns to focus.

Attention map. We visualize the concatenation of attention vectors for different sentences. It enables us to follow the change in activation patterns, even though it is a pure function of the normed embedding visualization.

Value transform. We visualize $VZ(x,t)$ as a point cloud. This allows us to understand how sequence embeddings are built and how the value matrix may overcome faulty attention patterns.

Sequence embeddings and transforms. We visualize the sequence embeddings ξ (or their transforms ζ) for a set of predefined sentences. These sentences are built by iterating over prefixes $(x_t)_{t\in[k]}$ and suffixes $(x_t)_{t\notin[k]}$. Sentences that share the same prefix have the same color. Sentences whose prefixes are equivalent up to a token position permutation have similar colors. Squares, circles, and triangles are used to distinguish between the classes of the sentences. The sequence embeddings visualization is a direct function of the value transformation and the normalized embeddings.

Transform level lines. We visualize the mapping from sentence embedding ξ to their associated learned probabilities $\hat{p}_\theta(y|x)$. We also plot the sentence embedding on the same plot to better understand the level line changes.

MLP receptors and assemblers. We visualize the $w_i \in \mathbb{R}^2$ (and $u_i \in \mathbb{R}^2$) that define the MLP transform as a point cloud in $\mathbb{R}^2$. A consistent color scheme is used to link receptors with the corresponding assemblers.

Loss and accuracy. We visualize the current train loss, test loss, and accuracy. It is interesting to put those classical quantities in relation to the other visualizations to better understand the loss spikes, loss plateaus, and phase transitions.

4.2. Clustering heads implementation

The *clustering head* described in Section 3 can be visualized in Fig. 1, with the model focusing on the informative tokens to regroup them by values with the attention module before computing the parity of the sequence with the feedforward. In the previous section, we showed that these pathways could be implemented by gradient descent. Since transformers can perform similar operations in many different ways, we observe in practice variations from the idealized circuit, categorized below.

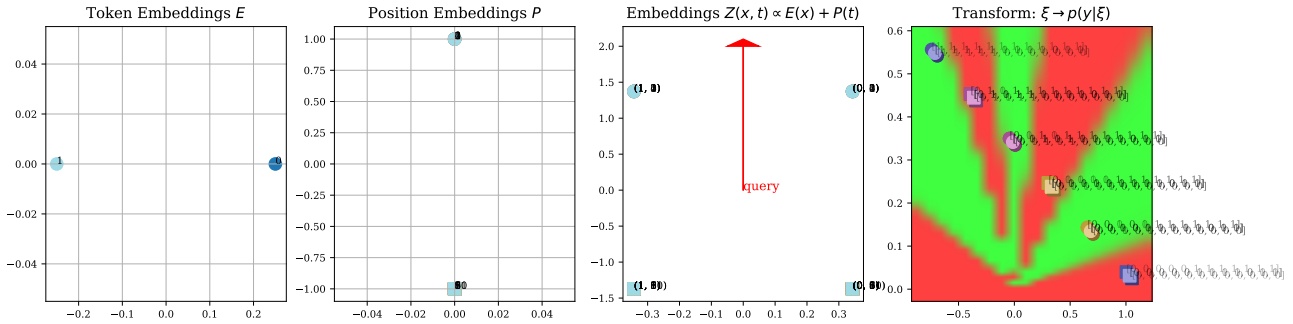


Figure 1. **Clustering Head**. Implementation of an idealized circuit that captures the invariants of the problem. *From left to right*: we first plot the two token embeddings $E(x)$ for $x \in [p]$ with $p = 2$. We then plot the twelve position embeddings $P(t)$. The spurious (resp. non-spurious) positions are represented with squares (resp. circles), and all collapse to a single point. The normalized embeddings $Z(x,t)$ are plotted in the third frame, annotated with (x,t), with the query vector q represented as a red arrow. This arrow points in the direction of the embedding $Z(x,t)$ for $t \in [k]$, allowing the attention mechanism to focus exclusively on non-spurious tokens. Finally, we plot the sequence embeddings ξ for some (x_t), and the output of the feedforward transform. The embeddings are clustered and respect the invariance, and the feedforward layer can then map each cluster to a probability vector $\hat{p}_\theta$. Each output class is represented by a color.

Faulty attention corrected by value. In many instances, we observe that the attention scores are not fully concentrated on the first five tokens but are compensated by the value matrix, which effectively collapses the embeddings of the spurious tokens that are attended to. This allows the sequence embeddings to remain invariant to the suffix of the sequence, as can be seen in Fig. 2.

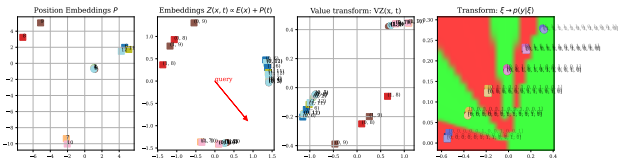


Figure 2. Faulty attention corrected by value collapse.

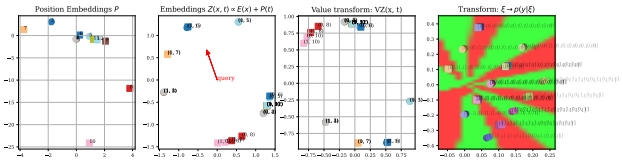


Figure 3. Embeddings having only learned some invariants.

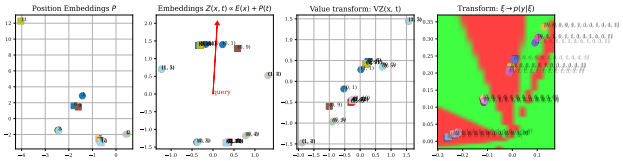


Figure 4. Fuzzy construction.

Partially learned invariants. Frequently, the sequence embeddings have not fully learned all the suffix and prefix invariants, resulting in more than six clusters of sequence embeddings[2]. Specifically, Fig. 3 illustrates a scenario where the sequence embeddings are not invariant to the value x_6, as evidenced by the positions of the blue squares $(0, 6)$ and $(1, 6)$ on the plot. They also lack invariance to permutations of the token in the first or fifth positions with another of the non-spurious tokens. This results in a sequence embedding that presents more clusters than the idealized model, leading to a greater number of connected decision regions in the feedforward layer.

Fuzzy constructions. Occasionally, we encounter fuzzy constructions where the sequence embeddings are clustered according to unconventional patterns that nonetheless generalize to unseen data. Such a construction is presented in Fig. 4.

[2]This corresponds to $\binom{k+p-1}{k}$ for $p = 2, k = 5$.

From initialization to transferability of circuits. We observe in Fig. 5 the strong similarity between the attention patterns at the start and at the end of the training, indicating that initialization plays a crucial role in the concrete realizations of clustering heads. We notice that the transformer learns to focus on the 1 among the non-spurious tokens, before counting them and deducing the number of 0 to make its final prediction. Among the lessons of training very large models is that some sources of data may facilitate the learning of certain skills (Cabannes et al., 2024a; Llama Team, 2024; Touvron et al., 2023b), suggesting the transferability of circuits. We illustrate this phenomenon in Fig. 6, which shows that training first a model on a simpler task (e.g., $p = 2$) helps it better solve a more challenging one (e.g., $p = 3$), as can be seen in along with the impact of initialization (more details in Section A.2).

Connectivity of neurons. Motivated by recent works studying the sparsity of transformers' activations (Li et al., 2023a; Mirzadeh et al., 2024), we investigate how neurons

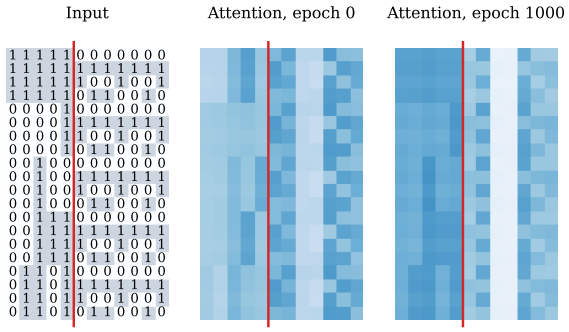


Figure 5. **Impact of the initialization.** For each input sequence (**left**), we plot the attention weights at the start of training (**middle**) and at the end of training (**right**); the darker, the higher. Final attention maps correlate with the original one, illustrating the impact of the original configuration on the implemented circuit.

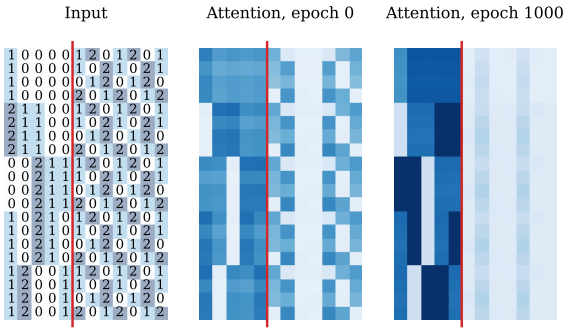


Figure 6. **Transferability of circuits**. This study is akin to Fig. 5 when the model is first pre-trained with $p = 2$ and then finetuned with $p = 3$. This illustrates that the final variations of the original circuit depend on the original weight configuration. We notice that the model learns better to focus on the non-spurious first 5 tokens.

connect to implement *clustering heads*. We show in Fig. 7 that the models that successfully solve the task have denser activation, suggesting that many neurons are necessary to correctly implement a clustering head and thus capturing the invariance of the task (more details in Section A.1).

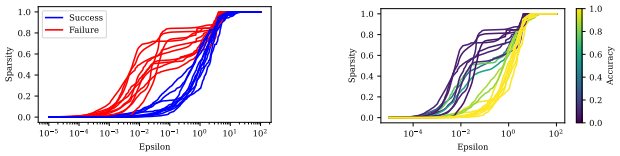


Figure 7. Connection between performance and sparsity. Since the activation is the GELU that does not map all the negative entries to 0 (contrary to the ReLU), we define the activation sparsity as the percentage of entries with values lower than $\varepsilon > 0$. We display the evolution of the activation sparsity of 20 trained models with $\varepsilon \in [10^{-5}, 10^{2}]$. **Left:** Successful models (i.e., with test accuracy above 0.9) in blue have less sparse activation than failed models in red. **Right:** The color indicates the models' test accuracy (the lighter, the better). The performance increases as the activation sparsity decreases.

4.3. Two-stage learning: escaping saddle points

To better understand how clustering heads emerge, we observe the evolution of the tokens along with the loss landscape during training. We notice a two-stage learning process with the model first clustering the sequence embeddings, before fitting a classifier on top. As displayed in Fig. 8 (right), the loss curves present two drops, corresponding to the learning of different parts of the network. Our findings resonate with theoretical arguments in the deep learning theory literature on saddle point (Chi et al., 2019a; Dauphin et al., 2014; Du et al., 2017) and stage-wise transitions of in-context learning of transformers (Edelman et al., 2024; Hoffmann et al., 2024; Minegishi et al., 2025; Olsson et al., 2022; Reddy, 2024; Varre et al., 2025).

First loss drop: learning of the sequence embeddings. The first loss drop coincides with the learning of the sequence embeddings. It corresponds to a phase change in the dynamics of the weights. Before the first phase change, the weights seem to wander as if trapped in a saddle point, waiting for a clear signal to escape it. At one point, they all move quite rapidly to create a relatively definitive structure for the sequence embeddings. Interestingly, we also notice that the time it takes for this phase change to occur when changing the training hyperparameters can vary quite a lot, reflecting the highly unpredictable time needed to escape from the saddle point. This is illustrated in Section A.3 where Fig. 16a shows the high variability of the test accuracy after 1000 epochs when slightly varying one hyperparameter (Fig. 16b shows the same phenomenon with more regularity when averaging over the runs). The exit from a saddle point can also be understood via the gradient norms, as discussed at the end of this section.

Second loss drop: fitting of the MLP. The second loss drop is due to the learning of the feedforward network. This change is about fitting the MLP weights to assign the correct classes to the different clusters created during the learning of the sequence embeddings. Interestingly, this second loss drop does not correspond to a clear phase change in the dynamics of the weights. The MLP weights seem to evolve at a continuous speed, although the corresponding decision frontiers change relatively strongly. We notice that this second loss drop appears soon after the first one, if not simultaneously. When the model's training stagnates, the loss plateaus for several epochs before decreasing again. This behavior may be understood by the optimizer reaching a saddle point in the loss function w.r.t. the model's param-

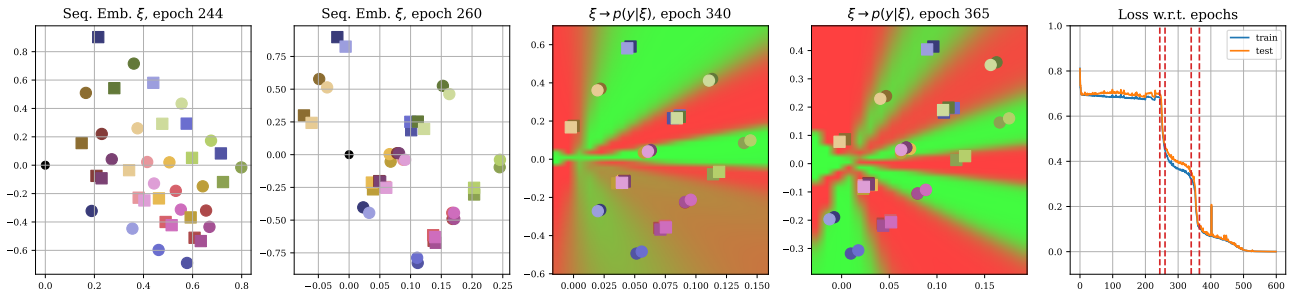


Figure 8. **Two-phase learning**. *From left to right:* **(1)** During the first snapshot, the sequence embeddings lack any clear structure. **(2)** They suddenly become clustered after the first loss drops, as seen in the second snapshot. **(3)** At this point, the MLP already classifies some clusters correctly (third snapshot). **(4)** A second loss drop occurs as the MLP gets fitted (last snapshot). *Right:* Loss profile featuring two significant drops in loss, marked by four red dashed lines at key snapshots.

eter, that is, the gradient of the loss is (almost) zero, but the optimization has not yet reached a local minimum. The connection between gradient norms and learning phases is salient in Fig. 9, where we can see that loss drops occur in tandem with high gradient norms for each layer.

Theoretical insights. Proposition 2 characterize how the gradient magnitude is controlled along the training (the full statement is deferred to Proposition 4).

Proposition 2 (Gradient upper-bound, informal)**.** *Let θ be the transformer's parameters. Assuming bounded token embeddings, the gradient of the loss verifies*

$$\|\nabla \mathcal{L}(\theta)\| = \mathcal{O}\Big(\tilde{B} \cdot \sqrt{\mathscr{E}}\Big),$$

where $\mathscr{E} = \mathbb{E}_x[d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))]$ captures the ability of the transformer to solve the task and $\tilde{B}$ depends solely on the spectral norm of the model's parameters.

We see that the term $\mathscr{E}$ captures the ability of the model to correctly solve the task. In the early iterations, the model behaves poorly, which imposes weak control on the gradient norm. This allows for large gradient norms and hence for bigger updates to the weights. The loss can vary rapidly, as can be seen by the steep descent of Fig. 8. During training, the model improves as can be seen by the increase in accuracy (both train and test), leading to a smaller $\mathscr{E}$ and strengthening the control on the norm. In parallel, the gradient updates are of smaller magnitude as well as the loss variations. This implies a less steep descent in the loss as observed in Fig. 8.

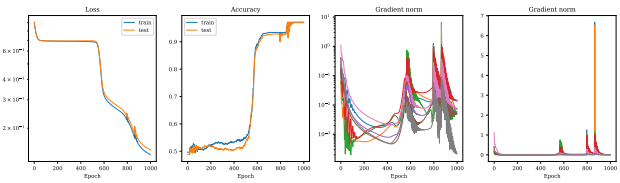


Figure 9. **Escaping saddle points**. *From left to right:* Evolution of train and test losses, the corresponding accuracies, the evolution of gradient norms for each layer in log-scale, and the similar evolution in linear scale in full-batch. We see that the learning phases of Section 4.3 appear in tandem with high gradient norms. This can be seen in the last subfigure, where the three peaks correspond to the loss drops and their corresponding plateaus.

4.4. Occurrence of loss spikes

Better understanding why loss spikes occur is of great interest as they can be detrimental to the training stability, particularly with very large-scale transformers such as large language models (Chowdhery et al., 2023). While some know-how exists for mitigating the impact of peak losses, it is often experimentally motivated and requires a lot of computational resources to achieve operational success (Grattafiori et al., 2024). Our visual inspection showcases two aspects leading to loss spikes: the high curvature of the RMS normalization layer near the origin, as well as the high curvature of multi-layer perceptrons with heavy weights, or with numerous small correlated weights.

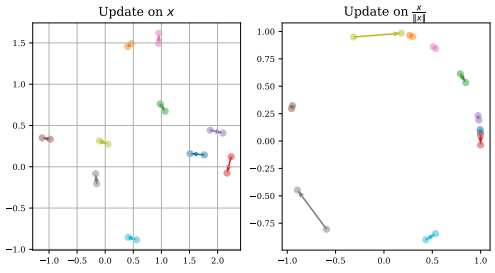


Figure 10. **High-curvature normalization**. Loss spikes are linked to the high curvature of internal network functions. A small update to an element x can result in a substantial change to its normalized version $x/\|x\|$, significantly altering the network's subsequent behavior.

High-curvature normalization. Fig. 10 illustrates that a small modification to an element can result in a disproportionately large change in its normalized version. In theory,

the gradient points towards directions that would reduce the training loss. However, considering a large step size in these directions could be counterproductive. This is especially true for functions with high curvature, such as the normalization layer $f(x) = x/\|x\|$ near the origin. At any point x_t, the gradient descent update rule suggests that one can update x_{t+1} as $x_t - \eta_t u_t$ without changing $f(x_t)$, where $u_t = x_t/\|x_t\|$ and η_t is the learning rate. This holds only if the learning is small enough, $\eta_t < \|x_t\|$. When x_t is close to zero, ensuring $\eta_t < \|x_t\|$ becomes challenging, particularly if the step size η_t was predetermined by some scheduler. This behavior is related to the "edge-of-stability" phenomenon highlighted by Cohen et al. (2022), further theoretical insights being provided by Cabannes et al. (2024c). Interestingly, this analysis suggests removing some loss spikes by smoothing out the normalization layer. For example, consider using $f(x) = \sigma(\|x\|)x/\|x\|$ where σ is a smooth function with $\sigma(0) = 0$ and $\sigma([1, \infty)) = \{1\}$. This demonstrates the usefulness of our visual sandbox in gaining insights and building intuition, which can then be validated on a larger scale in subsequent works.

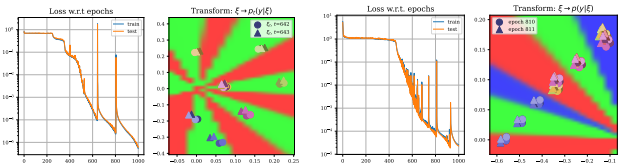


Figure 11. **High-curvature feed-forward**. Loss spikes (both left) resulting from a small change from one iteration to another in sequence embeddings that are close to the decision boundaries of the subsequent feedforward layer (both right).

