Pay Attention to What Matters

Pedro Luiz Silva[1,2] **Antonio de Domenico**[1] **Ali Maatouk**[3] **Fadhel Ayed**[1]

[1]Huawei Technologies Paris, France
[2]École Polytechnique, Palaiseau, France
[3]Yale University, New Haven, CT, USA

Abstract

Despite the remarkable success of Large Language Models (LLMs), they still exhibit a limited capability to align their outputs to the user instructions. In this work, we introduce a simple and effective method, which we name GUIDE, that mechanistically increases attention scores in instruction tokens. To support this operation, we present *Influence*, a novel metric that highlights how the user's instructions propagate through the transformer layers and impact the LLM output. Our results show that GUIDE improves the accuracy of following certain instructions from 29.4% to 60.4%, outperforming natural prompting alternatives.[1]

1 Introduction

Large Language Models (LLMs) are currently the state-of-the-art for Natural Language Processing (NLP) tasks. Despite this success, pretrained LLMs sometimes struggle to accurately interpret diverse users' instructions and may generate outputs that do not align with human expectations. Additionally, LLMs may produce biased or hallucinated facts, which can limit their practical usefulness. Since ensuring that LLMs consistently adhere to given instructions is paramount, researchers have designed methodologies to align the outputs of LLMs with human preferences.

Model fine-tuning usually aligns the output of the LLMs with human intents using Reinforcement Learning with Human Feedback (RLHF) [15], Reinforcement Learning with AI Feedback (RLAIF) [13] or Direct Preference Optimization (DPO) [16]. However, these methods have three significant constraints: they require specialized datasets, often with human annotations, thus reducing efficiency; they involve substantial computational complexity and cost due to the need for additional training; and they demand specialized expertise, as successfully implementing the process can be challenging. Given these constraints, this type of fine-tuning is typically reserved for general-purpose alignment, ensuring that models are Helpful, Harmless, and Honest [19] and are not suited to address end users' specific needs. Supervised fine-tuning (SFT) with techniques such as low-rank adapters (LoRA [8]) offers a more accessible way to customize a model for individual user requirements. However, these techniques still face the same three limitations, albeit to a lesser extent. Consequently, SFT is typically utilized only for very targeted use cases if used at all.

Utilizing LLMs for automated prompt engineering has demonstrated remarkable performance. Black-Box Prompt Optimization (BPO) is a sophisticated framework that automatically refines human-written prompts, often unstructured or ambiguous [5]. Similarly, the PE2 framework [24] enhances prompt performance by refining human-written prompts through a comprehensive search process. Although PE2 avoids additional model training, it increases complexity, latency, and cost, limiting its scalability. Both BPO and PE2 are generally designed for broad enhancements in prompt writing. They are not tailored to meet individual users' specific intentions or needs.

[1]Our experiments are available on `https://github.com/netop-team/pay_attention_experiment`.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

