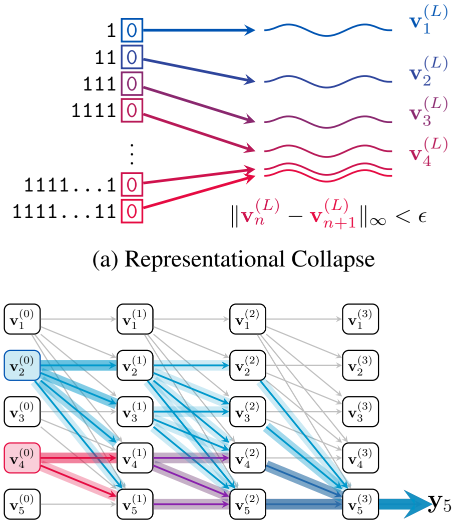
Transformers need glasses! 👓 Information over-squashing in language tasks

Federico Barbero [1 2] Andrea Banino [1] Steven Kapturowski [1] Dharshan Kumaran [1] João G.M. Araújo [1]
Alex Vitvitskyi [1] Razvan Pascanu [1] Petar Veličković [1]

Abstract

We study how information propagates in decoder-only Transformers, which are the architectural backbone of most existing frontier large language models (LLMs). We rely on a theoretical signal propagation analysis—specifically, we analyse the representations of the last token in the final layer of the Transformer, as this is the representation used for next-token prediction. Our analysis reveals a *representational collapse* phenomenon: we prove that certain distinct sequences of inputs to the Transformer can yield arbitrarily close representations in the final token. This effect is exacerbated by the low-precision floating-point formats frequently used in modern LLMs. As a result, the model is provably unable to respond to these sequences in different ways—leading to errors in, e.g., tasks involving counting or copying. Further, we show that decoder-only Transformer language models can lose sensitivity to specific tokens in the input, which relates to the well-known phenomenon of *over-squashing* in graph neural networks. We provide empirical evidence supporting our claims on contemporary LLMs. Our theory also points to simple solutions towards ameliorating these issues. **For the most up-to-date version**, we point to `https://arxiv.org/abs/2406.04267`.


1 0
11 0
111 0
1111 0
⋮
1111...1 0
1111...11 0
$\mathbf{v}_1^{(L)}$
$\mathbf{v}_2^{(L)}$
$\mathbf{v}_3^{(L)}$
$\mathbf{v}_4^{(L)}$
$\|\mathbf{v}_n^{(L)} - \mathbf{v}_{n+1}^{(L)}\|_\infty < \epsilon$
$\mathbf{v}_1^{(0)}$ $\mathbf{v}_2^{(0)}$ $\mathbf{v}_3^{(0)}$ $\mathbf{v}_4^{(0)}$ $\mathbf{v}_5^{(0)}$
$\mathbf{v}_1^{(1)}$ $\mathbf{v}_2^{(1)}$ $\mathbf{v}_3^{(1)}$ $\mathbf{v}_4^{(1)}$ $\mathbf{v}_5^{(1)}$
$\mathbf{v}_1^{(2)}$ $\mathbf{v}_2^{(2)}$ $\mathbf{v}_3^{(2)}$ $\mathbf{v}_4^{(2)}$ $\mathbf{v}_5^{(2)}$
$\mathbf{v}_1^{(3)}$ $\mathbf{v}_2^{(3)}$ $\mathbf{v}_3^{(3)}$ $\mathbf{v}_4^{(3)}$ $\mathbf{v}_5^{(3)}$
$\mathbf{y}_5$

(a) Representational Collapse

(b) Over-squashing

Figure 1. **(a) Representational Collapse** (Theorem 4.2). Sequences given to Transformer architectures comprising repeated `1` tokens with a single `0` token at the end. The color and proximity of the curved lines illustrate how these representations converge as sequence length increases. **(b) Over-squashing** (Theorem 5.1). Tokens earlier in their input sequence will have significantly more paths through which their data can reach the representation used for next-token prediction, leading to 'over-squashing'. This effect is depicted here for an early token (blue) and later token (red) in a five-token sequence.

1. Introduction

In recent years the field of Natural Language Processing (NLP) has been revolutionised through the introduction of Transformer-based architectures (Vaswani et al., 2017). Large Transformers trained on some version of next-token prediction, known as *Large* Language Models (LLMs), have demonstrated impressive performance across different tasks, including conversational agents (Gemini, 2023; OpenAI, 2023), understanding multi-modal inputs (Alayrac et al., 2022), and code completion (Li et al., 2022). Most contemporary LLMs specifically focus on the decoder part of the original Transformer architecture, and are commonly referred to as *decoder-only* Transformers. Consequently, we focus primarily on such models in this paper.

In this work, we study what information *can* be contained in the representation of the last token at the last layer, as this is ultimately the information that will be used for next-token prediction — the fundamental mechanism through which modern Transformer LLMs perform training and inference. In particular, we show that for certain distinct sequences,

[*]Equal contribution [1]Google DeepMind, London [2]University of Oxford, Department of Computer Science, Oxford. Correspondence to: Federico Barbero <federico.barbero@cs.ox.ac.uk>.

Work presented at TF2M workshop at ICML 2024, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

their last-token representations can become arbitrarily close to each other. This leads to a *representational collapse*, exacerbated by the lower-precision floating point types typically used by modern LLM stacks. As a result Transformers incorrectly produce the same tokens on these sequence pairs — see Figure 1 (a). We also show that there is a related phenomenon of *over-squashing* — see Figure 1 (b), an effect that is well studied in graph neural networks (GNNs) (Di Giovanni et al., 2023; Giovanni et al., 2024), and related to vanishing gradients (Bengio et al., 1994).

In summary, our paper provides the following contributions:

- Theoretical analysis of decoder-only Transformer limitations: we formalise the concepts of 'representational collapse' (Section 4) and 'over-squashing' (Section 5) in the context of Transformer-based architectures.
- Impact of floating point precision: we explore how low floating-point precision exacerbate the identified theoretical issues, causing them to manifest even in relatively short input sequences.
- Empirical validation of theoretical analysis: our theoretical findings are supported by real-world experiments conducted on contemporary LLMs, demonstrating practical implications of the limitations we identified.

2. Background

In this work, we study a class of Transformers which we believe forms the basis for a large number of current LLMs. We let $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n \times d}$ be the query, key, and value matrices respectively on n tokens and d dimensions. We denote with $\mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$ the d-dimensional query, key, and value vectors of the i-th token. We let $\mathbf{p}_{ij} \in \mathbb{R}^{2e}$ be the $2e$-dimensional positional encoding information between tokens i and j. We focus on the case in which the positional encodings are *bounded*, which is the case for the large majority of positional encodings used in practice (Su et al., 2024; Vaswani et al., 2017). The Transformer model we consider computes the values, for a single head, of the i-th token at the ℓ-th Transformer layer $\mathbf{v}_i^{(\ell)}$ as [1]

$$\mathbf{z}_i^{(\ell)} = \sum_{j \leqslant i} \alpha_{ij}^{(\ell)} \operatorname{norm}_1^{(\ell)} \left(\mathbf{v}_i^{(\ell)} \right) + \mathbf{v}_i^{(\ell)},$$

$$\text{with } \alpha_{ij}^{(\ell)} = \frac{\exp\left(k\left(\mathbf{q}_i^{(\ell)}, \mathbf{k}_j^{(\ell)}, \mathbf{p}_{ij}\right)\right)}{\sum_{w \leqslant i} \exp\left(k\left(\mathbf{q}_i^{(\ell)}, \mathbf{k}_w^{(\ell)}, \mathbf{p}_{iw}\right)\right)}$$

$$\mathbf{v}_i^{(\ell+1)} = \boldsymbol{\psi}^{(\ell)}\left(\operatorname{norm}_2^{(\ell)}\left(\mathbf{z}_i^{(\ell)}\right)\right) + \mathbf{z}_i^{(\ell)}$$

for a function $k : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^{2e} \to \mathbb{R}$ mapping queries, key, and positional encoding information to a scalar value, an MLP $\boldsymbol{\psi} : \mathbb{R}^d \to \mathbb{R}^d$, and normalization functions at the ℓ-th layer $\operatorname{norm}_1^{(\ell)}$ and $\operatorname{norm}_2^{(\ell)}$. This specific interleaving of components is often referred to as a Pre-LN Transformer (Xiong et al., 2020). We can view the output of the ℓ-th layer of a Transformer as a sequence of d-dimensional vectors $\mathbf{v}^{(\ell)} = (\mathbf{v}_1^{(\ell)}, \ldots, \mathbf{v}_n^{(\ell)})$. Importantly, due to the causal attention mechanism, the vector $\mathbf{v}_j^{(\ell)}$, will only depend on elements $\mathbf{v}_i^{(\ell-1)}$ for $i \leqslant j$. We define the attention matrix at the ℓ-th layer element-wise as $\mathbf{\Lambda}_{ij}^{(\ell)} = \alpha_{ij}^{(\ell)}$, this is a row-stochastic triangular matrix that can also be interpreted as a probabilistic directed graph. After the last transformer block a normalization is applied to the token representations: $\mathbf{y}_i = \operatorname{norm}_3\left(\mathbf{v}_i^{(L)}\right)$. We note that the next-token prediction usually depends purely on $\mathbf{y}_n$—the final representation of the last token. We refer to the Appendix (Section A) for a literature review.