High-curvature feed-forward. Another source of loss spikes is illustrated in Fig. 11. They are due to the decision boundaries of the feedforward layer being quite close to the sequence embeddings, meaning that a small change in the sequence embedding can lead to a great change in their classification. This is again due to the high curvature of the MLP layer, as illustrated in Fig. 12. In particular, the heavy,

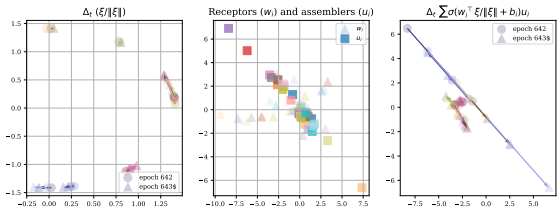


Figure 12. A small change in sequence embeddings (left) can lead to a big change in MLP response (right). This is due to heavy or small but heavily correlated assemblers (middle).

or the small but heavily correlated, weights in the MLP cause the response ζ to vary highly as a function of ζ. Once again, one can imagine different ways to regularize these types of loss spikes, with various regularization measures, or by ensuring that the capacity of the MLP is large enough for the MLP to avoid creating these heavy or highly correlated weights.

Connection to gradient norms. Loss spikes and high gradient norms appear in tandem as can be seen in Fig. 13, indicating that when a too-large step size deviates the model from its current small loss region, it is taken back to where it was with large updates. It motivates further study of the Hessian of the loss, which is promising in the literature (Foret et al., 2021; Gomes et al., 2025; Ilbert et al., 2024; Zhang et al., 2024c).

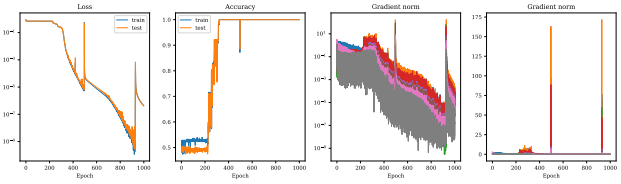


Figure 13. **Loss spikes and gradient norms.** Akin to Fig. 9 in mini-batch, this experiment makes the connection between gradient norm and loss spikes salient.

Experiments in larger dimensions. When the embedding dimension d is bigger, we cannot represent the tokens in the plane. However, we can plot the evolution of other metrics, such as the loss, the gradient norms, or the activation sparsity. We replicate the experiments of Section 4.3 with $d > 2$ in Section A.4 and obtain similar findings, which are also coherent with the existing literature in more practical settings (Ilbert et al., 2024; Li et al., 2023b; Mirzadeh et al., 2024; Zhang et al., 2024c).

Implication for large language models. Ensuring the stability of training is key to reducing budget and computational costs. A common issue during pretraining is the occurrence of loss spikes (Chowdhery et al., 2024; Li et al., 2025; Marin, 2025; Takase et al., 2025, see), which can lead to loss divergence. Better understanding their root cause is crucial to reducing the cost of training large models, e.g., through the design of novel optimizers. Our findings suggest that reducing gradient norms and avoiding embeddings with small norms can help stabilize the training. This resonates with practical heuristics such as QK-Norm (Dehghani et al., 2023; Wortsman et al., 2024) or MuonClip (Kimi Team et al., 2025) that can be seen as a way to enforce small gradient norms by avoiding exploding attention logits, while imposing a unit norm to embedding vectors, thus mitigating the high-curvature region of normalization layers, is beneficial for both the training stability and the sample efficiency (Loshchilov et al., 2025). Another promising

avenue is the use of second-order algorithms such as Shampoo (Gupta et al., 2018), Muon (Jordan et al., 2024), and AdaFisher (Gomes et al., 2025), which allow for better control over the gradient variations.

5. Related work

Mechanistic interpretability. Neural networks are often seen as "black boxes" whose internal computations are hard to decipher. Researchers have developed various methods, such as extracting meaningful features from neural network activations (Fel et al., 2023), and assessing the impact of perturbations on model inputs (Fel et al., 2021; Koh & Liang, 2017), among others. Methods have also been proposed to discover circuits in neural networks automatically (Conmy et al., 2023). Recently, the field of mechanistic interpretability has advocated for exposing the internal mechanisms of transformers to provide novel insights into their capabilities (Elhage et al., 2021; Olsson et al., 2022), by either relying on interventions on large models (Geva et al., 2023; Meng et al., 2022; Wang et al., 2023), or by conducting precise ablation studies in a controlled setup (Bietti et al., 2023; Cabannes et al., 2024a; Charton, 2022; Liu et al., 2022; Nanda et al., 2023). Our work aligns with the latter, aiming to make the internal behavior of transformers more explicit through carefully selected visualizations in controlled settings, with a focus on their training dynamics.

Training dynamics in neural networks. With non-convex losses and models of increasing sizes, understanding how models learn is more and more difficult. One approach to address this issue is to employ mathematical abstractions, as seen with neural tangent kernels (NTK) Chizat et al. (2020); Jacot et al. (2018) and mean-field analysis (Chen et al., 2024; Mei et al., 2018), among others (Abbe et al., 2022; Ahn et al., 2024; Cabannes et al., 2024c). Obtaining formal results often requires deviating from the practical implementation (Ahn et al., 2023; Bietti et al., 2023; Boix-Adsera et al., 2023; Cabannes et al., 2024c; Geshkovski et al., 2023a; Jelassi et al., 2022; Mahankali et al., 2024; Tian et al., 2023). In our work, we keep the main transformer block components intact to ensure that our findings have practical implications.

5.1. Discussion

This paper aims to advance our understanding of the transformer's training dynamics through a detailed study in a controlled and interpretable setting. We show how transformers trained with gradient descent can solve the sparse modular addition by capturing invariants, as is often the case in NLP and reasoning. Our visual sandbox allows us to monitor the evolution of tokens during training. We use it to uncover the stage-wise learning and the occurrence of loss spikes. By connecting our findings with more practical settings, we show the benefits of our sandbox to study complex behavior at a small scale. We hope it can be used to build intuition towards improving current training pipelines.

Limitations. Our work is a controlled study of a single transformer block, which allows both theory and experiments towards a better understanding of transformer training. In future work, we will aim to extend our visualization tools to larger models.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abbe, E., Boix-Adsera, E., and Misiakiewicz, T. The merged-staircase property: a necessary and nearly sufficient condition for SGD learning of sparse functions on two-layer neural networks, 2022.

Abbe, E., Cornacchia, E., and Lotfi, A. Provable advantage of curriculum learning on parity targets with mixed inputs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=9Ihu0VBOTq.

Agarwal, N., Allen-Zhu, Z., Bullins, B., Hazan, E., and Ma, T. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pp. 1195–1199, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055464. URL https://doi.org/10.1145/3055399.3055464.

Ahn, K., Cheng, X., Daneshmand, H., and Sra, S. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=LziniAXEI9.

Ahn, K., Cheng, X., Song, M., Yun, C., Jadbabaie, A., and Sra, S. Linear attention is (maybe) all you need (to understand transformer optimization). In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=0uI5415ry7.

Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Étienne Goffinet, Hesslow, D., Launay, J., Malartic, Q., Mazzotta, D., Noune, B.,

Pannier, B., and Penedo, G. The falcon series of open language models, 2023. URL https://arxiv.org/abs/2311.16867.

Ameisen, E., Lindsey, J., Pearce, A., Gurnee, W., Turner, N. L., Chen, B., Citro, C., Abrahams, D., Carter, S., Hosmer, B., Marcus, J., Sklar, M., Templeton, A., Bricken, T., McDougall, C., Cunningham, H., Henighan, T., Jermyn, A., Jones, A., Persic, A., Qi, Z., Ben Thompson, T., Zimmerman, S., Rivoire, K., Conerly, T., Olah, C., and Batson, J. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/methods.html.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization, 2016.

Barak, B., Edelman, B. L., Goel, S., Kakade, S., Malach, E., and Zhang, C. Hidden progress in deep learning: SGD learns parities near the computational limit, 2023.

Barth, A. L. and Poulet, J. F. Experimental evidence for sparse firing in the neocortex. *Trends in Neurosciences*, 35(6):345–355, 2012. ISSN 0166-2236. doi: https://doi.org/10.1016/j.tins.2012.03.008. URL https://www.sciencedirect.com/science/article/pii/S0166223612000513.

Bietti, A., Cabannes, V., Bouchacourt, D., Jegou, H., and Bottou, L. Birth of a transformer: A memory viewpoint. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=3X2EbBLNsk.

Boix-Adsera, E., Littwin, E., Abbe, E., Bengio, S., and Susskind, J. Transformers learn through gradual rank increase. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 24519–24551. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/4d69c1c057a8bd570ba4a7b71aae8331-Paper-Conference.pdf.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Cabannes, V., Arnal, C., Bouaziz, W., Yang, X. A., Charton, F., and Kempe, J. Iteration head: A mechanistic study of chain-of-thought. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024a. URL https://openreview.net/forum?id=D66dtunCnP.

Cabannes, V., Dohmatob, E., and Bietti, A. Scaling laws for associative memories, 2024b.

Cabannes, V., Simsek, B., and Bietti, A. Learning associative memories with gradient descent, 2024c.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Cascante-Bonilla, P., Tan, F., Qi, Y., and Ordonez, V. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 6912–6920, 2021.

Castin, V., Ablin, P., and Peyré, G. How smooth is attention? In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5817–5840. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/castin24a.html.

Charton, F. What is my math transformer doing? – three results on interpretability and generalization, 2022. URL https://arxiv.org/abs/2211.00170.

Charton, F. Learning the greatest common divisor: explaining transformer predictions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=cmcD05NPKa.

Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998. doi: 10.1137/S1064827596304010. URL https://doi.org/10.1137/S1064827596304010.

Chen, Y., Liu, F., Lu, Y., Chrysos, G., and Cevher, V. Generalization of scaled deep resnets in the mean-field regime. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tMzPZTvz2H.

Chi, Y., Lu, Y. M., and Chen, Y. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269, October 2019a. ISSN 1941-0476. doi: 10.1109/tsp.2019.2937282. URL http://dx.doi.org/10.1109/TSP.2019.2937282.

Chi, Y., Lu, Y. M., and Chen, Y. Nonconvex optimization meets low-rank matrix factorization: An overview. *Trans. Sig. Proc.*, 67(20):5239–5269, October 2019b. ISSN 1053-587X. doi: 10.1109/TSP.2019.2937282. URL https://doi.org/10.1109/TSP.2019.2937282.

Chistikov, D., Englert, M., and Lazic, R. Learning a neuron by a shallow reLU network: Dynamics and implicit bias for correlated inputs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xgY4QcOiEZ.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming, 2020. URL https://arxiv.org/abs/1812.07956.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., and et al., P. S. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023. URL http://jmlr.org/papers/v24/22-1144.html.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., and et al., G. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1), mar 2024. ISSN 1532-4435.

Cohen, J. M., Kaur, S., Li, Y., Kolter, J. Z., and Talwalkar, A. Gradient descent on neural networks typically occurs at the edge of stability, 2022. URL https://arxiv.org/abs/2103.00065.

Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., and et al., O. R. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=89ia77nZ8u.

Dauphin, Y., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, 2014. URL https://arxiv.org/abs/1406.2572.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A. P., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., and et al., R. R. Scaling vision transformers to 22 billion parameters. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7480–7512. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/dehghani23a.html.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Du, S. S., Jin, C., Lee, J. D., Jordan, M. I., Poczos, B., and Singh, A. Gradient descent can take exponential time to escape saddle points, 2017. URL https://arxiv.org/abs/1705.10412.

Dziri, N., Lu, X., Sclar, M., Li, X. L., Jiang, L., Lin, B. Y., Welleck, S., West, P., Bhagavatula, C., Bras, R. L., Hwang, J. D., Sanyal, S., Ren, X., Ettinger, A., Harchaoui, Z., and Choi, Y. Faith and fate: Limits of transformers on compositionality. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Fkckkr3ya8.

Edelman, E., Tsilivis, N., Edelman, B. L., eran malach, and Goel, S. The evolution of statistical induction heads: In-context learning markov chains. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=qaRT6QTIqJ.

Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.12.012. URL https://www.sciencedirect.com/science/article/pii/S0893608017302976. Special issue on deep reinforcement learning.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

Fel, T., Cadene, R., Chalvidal, M., Cord, M., Vigouroux, D., and Serre, T. Look at the variance! efficient black-box explanations with sobol-based sensitivity analysis. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 26005–26014. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/da94cbeff56cfda50785df477941308b-Paper.pdf.

Fel, T., Boutin, V., Béthune, L., Cadene, R., Moayeri, M., Andéol, L., Chalvidal, M., and Serre, T. A holistic approach to unifying automatic concept extraction and concept importance estimation. In *Thirty-seventh Conference*

on Neural Information Processing Systems, 2023. URL https://openreview.net/forum?id=MziFFGjpkb.

Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

Fukushima, K. Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4):322–333, 1969. doi: 10.1109/TSSC.1969.300225.

Geshkovski, B., Letrouit, C., Polyanskiy, Y., and Rigollet, P. The emergence of clusters in self-attention dynamics. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=aMjaEkkXJx.

Geshkovski, B., Letrouit, C., Polyanskiy, Y., and Rigollet, P. A mathematical perspective on transformers, 2023b.

Geshkovski, B., Rigollet, P., and Ruiz-Balet, D. Measure-to-measure interpolation using transformers, 2025. URL https://arxiv.org/abs/2411.04551.

Geva, M., Bastings, J., Filippova, K., and Globerson, A. Dissecting recall of factual associations in auto-regressive language models, 2023.

Gomes, D. M., Zhang, Y., Belilovsky, E., Wolf, G., and Hosseini, M. S. Adafisher: Adaptive second order optimization via fisher information. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=puTxuiK2q0.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., and et al., A. V. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Gribonval, R. and Nielsen, M. Sparse representations in unions of bases. *IEEE Transactions on Information Theory*, 49(12):3320–3325, 2003. doi: 10.1109/TIT.2003.820031.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., and et al., Z. G. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Gupta, V., Koren, T., and Singer, Y. Shampoo: Preconditioned stochastic tensor optimization. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1842–1850. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/gupta18a.html.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus), 2023. URL https://arxiv.org/abs/1606.08415.

Hoffmann, D. T., Schrodi, S., Bratulić, J., Behrmann, N., Fischer, V., and Brox, T. Eureka-moments in transformers: Multi-step tasks reveal softmax induced optimization problems. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=HssOwuZiaB.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Ilbert, R., Odonnat, A., Feofanov, V., Virmaux, A., Paolo, G., Palpanas, T., and Redko, I. SAMformer: Unlocking the potential of transformers in time series forecasting with sharpness-aware minimization and channel-wise attention. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20924–20954. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ilbert24a.html.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

Jelassi, S., Sander, M. E., and Li, Y. Vision transformers provably learn spatial structure. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=eMW9AkXaREI.

Jordan, K., Jin, Y., Boza, V., You, J., Cesista, F., Newhouse, L., and Bernstein, J. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Kimi Team, Bai, Y., Bao, Y., Chen, G., Chen, J., Chen, N., Chen, R., Chen, Y., Chen, Y., Chen, Y., Chen, Z., Cui,

J., Ding, H., Dong, M., Du, A., Du, C., Du, D., Du, Y., Fan, Y., Feng, Y., Fu, K., and et al., B. G. Kimi k2: Open agentic intelligence, 2025.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/koh17a.html.

Lee, N., Sreenivasan, K., Lee, J. D., Lee, K., and Papailiopoulos, D. Teaching arithmetic to small transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dsUB4bst9S.

Li, M. Q., Fung, B. C. M., and Huang, S.-C. Training dynamics of a 1.7b llama model: A data-efficient approach, 2025. URL https://arxiv.org/abs/2412.13335.

Li, Z., You, C., Bhojanapalli, S., Li, D., Rawat, A. S., Reddi, S. J., Ye, K., Chern, F., Yu, F., Guo, R., and Kumar, S. The lazy neuron phenomenon: On emergence of activation sparsity in transformers. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=TJ2nxciYCk-.

Li, Z., You, C., and et al., S. B. The lazy neuron phenomenon: On emergence of activation sparsity in transformers. In *ICLR*, 2023b. URL https://openreview.net/forum?id=TJ2nxciYCk-.

Lindsey, J., Gurnee, W., Ameisen, E., Chen, B., Pearce, A., Turner, N. L., Citro, C., Abrahams, D., Carter, S., Hosmer, B., Marcus, J., Sklar, M., Templeton, A., Bricken, T., McDougall, C., Cunningham, H., Henighan, T., Jermyn, A., Jones, A., Persic, A., Qi, Z., Thompson, T. B., Zimmerman, S., Rivoire, K., Conerly, T., Olah, C., and Batson, J. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Liu, Z., Kitouni, O., Nolte, N., Michaud, E. J., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=6at6rB3IZm.

Llama Team. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Loshchilov, I., Hsieh, C.-P., Sun, S., and Ginsburg, B. nGPT: Normalized transformer with representation learning on the hypersphere. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=se4vjm7h4E.

Lu, Y., Li, Z., He, D., Sun, Z., Dong, B., Qin, T., Wang, L., and yan Liu, T. Understanding and improving transformer from a multi-particle dynamic system point of view. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2019. URL https://openreview.net/forum?id=pxlqJa21C.

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

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 689–696, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553463. URL https://doi.org/10.1145/1553374.1553463.

Marin. Marin: developing foundation models. https://marin.community/, 2025. Accessed: 2025-09-30.

Mei, S., Montanari, A., and Nguyen, P.-M. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33), July 2018. ISSN 1091-6490. doi: 10.1073/pnas.1806579115. URL http://dx.doi.org/10.1073/pnas.1806579115.

Meng, K., Bau, D., Andonian, A. J., and Belinkov, Y. Locating and editing factual associations in GPT. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=-h6WAS6eE4.

Minegishi, G., Furuta, H., Taniguchi, S., Iwasawa, Y., and Matsuo, Y. Beyond induction heads: In-context meta learning induces multi-phase circuit emergence. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Xw01vF13aV.

Mirzadeh, S. I., Alizadeh-Vahid, K., Mehta, S., del Mundo, C. C., Tuzel, O., Samei, G., Rastegari, M., and Farajtabar, M. ReLU strikes back: Exploiting activation sparsity in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=osoWxY8q2E.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014. ISBN 1461346916.

Nichani, E., Lee, J. D., and Bietti, A. Understanding factual recall in transformers via associative memories, 2024. URL https://arxiv.org/abs/2412.06538.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Odonnat, A., Feofanov, V., and Redko, I. Leveraging ensemble diversity for robust self-training in the presence of sample selection bias. In Dasgupta, S., Mandt, S., and Li, Y. (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 595–603. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/odonnat24a.html.

Odonnat, A., Bouaziz, W., and Cabannes, V. Easing optimization paths: a circuit perspective. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10888894.

Olah, C. Mechanistic interpretability, variables, and the importance of interpretable bases. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2022/mech-interp-essay/index.html.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library, 2019.

Pesme, S. and Flammarion, N. Saddle-to-saddle dynamics in diagonal linear networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=iuqCXg1Gng.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022.

Press, O. and Wolf, L. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 157–163, Valencia, Spain, April 2017a. Association for Computational Linguistics.

Press, O. and Wolf, L. Using the output embedding to improve language models. In Lapata, M., Blunsom, P., and Koller, A. (eds.), *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 157–163, Valencia, Spain, April 2017b. Association for Computational Linguistics. URL https://aclanthology.org/E17-2025/.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023. URL https://arxiv.org/abs/1910.10683.

Reddy, G. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=aN4Jf6Cx69.

Robinson, J. and Wingate, D. Leveraging large language models for multiple choice question answering. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yKbprarjc5B.