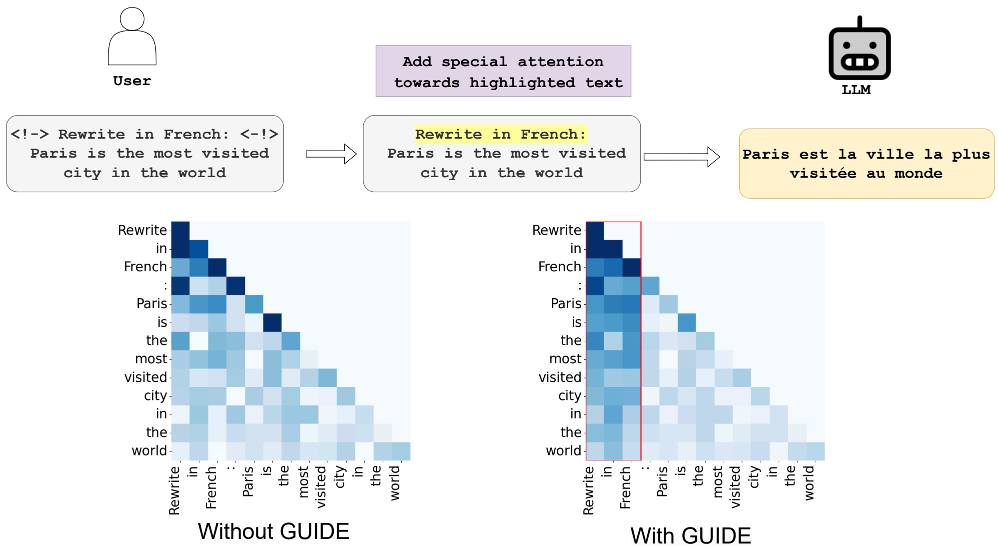


Figure 1: GUIDE uses tags (such as `<!-> <-!>`) to know where to focus. It then enhances the importance of highlighted tokens by biasing the attention scores toward them, as shown by the attention matrices above, where each entry represents the impact of a past token (x-axis) on the ongoing token (y-axis).

Due to its low cost and large accessibility, prompt engineering is extensively used to align the output of the LLMs with user preferences. This is clearly demonstrated by popular LLM frameworks like the system in [23], which empowers LM agents to tackle software engineering tasks. This system emphasizes crucial instructions through uppercase text and exclamation marks, like "PLEASE DO NOT DO THAT!" or "THE EDIT COMMAND REQUIRES PROPER INDENTATION." Similarly, the AI Scientist [14], a leading system for automated scientific discovery, uses strong directives such as "ABSOLUTELY DO NOT ADD IT AGAIN!!!" to steer the model's behavior. These examples, drawn from highly influential frameworks widely used, underscore the pressing need for end-users to signal what matters most to them in order to guide LLMs toward better alignment with their goals. Currently, users rely on prompt engineering and forceful language to achieve this alignment. However, this approach does not consistently deliver positive results as shown in [17].

In this work, we introduce GUIDE (**G**uided **U**nderstanding with **I**nstruction-**D**riven **E**nhancements), a novel and systematic approach that allows users to emphasize critical instructions in their prompts. GUIDE enables users to influence the attention given to specific tokens by simply enclosing important text within tags like `<!-> <-!>` (as shown in the top of Figure 1). These special tags directs the LLM's focus, which is done by adding a bias to the attention scores toward the tokens they enclose. Our implementation is open-source and designed for seamless integration. Our experiments demonstrate that GUIDE significantly increases the likelihood of the model following key instructions and retrieving crucial information designated by the user, outperforming natural prompting techniques.

2 GUIDE: Guided Understanding with Instruction-Driven Enhancements

In this section, we present GUIDE, a novel and systematic approach that enables users to highlight critical instructions within the text input provided to an LLM. To understand how GUIDE operates, it is essential first to revisit the core mechanism of self-attention, which drives the functioning of LLMs.

Each token k in the input text is initially represented by an embedding, denoted as $E_k^{(0)}$, which undergoes progressive refinement through stacked layers of attention. By the time it reaches the final layer, L, this embedding $E_k^{(L)}$ is expected to encapsulate all the semantic information required to predict the next token, $k+1$.

The process operates as follows: at each attention layer ℓ, the embedding of a token k is enhanced with the semantic information of past tokens ($i = 1, 2, \ldots, k-1$) and itself. This enrichment occurs through a residual connection, where the embedding $E_k^{(\ell)}$ is updated with the output of the attention layer, which consists of a weighted average of the values $V_i^{(\ell)}$ of the past tokens. The vectors V, known as values, are derived from a simple linear transformation of the embeddings $E_i^{(\ell)}$, for $i \leq k$, and are responsible for carrying the semantic information from the past tokens.