3. Motivating Examples

We start by providing motivating examples that show surprisingly simple failure cases of frontier LLMs specifically on copying (Section 3.1) and counting (Section 3.2) tasks. By copying we specifically mean tasks that involve the 'copying' or 'recalling' of a single or multiple tokens from the prompt. Instead, by *counting*, we mean the task of counting how many times a specific token appears in a sequence. We focus our evaluation on Gemini 1.5 (Gemini, 2023) as our frontier LLM (referred as Gemini) and later analyse the internal representations of the open-sourced Gemma model (Team et al., 2024). The goal is to showcase intriguing failure cases which will motivate our signal propagation analysis.

3.1. Copying

We study cases in which the LLM is prompted to copy tokens either at the *start* or at the *end* of a sequence. We avoid tasks that involve the copy of tokens at the 'n-th' position as most frontier LLMs do not have absolute positional information, making it very challenging for them to solve tasks that require absolute position. We focus specifically on sequences of zeros and ones growing in length with specific patterns. The design our experiments is meant to evaluate how well Transformers are able to capture and propagate information of *individual* tokens.

In Figure 2 (a), we prompt Gemini to return the last element of a sequences '$1 \ldots 10$' or the first element of a sequence '$01 \ldots 1$'. The answer for both is zero, but we progressively grow the number of ones. We observe how the task seems considerably easier when asked to return the first rather than

[1] Note that we rely on an abuse of notation. We ignore the linear projections used to compute the value $\mathbf{v}_i^{(l)}$ from the output of layer below $l-1$. This will not change our derivations, but would otherwise make notations cumbersome.

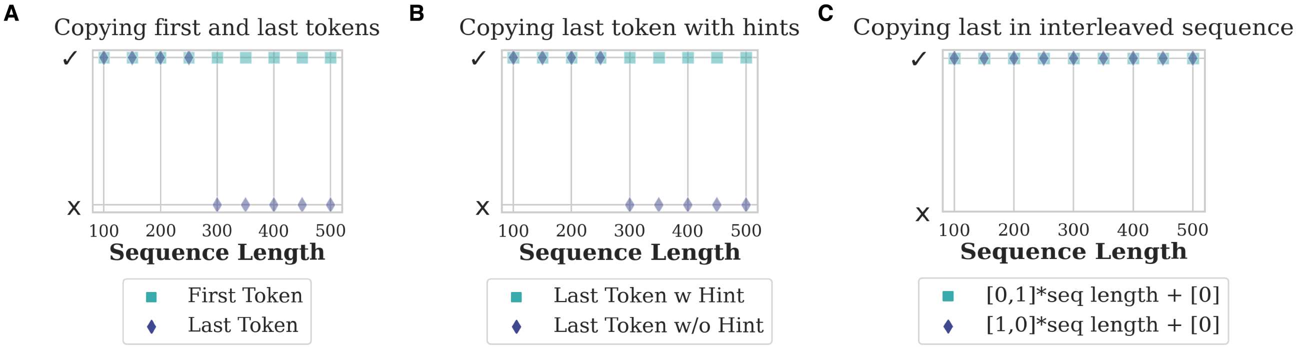


Figure 2. Results on simple copying tasks. (a). Gemini was prompted to predict the last token (diamond) of a sequences '1...10' or the first token (square) of a sequence '01...1'. (b). Same as (a) but with hints (see 3.2 for details) (c). Same as (a) but the sequences have interleaved 0s and 1s. See F.1 for extra details

the last element. Surprisingly, already at a sequence length of only 300 elements, Gemini incorrectly starts to output 'one' when trying to copy the last element. In Figure 2 (b), we show that providing hints in the form of: ' *Hint* It's not necessarily a 1, check carefully', helps significantly with the performance. Finally, in Figure 2 (c), we show that replacing the constant sequence of ones with alternating ones and zeros seems to also help. We refer to the Appendix (Section F.1) for further details on the experiments.

3.2. Counting

We consider four different tasks: (i) Summing $1 + \cdots + 1$, (ii) Counting the number of ones in a sequence of ones, (iii) Counting the number of ones in a sequence of ones and zeros, with ones being sampled with 70% probability, and (iv) Counting the number of times a specific word appears in a sentence. We consider predictions of an LLM which (1) Is instructed to only output the answer, (2) Is prompted to break down the problem (CoT-no-shot), and (3) Is prompted to break down the problem with few-shot in-context examples (CoT-few-shot). We refer to the Appendix (Section F.1) for a more detailed description of the tasks.

Results are presented in Figure 3. It is clear that the performance rapidly deteriorates with the sequence length. It is also interesting to see that the error seems to increase with the sequence very rapidly. For instance in task (i), the LLM is quite likely to predict the value of '100' once the sequence reaches a size around or larger than 100. Such an observation provides motivating evidence for the argument that Transformers may not be in fact mechanically counting but rather perform a type of crude subitizing. This explains why arguably 'common' numbers such as 100 are much more likely to be outputted by the LLM and why in tasks such as (i) and (ii) the values near 100 have relatively lower error. This does not happen in task (iii) as the response should actually be around 70% of the sequence length due to the sequence sampling procedure. This explains why the absolute error actually seems to increase around a sequence length of 100. In the Appendix (Section F.2), we show how 100 is by far the most common response — a surprising observation that points to the fact that LLMs may not be mechanically counting, but rather outputting a 'likely value'. In the Appendix (Section D), we provide a theoretical discussion of the counting problem which stems from the theory we develop in the next sections.

4. Representational Collapse

We start our theoretical analysis by showcasing a type of loss of information which we call *representational collapse*. More precisely, we show that under certain conditions, we can find distinct sequences such that their final representations of the last token at the last layer *become arbitrarily close* as the sequence length increases. As Transformer models operate over finite machine precision, this points to a fundamental representational incapacity of Transformers to distinguish certain prompts if the sequence is long enough. We start by showing Lemma 4.1, which is a statement about the increase of entropy in softmax layers with growing sequence size. We point to the Appendix (Lemma E.1) for the complete statement.

Lemma 4.1 (Informal)**.** *Consider a vector* $\mathbf{a} \in \mathbb{R}^{n-1}$ *and two scalars* $b, c \in \mathbb{R}$*. Let* $\mathbf{x} = [\mathbf{a}\ c]^T \in \mathbb{R}^n$ *and* $\mathbf{x}^* = [\mathbf{a}\ b\ c]^T \in \mathbb{R}^{n+1}$*. Then, the softmax value for the last token at* $\mathbf{x}^*$ *is strictly smaller than that of the last token of* $\mathbf{x}$*. Moreover, for large enough* n*, their difference is arbitrarily small.*

We now show, that we can find distinct sequences that will have arbitrarily close final representations. In particular, as language models often operate in low floating regimes, i.e. `bf16`, this can practically become catastrophic. The result is summarised in Theorem 4.2, which describes what we call representational collapse in this work. The complete statement is reported in the Appendix (Theorem E.4).

Theorem 4.2 (Representational Collapse – informal)**.** *Let* $\mathbf{v}^{(0)} \in \mathbb{R}^{n \times d}$ *be a sequence and* $\mathbf{v}^{*(0)} \in \mathbb{R}^{(n+1) \times d}$ *be an-*

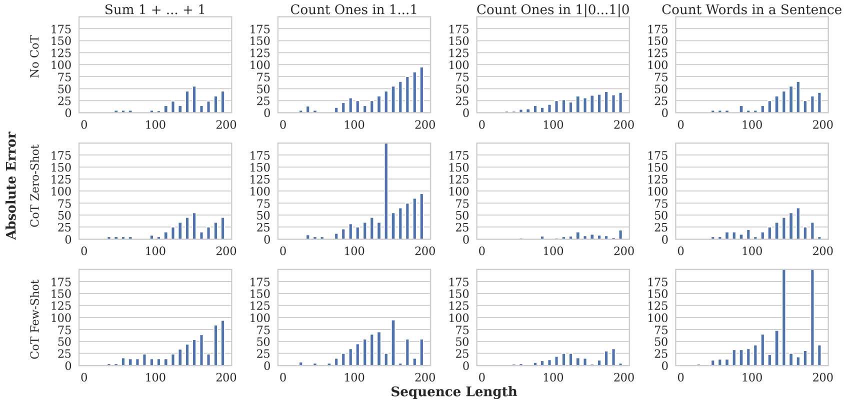


Figure 3. LLMs being prompted to (i) sum $1 + \cdots + 1$ (left),(ii) Count the number of ones in a sequence of 1s (center), and (iii) Count the number of ones in a sequence of ones and zeroes (the sequence is a Bernoulli sequence with probability of sampling a one being 0.7) (right).

other sequence equal to $\mathbf{v}^{(0)}$ with the last token of $\mathbf{v}^{(0)}$ repeated. Then, their representations become arbitrarily close as n increases.

Theorem 4.2 shows that it becomes increasingly challenging for a Transformer to distinguish two sequences that only differ via a repeated last token. We note that the repetition of the last token is a technical consideration to show this direct representational collapse. As we will later show in Section 5.1, it is particularly problematic *in general* to depend on the last token due to a type of topological 'squashing' present in decoder-only Transformers. In the Appendix (Section B), we provide a practical discussion on representational collapse, alongside experimental evidence showcasing expeirmental collapse occuring in Gemma (Team et al., 2024).

5. Over-squashing in Language Tasks

We study the quantity $\partial \mathbf{y}_n / \partial \mathbf{v}_i^{(0)}$ which measures how sensitive is the final token to an input token at position i. In graph neural network theory, the decay of such a partial derivative is often associated with the 'squashing' of information, leading to the phenomenon of *over-squashing* (Topping et al., 2022; Di Giovanni et al., 2023; Giovanni et al., 2024). The over-squashing analysis we carry out in this work is particularly challenging due to the flexible nature of the attention mechanism and the many components that are part of decoder-only Transformers. Consequently, we make two simplifying assumptions in our analysis: (i) We summarise the effect of layer normalisation via a constant β_i for the i-th layer norm component, and (ii) the attention weights are treated as independent of the input. Such simplifications are not strictly necessary for our analysis, but they greatly simplify the resulting bound we derive and do not detract from the two key takeaways: **(1) the sensitivity to an input token depends on its position in the sequence and (2) the sensitivity to an input token depends on the attention weights**. The result is summarised in Theorem 5.1. The full statement is reported in the Appendix (Theorem E.5).