Saxton, D., Grefenstette, E., Hill, F., and Kohli, P. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1gR5iR5FX.

Singh, J., Nambi, A., and Vineet, V. Exposing the achilles' heel: Evaluating LLMs ability to handle mistakes in mathematical reasoning. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27044–27065, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1313. URL https://aclanthology.org/2025.acl-long.1313/.

Takase, S., Kiyono, S., Kobayashi, S., and Suzuki, J. Spike no more: Stabilizing the pre-training of large language models. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=52YBEzcI0l.

Tian, Y., Wang, Y., Chen, B., and Du, S. S. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. In *Thirty-seventh Conference*

on Neural Information Processing Systems, 2023. URL https://openreview.net/forum?id=l3HUgVHqGQ.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023a.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023b. URL https://arxiv.org/abs/2302.13971.

Varre, A., Yüce, G., and Flammarion, N. Learning in-context n-grams with transformers: Sub-n-grams are near-stationary points. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=OMwdvGDeHL.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, K. R., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

Wortsman, M., Liu, P. J., Xiao, L., Everett, K. E., Alemi, A. A., Adlam, B., Co-Reyes, J. D., Gur, I., Kumar, A., Novak, R., Pennington, J., Sohl-Dickstein, J., Xu, K., Lee, J., Gilmer, J., and Kornblith, S. Small-scale proxies for large-scale transformer training instabilities. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=d8w0pmvXbZ.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Zhang, H., Lan, Y., Wang, L., and Liu, T.-Y. On layer normalization in the transformer architecture, 2020. URL https://openreview.net/forum?id=B1x8anVFPr.

Zhang, B. and Sennrich, R. Root mean square layer normalization. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/1e8a19426224ca89e83cef47f1e7f53b-Paper.pdf.

Zhang, R., Frei, S., and Bartlett, P. L. Trained transformers learn linear models in-context. *J. Mach. Learn. Res.*, 25 (1), January 2024a. ISSN 1532-4435.

Zhang, R., Frei, S., and Bartlett, P. L. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024b. URL http://jmlr.org/papers/v25/23-1042.html.

Zhang, Y., Chen, C., Ding, T., Li, Z., Sun, R., and Luo, Z.-Q. Why transformers need adam: A hessian perspective. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024c. URL https://openreview.net/forum?id=X6rqEpbnj3.

Appendix

Roadmap. We conduct additional experiments in Section A. In Section B, we discuss in detail the theoretical insights on the learning dynamics and provide the proofs in Section C.

Table of Contents

A Experimental results **17**

A.1 Connectivity of neurons 17

A.2 Transferability of circuits 17

A.3 Impact of the hyperparameters 19

A.4 Towards more practical setting 19

B Theoretical results **20**

B.1 Gradients derivations 21

B.2 Stationary points characterization 23

B.3 Bounding the gradients 24

C Proofs **24**

C.1 Proof of Lemma 1 24

C.2 Proof of Proposition 1 38

C.3 Proof of Proposition 2 39

A. Experimental results

In this section, we present additional results related to the study of the circuits implemented by the transformer. Notably, we investigate the concrete realizations to solve the task, the sparsity of the model's activation along training, and the transferability of circuits.

Implementation details. In our experiments, we trained our networks using 2^{11} data points, which were sampled uniformly with replacement from the p^N possible sentences. We utilized the Adam optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$ (Kingma & Ba, 2017), and we initialized the network weights using the default schemes provided by PyTorch (Paszke et al., 2019). Experiments were conducted on high-performance GPUs such as V100 and can be reproduced on a single device. The code will be open-sourced upon publication.

A.1. Connectivity of neurons

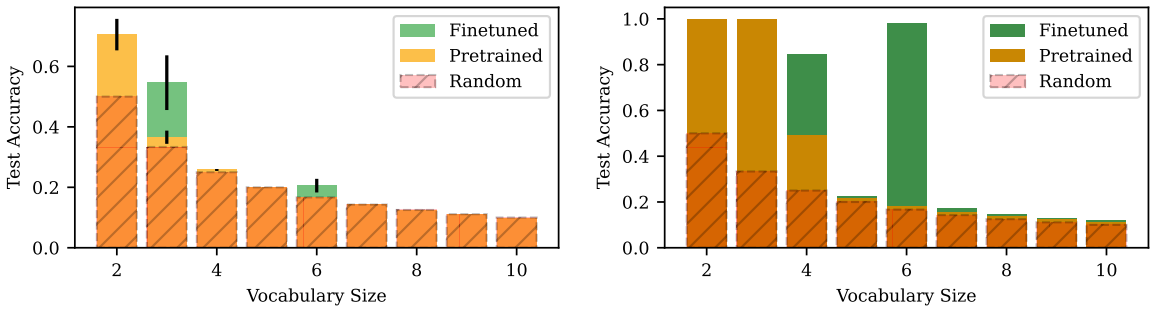
Sparsity is a phenomenon of interest in many fields such as signal processing, neuroscience, and machine learning (Barth & Poulet, 2012; Chen et al., 1998; Mairal et al., 2009). Recent studies focused on the sparsity in deep neural network activations. In particular, Li et al. (2023a) showed that trained transformers have sparse activations and concluded that it was caused by the training dynamics rather than by a compact representation of the training data, as commonly thought in computer vision and NLP. Mirzadeh et al. (2024) observed a similar phenomenon and showed how to leverage sparsity to reduce the inference cost of large language models. Inspired by this line of work, we study the activation sparsity of our model from a performance viewpoint. It should be noted that those works study deep transformers and identified that the sparsity increases with the depth, while we only consider a one-layer transformer. In our setting, the activation sparsity can help us better understand how neurons connect to implement *clustering heads*.

Successful models have dense activations. Following the framework from Li et al. (2023a), we recall that the activation sparsity corresponds to the percentage of non-zero entries of the feed-forward activation map. Without loss of generality, the feed-forward block is an MLP with weights W_1, W_2 and a non-linear activation σ that outputs for any input x a vector $z = W_1\sigma(W_2x)$. Formally, the activation sparsity is the percentage of non-zero entries in $\sigma(W_2x)$ and takes values in $[0, 1]$. In the classical setting with ReLU activation (Fukushima, 1969), this is equivalent to computing the percentage of non-negative neurons before the activation. However, some activations do not have non-negative outputs. This is the case of the SiLU (Elfwing et al., 2018), used in Llama models (Touvron et al., 2023a), and of the GeLU Hendrycks & Gimpel (2023) used in Falcon (Almazrouei et al., 2023), PaLM (Chowdhery et al., 2024), and in our transformer implementation. Instead of replacing such activations with a ReLU (Li et al., 2023a; Mirzadeh et al., 2024), we compute a smoothed sparsity. This is akin to using the ℓ_1-norm, respectively the nuclear norm, instead of the ℓ_0 quasi-norm, respectively the rank (Gribonval & Nielsen, 2003; Ilbert et al., 2024). as the percentage of entries with an absolute value lower than $\varepsilon > 0$. A sparsity of 1 means that all entries are ε-close to 0 (i.e., sparse activations), and a sparsity of 0 means that all entries are at least ε-away from 0 (i.e., dense activations).

Evolution of the sparsity with performance. To better understand the impact of sparsity, we train 20 independent models and display in Fig. 7 their sparsity after training for $\varepsilon \in [10^{-5}, 10^2]$ (the range is chosen such that the sparsity reaches its extremal values 0 and 1). Given the task's difficulty, achieving an accuracy above 0.9 is a success; otherwise, it is a failure. On the left, we plot successful models in blue and failed ones in red. We observe a striking separation between successful and failed training. In the permissible range $[10^{-3}, 10]$, successful models tend to have less sparse activation than failed ones. To further study this phenomenon, we plot in the right subplot of Fig. 7 the evolution of the sparsity with ε, and here, the color indicates the models' test accuracy (the lighter the color, the better the model). We can see that the sparsity decreases as the performance increases. This explains the sharp transition between failure and success in the left subplot. These experiments seem to indicate that, contrary to images and textual data (Li et al., 2023a; Mirzadeh et al., 2024), the sparse modular addition problem needs the involvement of many neurons during inference, and hence requires non-sparse activations.

A.2. Transferability of circuits

Among the lessons learned from training very large models is the importance of careful data engineering (Llama Team, 2024; Touvron et al., 2023b). Cabannes et al. (2024a) has highlighted that certain sources of data may facilitate the learning of invariances, while Abbe et al. (2023) discusses how data curation enables models to escape saddle points more quickly.



(a) Mean *(b)* Maximum

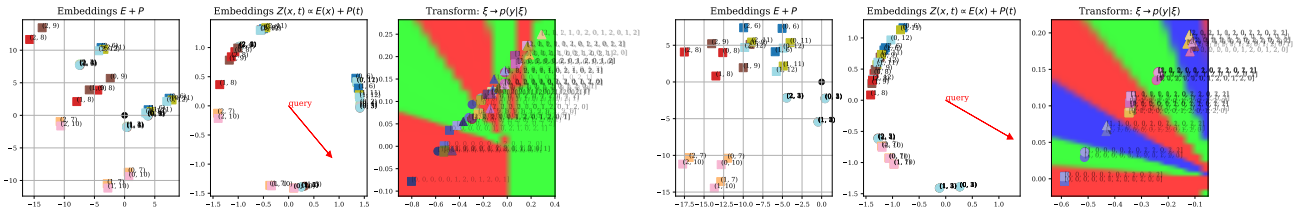
Figure 14. Accuracies obtained from pretraining only with $p \in [2, 10]$ and finetuning with $p \in [3, 10]$ starting from $n = 2$ for various vocabulary sizes. **Left:** Averaged accuracy; **Right:** Maximum accuracy. Finetuned models display better performance than pretrained-only models.

These insights are consistent with the observations we made regarding our problem. In the sparse modular addition problem, the number of unique sequences is equal to p^N, which increases rapidly with both p and N and makes the problem quite hard to learn. In particular, when setting $N = 12$ and limiting the training set to 2048 data points, training for 1000 epochs does not result in any learning for $p > 4$.

Impact on the performance. As previously seen in Fig. 2, when training with $p = 2$, we often find circuits that capture both permutation and suffix invariants. These invariants generalize for any $p \in \mathbb{N}$ when k and N are fixed. Consequently, initializing models with these invariants makes learning the sparse modular addition problem much easier. This observation was made after conducting the following experiments: we first trained a model with sequences in $\mathbb{F}_p$ for $p = 2$ over 1000 epochs, before switching the dataset to sequences in $\mathbb{F}_p$ for $p = 3$ for another 1000 epochs. We found that this procedure significantly facilitates the learning of the sparse modular addition problem for $p = 3$, which we summarize in Fig. 14 where each bar plot is obtained by over 250 runs. Remarkably, we found that the only models that achieved 100% test accuracy were those that captured both the token and permutation invariances after the first 1000 epochs. Specifically, these were the models that created six sequence embedding clusters, as shown in Figures 1 and 2, rather than those depicted in Figures 3 or 4.

Impact on the learned circuits. Fig. 15b shows the final circuit found in one of our finetuning experiments. The training was initialized with the circuit in Fig. 2, after adding a token embedding to encode for $x = 2$, resulting in Fig. 15a. The final embeddings are not that far from the initial ones, with the transformer having learned to mainly pay attention to the non-spurious tokens that are not equal to 2. It also pays some attention to 0 and 1 in positions $t = 7$ and $t = 10$. However, this faulty attention is corrected by the value matrix. Once again, the final configuration seems somewhat close to the initial one, as shown by the attention pattern reported in Fig. 5. This is consistent with the observations made in the previous subsections.

Benefits of transfer. Overall, our transfer experiment highlights the transferability of circuits and the usefulness of a curriculum in facilitating the learning of challenging tasks by inducing effective circuits through tasks that are easier to solve. Although our experiments are performed in two stages, we hypothesize that the same type of mechanism can explain the importance of data curation. We also note that curriculum learning has shown its benefits in other settings, such as



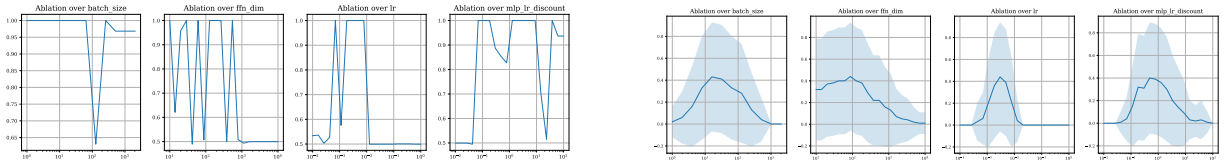
(a) Initialization with an additional token embedding for $p = 3$. *(b)* Circuit learned after 1000 epochs of finetuning with $p = 3$.

Figure 15. Transfer of clustering head. The obtained circuit is akin to Fig. 2 where the value corrects the faulty attention.

semi-supervised learning and self-training (Cascante-Bonilla et al., 2021; Odonnat et al., 2024).

A.3. Impact of the hyperparameters

In this section, we conduct ablation studies on the training hyperparameters. In Fig. 16a, we observe the high variability of the test accuracy for a fixed seed after 1000 epochs when varying the batch size, hidden dimension, learning rates, and MLP learning rates. Fig. 16b is similar with more regularity as it is averaged over 100 runs with different seeds. This study shows that the exit of saddle points is highly dependent on the hyperparameters.



(a) Single run with 1000 epochs. *(b)* Averaged over 100 runs with standard deviations.

Figure 16. Ablation studies regarding test accuracy as a function of batch size, hidden neurons, learning rates, and MLP learning rates discount factor for a single run with 1000 epochs. We ensured consistency in initial weights and batch designs when changing hyperparameters. We observe (mostly in Fig. 16b) some regularity in the effect of hyperparameters on the resulting test accuracy.

A.4. Towards more practical setting

Relying on our visual sandbox, we studied several phenomena on the training dynamics of neural networks from the learning of the different parts of the network in Section 4.3 to the loss spikes occurring during the optimization in Section 4.4 through the efficiency of transfer learning in Section A.2. We also analyzed the connection between saddle points, loss spikes, and gradient norms in Sections 4.3 and 4.4 and the impact of the activation sparsity on the models' performance in Section A.1. The low-dimensional embeddings allowed us to pair loss profiles with the visualization of each layer to decipher the training behaviors. However, we note that the studied behaviors and the insights we obtain can be analyzed independently of the embedding dimension d.

Motivation. Taking an embedding size $d = 2$ was mainly motivated by visualization purposes, and we saw that such a setting was already challenging enough to observe training phenomena of interest in more practical settings. For the sake of self-consistency, we extend our experiments with higher-dimensional embeddings and show that our insights remain valid in more practical settings with $d > 2$.

Two-phase learning and saddle points In this section, we extend the experiments of Section 4.3 for $d > 2$. In Fig. 17, we consider $d = 3$ (top) and $d = 64$ (bottom). We obtain similar conclusions to those in the case $d = 2$ in Fig. 9 where the loss drops occur in tandem with high gradient norms for each layer, hinting at the exit of saddle points.

Loss spikes and gradient norms In this section, we extend the experiments of Section 4.4 for $d > 2$. In Fig. 18, we consider $d = 3$ (top) and $d = 64$ (bottom). We observe a similar behavior as in the case $d = 2$ in Fig. 13 where the loss spikes and gradient norm spikes occur at the same time. This again highlights the connection between gradient norms and loss spikes.

Activation sparsity and performance Finally, we extend the experiments of Appendix A.1 with an embedding size $d \in \{2, 3, 4, 8, 16, 32\}$ in Fig. 19. We first note that the higher the embedding size, the more the model succeeds at the task. Especially, as of $d = 8$, all the models are successful, i.e., they all achieve an accuracy higher than 0.9 (as defined in Section A.1). This is expected given that imposing low-dimensional embeddings limits the expressiveness and the generalization power of our model. It should be noted that this was one of the many challenges of our study: obtaining a generalizable neural network with embeddings in R^2 for a mathematical reasoning task such as the sparse modular addition problem. We obtain similar conclusions than in Section A.1 with successful models having denser activations.

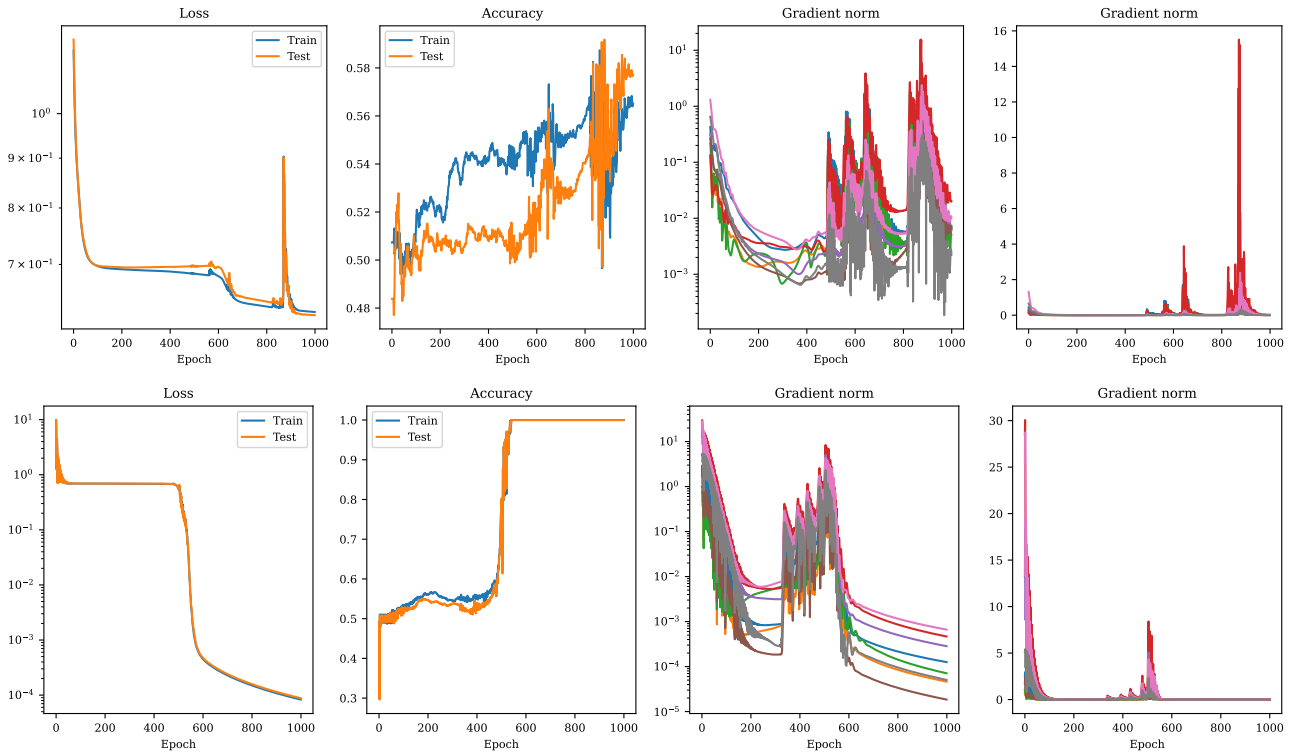


Figure 17. Connection to saddle points. This study is akin to Fig. 9 with $d = 3$ in the **top** plot and $d = 64$ in the **bottom** one. **From left to right:** Evolution of train and test losses, the corresponding accuracies, the evolution of gradient norms for each layer in log-scale, and the similar evolution in linear scale in full-batch. Akin to Fig. 9, we see that the learning phases studied in Section 4.3 appear in tandem with high gradient norms. This can be seen in the last subfigure, where the peaks correspond to the loss drops and spikes.