The extent to which previous tokens influence the semantic update of the token k is determined by attention logits, denoted as $w_{k,i}^{(\ell)}$. The logits are then passed through a softmax function, which normalizes them to sum to one. The resulting normalized weights are known as attention scores ($\mathbf{A}^{(\ell)}$) and quantify the degree of influence each past token has on the current token's semantic representation at a given layer. Denoting $U_k^{(\ell+1)} := \text{Attention}^{(\ell+1)}(\mathbf{E}_k^{(\ell)})$, the operations at layer ℓ can be summarized as follows[2]:

$$E_k^{(\ell+1)} = E_k^{(\ell)} + U_k^{(\ell+1)} = E_k^{(\ell)} + \sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell+1)} V_i^{(\ell)}. \tag{1}$$

The attention logits, and hence the attention scores, are automatically computed by the model. We argue that the end user should be able to influence the level of attention each token receives by explicitly signaling which instructions or pieces of information are critical. By doing so, the user can effectively guide the model to better align with their intention. We propose to achieve this by simply adding a bias, denoted by Δ, to the attention logits of the important tokens, i.e., $\bar{w}_{k,i}^{(\ell)} = w_{k,i}^{(\ell)} + \Delta$, for all tokens i indicated by the user. While this approach is direct, it proves to be highly effective, as demonstrated in the experimental results section.

The addition of Δ increases the attention the model pays to the tokens of interest, amplifying their influence on the generated output. However, because attention scores must sum to one, this adjustment reduces the attention given to other tokens. If Δ is set too high, the model might overly focus on the highlighted tokens, which could disrupt the generation process. Therefore, it is crucial to select an appropriate Δ that balances these effects.

Our experiments suggest that for the Mistral and Gemma-2 models, a Δ of 2 works well for emphasizing instructions, while a Δ of 1 is effective for highlighting specific information within the text. Besides, using Δ values greater than 5 often led to nonsensical outputs (see App. C.1 and E).

Although these default values improve performance, the optimal choice of Δ depends on various factors, including the model, the nature of the task, etc. The most precise way to determine an appropriate Δ is through hyperparameter tuning on a validation set.

In this work, we also introduce a heuristic approach for calibrating Δ with just a couple of forward passes. The idea is to match the influence increase from Δ to a "natural" level that could be achieved through conventional prompting, such as using uppercase. This calibration requires a metric that evaluates the importance of the selected tokens and tracks how this impact propagates both vertically across the stacked layers and horizontally across successive tokens.

To maintain simplicity and minimize computational cost, we avoid using gradient-based metrics to evaluate the impact of a subset of tokens on the overall sequence (for example, see [3],[4], and [18]). Instead, a more appropriate option appears to be the *Attention Rollout* method proposed in [1]. This metric can be easily computed during the forward pass, aligning well with our needs.

The *Attention Rollout* approach is based on a natural interpretation of attention scores. It postulates that the influence of a past token i on the update of the current token k is quantified by the attention score $\mathbf{A}_{k,i}^{(\ell)}$. The method addresses the residual connection by assuming that in the updated embedding $E_k^{(\ell+1)}$, both the previous embedding $E_k^{(\ell)}$ and the update vector $U_k^{(\ell+1)}$ contribute equally, each having an impact of $\frac{1}{2}$. The vertical and horizontal flow of the impact $\mathrm{R}_{\mathcal{U}}(E_k^{(l)})$ of a given token (or subsequence of tokens) of interest $\mathcal{U}$ on an embedding $E_k^{(l)}$ is hence characterized by the following recurrence:

$$\mathrm{R}_{\mathcal{U}}(E_k^{(\ell)}) = \frac{1}{2}\left[\mathrm{R}_{\mathcal{U}}(E_k^{(\ell-1)}) + \mathrm{R}_{\mathcal{U}}(U_k^{(\ell)})\right] = \frac{1}{2}\left[\mathrm{R}_{\mathcal{U}}(E_k^{(\ell-1)}) + \sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell)} \cdot \mathrm{R}_{\mathcal{U}}(E_i^{(\ell-1)})\right]$$

[2]For simplicity, we have excluded the normalization and feedforward layers from this explanation.

We argue that Attention Rollout inaccurately represents the flow of attention, particularly when handling the residual connection. The norm of the past embedding $E_k^{(\ell)}$ is typically about 100 times larger than that of the update vector $U_k^{(\ell+1)}$ (see Appendix A). By assuming equal contributions from these two vectors, Attention Rollout significantly overestimates the importance of past tokens. This error compounds as the context length increases, leading to an inflated impact estimate that increases with the context and hence negatively correlates with the model's likelihood of following the token of interest, such as adhering to a specific instruction (see Appendix C.4).