Theorem 5.1 (Over-squashing in Transformers)**.** *Consider an input sequence $\mathbf{v}_1^{(0)}, \dots, \mathbf{v}_n^{(0)}$. Let $C > 0$ be some constant and $\bar{\alpha}_{i,j}^{(\ell)} = \frac{\alpha_{i,j}^{(\ell)}}{\beta_2} + \delta_{i,j}$, then:*

$$\left\| \frac{\partial \mathbf{y}_n}{\partial \mathbf{v}_i^{(0)}} \right\| \leqslant C \sum_{k_1 \geqslant i} \cdots \sum_{k_L \geqslant k_{L-1}} \bar{\alpha}_{n,k_L}^{(L-1)} \prod_{\ell=2}^{L-1} \bar{\alpha}_{k_\ell,k_{\ell-1}}^{(\ell-1)} \bar{\alpha}_{k_1,i}^{(0)} \tag{1}$$

Theorem 5.1 provides intuition on how information propagates in a decoder-only Transformer. In particular, there is a topological aspect present in the bound which is directly controlled by the attention mechanism. More concretely, the sensitivity depends on the sum of the weighted paths between the token i at the input and the final layer. *In other words, for tokens coming sooner in the sequence, there will be more opportunity for their information to be preserved.* This is clear for instance for the last token, which will only be preserved by attention mechanism if the attention $n \to n$ is large at every layer L, i.e. there is only one path. The paths instead grow very quickly for tokens coming sooner in the sequence. In the Appendix (Section C), we provide a theoretical connection between over-squashing and the spectral theory of random walks, alongside a discussion on the observed 'U-shape' effect.

6. Conclusion and Future Work

In this work, we first present surprising failure cases of LLMs on simple copying and counting tasks. We then discussed how such failure cases can be explained by studying what *can be contained* inside the representation $\mathbf{y}_n$ and in particular how information may be lost. This lead to the unvealing of two phenomena : representational collapse and over-squashing. We showed how we can measure these phenomena in practice and proposed simple solutions to help alleviate such information loss. We hope that the findings may help better understand and improve language models available today.

References

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=i80OPhOCVH2.

Barbero, F., Velingker, A., Saberi, A., Bronstein, M. M., and Giovanni, F. D. Locality-aware graph rewiring in GNNs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4Ua4hKiAJX.

Bengio, Y., Simard, P., and Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Łukasz Kaiser. Universal transformers, 2019.

Delétang, G., Ruoss, A., Grau-Moya, J., Genewein, T., Wenliang, L. K., Catt, E., Cundy, C., Hutter, M., Legg, S., Veness, J., and Ortega, P. A. Neural networks and the chomsky hierarchy, 2023.

Di Giovanni, F., Giusti, L., Barbero, F., Luise, G., Lio, P., and Bronstein, M. M. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *International Conference on Machine Learning*, pp. 7865–7885. PMLR, 2023.

Dudzik, A. J., von Glehn, T., Pascanu, R., and Veličković, P. Asynchronous algorithmic alignment with cocycles. In *Learning on Graphs Conference*, pp. 3–1. PMLR, 2024.

Gemini, T. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Giovanni, F. D., Rusch, T. K., Bronstein, M., Deac, A., Lackenby, M., Mishra, S., and Veličković, P. How does over-squashing affect the power of GNNs? *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=KJRoQvRWNs.

Hahn, M. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020.

Herranz-Celotti, L. and Rouat, J. Stabilizing rnn gradients through pre-training, 2024.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Merrill, W. and Sabharwal, A. The expresssive power of transformers with chain of thought. *arXiv preprint arXiv:2310.07923*, 2023.

OpenAI, R. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5), 2023.

Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. In Dasgupta, S. and McAllester, D. (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1310–1318, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

Peng, B., Narayanan, S., and Papadimitriou, C. On limitations of the transformer architecture, 2024.

Pérez, J., Barceló, P., and Marinkovic, J. Attention is turing-complete. *Journal of Machine Learning Research*, 22 (75):1–35, 2021.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Team, G., Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M. S., Love, J., et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Topping, J., Di Giovanni, F., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Weiss, G., Goldberg, Y., and Yahav, E. Thinking like transformers, 2021.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Zhai, S., Likhomanenko, T., Littwin, E., Busbridge, D., Ramapuram, J., Zhang, Y., Gu, J., and Susskind, J. M. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pp. 40770–40803. PMLR, 2023.

A. Literature Review

Existing theory on Transformers. The theoretical representational capacity of Transformers has become a popular area of study, providing interesting results on what classes of problems they are able to model. In (Dehghani et al., 2019) the fact that transformers are not Turing Complete is first formally stated, and a simple proof is given, while the the focus of the paper is augmenting the transformer architecture to allow for it to compute any computable function. In (Pérez et al., 2021) the authors show that *the class* of transformers with hard attention, where instead of a softmax one uses one-hot vectors, and infinite precision is Turing Complete. This contrasts with our work, which focuses on the more standard setting of transformers using soft-attention and finite precision, and shows the limitations imposed by it.

Some papers have tried to study transformers capabilities through the lens of formal languages, such as (Weiss et al., 2021), which develops a computational model of what transformers can represent in an analogous way to how Recurrent Neural Networks are associated with finite automata, and then derive an implementable programming language that represents that model. Following that (Delétang et al., 2023) place transformers within the Chomsky Hierarchy, showing that they are quite limited and cannot learn the decision problem for simple languages, which prompted (Merrill & Sabharwal, 2023) to show that transformer LLMs can perform substantially better if they generate a number of decoding tokens linear in the problem input size, through scratch-pad, chain of thought or similar, but there are still more complex languages which they fail to learn the decision problem. Finally (Peng et al., 2024) shows that the transformer block with finite precision is fundamentally limited in its ability to represent compositional functions and solve simple problems that require it. Our work will similarly analyze transformer's inability to solve simple computational tasks, and proof that even with techniques like Chain-of-Thought that inability persists as it is inherent to the combination of architecture, next-token prediction, and limited floating point precision.

Decay in attention mechanisms. Others have studied the limitations of self-attention by showing that it can reach pathological states that limit what transformers are able to learn. In (Zhai et al., 2023) the authors show how a great reduction in the attention entropy can lead to unstable training if occurring early, but even when occurring later in training it can still lead to significantly lower performance. Theoretical Limitations of Self-Attention in Neural Sequence Models(Hahn, 2020) further shows how specific tokens can strongly concentrate attention, leading to transformers being unable to learn to process simple languages, like PARITY and DYCK, our work will similarly focus on showing how transformers end-up effectively ignoring many tokens in their input which leads them to fail to solve simple computational problems.

Over-squashing. Graph neural networks (GNNs) are neural networks designed to operate over graph structures. Importantly, Transformers, may be seen as types of attention-based GNNs operating over specific types of graphs. The difficulties of propagating information over a graph have been thoroughly analysed, with a notable phenomenon being that of *over-squashing* (Alon & Yahav, 2021; Topping et al., 2022; Di Giovanni et al., 2023; Barbero et al., 2024). Over-squashing refers to the fact that propagating information over certain graphs that exhbit 'bottlenecks' is likely to induce a 'squashing' of information. This can be made more precise by studying this effect via the notion of a *commute time* (Giovanni et al., 2024) — the expected number of steps that a random walk takes to travel from a node to another node and back. Information travelling between nodes with higher commute time will be squashed more.

A common way to measure over-squashing is by looking at how *sensitive* the representation $\mathbf{x}_v^{(L)}$ of a node v after L GNN layers is to the initial representation $\mathbf{x}_u^{(0)}$ of another node u. In particular, the partial derivative $\partial \mathbf{x}_v^{(L)} / \partial \mathbf{x}_u^{(0)}$ may be shown to decay, especially for nodes with high commute times between them. Our work may be seen as acting as a bridge between the well-studied phenomenon of over-squashing in GNNs and the loss of information we analyse in decoder-only Transformers specifically for language tasks. Note that this type of derivation is typical in the study of *vanishing gradients* for recurrent models as well (Hochreiter & Schmidhuber, 1997; Bengio et al., 1994; Pascanu et al., 2013; Herranz-Celotti & Rouat, 2024).

B. Representational Collapse Discussion

Measuring representational collapse. We report experiments showcasing representational collapse by measuring the internal representations of Gemma 7B (Team et al., 2024). For two sequences $\mathbf{v}^{(0)}$ and $\mathbf{v}^{*(0)}$ we report their difference in representation at the last layer $\left\|\mathbf{v}^{(L)} - \mathbf{v}^{*(L)}\right\|_\infty$ averaged out over each head. Figure 4 shows the collapse occuring on (a) prompting the model to count the number of ones in a sequence of ones, with one having an additional one, and (b) prompting the model to count the number of ones for a sequences with digits sampled uniformly ending with either a single one or two ones. The repeated digits seem to make the collapse occur much sooner with a sequence length of around 50 being near machine precision. The repeated digits seem to delay such a collapse, but a downward trend is maintained with respect to the sequence length.

Quantisation and Tokenisation. A common technique used to speedup the inference of an LLM is that of *quantisation*, a process that constructs an approximate version of an LLM that operates over lower precision datatypes. This helps drastically improve the inference speed of LLMs as modern accelerators produce significantly more FLOPs over lower precision datatypes. Of course quantisation usually comes at a cost. Our theoretical analysis points to a potentially catastrophic loss in representation due to quantisation. In particular, a lower machine precision will mean that the convergence of representations in Theorem 4.2 will occur much sooner, and that the LLM will not be able to distinguish even shorter sequences.