B. Theoretical results

In this section, we detail our theoretical insights into the learning dynamics. It should be noted that all our derivations are conducted with an embedding dimension of d and our transformer block contains the key components of its larger counterparts (see Section 2).

Notations. In this section, we recall the notations used in our theoretical results and proofs. For any integer $n > 0$, we denote $\{1, \cdots, n\}$ as $[n]$. The i-th row of a matrix $A \in \mathbb{R}^{n \times m}$ is denoted by A_i, its j-th column is denoted by $A_{\cdot,j}$ and its transpose is denoted by by $A^\top$. By abuse of notation, we denote by $\|A\|$ the entry-wise ℓ_2-norm of the matrix A, which amounts to computing the Euclidean norm of A flattened. Note that we have $\|A\| = \|A\|_{\mathrm{F}} = \sqrt{\sum_{i=1}^{\min\{n,m\}} \sigma_i(A)^2}$, where $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm of A and the $\sigma_i(A)$ are its singular values.The operator norm (also called spectral norm) of A writes $\|A\|_{\mathrm{op}} = \sigma_{\max}(A)$, with $\sigma_{\max}(A)$ the largest singular value of A. The max norm of A is denoted by $\|A\|_\infty = \max_{ij}|A_{ij}|$. The identity matrix of size n is denoted by $I_n \in \mathbb{R}^{n \times n}$. The indicator function is denoted by $\mathbb{1}$ and verifies $\mathbb{1}\{a = b\}$ with value 1 if $a = b$ and 0 otherwise. The transpose of a vector x is denoted by $x^\top$, and its Euclidean norm is denoted by $\|x\| = \sqrt{x^\top x}$. The total variation distance between two probability distributions P_1, P_2 is denoted by $d_{\mathrm{TV}}(P_1, P_2)$. Let d be the embedding dimension, h be the feedforward hidden dimension, and N be the input sequence length. For ease of notation, we define the following quantities for $\xi \in \mathbb{R}^d, W \in \mathbb{R}^{h \times d}, z \in \mathbb{R}^{d \times N}, q \in \mathbb{R}^d$ and the GeLU function $\sigma \colon \mathbb{R} \to \mathbb{R}$:

$$\bar{\xi} = \frac{\xi}{\|\xi\|} \in \mathbb{R}^d, \tag{1}$$

$$A_z = \mathrm{diag}\left(\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right) \in \mathbb{R}^{N \times N}, \tag{2}$$

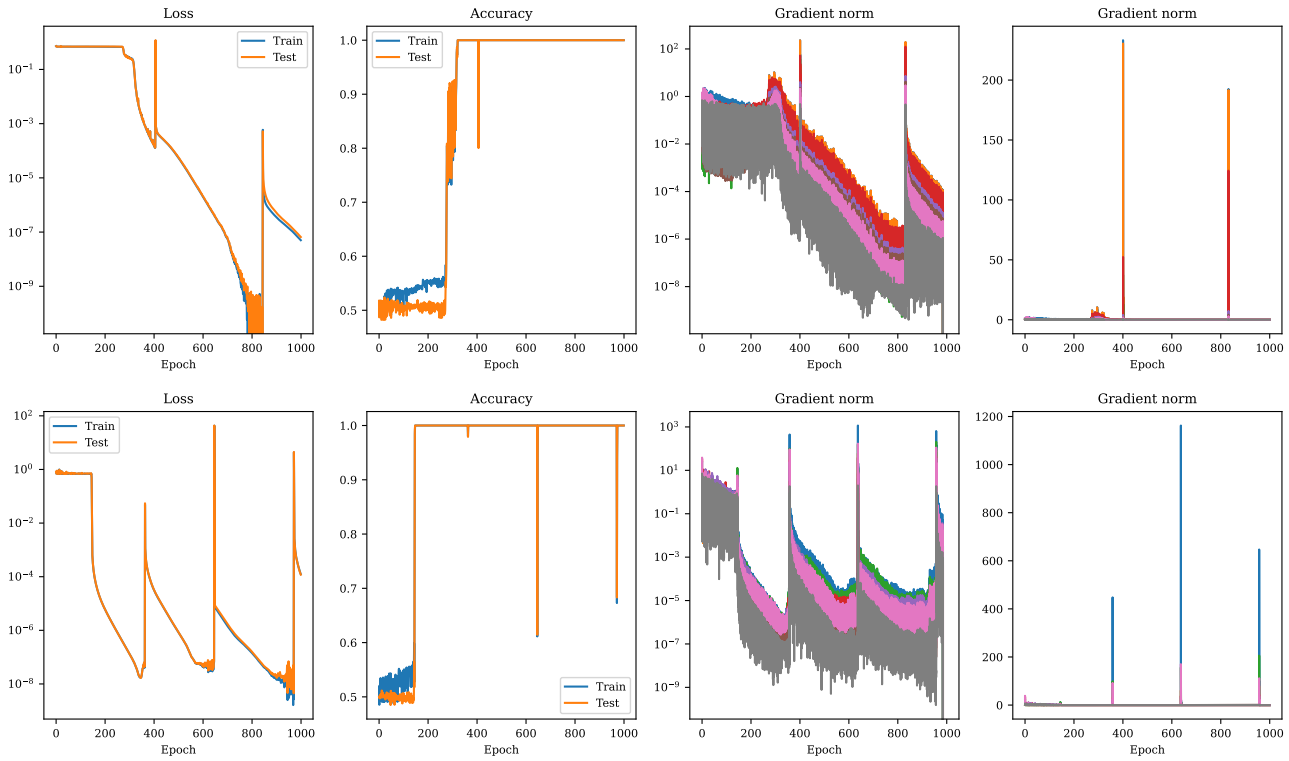


Figure 18. Loss spikes and gradient norms. This study is akin to Fig. 13 in mini-batch with $d = 3$ in the **top** plot and $d = 64$ in the **bottom** one. Again, we observe that the loss spikes and gradient norm spikes appear in tandem.

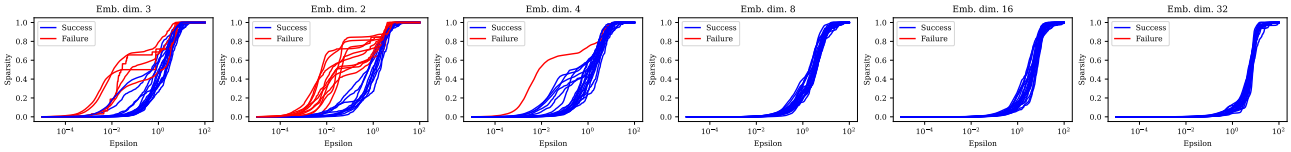


Figure 19. Connection between performance and sparsity. We display the evolution of the activation sparsity of 20 trained models with $\varepsilon \in [10^{-5}, 10^{2}]$ for $d \in \{2, 3, 4, 8, 16, 32\}$. Successful models (i.e., with test accuracy above 0.9) in blue have less sparse activation than failed models in red.

$$\Sigma_{\bar{\xi}} = \mathrm{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h \times h}. \tag{3}$$

$$\Delta_z = \frac{z}{\sqrt{d}} \left[I_N - \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \mathbb{1}_N^\top \right]. \tag{4}$$

B.1. Gradients derivations

In this section, we derive the gradient of the loss with respect to the neural network's weights. To avoid cumbersome derivations that do not fundamentally alter the analysis, and since we are primarily interested in the dynamics at work in the attention and feed-forward layers, we assume that the embedding matrices E and P are randomly initialized and do not participate in the optimization. It also concerns the unembedding matrix W_U thanks to weight tying. This is a common assumption in prior works studying the training dynamics of transformers (Bietti et al., 2023; Cabannes et al., 2024b,c; Nichani et al., 2024). To derive the loss gradients, we first recall the quantities involved and reformulate the loss.

1. **Input**: $x = [x_1, \ldots, x_N] \in (\mathbb{F}_p)^L$,
2. **Embedding**: $z_t := Z(x_t, t) := \frac{E(x_t)+P(t)}{\|E(x_t)+P(t)\|} \in \mathbb{R}^d$ with P, E learnable,

3. **Attention**: $\xi := (Vz)\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \in \mathbb{R}^d$ with $V \in \mathbb{R}^{d\times d}, q \in \mathbb{R}^d$ randomly fixed following (Bietti et al., 2023),
4. **Feed-forward**: $\psi := \xi + \sum_{i\in[h]} u_i \cdot \sigma\left(\frac{w_i^\top \xi}{\|\xi\|}\right) = \xi + U\sigma\left(\frac{W\xi}{\|\xi\|}\right) \in \mathbb{R}^d$ where $U \in \mathbb{R}^{d\times h}$ has columns $(u_i)_{i=1}^h$, $W \in \mathbb{R}^{h\times d}$ has rows $(w_i^\top)_{i=1}^h$ and U, W are learnable,
5. **Unembedding**: $\zeta = (E(v)^\top \psi)_{v\in\mathbb{F}_p} = W_U\psi \in \mathbb{R}^n$ where $W_U \in \mathbb{R}^{p\times d}$ has rows $E(v)^T$.

Gradient of the training loss. The vector θ encompasses all the model's parameters. More specifically, θ can be seen as the (flattened) concatenation of $q \in \mathbb{R}^d, V \in \mathbb{R}^{d\times d}, W \in \mathbb{R}^{h\times d}, U \in \mathbb{R}^{d\times h}$, i.e., as a vector of R^m with $m = d(1+d+2h)$. We recall that the cross-entropy loss can be reformulated as

$$\mathcal{L}(\theta) := \mathbb{E}_{x,y}[\ell(\theta; x, y],$$

with the expectation taken over training samples (x, y), and where we have

$$\ell(\theta; x, y) = -\sum_{i=1}^{p} \mathbb{1}\{y = i\} \log(\frac{\exp(\zeta_i)}{\sum_{n=1}^{p}\exp(\zeta_n)}) = \log(\sum_{n=1}^{p}\exp(\zeta_n)) - \sum_{i=1}^{p}\mathbb{1}\{y = i\}\cdot\zeta_i. \tag{5}$$

We can now proceed with computing the gradient of the loss function. The following lemma, whose proof is deferred to Section C.1, gives the closed-form expression of the gradient.

Lemma 2 (Full statement of Lemma 1)**.** *Let* $\Delta_z = \frac{z}{\sqrt{d}}\left[I_N - \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\right] \in \mathbb{R}^{d\times N}$, $A_z = \mathrm{diag}\left(\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right) \in \mathbb{R}^{N\times N}$, $\Sigma_{\bar\xi} = \mathrm{diag}(\sigma'(W\bar\xi)) \in \mathbb{R}^{h\times h}$, $M = \frac{1}{\|\xi\|}\left(I_d - \bar\xi\bar\xi^\top\right) \in \mathbb{R}^{d\times d}$ *and* $Q = U\Sigma_{\bar\xi}WM \in \mathbb{R}^{d\times d}$. *Assume that the embedding weights* E, P *are not trainable and let* $[\cdot]$ *indicate vector concatenation where matrices are flattened into vectors whenever needed. The gradient of the loss function is a vector* $\nabla_\theta\mathcal{L}(\theta) \in \mathbb{R}^m$ *with* $m = d(1 + d + 2h)$ *that writes*

$$\nabla\mathcal{L}(\theta) = \sum_{j=1}^{p}\mathbb{E}_x[(\hat p_\theta(y = j|x) - p(y = j|x))\cdot\nabla_\theta\zeta_j],$$

where

$$\begin{cases} \nabla_\theta\zeta_j &= [\nabla_q\zeta_j, \nabla_V\zeta_j, \nabla_W\zeta_j, \nabla_U\zeta_j] \in \mathbb{R}^m, \\ \nabla_q\zeta_j &= \Delta_z A_z (Vz)^\top\left(I_d + Q^\top\right)E(j) \in \mathbb{R}^d, \\ \nabla_V\zeta_j &= \left(I_d + Q^\top\right)E(j)\left[z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top \in \mathbb{R}^{d\times d}, \\ \nabla_W\zeta_j &= \Sigma_{\bar\xi}U^\top E(j)\bar\xi^\top \in \mathbb{R}^{h\times d}, \\ \nabla_U\zeta_j &= E(j)\sigma(W\bar\xi)^\top \in \mathbb{R}^{d\times h}. \end{cases}$$

We first notice the importance of the token embeddings since each gradient depends (linearly) on the $E(j)$. We can then see that the attention gradients are both controlled by $\left(I_d + Q^\top\right)E(j)$. We note that $M = \frac{1}{\|\xi\|}\left(I_d - \bar\xi\bar\xi^\top\right)$ projects any vector into $\bar\xi$'s orthogonal complementary space, which is included in the unit-sphere of R^d denoted by $\mathbb{S}^d = \{x \in \mathbb{R}^d \text{ s.t. } \|x\| = 1\}$. Similar orthogonal projections appear in the derivations of Tian et al. (2023) and are used to mimic normalization layers in Geshkovski et al. (2023a). With M symmetric, $Q^\top = M(U\Sigma_{\bar\xi}W)^\top$ simply perturbs the vector $E(j)$ before projecting it in the sphere, and $\left(I_d + Q^\top\right)E(j)$ can be seen as a perturbed residual connection. Finally, we observe that the gradients of the feed-forward weights have a linear dependency on U, W, which seems to allow implementing the decision boundaries once the sequence embeddings are clustered (see Section 4.3). Given that the gradient formulae in Lemma 2 are not straightforward to analyze due to the high dimension and the cross-dependency of the terms, we do not attempt to study the gradient updates, noting that a precise analysis is an important theoretical question for future work.

Remark B.1 (Faithful to practice)**.** *The derivations to obtain the gradients in closed form are somewhat involved due to the complexity of the network since we do not theoretically simplify the transformer architecture or the data contrary to most prior works that either model associative memories (Bietti et al., 2023; Cabannes et al., 2024b,c; Nichani et al., 2024), consider orthogonal token embeddings (Bietti et al., 2023; Tian et al., 2023), linearize the attention (Ahn et al., 2023; Mahankali et al., 2024; Nichani et al., 2024; Zhang et al., 2024b), discard residual connections (Geshkovski et al., 2023a; Jelassi et al., 2022; Tian et al., 2023) or the feed-forward block (Ahn et al., 2023; Edelman et al., 2024; Geshkovski et al., 2023a; Jelassi et al., 2022; Mahankali et al., 2024; Tian et al., 2023). Moreover, we preserve the classification setting with the cross-entropy loss used in practice instead of considering the regression case with a square loss (Chistikov et al., 2023; Pesme & Flammarion, 2023). While this ensures our analysis stays close to the practice, it also comes with technical challenges.*

B.2. Stationary points characterization

The neural networks considered in this work are learned via gradient descent, which converges to one of its fixed points (Nesterov, 2014). Such points are *stationary points* (or *critical points*) of the loss function (Chi et al., 2019b), formally defined below.

Definition 1 (Stationary points)**.** *The stationary points of $\mathcal{L}$ are the solutions of the Euler equation*

$$\nabla \mathcal{L}(\theta) = 0. \tag{6}$$

A stationary point $\bar{\theta}$ can be either a local minimum, a local maximum, or a saddle point of $\mathcal{L}$.

In deep learning, since most objectives are non-convex and in high dimensions, minima[3] are rarely global, and the gradient descent likely converges to a local minimum or a saddle point. In particular, saddle points are prevalent in deep neural networks' loss landscape (Dauphin et al., 2014), which can hinder the optimization.

Remark B.2 (Beyond stationary points)**.** *We note that distinguishing the critical points between local minimum, local maximum, or saddle points involves computing the Hessian of the training loss (Chi et al., 2019b), which is beyond the scope of this work.*

As a first step towards better understanding the training dynamics, we show in the following theorem that *clustering heads* can be learned by gradient descent. The proof is deferred to Section C.2.

Proposition 3 (Full statement of Proposition 1)**.** *Let* $\Delta_z = \frac{z}{\sqrt{d}}\left[I_N - \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\right] \in \mathbb{R}^{d\times N}$, $A_z = \mathrm{diag}\Big(\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\Big) \in \mathbb{R}^{N\times N}$, $\Sigma_{\bar{\xi}} = \mathrm{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h\times h}$, $M = \frac{1}{\|\xi\|}\big(I_d - \bar{\xi}\bar{\xi}^\top\big) \in \mathbb{R}^{d\times d}$, $Q = U\Sigma_{\bar{\xi}}WM \in \mathbb{R}^{d\times d}$, $\bar{Q} = \bar{U}\Sigma_{\bar{\xi}}\bar{W}M \in \mathbb{R}^{d\times d}$ *and* $\mathscr{C}_x = \sum_{j=1}^p (\hat{p}_\theta(y=j|x) - p(y=j|x))E(j)$. *The stationary points θ of $\mathcal{L}$ verify*

$$\begin{aligned}
&\mathbb{E}_x\left[\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big)\mathscr{C}_x\right] = 0,\\
&\mathbb{E}_x\left[\big(I_d + Q^\top\big)\mathscr{C}_x\left[z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right] = 0,\\
&\mathbb{E}_x\left[\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x \bar{\xi}^\top\right] = 0,\\
&\mathbb{E}_x\left[\mathscr{C}_x \sigma(W\bar{\xi})^\top\right] = 0.
\end{aligned} \tag{7}$$

Moreover, Eq. (7) holds for transformers that implement clustering heads. This implies that such circuits can be learned by gradient descent.

Proposition 3 allows us to access the final configuration of the model in the form of a system of equations. It involves the main quantities of the problem: the trainable weights q, V, W, U, the embedding tokens $E(j)$, and the training data (x, y). A particularly interesting observation is the linear dependency of each equation on the term $\mathscr{C}_x$. This term captures the model's prediction discrepancy with the ground truth through the $\hat{p}_\theta(y=j|x) - p(y=j|x)$ and plays an important role in the proof.

[3] We omit the mention of maxima since we are mostly interested in minimization problems.

In particular, once the stationary points are characterized, showing that clustering heads can be learned by gradient descent is rather straightforward using the $\mathscr{C}_x$. In addition, it shows that the natural way we proposed to solve the sparse modular addition problem also qualifies as a fixed-point of gradient descent.

Remark B.3 (Other pathways)**.** *While Proposition 3 shows that clustering heads are viable fixed points of the gradient descent, it does not provide conditions to ensure it is reached, nor qualify them as local minima, global minima, or saddle points. This interesting research direction is left for future work. We note that our experimental investigation suggests that clustering heads solve the sparse modular addition problem, but there may be other pathways to do so, too. In addition, as discussed in Section 4.2, the pathways learned in practice can differ from the idealized circuit, although the variations still capture most of the invariants of the problem.*

B.3. Bounding the gradients

In Section 4.3, we connect some of our findings with the evolution of gradient norms along the training. Leveraging the closed-form expressions given in Lemma 2, we can study upper-bound gradient norms to elucidate their dependency on the parameters and data. The following proposition, whose proof is deferred to Section C.3, makes those upper bounds explicit.