To address this issue, we introduce Influence, a new metric designed to quantify the impact flow of a token or a set of tokens of interest $\mathcal{U}$. This metric corrects Attention Rollout by weighting the contributions according to the norm of the vectors:

$$\mathrm{I}_{\mathcal{U}}(E_k^{(\ell+1)}) = \frac{\left(\|E_k^{(\ell)}\| \cdot \mathrm{I}_{\mathcal{U}}(E_k^{(\ell)}) + \|U_k^{(\ell+1)}\| \cdot \mathrm{I}_{\mathcal{U}}(U_k^{(\ell+1)})\right)}{\|E_k^{(\ell)}\| + \|U_k^{(\ell+1)}\|}$$

See Appendix B for detailed derivations. Although not flawless, our experiments demonstrate that Influence correlates positively with the likelihood of a set of tokens impacting the model's output, such as following an instruction (e.g., summarizing in French) or retrieving specific information (e.g., finding a needle in a haystack). Thus, Influence offers a tool of independent value that can be used, for example, to compare and predict the impact of different natural prompting techniques.

3 Experiments and analysis

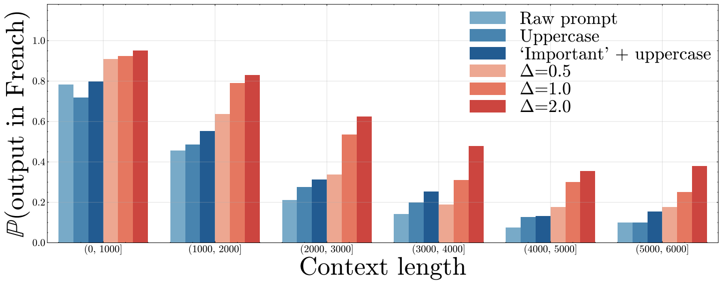
To evaluate the capability of GUIDE to support LLMs in producing outputs aligned with user's query, we perform experiments related to text translation and summarization. We obtain similar conclusions for JSON generation and "a needle in a haystack" tasks (see Appendix C).

In these experiments, we have used text from OpenWebText [6], chosen for its variety in context lengths. We have divided the dataset into groups based on context length, containing texts from a 500-token window, such as (0, 500], (500, 1000], and so on. From each group, we have randomly selected 20 texts and generated 10 summaries for each text using multinomial sampling [22].

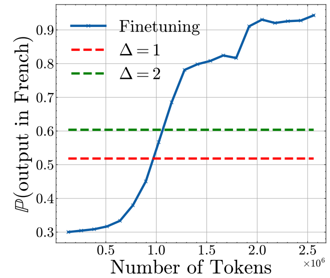
We conducted experiments using Mistral 7B Instruct[9] with GUIDE, biasing attention scores towards the instruction `Summarize in French`. Fig. 2(a) shows the observed probability that the LLM summary is in French when using GUIDE and compares the results achieved with the baseline model, with both uppercase and normal prompts, as well as the performance observed when including '`Important:`' before the prompt instruction.

Our findings show that GUIDE leads to an improvement from 29.4% to 60.4% with respect to the raw model, and that the best result is achieved with $\Delta = 2$.

As baselines, we compare the performance of GUIDE to prompt engineering and Supervised Fine-Tuning (SFT) using LORA (the hyperparameters can be found in Appendix D). Figure 2(a) show that using uppercase or adding '`Important`' on the instruction does not provides notable improvements, consistently underperforming GUIDE, while Figure 2(b) shows that GUIDE outperforms SFT until 1M training tokens. These results confirms that our method is an effective solution for aligning LLMs to instruction following that does not require additional training.



(a) Probability of outputting a summary in French.



(b) Performance of SFT over number of tokens (in millions) used during training.

Figure 2: Summarization results: (a) GUIDE outperforms prompt engineering techniques like using uppercase text, and (b) GUIDE demonstrates greater accuracy than SFT up to 1 million training tokens.