In practice, the direct application of theoretical results is made more complicated due to the tokeniser. In particular, a sequence of repeated tokens '11111' for instance may not be necessarily tokenised into 5 distinct '1' tokens. In principle, this should help alleviate the direct collapse of the representations. Tokenisation in general makes it more challenging to study such phenomena as it adds an additional layer of complexity to the analysis.

A simple solution to representational collapse. An important consequence of Theorem 4.2 is that *it is challenging for a Transformer to deal with a long sequence of repeated tokens*. A practical solution is to this issue is to introduce additional tokens throughout the sequence to help keep the representations distant. We provide direct evidence of this in Figure 4 (c,d), where we prompt the model on a simple copying task of a long string of ones. While the representations collapse for the sequence of ones (c), adding commas every third digit (d) helps to keep the representations well-separated.

C. Over-squashing Discussion

The over-squashing analysis leads to an interesting limiting case described in Proposition C.1, that shows a type of exponential vanishing that can occur in some degenerate cases in which $\mathbf{y}_n$ depends only on the starting input token $\mathbf{v}_1^{(0)}$. Fortunately, there are many mechanisms which prevent this from happening, but regardless we found this to be an interesting consequence of the topology of the causal attention mechanism. Further, it provides an interesting connection between the spectral theory of directed graphs and causal attention mechanisms. We report the formal statement in the Appendix (Proposition E.8).

Proposition C.1 (Informal)**.** *Under certain assumptions on the effect of the normalisation and on the attention weights, in the limit of layers $L \to \infty$ the output representation will only depend on the first input token.*

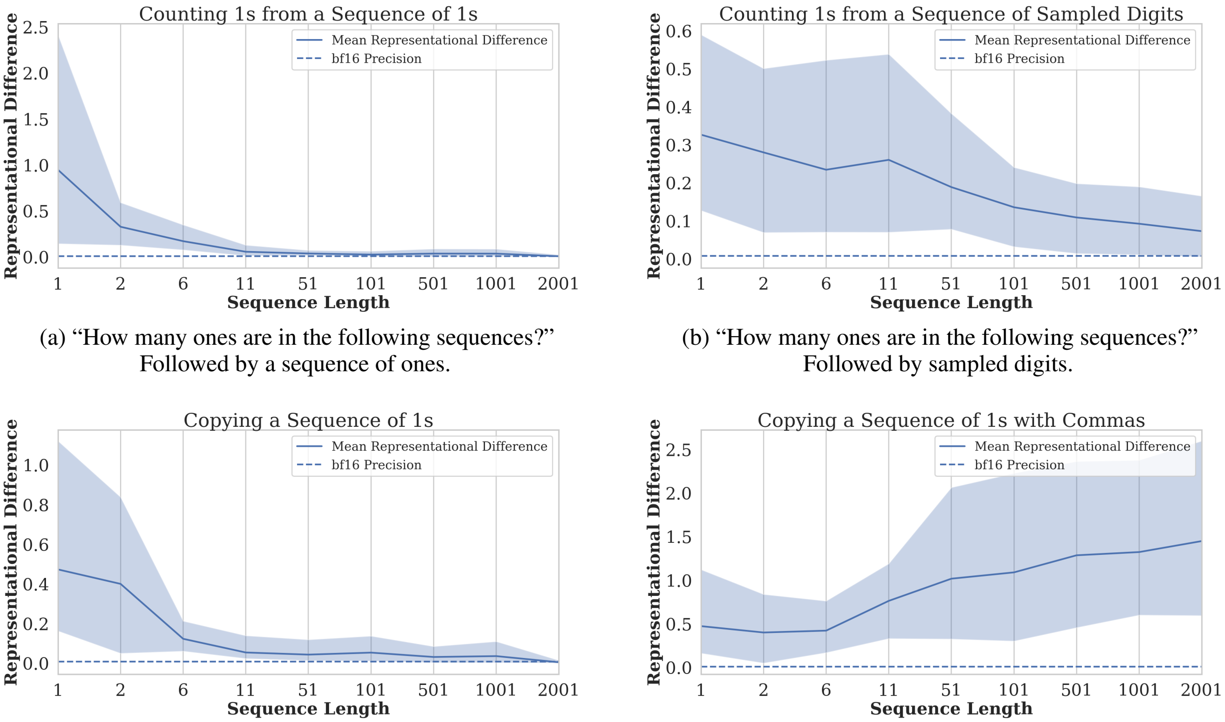
U-shape effect. Theorem 5.1 in part also helps to explain the empirically observed *U-shape effect*—the observation that LLMs seem to perform better at retrieval tasks when the information to be retrieved is located either near the start or the end of the sequence. In fact, due to the topology of the causal mechanism, we find from Theorem 5.1 that tokens at the start of the sequence have more opportunity for the information to be maintained at the end. The final tokens being also easier instead can be explained from the recency bias that is learnt by the attention mechanism during training. In auto-regressive next-token prediction, it is in fact reasonable to assume that tokens that are closer to the end will be more important and this is likely a bias that is picked-up throughout training by the LLM.

D. Counting

We highlight another representational problem that arises specifically in counting problems. We believe that our analysis points to a fundamental difficulty that emerges from the normalisation of the softmax. In particular, the normalisation of the softmax makes it hard for a model to take into account the *length* of a sequence. This is exacerbated by the fact that positional encodings are often normalised and thus relative, meaning that they also do not hold absolute positional information. Intuitively, counting is a problem that requires some notion of 'unboundedness' of the representations, whilst the normalisations used inside a Transformer work against this.

We start by showing that without causal masking and positional embeddings, a Transformer is immediately unable to count the number of tokens in a sequence, highlighting a pathological issue which stems directly from the softmax normalisation. We note that similar issues have been already pointed out (Pérez et al., 2021). We show the result in Proposition D.1 and report the full statement in the Appendix (Proposition E.9).

Proposition D.1. *A Transformer without positional encodings and a causal attention mechanism is immediately unable to count.*



(a) "How many ones are in the following sequences?" Followed by a sequence of ones.

(b) "How many ones are in the following sequences?" Followed by sampled digits.

(c) "Can you copy the following number?" Followed by a sequence of ones.

(d) "Can you copy the following number?" Followed by a sequence of ones with commas.

Figure 4. Representational collapse for counting (a, b) and copying (c, d) tasks.

While causal mechanisms and positional encodings help to break such representational issues, they break the permutation invariance of the Transformer, meaning that the representations will be heavily miss-aligned with the task, something which has been shown to hinder performance (Dudzik et al., 2024). As permutations grow factorially with sequence length, this makes it practically very challenging for a decoder-only Transformer to learn such a property simply from the data. This explains the extreme incapacity of counting highlighted in Section 3. Further, as a corollary of Theorem 4.2, we have that even if a model would be able to generalise completely, the problem of representational collapse points to an impossibility result in counting regardless. The result is summarised in Corollary D.2, with the full statement in the Appendix (Corollary E.10).

Corollary D.2 (Informal)**.** *Counting in certain situations becomes impossible due to representational collapse.*

E. Proofs

We provide formal statements and proofs for the results shown in the main text. We follow the order in which they are presented in the main text. In Section E.1, we present the proofs on representational collapse (Section 4), in Section E.2 the proofs on over-squashing (Section 5), and finally in Section E.3 the proofs on counting (Section **??**).

E.1. Representational Collapse

We start by showing that adding a new element to a sequence ingested by a softmax layer results in the softmax value of a specific token to decrease. This is fundamentally a statement about the increase of entropy caused by the addition of a new token.

Lemma E.1. *Consider a vector* $\mathbf{a} \in \mathbb{R}^{n-1}$ *and two scalars* $b, c \in \mathbb{R}$*. Let* $\mathbf{x} = [\mathbf{a}\ c]^T \in \mathbb{R}^n$ *and* $\mathbf{x}^* = [\mathbf{a}\ b\ c]^T \in \mathbb{R}^{n+1}$*. Then,* $\mathrm{softmax}\left(\mathbf{x}\right)_n > \mathrm{softmax}\left(\mathbf{x}^*\right)_{n+1}$*. Moreover for any* $p > 0$ *we can find large enough* $n \in \mathbb{N}^+$ *such that* $|\,\mathrm{softmax}\left(\mathbf{x}\right)_n - \mathrm{softmax}\left(\mathbf{x}^*\right)_{n+1}| < p$*.*

Proof. We directly compute:

$$\begin{aligned}\operatorname{softmax}(\mathbf{x})_n &= \frac{\exp(c)}{\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(c)}\\ \operatorname{softmax}(\mathbf{x}^*)_{n+1} &= \frac{\exp(c)}{\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(b)+\exp(c)}\end{aligned}$$

As we assume that $b \in \mathbb{R}$ and in particular finite, we have that $\sum_{j=1}^{n-1}\exp(\mathbf{a}_j)+\exp(c) < \sum_{j=1}^{n-1}\exp(\mathbf{a}_j)+\exp(b)+\exp(c)$, therefore $\operatorname{softmax}(\mathbf{x})_n > \operatorname{softmax}(\mathbf{x}^*)_{n+1}$.