Proposition 4 (Full statement of Proposition 2)**.** *Assume the token embedding is bounded, i.e., there exists $B > 0$ such that $\|E\|_\infty \leq B$. We introduce $\mathscr{E} = \mathbb{E}_x[d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))]$ that captures the ability of the model to solve the sparse modular addition problem task. We have*

$$\|\nabla \mathcal{L}(\theta)\| = \mathcal{O}\Big(\tilde{B} \cdot \sqrt{\mathscr{E}}\Big),$$

where

$$\tilde{B} = B\sqrt{\|V\|_{\mathrm{op}}\big[1 + \|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}^2\big]^2 + \big[1 + \|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}\big]^2 + \|U\|_{\mathrm{op}}^2 + \|W\|_{\mathrm{op}}^2}.$$

This result is of interest for two main reasons. The first one is that it can be used to understand the loss profiles identified in Section 4.3. Notably, we see that the term $\mathscr{E}$ captures the ability of the model to correctly solve the task. In the early iterations, the model behaves poorly, which imposes weak control on the gradient norm. This allows for large gradient norms and hence for bigger updates to the weights. The loss can vary rapidly, as can be seen by the steep descent of Fig. 8. Along with the training, the model improves, as can be seen by the increase in accuracy (both train and test). This leads to a smaller $\mathscr{E}$ and strengthens the control on the norm. In parallel, the gradient updates are of smaller magnitude, as well as the loss variations. This goes along a less steep descent in the loss as observed in Fig. 8. The second benefit of Proposition 4 is that one could derive sufficient conditions on the term related to the weight matrices and on the term related to classification to achieve an approximate critical point (Chi et al., 2019b), that is parameters θ such that the gradient verifies $\|\nabla \mathcal{L}(\theta)\| \leq \varepsilon$ for some $\varepsilon > 0$. Since finding local approximate minima can be faster (and more beneficial) than finding critical points (Agarwal et al., 2017), practitioners could derive optimization algorithms that impose an appropriate bound on the two terms of Proposition 4 instead of using gradient descent. These interesting considerations are left for future work.

C. Proofs

In this section, we provide detailed proof of our theoretical results.

C.1. Proof of Lemma 1

We detail below the proof of Lemma 2.

Proof. The proof consists of three main steps:

1. Expressing the gradient of the loss with respect to the gradients of the attention embeddings,
2. Deriving the gradients for each trainable weight,
3. Putting everything together.

We first formulate the gradient of the loss with respect to the gradient of the attention embeddings. The next lemma provides

the formula.

Lemma 3 (Gradient compact formulation)**.** *Denoting the model's trainable parameters by $\theta \in \mathbb{R}^m$ and recalling that the attention embedding $\xi \in \mathbb{R}^d$ can be seen as a function of θ, we have*

$$\nabla_\theta \mathcal{L}(\theta) = \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_\theta \zeta_j] \tag{8}$$

Proof. Noting that the logits can be written as $\zeta = g(\theta; x, y)$ for some g, the loss[4] can then be written as $\ell(\theta; x, y) = \ell(\zeta) = \ell(g(\theta; x, y))$. Using the chain rule, with D denoting the Jacobian operator, we have

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= \mathbb{E}_{x,y}[\nabla_\theta \ell(\theta)] \\
&= \mathbb{E}_{x,y}[\nabla_\theta \ell(g(\theta))] \\
&= \mathbb{E}_{x,y}\left[(D_\theta g(\theta))^\top (D_{g(\theta)} \ell(g(\theta)))^\top\right] \\
&= \mathbb{E}_{x,y}[\nabla_\theta g(\theta) \nabla_\zeta \ell(\zeta)].
\end{aligned}$$

By definition, we have $\nabla_\theta g(\theta) = (\nabla_\theta \zeta_1, \dots, \nabla_\theta \zeta_p)$ and $\nabla_\zeta \ell(\zeta) = (\frac{\partial \ell}{\partial \zeta_j}(\zeta)) \in \mathbb{R}^p$. Hence, the matrix-vector product writes

$$\begin{aligned}
(\nabla_\theta g(\theta)) \nabla_\zeta \ell(\zeta) &= \sum_{j=1}^{p} \frac{\partial \ell}{\partial \zeta_j}(\zeta) \cdot [\nabla_\theta g(\theta)]_{\cdot,j} \\
&= \sum_{j=1}^{p} \frac{\partial \ell}{\partial \zeta_j}(\zeta) \cdot \nabla_\theta \zeta_j.
\end{aligned}$$

Using Eq. (5), we have

$$\begin{aligned}
\frac{\partial \ell}{\partial \zeta_j}(\zeta) &= \frac{\exp(\zeta_j)}{\sum_{n=1}^{p} \exp(\zeta_n)} - \mathbb{1}\{y=j\} \\
&= \mathrm{softmax}(\zeta)_j - \mathbb{1}\{y=j\} \\
&= \hat{p}_\theta(y=j|x) - \mathbb{1}\{y=j\},
\end{aligned}$$

Putting everything together, we obtain

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= \mathbb{E}_{x,y}\left[\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - \mathbb{1}\{y=j\}) \cdot \nabla_\theta \zeta_j\right] \\
&= \sum_{j=1}^{p} \mathbb{E}_{x,y}[(\hat{p}_\theta(y=j|x) - \mathbb{1}\{y=j\}) \cdot \nabla_\theta \zeta_j] \\
&= \sum_{j=1}^{p} \mathbb{E}_x[\mathbb{E}_y[(\hat{p}_\theta(y=j|x) - \mathbb{1}\{y=j\}) \cdot \nabla_\theta \zeta_j | x]],
\end{aligned}$$

where the first equality comes from the linearity of the expectation and the second stems from the property of the conditional expectation. Using the fact that $\hat{p}_\theta(y=j|x)$ and $\nabla_\theta \zeta_j$ do not depend on the random variable y leads us to

$$\begin{aligned}
&\mathbb{E}_y[(\hat{p}_\theta(y=j|x) - \mathbb{1}\{y=j\}) \cdot \nabla_\theta \zeta_j | x] \\
&\quad = (\hat{p}_\theta(y=j|x) - \mathbb{E}_y[\mathbb{1}\{y=j\}|x]) \cdot \nabla_\theta \zeta_j \\
&\quad = (\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_\theta \zeta_j,
\end{aligned}$$

[4]We keep denoting the loss by ℓ by abuse of notation.

where the last equality comes from the property of the conditional expectation and the fact that $x, y \sim p(\cdot)$. Indeed, we have

$$\mathbb{E}_y[\mathbb{1}\{y=j\}|x] = \sum_{l=1}^{p} \mathbb{1}\{l=j\}p(y=l|x) = p(y=j|x).$$

In summary, the gradient of the loss writes

$$\nabla_\theta \mathcal{L}(\theta) = \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_\theta \zeta_j].$$

□

Lemma 3 implies that deriving the $\nabla_\theta \zeta_j = \left(\frac{\partial \zeta_j}{\partial \theta_n}\right)_{n=1}^{m}$ allows us to obtain the final gradient formula. The following lemmas provide for a fixed j the gradients of ξ_j with respect to the different parts of the neural networks, namely the feed-forward weights U and W and the attention weights q and V.

C.1.1. Attention derivations

In this section, we derive the gradients with respect to the attention weights.

Attention query. The following lemma provides the formula of $\nabla_q \xi_j$.

Lemma 4 (Query). *Let* $\Delta_z = \frac{z}{\sqrt{d}}\left[I_N - \text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\right] \in \mathbb{R}^{d\times N}$ *following Eq. (4),* $A_z = \text{diag}\left(\text{softmax}(\frac{z^\top q}{\sqrt{d}})\right) \in \mathbb{R}^{N\times N}$ *following Eq. (2),* $\Sigma_{\bar{\xi}} = \text{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h\times h}$ *following Eq. (3),* $M = \frac{1}{\|\xi\|}\left(I_d - \bar{\xi}\bar{\xi}^\top\right) \in \mathbb{R}^{d\times d}$ *following Eq. (15) and* $Q = U\Sigma_{\bar{\xi}}WM \in \mathbb{R}^{d\times d}$*. The entries of* $\nabla_\theta \zeta_j$ *associated to the value* V *are the entries of* $\nabla_q \zeta_j \in \mathbb{R}^d$ *where we have*

$$\nabla_q \zeta_j = \Delta_z A_z[(I_d + Q)Vz]^\top E(j).$$

Proof. We want to compute the $\frac{\partial \zeta_j}{\partial \theta_n}$ for the θ_n that are entries of q. We first note that

$$\frac{\partial \zeta_j}{\partial \theta_n} = \sum_{l=1}^{p} (W_U)_{jl} \frac{\partial \psi_l}{\partial \theta_n}, \tag{9}$$

where we have thanks to weight tying, $(W_U)_{jl} = E(j)_l$ is fixed. Moreover, we have

$$\psi_l = \xi_l + \sum_{i=1}^{h} (u_i)_l \sigma(w_i^\top \bar{\xi}), \tag{10}$$

where we recall that $\bar{\xi} = \frac{\xi}{|\xi\|}$ following Eq. (1). Considering n such that $\theta_n = q_a$ for some $a \in [d]$, we have

$$\frac{\partial \psi_l}{\partial \theta_n} = \frac{\partial \psi_l}{\partial q_a} = \frac{\partial \xi_l}{\partial q_a} + \frac{\partial}{\partial q_a}(\sum_{i=1}^{h}(u_i)_l \sigma(w_i^\top \bar{\xi})) = \underbrace{\frac{\partial f_1(q_a)}{\partial q_a}}_{\text{LHS}} + \underbrace{\sum_{i=1}^{h}(u_i)_l \frac{\partial f_2^i(q_a)}{\partial q_a}}_{\text{RHS}}, \tag{11}$$

with

$$\begin{cases} f_1: & \mathbb{R} \to \mathbb{R}, \quad q_a \mapsto \xi_l \\ f_2^i: & \mathbb{R} \to \mathbb{R}, \quad q_a \mapsto (\sigma \circ f_3^i \circ f_4^i \circ f_5^i)(q_a) = \sigma(w_i^\top \bar{\xi}) \\ f_3^i: & \mathbb{R}^d \to \mathbb{R}, \quad \bar{\xi} \mapsto w_i^\top \bar{\xi} \\ f_4^i: & \mathbb{R}^d \to \mathbb{R}^d, \quad \xi \mapsto \bar{\xi} = \frac{\xi}{\|\xi\|} \\ f_5^i: & \mathbb{R} \to \mathbb{R}^d, \quad q_a \mapsto \xi. \end{cases}$$

We proceed in two steps and first compute the LHS of Eq. (11).

LHS of Eq. (11): we first note that

$$\xi_l = \sum_{t=1}^{N} (Vz)_{lt} \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}})_t.$$

This leads to

$$\frac{\partial f_1(q_a)}{\partial q_a} = \frac{\partial \xi_l}{\partial q_a} = \sum_{t=1}^{N} (Vz)_{lt} \frac{\partial s_t}{\partial q_a}, \tag{12}$$

where

$$s_t = \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}})_t = \frac{\exp\left(\frac{1}{\sqrt{d}} \sum_{o=1}^{d} z_{ot} q_o\right)}{\sum_{t'=1}^{N} \exp\left(\frac{1}{\sqrt{d}} \sum_{o=1}^{d} z_{ot'} q_o\right)}. \tag{13}$$

This can be rewritten as

$$s_t = \frac{\exp\left(\frac{1}{\sqrt{d}} z_{at} q_a + \frac{1}{\sqrt{d}} \sum_{o \neq a} z_{ot} q_o\right)}{\sum_{t'=1}^{N} \exp\left(\frac{1}{\sqrt{d}} z_{at'} q_a + \frac{1}{\sqrt{d}} \sum_{o \neq a} z_{ot'} q_o\right)} = \frac{g(q_a)}{\tilde{g}(q_a)},$$

where

$$\begin{cases} g\colon x \to \exp\left(\frac{1}{\sqrt{d}} z_{at} x + \frac{1}{\sqrt{d}} \sum_{o \neq a} z_{ot} q_o\right) \\ \tilde{g}\colon x \to \sum_{t'=1}^{N} \exp\left(\frac{1}{\sqrt{d}} z_{at'} x + \frac{1}{\sqrt{d}} \sum_{o \neq a} z_{ot'} q_o\right) \end{cases}.$$

This leads to

$$\frac{\partial s_t}{\partial q_a} = \frac{g'(q_a)\tilde{g}(q_a) - \tilde{g}'(q_a) g(q_a)}{[\tilde{g}(q_a)]^2}.$$

By simple application of the chain rule, we have

$$\begin{aligned} g'(q_a) &= \frac{1}{\sqrt{d}} z_{at} \exp\left(\frac{1}{\sqrt{d}} \sum_{o=1}^{d} z_{ot} q_o\right) \\ &= \frac{1}{\sqrt{d}} z_{at} \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t}^\top q\right). \end{aligned}$$

Similarly, we have

$$\tilde{g}'(q_a) = \sum_{t'=1}^{N} \frac{1}{\sqrt{d}} z_{at'} \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t'}^\top q\right).$$

It follows that

$$\begin{aligned} g'(q_a)\tilde{g}(q_a) &= \frac{1}{\sqrt{d}} z_{at} \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t}^\top q\right) \sum_{t'=1}^{N} \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t'}^\top q\right) \\ &= \frac{1}{\sqrt{d}} z_{at} \sum_{t'=1}^{N} \exp\left(\frac{1}{\sqrt{d}} \sum_{o=1}^{d} (z_{\cdot,t} + z_{\cdot,t'})^\top q\right) \end{aligned}$$

and similarly

$$\begin{aligned} g(q_a)\tilde{g}'(q_a) &= \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t}^\top q\right) \sum_{t'=1}^{N} \frac{1}{\sqrt{d}} z_{at'} \exp\left(\frac{1}{\sqrt{d}} z_{\cdot,t'}^\top q\right) \\ &= \frac{1}{\sqrt{d}} \sum_{t'=1}^{N} z_{at'} \exp\left(\frac{1}{\sqrt{d}} \sum_{o=1}^{d} (z_{\cdot,t} + z_{\cdot,t'})^\top q\right). \end{aligned}$$

Putting everything together, we obtain

$$\begin{aligned}
\frac{\partial s_t}{\partial q_a} &= \frac{1}{\sqrt{d}} \frac{\sum_{t'=1}^{N}(z_{at}-z_{at'})\exp\left(\frac{1}{\sqrt{d}}(z_{\cdot,t}+z_{\cdot,t'})^\top q\right)}{\left[\sum_{r=1}^{N}\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{or}q_o\right)\right]^2} \\
&= \frac{1}{\sqrt{d}}\sum_{t'=1}^{N}(z_{at}-z_{at'})\frac{\exp\left(\frac{1}{\sqrt{d}}z_{\cdot,t}^\top q\right)}{\sum_{r=1}^{N}\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{or}q_o\right)}\cdot\frac{\exp\left(\frac{1}{\sqrt{d}}z_{\cdot,t'}^\top q\right)}{\sum_{r=1}^{N}\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{or}q_o\right)} \\
&= \frac{1}{\sqrt{d}}\sum_{t'=1}^{N}(z_{at}-z_{at'})\frac{\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{ot}q_o\right)}{\sum_{r=1}^{N}\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{or}q_o\right)}\cdot\frac{\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{ot'}q_o\right)}{\sum_{r=1}^{N}\exp\left(\frac{1}{\sqrt{d}}\sum_{o=1}^{d} z_{or}q_o\right)} \\
&= \frac{1}{\sqrt{d}}\sum_{t'=1}^{N}(z_{at}-z_{at'})\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})_t\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})_{t'} \\
&= \frac{1}{\sqrt{d}}\sum_{t'=1}^{N}(z_{at}-z_{at'})s_t s_{t'}.
\end{aligned}$$

Using Eq. (12) leads to

$$\begin{aligned}
\frac{\partial f_1(q_a)}{\partial q_a} = \frac{\partial \xi_l}{\partial q_a} &= \sum_{t=1}^{N}(Vz)_{lt}\frac{1}{\sqrt{d}}\sum_{t'=1}^{N}(z_{at}-z_{at'})s_t s_{t'} \\
&= \frac{1}{\sqrt{d}}\sum_{tt'}(Vz)_{lt}z_{at}s_t s_{t'} - \frac{1}{\sqrt{d}}\sum_{tt'}(Vz)_{lt}z_{at'}s_t s_{t'} \\
&= \frac{1}{\sqrt{d}}\sum_{t=1}^{N}(Vz)_{lt}z_{at}s_t\underbrace{\sum_{t'=1}^{N}s_{t'}}_{=1} - \frac{1}{\sqrt{d}}\sum_{t=1}^{N}(Vz)_{lt}s_t\underbrace{\sum_{t=1}^{N}z_{at'}s_{t'}}_{=\mu(z_a)} \\
&= \frac{1}{\sqrt{d}}\sum_{t=1}^{N}(z_{at}-\mu(z_a)(Vz)_{lt}s_t \\
&= \sum_{t=1}^{N}\Delta_{at}(Vz)_{lt}s_t,
\end{aligned} \tag{14}$$

where $\Delta_{at} = \frac{z_{at}-\mu(z_a)}{\sqrt{d}}$ corresponds to the deviation from the mean embedding at dimension a, assuming the distribution probability follows the softmax $(s_t)_{t=1}^N$. We note that the Δ_{at} are the entries of

$$\Delta_z = \frac{z}{\sqrt{d}}\left[I_N - \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\right] \in \mathbb{R}^{d\times N}$$

defined in Eq. (4). Hence, $\frac{\partial f_1(q_a)}{\partial q_a}$ can be understood as the l-element of a sentence embedding where the deviation from the mean embedding has rescaled the value matrix. We now proceed to the RHS of Eq. (11).