4 Conclusion

Although LLMs achieve state-of-the-art performance in most NLP tasks, they often fail to align to user instructions. To deal with this problem, we have presented GUIDE, a simple and effective approach for instruction alignment that does not require neither prompt optimization nor model fine-tuning. Our paper has demonstrated the benefits of GUIDE, in particular with respect to baseline prompt engineering, and introduced Influence, a novel metric for Transformer explainability, which provides new insights on the relations between sequences of tokens in the user's query and how they relate to the LLM output.

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.385. URL https://aclanthology.org/2020.acl-main.385.

[2] AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[3] Hila Chefer, Shir Gur, and Lior Wolf. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers, 2021. URL https://arxiv.org/abs/2103.15679.

[4] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization, 2021. URL https://arxiv.org/abs/2012.09838.

[5] Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. Black-box prompt optimization: Aligning large language models without model training, 2024. URL https://arxiv.org/abs/2311.04155.

[6] Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

[7] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[8] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[9] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

[10] Gregory Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023.

[11] Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. In search of needles in a 11m haystack: Recurrent memory finds what llms miss, 2024. URL https://arxiv.org/abs/2402.10790.

[12] British Library Labs. Digitised books. c. 1510 - c. 1900. jsonl (ocr derived text + metadata). https://doi.org/10.23636/r7w6-zy15, 2021.

[13] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. Rlaif: Scaling reinforcement learning from human feedback with ai feedback, 2023. URL https://arxiv.org/abs/2309.00267.

[14] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

[15] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

[16] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

[17] Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting, 2024. URL https://arxiv.org/abs/2310.11324.

[18] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, October 2019. ISSN 1573-1405. doi: 10.1007/s11263-019-01228-7. URL http://dx.doi.org/10.1007/s11263-019-01228-7.

[19] Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. Large language model alignment: A survey, 2023. URL https://arxiv.org/abs/2309.15025.

[20] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

[22] Gian Wiher, Clara Meister, and Ryan Cotterell. On decoding strategies for neural text generators, 2022. URL https://arxiv.org/abs/2203.15721.

[23] John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024. URL https://arxiv.org/abs/2405.15793.

[24] Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. Prompt engineering a prompt engineer, 2024. URL https://arxiv.org/abs/2311.05661.

[25] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020. URL https://arxiv.org/abs/1904.09675.

A Motivation behind *Influence*

In this section, we illustrate the motivation behind influence. As described in Section 2, we argue that Attention Rollout does not properly represent the impact of the attention flow to the transformer output because it does not take into account the difference of norms between residual connections and the outputs of attention.

Indeed, as described in the left-side of Fig. 3, $\|\mathbf{E}^{(\ell)}\|$ and $\|\text{Attention}^{(\ell+1)}(\mathbf{E}^{(\ell)})\|$ differs by a factor of approximately 100.

To stress the benefits of Influence with respect to attention rollout, we show in the right-side of Figure 3 the behaviour of these metrics (in log scale) as a function of the context length in two generic layers of a transformer, considering different types of user' queries. The right side of Figure 3 shows the attention rollout increases with the context length while influence is characterized by the opposite behaviour. These results highlight that, in contrast to attention rollout, influence can efficiently explain the decreasing capability of a transformer to align to the instruction when context length grows.

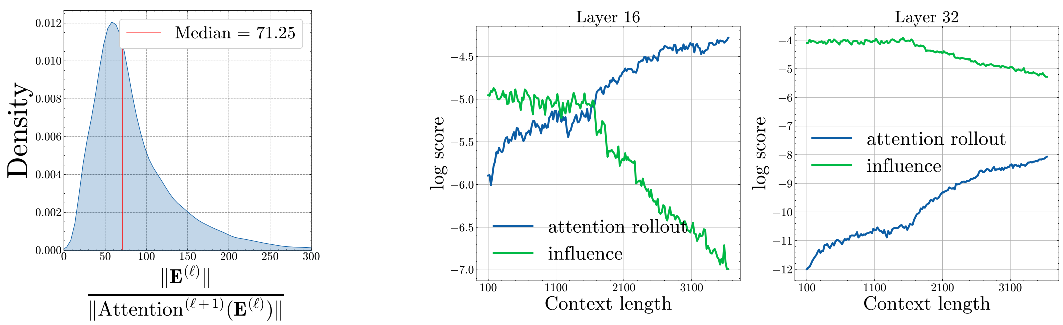


Figure 3: Left : Distribution of ratio between norms of token embeddings before and after attention; Right: Attention rollout and Influence trends in log scale over context length.