For the second part of the statement, we compute:

$$\begin{aligned}|\operatorname{softmax}(\mathbf{x})_n - \operatorname{softmax}(\mathbf{x}^*)_{n+1}| &= \left|\frac{\exp(c)}{\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(c)} - \frac{\exp(c)}{\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(b)+\exp(c)}\right|\\ &= \left|\frac{\exp(b+c)}{\left(\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(b)+\exp(c)\right)\left(\sum_{k=1}^{n-1}\exp(\mathbf{a}_k)+\exp(c)\right)}\right|\end{aligned}$$

We therefore have that $|\operatorname{softmax}(\mathbf{x})_n - \operatorname{softmax}(\mathbf{x}^*)_{n+1}| \to 0$ as $n \to \infty$ and for large enough n $|\operatorname{softmax}(\mathbf{x})_n - \operatorname{softmax}(\mathbf{x}^*)_{n+1}| < p$ for any $p > 0$ due to the previous statement. □

We now show a slightly more general result that will help us deal with positional encodings.

Lemma E.2. *Let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{x}^* \in \mathbb{R}^{n+1}$ be two vectors, such that $\mathbf{x}_{max} - \mathbf{x}_{min} \leqslant \delta$ and $\mathbf{x}^*_{max} - \mathbf{x}^*_{min} \leqslant \delta$. Let $\mathbf{y} = \operatorname{softmax}(\mathbf{x})$ and similarly $\mathbf{y}^* = \operatorname{softmax}(\mathbf{x}^*)$. Then, $|\mathbf{y}_i - \mathbf{y}^*_j| \leqslant O(\frac{1}{n})$ for $i \leqslant n$ and $j \leqslant n+1$.*

Proof. Let $\mathbf{y}_k = e^{\mathbf{x}_k}/Z$ for some partition function Z. We can bound $\mathbf{y}_k$ from above as:

$$\mathbf{y}_k \leqslant \frac{e^{\mathbf{x}_{max}}}{Z} \leqslant \frac{e^{\mathbf{x}_{max}}}{ne^{\mathbf{x}_{min}}} = \frac{1}{n}e^{\delta}$$

Similarly for $\mathbf{y}^*_k$:

$$\mathbf{y}^*_k \leqslant \frac{e^{\mathbf{x}^*_{max}}}{Z^*} \leqslant \frac{e^{\mathbf{x}^*_{max}}}{(n+1)e^{\mathbf{x}^*_{min}}} = \frac{1}{n+1}e^{\delta}$$

Through the same process, we can bound from below achieving:

$$\begin{aligned}\frac{1}{n}e^{-\delta} \leqslant &\mathbf{y}_k \leqslant \frac{1}{n}e^{\delta},\\ \frac{1}{n+1}e^{-\delta} \leqslant &\mathbf{y}^*_k \leqslant \frac{1}{n+1}e^{\delta}\end{aligned}$$

Now to upper bound the difference $|\mathbf{y}_i - \mathbf{y}^*_j|$, we start by taking $\mathbf{y}_i$ as a maximum and $\mathbf{y}^*_j$ as a minimum, yielding:

$$\left|\frac{1}{n}e^{\delta} - \frac{1}{n+1}e^{-\delta}\right| = \left|\frac{(n+1)e^{\delta} - ne^{-\delta}}{n^2+n}\right| = O\left(\frac{1}{n}\right)$$

The other direction gives a similar result:

$$\left|\frac{1}{n}e^{-\delta} - \frac{1}{n+1}e^{\delta}\right| = \left|\frac{(n+1)e^{-\delta} - ne^{\delta}}{n^2+n}\right| = O\left(\frac{1}{n}\right).$$

This gives us that:

$$\left|\mathbf{y}_i - \mathbf{y}_j^*\right| \leqslant O\left(\frac{1}{n}\right).$$

□

Lemma E.3. *Total variation between $\mathbf{x}_i$ and $\mathbf{x}_i + \mathbf{p}_i$ for $\mathbf{p}_i > 0$ for $i \geqslant n-k$ and $\mathbf{p}_i = 0$ for $i < n-k$ goes to 0 as $n \to \infty$.*

Proof. Let Z and Z^* be the partition functions for $\mathbf{y}$ and $\mathbf{y}^*$, respectively. The total variation is then:

$$\sum_i \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i^*}}{Z^*}\right| = \sum_{i<n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i^*}}{Z^*}\right| + \sum_{i \geqslant n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i^*}}{Z^*}\right| + \mathbf{y}_{n+1}^*$$

We start by showing that the first term vanishes as $n \to \infty$.

$$\begin{aligned}
\sum_{i<n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i^*}}{Z^*}\right| &= \sum_{i<n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i}}{Z^*}\right| \\
&\leqslant (n-k)e^{\mathbf{x}_{max}} \left|\frac{1}{Z} - \frac{1}{Z^*}\right| \\
&= (n-k)e^{\mathbf{x}_{max}} \left|\frac{Z - Z^*}{ZZ^*}\right| \\
&\leqslant (n-k)e^{\mathbf{x}_{max}} \frac{ke^{\mathbf{x}_{max}}}{n^2 e^{\mathbf{x}_{min}}} \\
&= \frac{nk - k^2}{n^2} \frac{e^{2\mathbf{x}_{max}}}{e^{\mathbf{x}_{min}}} \to 0
\end{aligned}$$

We now consider the second term in a similar manner:

$$\sum_{i \geqslant n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i^*}}{Z^*}\right| = \sum_{i \geqslant n-k} \left|\frac{e^{\mathbf{x}_i}}{Z} - \frac{e^{\mathbf{x}_i + \mathbf{p}_i}}{Z^*}\right| \leqslant O\left(\frac{1}{n}\right) \to 0$$

and finally, the last term $\mathbf{y}_{n+1}^*$ also vanishes as $\mathbf{y}_{n+1}^* \leqslant e^{\delta}/n$.

□

We now show one of the main results on representational collapse.

Theorem E.4 (Representational Collapse)**.** *Let $\mathbf{x} \in \mathbb{R}^{n-1 \times d}$ be an underlying growing token sequence. Let $\mathbf{v}^{(0)} = [\mathbf{v}\ \mathbf{v}_a]^T \in \mathbb{R}^{n \times d}$ and $\mathbf{v}^{*(0)} = [\mathbf{v}\ \mathbf{v}_a\ \mathbf{v}_a]^T \in \mathbb{R}^{n+1 \times d}$ be two sequences for some additional token $\mathbf{x}_a \in \mathbb{R}^d$. Then, for large enough $n \in \mathbb{N}^+$, we have that the representations are under any ϵ:*

$$||\mathbf{v}_n^{(L)} - \mathbf{v}_{n+1}^{*(L)}||_\infty < \epsilon.$$

Proof. Here we assume that the maximum difference in activations going into the softmax is bounded by δ. This assumption is realistic as query and keys are bounded to for training thanks to the normalisation layers stability and the most popular positional encodings such as sinusoidal or RoPE are also bounded. This assumption is needed to allow us to apply Lemma E.2.

We further make the assumption that positional encodings decay up to a specific distance k, after which they can be ignored. We point that this behaviour is often wanted by design, for instance with the popular RoPE encodings (Su et al., 2024).

We now compare the n-th element of $\mathbf{z}^{(0)}$ with the $n+1$-th element of $\mathbf{z}^{*(0)}$:

$$\begin{aligned}
\left\|\mathbf{z}_n^{(0)} - \mathbf{z}_{n+1}^{*(0)}\right\|_\infty &= \left\|\sum_{i<n} \alpha_{n,i}^{(0)} \mathbf{v}_i^{(0)} + \alpha_{n,n}^{(0)} \mathbf{v}_a^{(0)} - \left(\sum_{i<n} \alpha_{n+1,i}^{*(0)} \mathbf{v}_i^{(0)} + \left(\alpha_{n+1,n}^{*(0)} + \alpha_{n+1,n+1}^{*(0)}\right) \mathbf{v}_a^{(0)}\right)\right\|_\infty \\
&= \left\|\sum_{i<n} \left(\alpha_{n,i}^{(0)} - \alpha_{n+1,i}^{*(0)}\right) \mathbf{v}_i^{(0)} + \left(\alpha_{n,n}^{(0)} - \alpha_{n+1,n}^{*(0)} - \alpha_{n+1,n+1}^{*(0)}\right) \mathbf{v}_a^{(0)}\right\|_\infty \\
&\leqslant \sum_{i\leqslant n} \left|\alpha_{n,i}^{(0)} - \alpha_{n+1,i}^{*(0)}\right| + \left|\alpha_{n+1,n+1}^{*(0)}\right| \\
&= \sum_{i<n-k} \left|\alpha_{n,i}^{(0)} - \alpha_{n+1,i}^{*(0)}\right| + \sum_{n-k\leqslant i\leqslant n} \left|\alpha_{n,i}^{(0)} - \alpha_{n+1,i}^{*(0)}\right| + \left|\alpha_{n+1,n+1}^{*(0)}\right| \\
&\leqslant \frac{k}{n} + \frac{1}{n+1} e^{\delta} < \epsilon
\end{aligned}$$

Where in the penultimate step we use our assumption that the positional encoding effects are neglible for tokens coming before token $n-k$ from our assumptions. In the last step, we instead apply Lemma E.2. We also assume for simplicity that the values are unit norm. This is not crucial as this would equivalently just give additional constant factors. We note that each of these components will fall for large enough n under machine precision.

As $\mathbf{v}_i^{(\ell+1)} = \boldsymbol{\psi}^{(\ell)}\left(\mathrm{norm}_2^{(\ell)}\left(\mathbf{z}_i^{(\ell)}\right)\right) + \mathbf{z}_i^{(\ell)}$ and importantly the sequences have the same final token, their representations will be closer when entering the next layer. The result then follows via a simple induction argument on the number of layers up to L. □

To provide further experimental evidence with other positional encodings, we experiment using the original sinusoidal embeddings from (Vaswani et al., 2017). We sample key, query, and values from a Gaussian distribution with variance $\sigma^2 = 1/d$, with d the dimension of the embeddings. We set $d = 64$ and otherwise follow the exact structure of the decoder-only Transformer presented in the original Transformer paper. We experiment with a single attention layer and check the convergence of the representations of the final token between a sequence of length n and a sequence of length $n+1$ in which we simply copy the final token. We present the results in Figure 5. We see how also for sinusoidal PEs with key, queries, and values randomly sampled, the convergence still occurs.