RHS of Eq. (11): a simple application of the chain rule gives

$$\frac{\partial f_2^i(q_a)}{\partial q_a} = \underbrace{\nabla f_5^i(q_a)}_{\in\mathbb{R}^{1\times d}}\underbrace{\nabla f_4^i(\xi)}_{\in\mathbb{R}^{d\times d}}\underbrace{\nabla f_3^i(\bar{\xi})}_{\in\mathbb{R}^{d}}\underbrace{\sigma'(w_i^\top\bar{\xi})}_{\in\mathbb{R}},$$

where we recall that $\bar{\xi} = \xi/\|\xi\|$. We have

$$\nabla f_5^i(q_a) = \frac{\partial \xi}{\partial q_a} = \left[\frac{\partial \xi_1}{\partial q_a},\ldots,\frac{\partial \xi_d}{\partial q_a}\right],$$

and

$$\nabla f_3^i(\bar{\xi}) = w_i.$$

For f_4^i, we note that $f_4^i(\xi) = (h_1 \circ h_2)(\xi) \cdot \xi$, with $h_1 \colon \mathbb{R} \to \mathbb{R}, t \mapsto \frac{1}{t}$ and $h_2 \colon \mathbb{R}^d \to \mathbb{R}, x \mapsto \|x\|$. It leads to

$$\begin{aligned}\nabla f_4^i(\xi) &= \frac{\partial (h_1 \circ h2)(\xi)}{\partial \xi} \cdot \xi^\top + (h_1 \circ h_2)(\xi) \cdot \frac{\partial \xi}{\partial \xi} \\ &= \frac{\partial (h_1 \circ h2)(\xi)}{\partial \xi} \cdot \xi^\top + \frac{1}{\|\xi\|} I_d.\end{aligned}$$

Since we have $h_2(\xi) = \|\xi\| = \sqrt{\xi^\top \xi}$, we know that

$$\frac{\partial h2(\xi)}{\partial \xi} = \frac{1}{2\sqrt{\xi^\top \xi}} \cdot 2\xi = \frac{\xi}{\|\xi\|}.$$

It leads to

$$\frac{\partial (h_1 \circ h2)(\xi)}{\partial \xi} = -\frac{1}{\|\xi\|^2} \cdot \frac{\xi}{\|\xi\|} = -\frac{\xi}{\|\xi\|^3}.$$

Putting everything together, we obtain

$$\nabla f_4^i(\xi) = -\frac{\xi}{\|\xi\|^3} \xi^\top + \frac{1}{\|\xi\|} I_d = \frac{1}{\|\xi\|} \big(I_d - \bar{\xi}\bar{\xi}^\top\big).$$

Going back to $\frac{\partial f_2^i(q_a)}{\partial q_a}$, we have

$$\frac{\partial f_2^i(q_a)}{\partial q_a} = \left[\frac{\partial \xi_1}{\partial q_a}, \dots, \frac{\partial \xi_d}{\partial q_a}\right] \frac{1}{\|\xi\|} \big(I_d - \bar{\xi}\bar{\xi}^\top\big) w_i \sigma'(w_i^\top \bar{\xi}) = \lambda M w_i \alpha_i,$$

where we introduce

$$\lambda = \left[\frac{\partial \xi_1}{\partial q_a}, \dots, \frac{\partial \xi_d}{\partial q_a}\right] \in \mathbb{R}^{1 \times d}, \quad M = \frac{1}{\|\xi\|} \big(I_d - \bar{\xi}\bar{\xi}^\top\big) \in \mathbb{R}^{d \times d}, \quad \alpha_i = \sigma'(w_i^\top \bar{\xi}), \tag{15}$$

to ease notations. Developing the matrix multiplication, we obtain

$$\begin{aligned}\lambda M w_i \alpha_i &= \sum_{r=1}^{d} \frac{\partial \xi_r}{\partial q_a} [M w_i]_r \alpha_i \\ &= \alpha_i \sum_{r=1}^{d} \sum_{t=1}^{N} \Delta_{at} (Vz)_{rt} s_t [M w_i]_r && \text{(using Eq. (14))} \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{r=1}^{d} (Vz)_{rt} \sum_{o=1}^{d} M_{ro} (w_i)_o \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{o=1}^{d} (w_i)_o \sum_{r=1}^{d} (Vz)_{rt} M_{ro} \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{o=1}^{d} (w_i)_o \sum_{r=1}^{d} M_{ro} (Vz)_{rt} \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{o=1}^{d} (w_i)_o \sum_{r=1}^{d} M_{or}^\top (Vz)_{rt} \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{o=1}^{d} (w_i)_o \big[M^\top V z\big]_{ot} \\ &= \alpha_i \sum_{t=1}^{N} \Delta_{at} s_t \sum_{o=1}^{d} (w_i)_o \tilde{M}_{ot},\end{aligned}$$

where we introduce

$$\tilde{M} = M^\top V z \in \mathbb{R}^{d\times N}. \tag{16}$$

Now multiplying by $(u_i)_l$ and summing over $i \in [h]$, we obtain

$$\begin{aligned}\sum_{i=1}^{h}(u_i)_l \frac{\partial f_2^i(q_a)}{\partial q_a} &= \sum_{i=1}^{h}(u_i)_l \alpha_i \sum_{t=1}^{N}\Delta_{at} s_t \sum_{o=1}^{d}(w_i)_o \tilde{M}_{ot} \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{o=1}^{d}\tilde{M}_{ot}\sum_{i=1}^{h}\alpha_i (u_i)_l (w_i)_o.\end{aligned}$$

We can now put everything together to rewrite Eq. (11).

LHS + RHS of Eq. (11): We have

$$\frac{\partial \psi_l}{\partial q_a} = \frac{\partial \xi_l}{\partial q_a} + \sum_{t=1}^{N}\Delta_{at} s_t \sum_{o=1}^{d}\tilde{M}_{ot}\sum_{i=1}^{h}\alpha_i (u_i)_l (w_i)_o.$$

and Eq. (9) leads to

$$\frac{\partial \zeta_j}{\partial q_a} = \underbrace{\sum_{l=1}^{p}(W_U)_{jl}\frac{\partial \xi_l}{\partial q_a}}_{=\text{LHS}} + \underbrace{\sum_{l=1}^{p}(W_U)_{jl}\sum_{t=1}^{N}\Delta_{at} s_t \sum_{o=1}^{d}\tilde{M}_{ot}\sum_{i=1}^{h}\alpha_i (u_i)_l (w_i)_o}_{=\text{RHS}}. \tag{17}$$

The LHS can be rewritten as

$$\begin{aligned}\text{LHS} &= \sum_{l=1}^{p}(W_U)_{jl}\sum_{t=1}^{N}\Delta_{at}(Vz)_{lt} s_t \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}(W_U)_{jl}(Vz)_{lt} \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}E(j)_l (Vz)_{lt} \\ &= \sum_{t=1}^{N}\Delta_{at} s_t E(j)^\top (Vz)_{\cdot,t} \\ &= \left[\sum_{t=1}^{N}\Delta_{at} s_t [(Vz)_{\cdot,t}]^\top\right] E(j).\end{aligned}$$

The RHS can be rewritten as

$$\begin{aligned}\text{RHS} &= \sum_{l=1}^{p}(W_U)_{jl}\sum_{t=1}^{N}\Delta_{at} s_t \sum_{o=1}^{d}\tilde{M}_{ot}\sum_{i=1}^{h}\alpha_i (u_i)_l (w_i)_o \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}(W_U)_{jl}\sum_{i=1}^{h}\alpha_i (u_i)_l \sum_{o=1}^{d}\tilde{M}_{ot}(w_i)_o \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}(W_U)_{jl}\sum_{i=1}^{h}\alpha_i (u_i)_l \sum_{o=1}^{d}(w_i)_o \tilde{M}_{ot} \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}(W_U)_{jl}\sum_{i=1}^{h}\alpha_i (u_i)_l \sum_{o=1}^{d}W_{io}\tilde{M}_{ot} \\ &= \sum_{t=1}^{N}\Delta_{at} s_t \sum_{l=1}^{p}(W_U)_{jl}\sum_{i=1}^{h}\alpha_i (u_i)_l (W\tilde{M})_{it}.\end{aligned}$$

We now introduce the matrix

$$\tilde{P} = U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} W\tilde{M} \in \mathbb{R}^{d \times N}. \tag{18}$$

Its entry in row l and column t writes

$$\begin{aligned}
\tilde{P}_{lt} &= \left[U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} W\tilde{M} \right]_{lt} \\
&= \sum_{i=1}^{h} \left[U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} \right]_{li} (W\tilde{M})_{it} \\
&= \sum_{i=1}^{h} \alpha_i U_{li} (W\tilde{M})_{it} \\
&= \sum_{i=1}^{h} \alpha_i (u_i)_l (W\tilde{M})_{it}. && \text{(as } U \text{ has columns } u_i\text{)}
\end{aligned}$$

Plugging this term into the computation of the RHS leads to

$$\begin{aligned}
\text{RHS} &= \sum_{t=1}^{N} \Delta_{at} s_t \sum_{l=1}^{p} (W_U)_{jl} \tilde{P}_{lt} \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \sum_{l=1}^{p} \tilde{P}_{lt} (W_U)_{jl} \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \sum_{l=1}^{p} \tilde{P}_{lt} E(j)_l && \text{(as } W_U \text{ has rows } E(j)^\top\text{)} \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \sum_{l=1}^{p} \tilde{P}_{tl}^\top E(j)_l \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \tilde{P}_t^\top E(j).
\end{aligned}$$

Going back to Eq. (17), we have

$$\begin{aligned}
\frac{\partial \zeta_j}{\partial q_a} &= \left[\sum_{t=1}^{N} \Delta_{at} s_t [(Vz)_{\cdot,t}]^\top\right] E(j) + \sum_{t=1}^{N} \Delta_{at} s_t \tilde{P}_t^\top E(j) \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \Big[(Vz)_{\cdot,t} + \tilde{P}_t^\top\Big] E(j) \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \Big[(Vz)_{\cdot,t} + \tilde{P}_{\cdot,t}\Big] E(j) \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \Big[Vz + \tilde{P}\Big]_{\cdot,t} E(j) \\
&= \sum_{t=1}^{N} \Delta_{at} s_t \Big[Vz + \tilde{P}\Big]_{t}^\top E(j) \\
&= \sum_{t=1}^{N} \Delta_{at} \left[\begin{pmatrix} s_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & s_N \end{pmatrix} \Big[Vz + \tilde{P}\Big]^\top\right]_t E(j) \\
&= \Delta_a \begin{pmatrix} s_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & s_N \end{pmatrix} \Big[Vz + \tilde{P}\Big]^\top E(j),
\end{aligned}$$

where Δ_a is the a-th column of Δ defined in Eq. (4). Using Eq. (2) and Eq. (13), we notice that

$$A_z = \mathrm{diag}(\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})) = \mathrm{diag}((s_t)_{t=1}^N) = \begin{pmatrix} s_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & s_N \end{pmatrix} \in \mathbb{R}^{N\times N}.$$

Using Eq. (3) and Eq. (15), we notice that

$$\Sigma_{\bar{\xi}} = \mathrm{diag}(\sigma'(W\bar{\xi}) = \mathrm{diag}((\sigma'(w_i^\top \bar{\xi})_{i=1}^h) = \mathrm{diag}((\alpha_i)_{i=1}^h) = \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} \in \mathbb{R}^{N\times N}.$$

Using Eq. (3) and Eq. (18), we thus have

$$\tilde{P} = U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} W\tilde{M} = U\Sigma_{\bar{\xi}} M\tilde{W}$$

and we obtain

$$\begin{aligned}
Vz + \tilde{P} &= Vz + U\Sigma_{\bar{\xi}} M\tilde{W} \\
&= Vz + U\Sigma_{\bar{\xi}} WM^\top Vz && \text{(using Eq. (16))} \\
&= Vz + U\Sigma_{\bar{\xi}} WMVz && (M \text{ symmetric from Eq. (15)}) \\
&= \big(I_d + U\Sigma_{\bar{\xi}} WM\big)Vz.
\end{aligned}$$

It follows that

$$\frac{\partial \zeta_j}{\partial q_a} = \Delta_a A_z \big[\big(I_d + U\Sigma_{\bar{\xi}} WM\big)Vz\big]^\top E(j).$$

It leads to $\nabla_q \zeta_j = \left(\frac{\partial \zeta_j}{\partial q_a}\right)_a \in \mathbb{R}^d$ that writes

$$\nabla_q \zeta_j = \Delta_z A_z \big[\big(I_d + U\Sigma_{\bar{\xi}} WM\big)Vz\big]^\top E(j).$$

The corresponding entries of $\nabla_\theta \zeta_j$ are the entries of $\nabla_q \zeta_j$. □

Attention value. The following lemma provides the formula of $\nabla_V \xi_j$.

Lemma 5 (Value)**.** *Let $\Sigma_{\bar{\xi}} = \mathrm{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h\times h}$ following Eq. (3), $M = \frac{1}{\|\xi\|}\big(I_d - \bar{\xi}\bar{\xi}^\top\big) \in \mathbb{R}^{d\times d}$ following Eq. (15) and $Q = U\Sigma_{\bar{\xi}} W M \in \mathbb{R}^{d\times d}$. The entries of $\nabla_\theta \zeta_j$ associated to the value V are the entries of $\nabla_U \zeta_j \in \mathbb{R}^{d\times d}$ flattened in $\mathbb{R}^{d^2}$ where we have*

$$\nabla_V \zeta_j = \big(I_d + Q^\top\big) E(j) z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}).$$

Proof. We want to compute the $\frac{\partial \zeta_j}{\partial \theta_n}$ for the θ_n that are entries of V. Using Eq. (9) we first need to derive $\frac{\partial \psi_l}{\partial \theta_n}$. As θ encompasses all the learnable weights, we will derive it depending on the value of n. Considering n such that $\theta_n = V_{bc}$ for some $b, c \in [d]$ and using Eq. (10), we have

$$\frac{\partial \psi_l}{\partial \theta_n} = \frac{\partial \psi_l}{\partial V_{bc}} = \frac{\partial \xi_l}{\partial V_{bc}} + \frac{\partial}{\partial V_{bc}}(\sum_{i=1}^{h}(u_i)_l \sigma(w_i^\top \bar{\xi})) = \underbrace{\frac{\partial \beta_1(V_{bc})}{\partial V_{bc}}}_{\text{LHS}} + \underbrace{\sum_{i=1}^{h}(u_i)_l \frac{\partial \beta_2^i(V_{bc})}{\partial V_{bc}}}_{\text{RHS}}, \tag{19}$$

with

$$\begin{cases} \beta_1: & \mathbb{R}\to\mathbb{R}, \quad V_{bc}\mapsto \xi_l \\ \beta_2^i: & \mathbb{R}\to\mathbb{R}, \quad V_{bc}\mapsto (\sigma\circ\beta_3^i\circ\beta_4^i\circ\beta_5^i)(V_{bc}) = \sigma(w_i^\top\bar{\xi}) \\ \beta_3^i: & \mathbb{R}^d\to\mathbb{R}, \quad \bar{\xi}\mapsto w_i^\top\bar{\xi} \\ \beta_4^i: & \mathbb{R}^d\to\mathbb{R}^d, \quad \xi\mapsto\bar{\xi} = \frac{\xi}{\|\xi\|} \\ \beta_5^i: & \mathbb{R}\to\mathbb{R}^d, \quad V_{bc}\mapsto\xi. \end{cases}$$

We proceed in two steps and first compute the LHS of Eq. (19).

LHS of Eq. (19): we first note that

$$\xi_l = \sum_{t=1}^{N}(Vz)_{lt}s_t = \sum_{t=1}^{N}\sum_{c=1}^{d} V_{lc}z_{ct}s_t = \sum_{c=1}^{d} V_{lc}\sum_{t=1}^{N} z_{ct}s_t,$$

where $s_t = \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})_t$. We notice that ξ_l depends only on the (V_{lc}), hence the gradients with respect to V_{bc} for some $c\in[d]$ and $b\neq l$ equals zero. It follows that

$$\frac{\partial \beta_1(V_{bc})}{\partial V_{bc}} = \frac{\partial \xi_l}{\partial V_{bc}} = \begin{cases} \sum_{t=1}^{N} z_{ct}s_t & \text{if } b = l \\ 0 & \text{otherwise} \end{cases}. \tag{20}$$

RHS of Eq. (19): a simple application of the chain rule gives

$$\frac{\partial \beta_2^i(V_{bc})}{\partial V_{bc}} = \underbrace{\nabla\beta_5^i(V_{bc})}_{\in\mathbb{R}^{1\times d}}\underbrace{\nabla\beta_4^i(\xi)}_{\in\mathbb{R}^{d\times d}}\underbrace{\nabla\beta_3^i(\bar{\xi})}_{\in\mathbb{R}^{d}}\underbrace{\sigma'(w_i^\top\bar{\xi})}_{\in\mathbb{R}},$$

where we recall that $\bar{\xi} = \xi/\|\xi\|$. We have, using similar arguments as the computation of $\frac{\partial \xi_l}{\partial V_{bc}}$ in Eq. (20) that

$$\nabla\beta_5^i(V_{bc}) = \frac{\partial \xi}{\partial V_{bc}} = \left[\frac{\partial \xi_1}{\partial V_{bc}}, \dots, \frac{\partial \xi_d}{\partial V_{bc}}\right] = \left[0, \dots, 0, \sum_{t=1}^{N} z_{ct}s_t, 0, \dots, 0\right],$$

where the only non-zero entry is in position b of the row-vector $\nabla\beta_5^i(V_{bc}) \in \mathbb{R}^{1\times d}$. We notice that the functions β_3^i and β_4^i of Eq. (19) corresponds to the functions f_3^i and f_4^i of Eq. (11). Hence, the gradients are the same and we have

$$\nabla\beta_3^i(\bar{\xi}) = \nabla f_3^i(\bar{\xi}) = w_i$$

and

$$\nabla \beta_4^i(\bar{\xi}) = \nabla f_4^i(\bar{\xi}) = \frac{1}{\|\xi\|}\big(I_d - \bar{\xi}\bar{\xi}^\top\big).$$

Going back to $\frac{\partial \beta_2^i(V_{bc})}{\partial V_{bc}}$, we have

$$\frac{\partial \beta_2^i(V_{bc})}{\partial V_{bc}} = \left[0, \dots, 0, \sum_{t=1}^N z_{ct} s_t, 0, \dots, 0\right] \frac{1}{\|\xi\|}\big(I_d - \bar{\xi}\bar{\xi}^\top\big) w_i \sigma'(w_i^\top \bar{\xi}) = \mu M w_i \alpha_i,$$

where we introduce

$$\mu = \left[0, \dots, 0, \sum_{t=1}^N z_{ct} s_t, 0, \dots, 0\right] \in \mathbb{R}^{1\times d}, \quad M = \frac{1}{\|\xi\|}\big(I_d - \bar{\xi}\bar{\xi}^\top\big) \in \mathbb{R}^{d\times d}, \quad \alpha_i = \sigma'(w_i^\top \bar{\xi}), \tag{21}$$

to ease notations, with M, α_i first defined in Eq. (15). Developing the matrix multiplication, we obtain

$$\mu M w_i \alpha_i = \alpha_i \sum_{t=1}^N z_{ct} s_t [M w_i]_b = \alpha_i \sum_{t=1}^N z_{ct} s_t \sum_{o=1}^d M_{bo} (w_i)_o.$$

It follows that

$$\sum_{i=1}^h (u_i)_l \frac{\partial \beta_2^i(V_{bc})}{\partial V_{bc}} = \sum_{i=1}^h (u_i)_l \alpha_i \sum_{t=1}^N z_{ct} s_t \sum_{o=1}^d M_{bo}(w_i)_o = \sum_{t=1}^N z_{ct} s_t \sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d M_{bo}(w_i)_o.$$

We can now put everything together to rewrite Eq. (19).