E.2. Over-squashing

We now derive our over-squashing results. In our derivations, we assume that the attention coefficients are independent of the values and that we can summarise the effect of the layer norms via a constant factor. These assumptions are not necessary for the same derivation to hold, but they greatly simplify the obtained bound and help more clearly point out the main take-aways.

Theorem E.5 (Over-squashing in Transformers). *Consider an input sequence $\mathbf{v}_1^{(0)}, \ldots, \mathbf{v}_n^{(0)}$ (including CoT). Let $\sigma_{\boldsymbol{\psi}}$ be the maximal Lipschitz constant of any $\boldsymbol{\psi}^{(\ell)}$, and $\bar{\alpha}_{j,i}^{(\ell)} = \frac{1}{\beta^{(\ell)}}\left(\alpha_{j,i}^{(\ell)} + \delta_{j,i}\right)$ the normalized attention coefficient, then:*

$$\left\|\frac{\partial \mathbf{y}_n}{\partial \mathbf{v}_i^{(0)}}\right\| \leqslant \sigma_{\boldsymbol{\psi}}^L \sum_{k_1\geqslant i} \cdots \sum_{k_L\geqslant k_{L-1}} \bar{\alpha}_{n,k_L}^{(L-1)} \prod_{\ell=2}^{L-1} \bar{\alpha}_{k_\ell,k_{\ell-1}}^{(\ell-1)} \bar{\alpha}_{k_1,i}^{(0)} \tag{2}$$

Proof. Note that for $j \geqslant i$ we have:

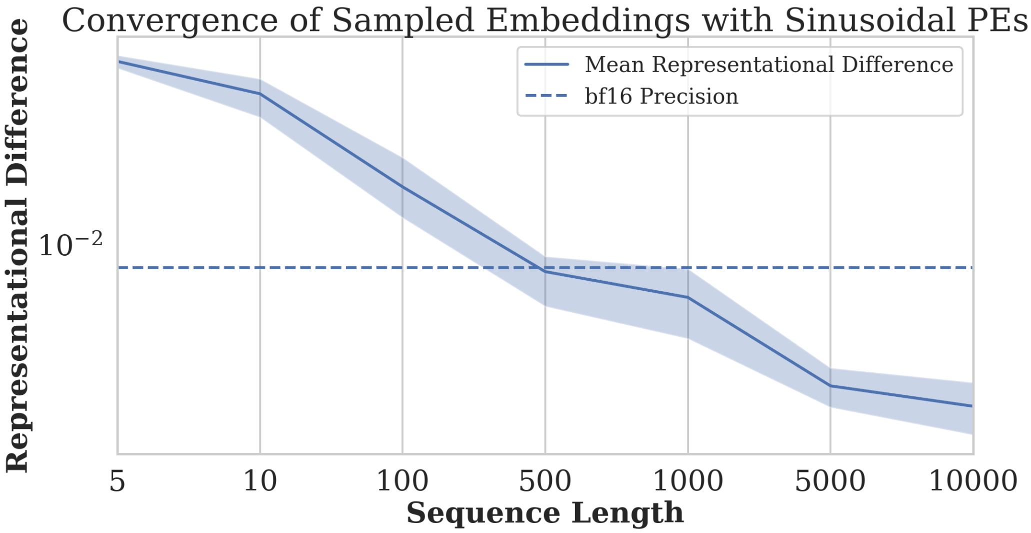


Figure 5. Convergence behaviour with a synthetic Transformer experiment. We sample the key, query, and values from a Gaussian distribution and apply the traditional sinusoidal PEs from (Vaswani et al., 2017). We apply a logarithmic scale on the y-axis.

$$\begin{aligned}
\left\|\frac{\partial \mathbf{v}_j^{(\ell+1)}}{\partial \mathbf{v}_i^{(\ell)}}\right\| &= \left\|\frac{\partial}{\partial \mathbf{v}_j^{(\ell)}}\left[\boldsymbol{\psi}^{(\ell)}\left(\mathrm{norm}_2^{(\ell)}\left(\mathbf{z}_j^{(\ell)}\right)\right)+\mathbf{z}_j^{(\ell)}\right]\right\| \\
&\leqslant \left(\frac{\sigma_{\psi^{(\ell)}}}{\beta_2^{(\ell)}}+1\right)\frac{\partial \mathbf{z}_j^{(\ell)}}{\partial \mathbf{v}_i^{(\ell)}} \\
&= \left(\frac{\sigma_{\psi^{(\ell)}}}{\beta_2^{(\ell)}}+1\right)\frac{\partial}{\partial \mathbf{v}_i^{(\ell)}}\left[\sum_{j\leqslant i}\alpha_{ij}^{(\ell)}\,\mathrm{norm}_1^{(\ell)}\left(\mathbf{v}_i^{(\ell)}\right)+\mathbf{v}_i^{(\ell)}\right] \\
&= \left(\frac{\sigma_{\psi^{(\ell)}}}{\beta_2^{(\ell)}}+1\right)\left(\frac{\alpha_{j,i}^{(\ell)}}{\beta_1^{(\ell)}}+\delta_{j,i}\right)
\end{aligned}$$

where we let $\beta_i^{(\ell)}$ represent the effect of layer normalization i at the ℓ-th layer and $\sigma_{\boldsymbol{\psi}^{(\ell)}}$ the Lipschitz constant of $\boldsymbol{\psi}^{(\ell)}$. For the case when $j<i$ due to the causal mechanism we have that $\partial\mathbf{v}_j^{(\ell)}/\partial\mathbf{v}_i^{(\ell-1)}=0$. We compute the following bound:

$$\begin{aligned}
\left\|\frac{\partial \mathbf{y}_n}{\partial \mathbf{v}_i^{(0)}}\right\| &= \left\|\frac{1}{\beta_3}\sum_{k_1}\cdots\sum_{k_L}\frac{\partial \mathbf{v}_n^{(L)}}{\partial \mathbf{v}_{k_L}^{(L-1)}}\prod_{\ell=2}^{L-1}\frac{\partial \mathbf{v}_{k_\ell}^{(\ell)}}{\partial \mathbf{v}_{k_{\ell-1}}^{(\ell-1)}}\frac{\partial \mathbf{v}_{k_1}^{(1)}}{\partial \mathbf{v}_i^{(0)}}\right\| \\
&= \left\|\frac{1}{\beta_3}\sum_{k_1\geqslant i}\cdots\sum_{k_L\geqslant k_{L-1}}\frac{\partial \mathbf{v}_n^{(L)}}{\partial \mathbf{v}_{k_L}^{(L-1)}}\prod_{\ell=2}^{L-1}\frac{\partial \mathbf{v}_{k_\ell}^{(\ell)}}{\partial \mathbf{v}_{k_{\ell-1}}^{(\ell-1)}}\frac{\partial \mathbf{v}_{k_1}^{(1)}}{\partial \mathbf{v}_i^{(0)}}\right\| \\
&\leqslant \frac{1}{\beta_3}\prod_{\ell=1}^{L}\left(\frac{\sigma_{\boldsymbol{\psi}}}{\beta_2^{(\ell)}}+1\right)\sum_{k_1\geqslant i}\cdots\sum_{k_L\geqslant k_{L-1}}\bar{\alpha}_{n,k_L}^{(L-1)}\prod_{\ell=2}^{L-1}\bar{\alpha}_{k_\ell,k_{\ell-1}}^{(\ell-1)}\bar{\alpha}_{k_1,i}^{(0)} \\
&= C\sum_{k_1\geqslant i}\cdots\sum_{k_L\geqslant k_{L-1}}\bar{\alpha}_{n,k_L}^{(L-1)}\prod_{\ell=2}^{L-1}\bar{\alpha}_{k_\ell,k_{\ell-1}}^{(\ell-1)}\bar{\alpha}_{k_1,i}^{(0)}
\end{aligned}$$

where we let $\bar{\alpha}_{j,i}^{(\ell)} = \frac{\alpha_{j,i}^{(\ell)}}{\beta_1^{(\ell)}} + \delta_{j,i}$ and $C = \frac{1}{\beta_3} \prod_{\ell=1}^{L} \left(\frac{\sigma_\psi}{\beta_2^{(\ell)}} + 1 \right)$. □

We note that in this derivation, we use simplifying assumptions on the layer norms and attention coefficients, more specifically we assume that they are independent of the $\mathbf{v}_i$s. Of course, there is nothing stopping us from avoiding such assumptions and pushing the partial derivatives inside these components as well. The drawback is that this would add a great deal of additional complexity to the result and potentially distract from what we believe are the two key takeaways: (1) the position of the token matters, and (2) the attention coefficients matter.

We now show some basic results on the spectral theory of matrices which relate to causal attention mechanisms. We emphasize that in this work, we view causal attention mechanisms as triangular row-stochastic matrices. We show that these matrices have interesting spectral properties.

Lemma E.6. *A row-stochastic triangular matrix* $\mathbf{A}$ *has* 1 *as its largest eigenvalue. Moreover, such eigenvalue has multiplicity* 1 *if each row except the first has at least* 2 *non-zero entries.*

Proof. We start by showing that $\mathbf{A}$ cannot have eigenvalues $\lambda > 1$. We then provide an eigenvector with eigenvalue 1. We finally show that such an eigenvector is unique if each row has at least 2 non-zero entries.

Assume $\lambda > 1$ for some eigenvector ϕ, we then have that $\mathbf{A}\phi = \lambda\phi$. Consider $\phi_i = \max_k \phi_k > 0$. Now $(\mathbf{A}\phi)_i = \sum_{j \leqslant i} \mathbf{A}_{ij}\phi_j = \lambda\phi_i$. As the sum is a convex combination, the result cannot be larger than the already maximal element ϕ_i. As $\lambda > 1$, we however have that $\lambda\phi_i > \phi_i$ which is a contradiction and we conclude that $\lambda \leqslant 1$.

It is easy to find an eigenvector that always has eigenvalue 1. Consider a vector $\mathbf{x}$ which is a constant vector of 1s. Then $(\mathbf{A}\mathbf{x})_i = \sum_{j \leqslant i} \mathbf{A}_{ij} = \mathbf{x}_i$, therefore $\mathbf{x}$ is an eigenvector with eigenvalue 1.

Finally, we show that when each row is non-zero, the only eigenvector is the constant-valued eigenvector. Consider the largest entry $\mathbf{y}_i > 0$, then we have that $(\mathbf{A}\mathbf{y})_i = \sum_{j \leqslant i} \mathbf{A}_{ij}\mathbf{y}_j = \mathbf{y}_i$. Again, as this defines a convex combination, we must have that all tokens that i points (i.e. the non-zero entries) are also equal to $\mathbf{y}_i$. The condition that each row has at least two non-zero entries is important as it means that the condition $\mathbf{y}_i = \mathbf{y}_j$ is true for all tokens. □

Lemma E.7. *The product of two row-stochastic matrices is again row-stochastic. Moreover, the product of two triangular row-stochastic matrices is a triangular row-stochastic matrix.*

Proof. Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ be two row-stochastic matrices. We compute:

$$\sum_j (\mathbf{AB})_{ij} = \sum_j \sum_k \mathbf{A}_{ik}\mathbf{B}_{kj} = \sum_k \mathbf{A}_{ik} \sum_j \mathbf{B}_{kj} = 1$$

The final statement follows immediately from the fact that the product of two triangular matrices is triangular. □

We now show that under specific conditions, our over-squashing bound converges to a steady state in which the final token $\mathbf{y}_n$ only depends on the initial input token $\mathbf{v}_1^{(0)}$.

Proposition E.8. *Let* $\beta_1^{(\ell)}, \beta_2^{(\ell)} = 1$, $\beta_3^{1/L} = 4$, σ_ψ *= 1. Furthermore, for simplicity, let the attention coefficients be equal at each layer and such that each row except the first of the causal mechanism has at least two non-zero elements. Then, we have as* $L \to \infty$ *that* $\partial \mathbf{y}_n / \partial \mathbf{v}_i^{(0)} = 0$ *when* $i \neq 1$ *and* $\partial \mathbf{y}_n / \partial \mathbf{v}_i^{(0)} = 1$ *when* $i = 1$. *In other words,* $\mathbf{y}_n$ *will only be sensitive to the first token.*

Proof. Let the associated attention matrix be $\mathbf{\Lambda}$. We start by re-writing the following:

$$\begin{aligned}\left\|\frac{\partial \mathbf{y}_n}{\partial \mathbf{v}_i^{(0)}}\right\| &\leqslant \frac{1}{\beta_3}\prod_{\ell=1}^{L}\left(\frac{\sigma_{\boldsymbol{\psi}}}{\beta_2^{(\ell)}}+1\right)\sum_{k_1\geqslant i}\cdots\sum_{k_L\geqslant k_{L-1}}\bar{\alpha}^{(L-1)}_{n,k_L}\prod_{\ell=2}^{L-1}\bar{\alpha}^{(\ell-1)}_{k_\ell,k_{\ell-1}}\bar{\alpha}^{(0)}_{k_1,i}\\ &= \left(\prod_{\ell=1}^{L}\left[\frac{1}{\beta_3^{1/L}}\left(\frac{\sigma_{\boldsymbol{\psi}}}{\beta_2}+1\right)\left(\frac{1}{\beta_1}\mathbf{\Lambda}+\mathbf{I}\right)\right]\right)_{n,i}\\ &= \left(\left[\frac{1}{2}\left(\mathbf{\Lambda}+\mathbf{I}\right)\right]^{L}\right)_{n,i}\end{aligned}$$

We now point out that $\tilde{\mathbf{\Lambda}} = \frac{1}{2}\left(\mathbf{\Lambda}+\mathbf{I}\right)$ is row-stochastic and with our assumptions is diagonalizable into $\tilde{\mathbf{\Lambda}} = \Psi\Sigma\Phi$. In particular, by Lemma E.7, also $\tilde{\mathbf{\Lambda}}^L$ is row-stochastic and each entry is non-negative. We now apply the Perron-Frobenius theorem for non-negative matrices, which guarantees us that all eigenvalues λ_k of $\tilde{\mathbf{\Lambda}}$ are bounded such that $|\lambda_k| \leqslant 1$. In particular, thanks to Lemma E.6, we know that there is a unique eigenvector (the constant eigenvector ψ_n) with eigenvalue $\lambda_n = 1$. Denote the left eigenvectors by ψ_k and the right eigenvectors ϕ_n, we therefore have:

$$\lim_{L\to\infty}\tilde{\mathbf{\Lambda}}^L = \sum_k \lambda_k^L\psi_k\phi_k^T = \psi_n\phi_n^T.$$

One can check that $\phi_n = [1\;0\ldots\;0]$, meaning that $\psi_n\phi_n^T$ has as first column a constant vector of 1s and every other entry 0. This completes the proof. □

E.3. Counting

Proposition E.9. *A Transformer without positional encodings and a causal attention mechanism is immediately unable to solve the counting problem.*

Proof. We prove this statement by showing that such an model will produce equivalent embeddings up to the ratio of the elements in the sequence and will therefore produce the same output for different sequences having the same ratio of elements. Of course having the same ratio of elements does not mean that the count will be different, for instance the sequences '10' and '1100' have the same ratio of digits but clearly different counts. Consider a sequence of two values, $\mathbf{v}^{(0)}_{zero}$ and $\mathbf{v}^{(0)}_{one}$, with n_0 and n_1 being the number of zeros and ones respectively. We ignore in our calculations the MLPs ψ and the normalizations norm as these don't affect the argument. As the attention mechanism is permutation equivariant, the initial zero tokens will all be mapped to:

$$\begin{aligned}\mathbf{z}^{(1)}_{zero} &= \sum_j \frac{\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_j\right)}{\sum_w \exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_w\right)}\mathbf{v}^{(0)}_j + \mathbf{v}^{(0)}_{zero}\\ &= \frac{n_0\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)}{n_0\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)+n_1\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}\mathbf{v}^{(0)}_{zero} + \frac{n_1\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}{n_0\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)+n_1\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}\mathbf{v}^{(0)}_{one} + \mathbf{v}^{(0)}_{zero}\\ &= \frac{\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)}{\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)+\frac{n_1}{n_0}\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}\mathbf{v}^{(0)}_{zero} + \frac{\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}{\frac{n_0}{n_1}\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{zero}\right)+\exp\left(\mathbf{q}^{(0)T}_{zero}\mathbf{k}^{(0)}_{one}\right)}\mathbf{v}^{(0)}_{one} + \mathbf{v}^{(0)}_{zero}\end{aligned}$$

Similarly, the ones will be mapped to:

$$\begin{aligned}
\mathbf{z}_{one}^{(1)} &= \sum_j \frac{\exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_j^{(0)}\right)}{\sum_w \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_w^{(0)}\right)}\mathbf{v}_j^{(0)} + \mathbf{v}_{one}^{(0)} \\
&= \frac{n_0 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{zero}^{(0)}\right)}{n_0 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{zero}^{(0)}\right) + n_1 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}\mathbf{v}_{zero}^{(0)} + \frac{n_1 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}{n_0 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{zero}^{(0)}\right) + n_1 \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}\mathbf{v}_{one}^{(0)} + \mathbf{v}_{one}^{(0)} \\
&= \frac{\exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{zero}^{(0)}\right)}{\exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{zero}^{(0)}\right) + \frac{n_1}{n_0} \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}\mathbf{v}_{zero}^{(0)} + \frac{\exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}{\frac{n_0}{n_1} \exp\left(\mathbf{q}_{zero}^{(0)T}\mathbf{k}_{zero}^{(0)}\right) + \exp\left(\mathbf{q}_{one}^{(0)T}\mathbf{k}_{one}^{(0)}\right)}\mathbf{v}_{one}^{(0)} + \mathbf{v}_{one}^{(0)}
\end{aligned}$$

Assuming that $\mathbf{z}_{zero}^{(1)} \neq \mathbf{z}_{one}^{(1)}$ (to avoid the trivial case), we notice that the attention mechanism alongside the MLP $\boldsymbol{\psi}$ define an isomorphism between sequences at different layers, updating all zeros and ones to a different value vector. The critical fact is that the representations *only depend on the ratio between* n_0 *and* n_1, meaning that sequences of different lengths (therefore different counts) will have the exact same representation. This is respected at each layer, meaning that the LLM after L layers will assign the same representation to different sequences as long as they have the same ratio. This points to a loss of representation for the counting problem. □

Corollary E.10. *Consider a task in which the goal is to count how many* $\mathbf{v}_a$ *tokens there in the sequence. Let* $\mathbf{v}^{(0)} = [\mathbf{v}\ \mathbf{v}_a]^T \in \mathbb{R}^{n \times d}$ *and* $\mathbf{v}*^{(0)} = [\mathbf{v}\ \mathbf{v}_a\ \mathbf{v}_a] \in \mathbb{R}^{(n+1) \times d}$*. Due to representational collapse, at least one sequence will be given the wrong count for large enough finite* n.