<u>LHS + RHS of Eq. (19)</u>: We have

$$\frac{\partial \psi_l}{\partial V_{bc}} = \frac{\partial \xi_l}{\partial V_{bc}} + \sum_{t=1}^N z_{ct} s_t \sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d M_{bo}(w_i)_o$$

and Eq. (9) leads to

$$\frac{\partial \zeta_j}{\partial V_{bc}} = \underbrace{\sum_{l=1}^p (W_U)_{jl} \frac{\partial \xi_l}{\partial V_{bc}}}_{=\text{LHS}} + \underbrace{\sum_{l=1}^p (W_U)_{jl} \sum_{t=1}^N z_{ct} s_t \sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d M_{bo}(w_i)_o}_{=\text{RHS}}. \tag{22}$$

Using Eq. (20), the LHS can be rewritten as

$$\text{LHS} = (W_U)_{jb} \sum_{t=1}^N z_{ct} s_t = E(j)_b \sum_{t=1}^N z_{ct} s_t.$$

The RHS can be rewritten as

$$\begin{aligned}
\text{RHS} &= \sum_{l=1}^p (W_U)_{jl} \sum_{t=1}^N z_{ct} s_t \sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d M_{bo}(w_i)_o \\
&= \sum_{t=1}^N z_{ct} s_t \sum_{l=1}^p (W_U)_{jl} \left[\sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d M_{bo}(w_i)_o\right] \\
&= \sum_{t=1}^N z_{ct} s_t \sum_{l=1}^p (W_U)_{jl} \left[\sum_{i=1}^h \alpha_i (u_i)_l \sum_{o=1}^d W_{io} M_{bo}\right] && \text{(as } W \text{ has rows } w_i^\top\text{)} \\
&= \sum_{t=1}^N z_{ct} s_t \sum_{l=1}^p (W_U)_{jl} \left[\sum_{i=1}^h \alpha_i (u_i)_l (WM)_{ib}\right].
\end{aligned}$$

We now introduce the matrix

$$Q = U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} WM \in \mathbb{R}^{d\times d}. \tag{23}$$

Its entry in row l and column b writes

$$\begin{aligned} Q_{lt} &= \left[U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} WM \right]_{lb} \\ &= \sum_{i=1}^{h} \left[U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} \right]_{li} (WM)_{ib} \\ &= \sum_{i=1}^{h} \alpha_i U_{li} (WM)_{ib} \\ &= \sum_{i=1}^{h} \alpha_i (u_i)_l (WM)_{ib}^\top. && \text{(as } W \text{ has rows } w_i\text{)} \end{aligned}$$

Plugging this term in the computation of the RHS leads to

$$\begin{aligned} \text{RHS} &= \sum_{t=1}^{N} z_{ct} s_t \sum_{l=1}^{p} (W_U)_{jl} Q_{lb} \\ &= \sum_{t=1}^{N} z_{ct} s_t \sum_{l=1}^{p} E(j)_l Q_{lb} \\ &= \sum_{t=1}^{N} z_{ct} s_t \sum_{l=1}^{p} Q_{bl}^\top E(j)_l \\ &= \sum_{t=1}^{N} z_{ct} s_t \left[Q^\top E(j) \right]_b \end{aligned}$$

Going back to Eq. (22), we have

$$\begin{aligned} \frac{\partial \zeta_j}{\partial V_{bc}} &= E(j)_b \sum_{t=1}^{N} z_{ct} s_t + \sum_{t=1}^{N} z_{ct} s_t \left[Q^\top E(j) \right]_b \\ &= \left[E(j) + Q^\top E(j) \right]_b \sum_{t=1}^{N} z_{ct} s_t \\ &= \left[\left(I_d + Q^\top \right) E(j) \right]_b \sum_{t=1}^{N} z_{ct} s_t. \end{aligned}$$

We introduce the vector $z \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}}) \in \mathbb{R}^d$ whose entries writes

$$\left[z \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}}) \right]_c = \sum_{t=1}^{N} z_{ct} \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}})_t = \sum_{t=1}^{N} z_{ct} s_t,$$

where we used the fact that $s_t = \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}})_t$ following Eq. (13). It follows that

$$\frac{\partial \zeta_j}{\partial V_{bc}} = \left[\left(I_d + Q^\top \right) E(j) \right]_b \left[z \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}}) \right]_c.$$

Using Eq. (3) and Eq (23), we have

$$Q = U \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} WM = U\Sigma_{\bar{\xi}} WM.$$

Hence, $\nabla_V \zeta_j = \left(\frac{\partial \zeta_j}{\partial V_{bc}}\right)_{b,c} \in \mathbb{R}^{d\times d}$ is a matrix equal to

$$\nabla_V \zeta_j = \left(I_d + \left[U\Sigma_{\bar{\xi}} WM\right]^\top\right) E(j) \left[z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top.$$

The corresponding entries of $\nabla_\theta \zeta_j$ amounts to the entries of $\nabla_V \zeta_j$ flattened in $\mathbb{R}^{d^2}$. This concludes the proof. □

Feed-forward receptors. The following lemma provides the formula of $\nabla_W \xi_j$.

Lemma 6 (Receptors)**.** *Let $\Sigma_{\bar{\xi}} = \mathrm{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h\times h}$ following Eq. (3). The entries of $\nabla_\theta \zeta_j$ associated to the assemblers $(w_i)_{i=1}^h$ are the entries of $\nabla_W \zeta_j \in \mathbb{R}^{htimesd}$ flattened in $\mathbb{R}^{dh}$ where we have*

$$\nabla_W \zeta_j = \Sigma_{\bar{\xi}} U^\top E(j) \bar{\xi}^\top.$$

Proof. We want to compute the $\frac{\partial \zeta_j}{\partial \theta_n}$ for the θ_n that are entries of W. Using Eq. (9) we first need to derive $\frac{\partial \psi_l}{\partial \theta_n}$. As θ encompasses all the learnable weights, we will derive it depending on the value of n. Here, ψ_l depends on all the $(w_i)_e$. Considering n such that $\theta_n = (w_i)_e$ for some $i \in [h]$ and some $e \in [d]$ and using Eq. (10), we have

$$\frac{\partial \psi_l}{\partial \theta_n} = \frac{\partial \psi_l}{\partial (w_i)_e} = (u_i)_l \frac{\partial \sigma(w_i^\top \bar{\xi})}{\partial (w_i)_e}.$$

Noting that $w_i^\top \xi = \sum_{f=1}^d (w_i)_f \xi_f$ and using the chain rule, we have

$$\begin{aligned} \frac{\partial \sigma(w_i^\top \xi / \|\xi\|)}{\partial (w_i)_e} &= \frac{\partial (w_i^\top \bar{\xi})}{\partial (w_i)_e} \sigma'(w_i^\top \bar{\xi}) \\ &= \frac{\partial (\sum_{f=1}^d (w_i)_f \bar{\xi}_f)}{\partial (w_i)_e} \sigma'(w_i^\top \bar{\xi}) \\ &= \frac{\partial ((w_i)_e \bar{\xi}_e)}{\partial (w_i)_e} \sigma'(w_i^\top \bar{\xi}) \\ &= \bar{\xi}_e \sigma'(w_i^\top \bar{\xi}). \end{aligned}$$

We recall that σ is the GeLU activation defined as $\sigma\colon x \to x\varphi(x)$ with φ the cumulative distribution function of the standard normal distribution. By the fundamental theorem of analysis, it leads to

$$\sigma'(x) = \varphi(x) + x\varphi'(x) = \varphi(x) + xf(x),$$

with $f\colon u \to \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right)$ the probability density function of the standard normal distribution. Putting everything together, we have

$$\frac{\partial \psi_l}{\partial (w_i)_e} = (u_i)_l \bar{\xi}_e \sigma'(w_i^\top \bar{\xi}).$$

Using Eq. (9), we have

$$\begin{aligned}\frac{\partial \zeta_j}{\partial (w_i)_e} &= \sum_{l=1}^{p} (W_U)_{jl} (u_i)_l \bar{\xi}_e \sigma'(w_i^\top \bar{\xi}) \\ &= \sum_{l=1}^{p} E(j)_l (u_i)_l \bar{\xi}_e \sigma'(w_i^\top \bar{\xi}) \\ &= E(j)^\top u_i \bar{\xi}_e \sigma'(w_i^\top \xi / \|\xi\|) \\ &= \sigma'(w_i^\top \bar{\xi}) u_i^\top E(j) \bar{\xi}_e.\end{aligned}$$

To ease the notations, we denote $\alpha_i = \sigma'(w_i^\top \bar{\xi})$. Recalling that σ and thus σ' is entry-wise, it leads to $\nabla_W \zeta_j = (\frac{\partial \zeta_j}{\partial (w_i)_e})_{i,e} \in \mathbb{R}^{h \times d}$ that writes

$$\begin{aligned}\nabla_W \zeta_j &= \begin{pmatrix} \alpha_1 u_1^\top E(j) \\ \cdots \\ \alpha_h u_h^\top E(j) \end{pmatrix} \bar{\xi}^\top \\ &= \begin{pmatrix} \alpha_1 u_1^\top \\ \cdots \\ \alpha_h u_h^\top \end{pmatrix} E(j) \bar{\xi}^\top \\ &= \begin{pmatrix} \alpha_1 & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \alpha_h \end{pmatrix} U^\top E(j) \bar{\xi}^\top \\ &= \begin{pmatrix} \sigma'(w_1^\top \bar{\xi}) & 0 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & 0 & \sigma'(w_h^\top \bar{\xi}) \end{pmatrix} U^\top E(j) \bar{\xi}^\top \\ &= \mathrm{diag}(\sigma'(W\bar{\xi})) U^\top E(j) \bar{\xi}^\top,\end{aligned}$$

where $\mathrm{diag}(\sigma'(W\bar{\xi})) \in \mathbb{R}^{h \times h}$ is diagonal with entries $(\sigma'(w_i^\top \bar{\xi}))_{i=1}^h$. Using Eq. (3), it finally leads to

$$\nabla_W \zeta_j = \Sigma_{\bar{\xi}} U^\top E(j) \bar{\xi}^\top,$$

and the corresponding entries of $\nabla_\theta \zeta_j$ amounts to the entries of $\nabla_W \zeta_j$ flattened in $\mathbb{R}^{hd}$. This concludes the proof. □

Feed-forward assemblers. The following lemma provides the formula of $\nabla_U \xi_j$.

Lemma 7 (Assemblers)**.** *The entries of $\nabla_\theta \zeta_j$ associated to the assemblers $(u_i)_{i=1}^h$ are the entries of $\nabla_U \zeta_j \in \mathbb{R}^{d \times h}$ flattened in $\mathbb{R}^{dh}$ where we have*

$$\nabla_U \zeta_j = E(j) \sigma(W\bar{\xi})^\top \in R^{d \times h}.$$

Proof. We want to compute the $\frac{\partial \zeta_j}{\partial \theta_n}$ for the θ_n that are entries of U. Using Eq. (9) we first need to derive $\frac{\partial \psi_l}{\partial \theta_n}$. As θ encompasses all the learnable weights, we will derive it depending on the value of n. As ψ_l only depends on the l-elements of the (u_i), we only consider indices n such that $\theta_n = (u_i)_l$ for some $i \in [h]$ as the other values of n will lead to a null gradient. Using Eq. (10), we have

$$\frac{\partial \psi_l}{\partial \theta_n} = \frac{\partial \psi_l}{\partial (u_i)_l} = \frac{\partial (u_i)_l}{\partial (u_i)_l} \sigma(w_i^\top \bar{\xi}) = \sigma(w_i^\top \bar{\xi}).$$

Using Eq. (9), we have

$$\frac{\partial \zeta_j}{\partial (u_i)_l} = (W_U)_{jl} \sigma(w_i^\top \bar{\xi}) = E(j)_l \sigma(w_i^\top \bar{\xi}).$$

Hence, recalling that σ is applied element-wise, $\nabla_U \zeta_j = (\frac{\partial \zeta_j}{\partial (u_i)_l})_{l,i} \in \mathbb{R}^{d \times h}$ is a matrix equal to

$$\nabla_U \zeta_j = E(j) \sigma(W\bar{\xi})^\top,$$

and the corresponding entries of $\nabla_\theta \zeta_j$ amounts to the entries of $\nabla_U \zeta_j$ flattened in $\mathbb{R}^{dh}$. This concludes the proof. □

We can now put everything together. Using Lemma 3 in combination with the gradientsderived in Lemma 4, Lemma 5, Lemma 6 and Lemma 7 concludes the proof of Lemma 2. □

C.2. Proof of Proposition 1

We detail below the proof of Proposition 3.

Proof. We assume the gradient descent continues until convergence at a critical point and discard other practical considerations, such as early stopping. Following Chi et al. (2019b) and using Definition 1, we know that the parameters θ of the transformer verify

$$\nabla_\theta \mathcal{L}(\theta) = 0 \tag{24}$$

From Lemma 2, we know that $\nabla_\theta \mathcal{L}(\theta) \in \mathbb{R}^m$ writes

$$\nabla_\theta \mathcal{L}(\theta) = [\nabla_q \mathcal{L}(\theta), \nabla_V \mathcal{L}(\theta), \nabla_W \mathcal{L}(\theta), \nabla_U \mathcal{L}(\theta)], \tag{25}$$

using $[\cdot]$ as the vector concatenation and flattening matrices whenever needed. We now reformulate the gradients of each weight group. We have

$$\begin{aligned}
\nabla_q \mathcal{L}(\theta) &= \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_q \zeta_j] \\
&= \sum_{j=1}^{p} \mathbb{E}_x\big[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big) E(j)\big)\big] \\
&= \mathbb{E}_x\left[\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big) E(j)\big)\right] && \text{(by linearity of the expectation)} \\
&= \mathbb{E}_x\left[\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big) \cdot \sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) E(j)\right] && (\hat{p}_\theta(y=j|x) - p(y=j|x) \in \mathbb{R}) \\
&= \mathbb{E}_x\big[\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big) \mathscr{C}_x\big]. && \text{(by definition of } \mathscr{C}_x)
\end{aligned}$$

The same arguments lead to

$$\begin{aligned}
\nabla_V \mathcal{L}(\theta) &= \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_V \zeta_j] \\
&= \sum_{j=1}^{p} \mathbb{E}_x\left[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \left(\big(I_d + Q^\top\big) E(j) \left[z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right)\right] \\
&= \mathbb{E}_x\left[\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \left(\big(I_d + Q^\top\big) E(j) \left[z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right)\right] \\
&= \mathbb{E}_x\left[\big(I_d + Q^\top\big) \cdot \left(\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) E(j)\right) \cdot \left[z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right] \\
&= \mathbb{E}_x\left[\big(I_d + Q^\top\big) \mathscr{C}_x \left[z \,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right],
\end{aligned}$$

then to

$$\begin{aligned}
\nabla_W \mathcal{L}(\theta) &= \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_W \zeta_j] \\
&= \sum_{j=1}^{p} \mathbb{E}_x\big[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(\Sigma_{\bar{\xi}} U^\top E(j) \bar{\xi}^\top\big)\big] \\
&= \mathbb{E}_x\left[\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(\Sigma_{\bar{\xi}} U^\top E(j) \bar{\xi}^\top\big)\right] \\
&= \mathbb{E}_x\left[\Sigma_{\bar{\xi}} U^\top \cdot \left(\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) E(j)\right) \cdot \bar{\xi}^\top\right] \\
&= \mathbb{E}_x\big[\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x \bar{\xi}^\top\big],
\end{aligned}$$

and finally to

$$\begin{aligned}
\nabla_U \mathcal{L}(\theta) &= \sum_{j=1}^{p} \mathbb{E}_x[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \nabla_U \zeta_j] \\
&= \sum_{j=1}^{p} \mathbb{E}_x\big[(\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(E(j) \sigma(W\bar{\xi})^\top\big)\big] \\
&= \mathbb{E}_x\left[\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) \cdot \big(E(j) \sigma(W\bar{\xi})^\top\big)\right] \\
&= \mathbb{E}_x\left[\left(\sum_{j=1}^{p} (\hat{p}_\theta(y=j|x) - p(y=j|x)) E(j)\right) \cdot \sigma(W\bar{\xi})^\top\right] \\
&= \mathbb{E}_x\big[\mathscr{C}_x \sigma(W\bar{\xi})^\top\big].
\end{aligned}$$

The equality to 0 of the vector in $\mathbb{R}^m$ is equivalent to each of its components being equal to 0, and in particular, this is the case of the gradients with respect to q, V, W, and U. Combining Eq. (24) with Lemma 2 leads to the following stationarity conditions

$$\begin{cases}
\mathbb{E}_x\big[\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big) \mathscr{C}_x\big] = 0 \\
\mathbb{E}_x\left[\big(I_d + Q^\top\big) \mathscr{C}_x \left[z \operatorname{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right] = 0 \\
\mathbb{E}_x\big[\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x \bar{\xi}^\top\big] = 0 \\
\mathbb{E}_x\big[\mathscr{C}_x \sigma(W\bar{\xi})^\top\big] = 0.
\end{cases}$$

Once those conditions are established, the rest of the proof is straightforward. Indeed, it suffices to recall that the clustering heads described in Section 3 capture all the invariants of the problem. This leads to a perfect separation of the clusters that can be seen in Fig. 1. In particular, the transformer correctly classifies all the sequences, which amounts to the $\hat{p}_\theta(y=j|x)$ and the $p(y=j|x)$ being aligned. By definition, this implies $\mathscr{C}_x = 0$ for all x, and noticing the linear dependency of the gradients with $\mathscr{C}_x$ ensures that the stationary conditions previously stated are verified. Hence, a transformer that implements a circuit head has parameters θ for which $\nabla \mathcal{L}(\theta) = 0$ holds and that can thus be reached via gradient descent. This implies circuit heads can emerge during the training of the transformer. This concludes the proof. □

C.3. Proof of Proposition 2

We detail below the proof of Proposition 2 whose formal statement is given in Proposition 4.

Proof. We first decompose the gradient of the loss using Eq. (25). Gathering the terms by group leads to

$$\|\nabla_\theta \mathcal{L}(\theta)\|^2 = \sum_{n=1}^{m} \left| \frac{\partial \mathcal{L}(\theta)}{\partial \theta_n} \right|^2 = \|\nabla_q \mathcal{L}(\theta)\|^2 + \|\nabla_V \mathcal{L}(\theta)\|^2 + \|\nabla_W \mathcal{L}(\theta)\|^2 + \|\nabla_U \mathcal{L}(\theta)\|^2, \tag{26}$$

where the norm is the Euclidean one with matrices flattened as vectors. Similarly to the proof of Proposition 3, we have

$$\begin{cases} \nabla_q \mathcal{L}(\theta) = \mathbb{E}_x\left[\Delta_z A_z (Vz)^\top \left(I_d + Q^\top\right)\mathscr{C}_x\right] \\ \nabla_V \mathcal{L}(\theta) = \mathbb{E}_x\left[\left(I_d + Q^\top\right)\mathscr{C}_x\left[z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right] \\ \nabla_W \mathcal{L}(\theta) = \mathbb{E}_x\left[\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x \bar{\xi}^\top\right] \\ \nabla_U \mathcal{L}(\theta) = \mathbb{E}_x\left[\mathscr{C}_x \sigma(W\bar{\xi})^\top\right]. \end{cases}$$

We now proceed with upper-bounding each term of Eq. (26). For clarity of the proofs, we first recall the following basic relations.

Lemma 8 (Properties)**.** *Let $a \in \mathbb{R}^m, b \in \mathbb{R}^n, A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times l}$. We have*

1. $\|ab^\top\| = \|a\| \cdot \|b\|$.
2. $\|Ab\| \leq \|A\|_{\mathrm{op}}\|b\|$.
3. *The operator norm (resp. the Frobenius norm) is sub-multiplicative, i.e.,* $\|AB\|_{\mathrm{op}} \leq \|A\|_{\mathrm{op}}\|B\|_{\mathrm{op}}$.
4. *If A has elements bounded by $C > 0$, then* $\|A\|_{\mathrm{op}} \leq C\sqrt{nm}$.
5. *If P be an orthogonal projection, then, we have* $\|P\|_{\mathrm{op}} = 1$.
6. *The singular values of A and $A^\top$ coincides. In particular,* $\|A\|_{\mathrm{op}} = \|A^\top\|_{\mathrm{op}}$.

Proof. The first point follows from

$$\|ab^\top\|^2 = \sum_{i,j}(ab^\top)_{ij}^2 = \sum_{i,j}(a_i b_j)^2 = \left(\sum_i a_i^2\right) \cdot \left(\sum_j b_j^2\right) = \|a\|^2 \cdot \|b\|^2,$$

where taking the square root gives the desired equality.

The second point comes from the definition of the operator norm

$$\|A\|_{\mathrm{op}} = \sup\{\frac{\|Ax\|}{\|x\|} : x \in \mathbb{R}^m, x \neq 0\} = \sup\{\|Ax\| : x \in \mathbb{R}^m, \|x\| = 1\}.$$

It implies that for any $x \in \mathbb{R}^m$ non-zero, by definition of the sup, we have

$$\|Ax\| = \frac{\|Ax\|}{\|x\|} \cdot \|x\| \leq \|A\|_{\mathrm{op}}\|x\|.$$

The inequality holds (with equality) for $x = 0$.