Proof. This statement is a direct consequence of representational collapse. In particular, as $\mathbf{y}_n$ and $\mathbf{y}_n^*$ will be indistinguishable for large enough n, the Transformer will be forced to make a mistake for at least one of them. This points to an impossibility result of counting on certain sequences due to floating point error. □

F. Experiments

The prompting done on Gemini 1.5 in our work does not require custom resources as we use hosted Gemini instances. We run a local version of Gemma 7B on modest hardware to analyse the internal representations.

F.1. Experimental Details

We detail the way in which we execute the prompting for the various experiments.

Counting experiments. For the sum experiment we prompt as:

```
Please perform the following sum:  seq.  Please give the answer on the final line
exactly as ’The final answer to your maths question is:  xxxx’, where ’xxxx’ is
your answer..
```

For the ones and zero sequences, we similarly prompt as

```
Please count the number of ones in the following sequence:seq.  Please give the
answer on the final line exactly as ’The final answer to your maths question is:
xxxx’, where ’xxxx is your answer.
```

For the word counting experiment, we prompt as

```
Please count the number of times ‘car’ appears in the following sentence:  ’seq’.
Please give the answer on the final line exactly as ’The final answer to your
maths question is:  xxxx’, where ’xxxx’ is your answer.
```

For the CoT experiments, we supply examples of the form:

```
Let’s think step by step, showing me your reasoning.  Here are a few examples:
Please perform the following sum:  1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1
```

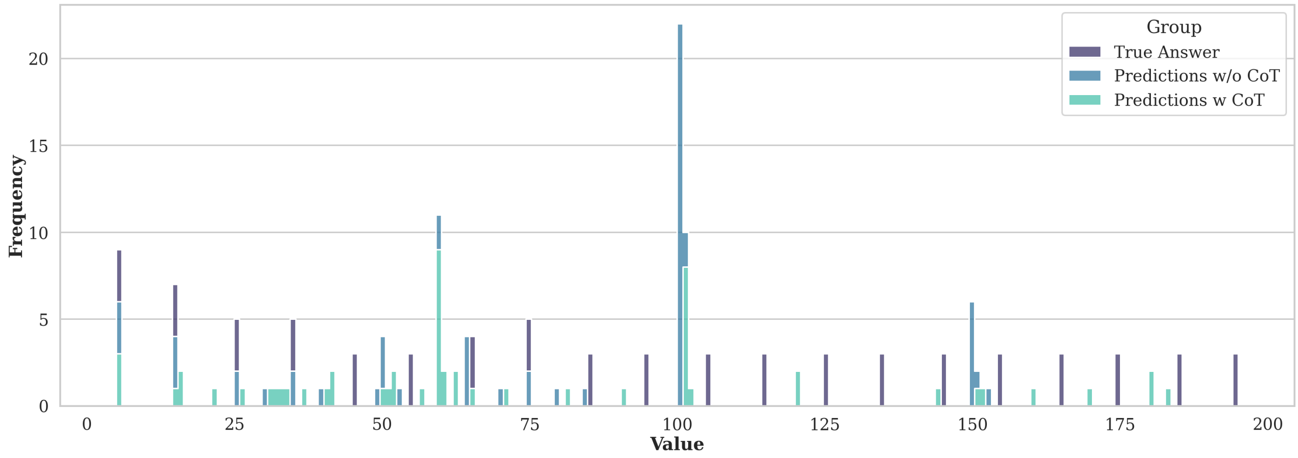


Figure 6. Frequency of different outputted values for Gemini 1.5.

```
We divide the sum into groups of 5.  (1 + 1 + 1 + 1 + 1) + (1 + 1 + 1 + 1 + 1) +
1 + 1
The answer is then 2 * 5 + 2 = 12
The final answer to your maths question is:  12
Please perform the following sum:  1 + 1 + 1 + 1 + 1 + 1
We divide the sum into groups of 5.
(1 + 1 + 1 + 1 + 1) + 1
The answer is then 1 * 5 + 1 = 6
The final answer to your maths question is:  6
```

With similar strategies for the 4 experiments.

Copying experiments. For the copying experiments, we use the following prompt:

```
Consider the following sequence:  seq.  What is the last digit in this sequence?
Please answer exactly as 'The answer to your question is:  <ANSWER>
```

and change appropriately the sequence as described.

We commit to releasing the code we have used to generate the prompts for the camera ready version.

F.2. Additional Experiments

We report similar results for the counting experiments using Gemma (Team et al., 2024). Compared to Gemini 1.5, Gemma seems to answer less accurately on the counting prompts.

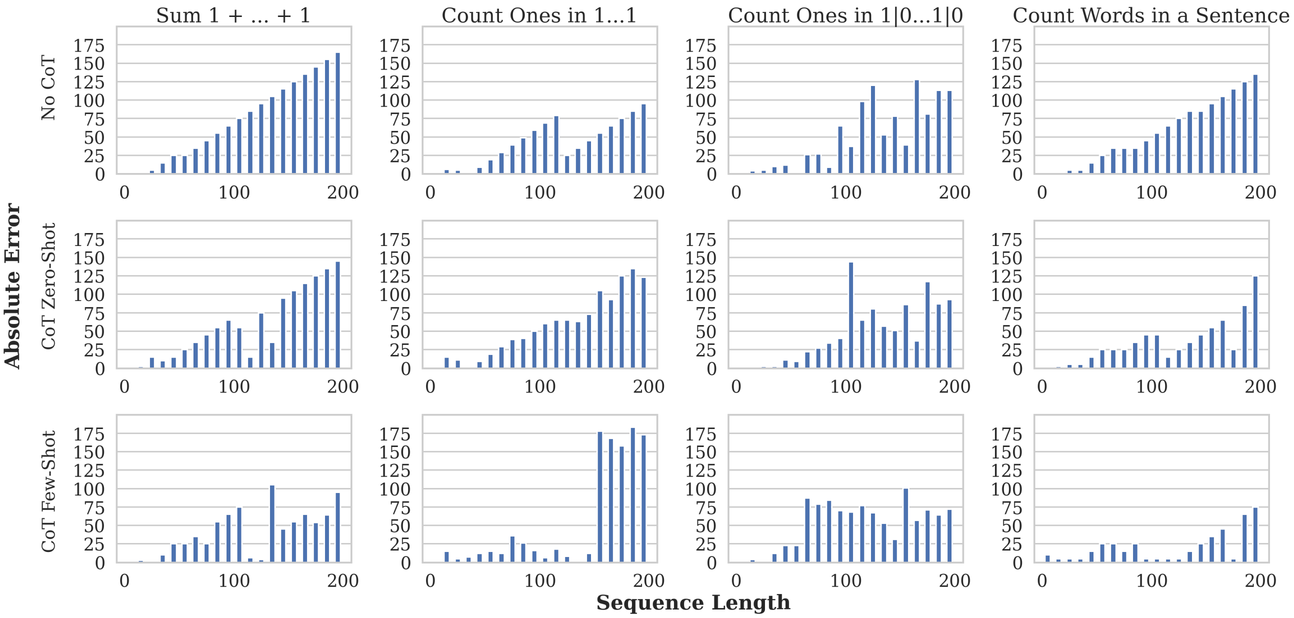


Figure 7. Gemma 7B LLMs being prompted to (i) sum $1 + \cdots + 1$ (left), (ii) Count the number of ones in a sequence of 1s (center), and (iii) Count the number of ones in a sequence of ones and zeroes (the sequence is a Bernoulli sequence with probability of sampling a one being 0.7) (right).

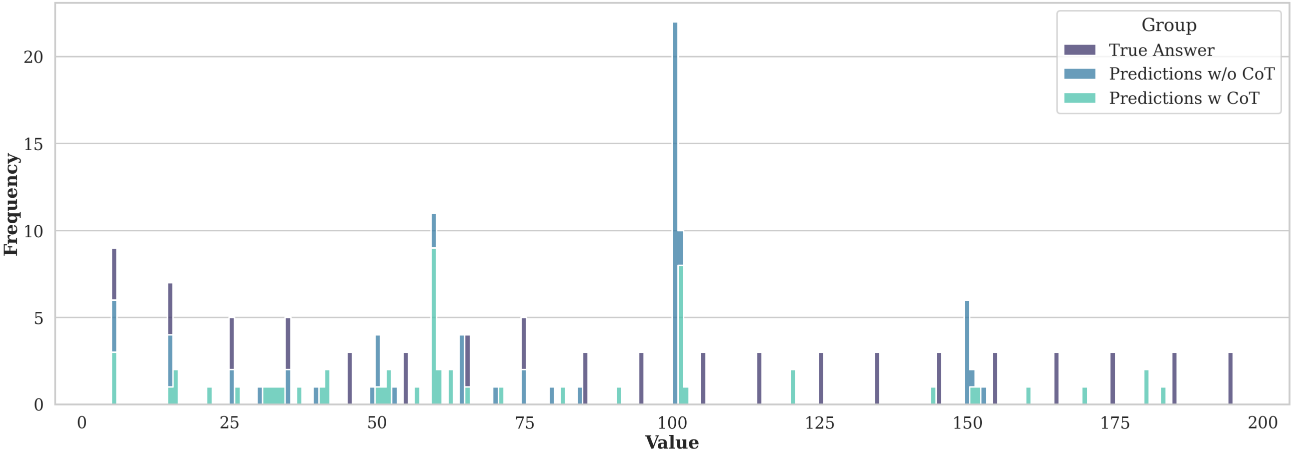


Figure 8. Frequency of different outputs for Gemma 7B