The third point is simply a property of matrix norms.

The fourth point comes from the definition of operator norm. Let $x \in \mathbb{R}^m$ with unit norm. We have

$$\|Ax\|^2 = \sum_{i=1}^{n}\left(\sum_{j=1}^{m} A_{ij}x_j\right)^2 \leq \sum_{i=1}^{n}\left(\sum_{j=1}^{m} A_{ij}^2\right)\left(\sum_{j=1}^{m} x_j^2\right) \leq nm \cdot C^2, \tag{27}$$

using the fact that $\|x\|^2 = \sum_{j=1}^m x_j^2 = 1$ and the bound on the entries of A for the last inequality. Taking the root square and the sup on the left-hand term gives the desired formula.

For the fifth point, we recall that an orthogonal projection is symmetric. Hence, we have $P = P^2 = p^\top P$, which also ensures P to be semi-definite positive, i.e., with nonnegative eigenvalues. Let λ be such an eigenvalue and v an eigenvector. We must have $Pv = \lambda v$ but applying P on this relation also leads to

$$Pv = P^2 v = P(Pv) = P(\lambda v) = \lambda^2 v.$$

Taking the norm forces $\lambda = \lambda^2$ which implies $\lambda \in \{0, 1\}$. As the singular values of P are its eigenvalues (by the symmetry of P), it implies that $\|P\|_{\text{op}} \leq 1$. Moreover, writing the singular value decomposition of P leads to $P = \sum_i \sigma_i u_i v_i^\top$. Let u_k be in the range of P (it suffices to show that $Pv_k = \sigma_k u_k$ and since $P \neq 0$, there is at least one σ_k non-zero which provides at least one u_k in the range of P). Using $P^2 = P$ implies that

$$Pu_k = u_k \iff \sum_i \sigma_i u_i v_i^\top u_k = u_k \iff \sum_i (\sigma_i v_i^\top u_k) u_i = u_k.$$

Since the (u_i) forms an orthogonal basis, it forces $(\sigma_k v_k^\top u_k) = 1$. A simple application of Cauchy-Schwartz (recalling that the u_i, v_i have unit norm) leads to $v_k^\top u_k \leq 1$, and finally it implies that $\sigma_k \geq 1$. This forces by definition of the operator norm $\|P\|_{\text{op}} \geq 1$. Putting everything together gives $\|P\|_{\text{op}} = 1$.

For the last point, assume A has a singular value decomposition that writes

$$A = \sum_{i=1}^{\min\{n,m\}} \sigma_i u_i v_i^\top,$$

where the $u_i \in \mathbb{R}^n, v_i \in \mathbb{R}^m$ are orthogonal basis. Hence, we have

$$A^\top = \sum_{i=1}^{\min\{n,m\}} \sigma_i v_i u_i^\top,$$

which shows that the singular values are the same. Since the operator norm computes the largest singular value, we have the last equality. □

In the rest, we upper-bound all the terms by extensively using the monotonicity of the expectation.

Query term. We have

$$\begin{aligned}
\|\nabla_q \mathcal{L}(\theta)\|^2 &= \|\mathbb{E}_x\big[\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big)\mathscr{C}_x\big](\theta)\|^2 \\
&\leq \mathbb{E}_x\big[\|\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big)\mathscr{C}_x\|^2\big] && \text{(Jensen inequality since } \|\cdot\|^2 \text{ is convex)} \\
&\leq \mathbb{E}_x\big[\|\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big)\|_{\text{op}}^2 \|\mathscr{C}_x\|^2\big] && \text{(Point (2) of Lemma 8)}
\end{aligned}$$

Using the fact that operator norms are sub-multiplicative, we have

$$\|\Delta_z A_z (Vz)^\top \big(I_d + Q^\top\big)\|_{\text{op}} \leq \|\Delta_z\|_{\text{op}} \|A_z\|_{\text{op}} \|(Vz)^\top\|_{\text{op}} \|\big(I_d + Q^\top\big)\|_{\text{op}}.$$

Recalling from Eq. (4) that

$$\Delta_z = \frac{z}{\sqrt{d}}\left[I_N - \text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\right],$$

and using the fact that the embeddings $z \in \mathbb{R}^{d \times M}$ have columns in $\mathbb{S}^d$ (thanks to the RMS-norm), we have that $I_N - \text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top$ has entries bounded by 2. Indeed, given that

$$\forall t \in [N], \|z_{\cdot,t}\|^2 = \sum_{i=1}^d z_{dt}^2 = 1,$$

we know that the z_{dt} are bounded by one. As the softmax outputs a probability vector, the z_{dt} are reweighted by elements in $[0, 1]$. This implies that the entries of $\text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top$ (a matrix with rows equal to $\text{softmax}(\frac{z^\top q}{\sqrt{d}})$) are bounded by 1. Taking the minus sign into account leads to $I_N - \text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top$ with entries bounded by 2 and using the point (3) and (4) of Lemma 8, we have

$$\begin{aligned}
\|\Delta_z\|_{\text{op}} &\leq \frac{1}{\sqrt{d}}\|z\|_{\text{op}} \|I_N - \text{softmax}(\frac{z^\top q}{\sqrt{d}})\mathbb{1}_N^\top\|_{\text{op}} \\
&\leq \frac{1}{\sqrt{d}}\|z^\top\|_{\text{op}} \cdot (2N) && \text{(last point of Lemma 8)}
\end{aligned}$$

Moreover, using a proof similar to the point (4) of Lemma 8 (in particular involving Eq. (27)), one can show that for any $A \in \mathbb{R}^{n\times m}$,

$$\|A\|_{\text{op}} \leq \sqrt{m} \max_{i\in[m]} \|A_i\|.$$

Applying the previous inequality to $z^\top \in \mathbb{R}^{N\times d}$ that has rows $z_{\cdot,t}$ with unit norm leads to

$$\|z^\top\|_{\text{op}} \leq \sqrt{d}. \tag{28}$$

We thus obtain $\|\Delta_z\|_{\text{op}} \leq 2N$. Recalling from (2) that

$$A_z = \text{diag}\left(\text{softmax}(\frac{z^\top q}{\sqrt{d}})\right)$$

is diagonal with entries that form a probability vector, we know that

$$\|A_z\|_{\text{op}} = \sigma_{\max}(A_z) = \lambda_{\max}(A_z) = \max_{t\in[N]}\{\text{softmax}(\frac{z^\top q}{\sqrt{d}})_t\} \leq 1.$$

Then, using Eq. (28), we notice that

$$\|(Vz)^\top\|_{\text{op}} = \|z^\top V^\top\|_{\text{op}} \leq \|z^\top\|_{\text{op}}\|V^\top\|_{\text{op}} \leq \sqrt{d}\|V^\top\|_{\text{op}} = \sqrt{d}\|V\|_{\text{op}},$$

where the last equality comes from the last point of Lemma 8.

We now proceed to $I + Q^\top$. Using the triangular inequality leads to $\|I + Q^\top\|_{\text{op}} \leq \|I\|_{\text{op}} + \|Q^\top\|_{\text{op}}$. We know that $\|I\|_{\text{op}} = 1$. Moreover, we have

$$\|Q^\top\|_{\text{op}} = \|M(U\Sigma_{\bar{\xi}}W)^\top\|_{\text{op}} \leq \|M\|_{\text{op}} \cdot \|(U\Sigma_{\bar{\xi}}W)^\top\|_{\text{op}}.$$

Using the fact that M is an orthogonal projection ensures from the point (6) of Lemma 8 that $\|M\|_{\text{op}} = 1$. Moreover, using the points (3) and (6) of Lemma 8 leads to

$$\|(U\Sigma_{\bar{\xi}}W)^\top\|_{\text{op}} = \|U\Sigma_{\bar{\xi}}W\|_{\text{op}} \leq \|U\|_{\text{op}}\|\Sigma_{\bar{\xi}}\|_{\text{op}}\|W\|_{\text{op}}\|.$$

Noting that $\Sigma_{\bar{\xi}}$ is diagonal with entries $\sigma'(w_i^\top\bar{\xi})$ (Eq. (3)) and recalling that $\sigma'(x) = \psi(x) + xf(x)$ with φ the cumulative distribution function of the normal distribution and f its probability density function (in particular, φ has values in $[0, 1]$ and f in $[0, 1/\sqrt{2\pi}]$), we know that $|\sigma'(x)| \leq 1 + (1/\sqrt{2\pi})|x|$. Since σ' is applied entry-wise, using again the point (2) of Lemma 8 leads to

$$\|\Sigma_{\bar{\xi}}\|_{\text{op}} \leq 1 + \frac{1}{\sqrt{2\pi}}\|W\bar{\xi}\| \leq 1 + \frac{1}{\sqrt{2\pi}}\|W\|_{\text{op}}.$$

Putting everything together leads to

$$\|\Delta_z A_z(Vz)^\top\left(I_d + Q^\top\right)\|_{\text{op}} \leq 2n\sqrt{d}\|V\|_{\text{op}}(1 + \sqrt{\frac{2}{\pi}}\|W\|_{\text{op}}^2\|U\|_{\text{op}}).$$

Since the weight matrices are randomly initialized and learned by gradient descent whose updates have the form $\mathbb{E}_x[\varphi(\text{parameters})]$, they do not depend on the expectation with respect to the training data distribution. We can thus take those terms outside of the expectation which leads to

$$\|\nabla_q \mathcal{L}(\theta)\|^2 \leq \left[2n\sqrt{d}\|V\|_{\text{op}}(1 + \sqrt{\frac{2}{\pi}}\|W\|_{\text{op}}^2\|U\|_{\text{op}})\right]^2 \cdot \mathbb{E}_x\left[\|\mathscr{C}_x\|^2\right]. \tag{29}$$

Value term. We have

$$\begin{aligned}
\|\nabla_V \mathcal{L}(\theta)\|^2 &\leq \mathbb{E}_x\left[\|\underbrace{\left(I_d + Q^\top\right)\mathscr{C}_x}_{\in\mathbb{R}^d}\underbrace{\left[z\,\text{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top}_{\in\mathbb{R}^{1\times d}}\|^2\right] && \text{(Jensen inequality)}\\
&\leq \mathbb{E}_x\left[\|\left(I_d + Q^\top\right)\mathscr{C}_x\|^2\left\|\left[z\,\text{softmax}(\frac{z^\top q}{\sqrt{d}})\right]^\top\right\|^2\right] && \text{(Point (1) of Lemma 8)}
\end{aligned}$$

Again, using point (2) of Lemma 8 and the derivations done for the query term, we have

$$\|\big(I_d + Q^\top\big)\mathscr{C}_x\| \leq \|\big(I_d + Q^\top\big)\|_{\mathrm{op}} \cdot \|\mathscr{C}_x\| \leq (1 + 2\|W\|_{\mathrm{op}}\|U\|_{\mathrm{op}}) \cdot \|\mathscr{C}_x\|.$$

Similarly, we have

$$\left\| \left[z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \right]^\top \right\| = \left\| z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \right\| \leq \|z\|_{\mathrm{op}} \|\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})\|.$$

As shown in the proof of the query term, we have $\|z\|_{\mathrm{op}} = \|z^\top\|_{\mathrm{op}} \leq \sqrt{d}$ and $p = \mathrm{softmax}(\frac{z^\top q}{\sqrt{d}})$ is a probability vector which implies

$$\|p\|^2 = \sum_{i=1}^{N} p_i^2 \leq \sum_{i=1}^{N} p_i = 1,$$

where the first inequality comes from $x \mapsto x^2 \leq x \mapsto x$ on $[0, 1]$. It follows that

$$\left\| \left[z\,\mathrm{softmax}(\frac{z^\top q}{\sqrt{d}}) \right]^\top \right\| \leq \sqrt{d}.$$

In summary, with the same arguments as before, we can take the weight terms outside of the expectation and we obtain

$$\|\nabla_V \mathcal{L}(\theta)\|^2 \leq d(1 + 2\|W\|_{\mathrm{op}}\|U\|_{\mathrm{op}})^2 \cdot \mathbb{E}_x\big[\|\mathscr{C}_x\|^2\big]. \tag{30}$$

Receptors terms. With similar arguments, we have

$$\begin{aligned}
\|\nabla_W \mathcal{L}(\theta)\|^2 &= \|\mathbb{E}_x\big[\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x \bar{\xi}^\top\big]\|^2 \\
&\leq \mathbb{E}_x\left[\|\underbrace{\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x}_{\in \mathbb{R}^d} \underbrace{\bar{\xi}^\top}_{\in \mathbb{R}^{1\times d}}\|^2 \right] && \text{(Jensen)} \\
&\leq \mathbb{E}_x\big[\|\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x\|^2 \|\bar{\xi}^\top\|^2\big] && \text{(Point (1) of Lemma 8)}
\end{aligned}$$

By definition, $\bar{\xi} = \xi/\|\xi\|$ has unit norm. Moreover, using the point (2) of Lemma 8, we know that

$$\|\Sigma_{\bar{\xi}} U^\top \mathscr{C}_x\| \leq \|\Sigma_{\bar{\xi}} U^\top\|_{\mathrm{op}} \cdot \|\mathscr{C}_x\| \leq \|\Sigma_{\bar{\xi}}\|_{\mathrm{op}} \cdot \|U^\top\|_{\mathrm{op}} \cdot \|\mathscr{C}_x\|.$$

where the last inequality comes from the sub-multiplicity of the operator norm. As we showed in the deviation of the query term, we have $\|\Sigma_{\bar{\xi}}\|_{\mathrm{op}} \leq 2$ and we know that $\|U^\top\|_{\mathrm{op}} = \|U\|_{\mathrm{op}}$. In summary, taking the weight terms outside of the expectation, we have

$$\|\nabla_W \mathcal{L}(\theta)\|^2 \leq 4\|U\|_{\mathrm{op}}^2 \mathbb{E}_x\big[\|\mathscr{C}_x\|^2\big]. \tag{31}$$

Assemblers terms. The same arguments as before lead to

$$\|\nabla_U \mathcal{L}(\theta)\|^2 = \|\mathbb{E}_x\big[\mathscr{C}_x \sigma(W\bar{\xi})^\top\big]\|^2 \leq \mathbb{E}_x\big[\|\mathscr{C}_x \sigma(W\bar{\xi})^\top\|^2\big] \leq \mathbb{E}_x\big[\|\mathscr{C}_x\|^2 \cdot \|\sigma(W\bar{\xi})^\top\|^2\big]$$

where the last inequality comes from the first point of Lemma 8. Recalling that the GeLU function is defined as $x \leq x\varphi(x)$ with φ the cumulative distribution function of the standard normal distribution (in particular, φ outputs values in [0, 1]), we know that for any real number x, $|\sigma(x)| \leq |x|$. Using again the point (2) of Lemma 8 and that $\bar{\xi}$ has unit norm, since σ is applied entry-wise, we have

$$\|\sigma(W\bar{\xi})^\top\| = \|\sigma(W\bar{\xi})\| \leq \|W\bar{\xi}\| \leq \|W\|_{\mathrm{op}}.$$

In summary, taking the weight terms outside of the expectation, we have

$$\|\nabla_U \mathcal{L}(\theta)\|^2 \leq \|W\|_{\mathrm{op}}^2 \mathbb{E}_x\big[\|\mathscr{C}_x\|^2\big]. \tag{32}$$

Conclusion. Putting (29), (30), (31) and (32) together and using (26) leads to

$$\|\nabla_\theta \mathcal{L}(\theta)\|^2 \leq \left[\left[2n\sqrt{d}\|V\|_{\mathrm{op}}(1+\sqrt{\frac{2}{\pi}}\|W\|_{\mathrm{op}}^2\|U\|_{\mathrm{op}})\right]^2 + d(1+2\|W\|_{\mathrm{op}}\|U\|_{\mathrm{op}})^2 + 4\|U\|_{\mathrm{op}}^2 + \|W\|_{\mathrm{op}}^2\right] \cdot \mathbb{E}_x\left[\|\mathscr{C}_x\|^2\right]. \tag{33}$$

Finally, denoting $\gamma_j = \hat{p}_\theta(y=j|x) - p(y=j|x)$ to ease notations, we notice that

$$\begin{aligned}
\|\mathscr{C}_x\|^2 &= \left\|\sum_{j=1}^p (\hat{p}_\theta(y=j|x) - p(y=j|x))E(j)\right\|^2 \\
&= \left\|\sum_{j=1}^p \gamma_j E(j)\right\|^2 \\
&= \sum_{l=1}^d \left(\sum_{j=1}^p \gamma_j E(j)_l\right)^2 \\
&\leq \sum_{l=1}^d \left(\sum_{j=1}^p |\gamma_j||E(j)_l|\right)^2 \\
&\leq \left(\sum_{l=1}^d \max_{j\in[p]} |E(j)_l|^2\right)\left(\sum_{j=1}^p |\gamma_j|\right)^2 \\
&\leq dB^2\left(\sum_{j=1}^p |\gamma_j|\right)^2. && \text{(using } \|E\|_\infty \leq B\text{)}
\end{aligned}$$

Recalling that the total variation distance verifies for any probability vectors p, q

$$d_{\mathrm{TV}}(p,q) = \frac{1}{2}\|p-q\|_1 = \frac{1}{2}\sum_{\omega\in\Omega}|p(\omega)-q(\omega)|,$$

with $\|\cdot\|_1$ the ℓ_1-norm. We notice that

$$\sum_{j=1}^p |\gamma_j| = \sum_{j=1}^p |\hat{p}_\theta(y=j|x) - p(y=j|x)| = \|\hat{p}_\theta(\cdot|x) - p(\cdot|x)\|_1 = 2\cdot d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x)).$$

Recalling that the total variation distance takes values in $[0,1]$ and that $x \mapsto x^2 \leq x \mapsto x$ on this interval, we know that $d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))^2 \leq d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))$. It leads to

$$\mathbb{E}_x\left[\|\mathscr{C}_x\|^2\right] \leq 4dB^2\mathbb{E}_x[d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))].$$

Rearranging the terms in front of the expectation in (33) finally leads to

$$\|\nabla_\theta \mathcal{L}(\theta)\|^2 \leq \hat{B}\cdot\mathbb{E}_x[d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))]$$

where $\hat{B} = 4dB^2\left[\left[2n\sqrt{d}\|V\|_{\mathrm{op}}(1+\sqrt{\frac{2}{\pi}}\|W\|_{\mathrm{op}}^2\|U\|_{\mathrm{op}})\right]^2 + d(1+2\|W\|_{\mathrm{op}}\|U\|_{\mathrm{op}})^2 + 4\|U\|_{\mathrm{op}}^2 + \|W\|_{\mathrm{op}}^2\right]$. Noticing that $\hat{B} = \mathcal{O}\Big(B^2\Big[\|V\|_{\mathrm{op}}\big[1+\|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}^2\big]^2 + [1+\|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}]^2 + \|U\|_{\mathrm{op}}^2 + \|W\|_{\mathrm{op}}^2\Big]\Big)$ and taking the square root leads to

$$\|\nabla_\theta \mathcal{L}(\theta)\| = \mathcal{O}\left(B\sqrt{\|V\|_{\mathrm{op}}\big[1+\|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}^2\big]^2 + [1+\|U\|_{\mathrm{op}}\|W\|_{\mathrm{op}}]^2 + \|U\|_{\mathrm{op}}^2 + \|W\|_{\mathrm{op}}^2}\cdot\sqrt{\mathbb{E}_x[d_{\mathrm{TV}}(\hat{p}_\theta(\cdot|x), p(\cdot|x))]}\right),$$

which concludes the proof. □