B Influence

In this section, we formally define Influence and explain how it propagates through the transformer.

B.1 Principles

Let us denote by $\bar{\mathcal{U}} = (x_1, \ldots, x_n)$ the overall sequence of tokens associated with the user's query and with $\mathcal{U} = (x_i, \ldots, x_j)$ the tokens related to the instruction that the user desires to highlight. Based on this analysis, we introduce Influence ($\text{I}_{\mathcal{U}} : \mathbb{R}^{dH} \to \mathbb{R}^{+}$) (where d is the attention head dimension and H is the number of attention heads), a transformer interpretability metric designed to quantify how sequences of tokens in the user's query impact each other and relate to the LLM output, designed based on the following principles:

Initialization: We initialize the Influence value as 1 for tokens within the instruction $\mathcal{U}$, and as 0 elsewhere. Let $E_k^0 \in \mathbb{R}^{dH}$ be the embedding of token x_k. Then, the Influence initialization can be formally defined as:

$$\text{I}_{\mathcal{U}}(E_k^0) = \mathbb{1}_{\{x_k \in \mathcal{U}\}}. \tag{2}$$

Propagation Rules: Given m embedding vectors $E_1, \ldots, E_m \in \mathbb{R}^{dH}$, the joint Influence of the instruction tokens $\text{I}_{\mathcal{U}} : \mathbb{R}^{dH} \times \cdots \times \mathbb{R}^{dH} \to \mathbb{R}_{+}$ is calculated as the average of each individual Influence, weighted by the norms of each embedding, as follows:

$$\mathrm{I}_{\mathcal{U}}(E_1, E_2, \ldots, E_m) = \frac{\sum_{i=1}^{m} \mathrm{I}_{\mathcal{U}}(E_i)\|E_i\|}{\sum_{i=1}^{m} \|E_i\|}. \tag{3}$$

Additionally, we maintain the invariance of Influence to function composition, i.e.,

$$\mathrm{I}_{\mathcal{U}}(f(E)) = \mathrm{I}_{\mathcal{U}}(E). \tag{4}$$

Then, the Influence of the instruction tokens at the ℓth layer of the transformer can be computed as follows:

$$\mathrm{I}_{\mathcal{U}}(E_k^{(\ell)}) = \frac{\mathrm{I}_{\mathcal{U}}(E_k^{(\ell-1)})}{1 + r_k^{(\ell-1)}} r_k^{(\ell-1)} + \frac{\sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell)} \mathrm{I}_{\mathcal{U}}(E_i^{(\ell-1)})}{1 + r_k^{(\ell-1)}}, \tag{5}$$

where $r_k^{(\ell-1)} := \frac{\|E_k^{(\ell-1)}\|}{\|\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k\|}$.

B.2 Detailed derivations

Let us denote by d the transformer head dimension, with H the number of attention heads, and with n the context length. Following the propagation of a transformer layer [21], the embedding on layer ℓ, denoted by $\mathbf{E}^{(\ell)}$, is computed as follows:

$$\mathbf{E}^{(\ell)} = \text{Linear}\left(\text{Norm}\left(\mathbf{E}^{(\ell-1)} + \text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})\right)\right), \tag{6}$$

$$\text{Attention}^{(\ell)}(\mathbf{E}) = \mathbf{A}^{(\ell)} \cdot V^{(\ell)}(\mathbf{E}), \tag{7}$$

where, $\mathbf{E}^{(\ell-1)} \in \mathbb{R}^{dH \times s}$ is the embedding on layer $\ell - 1$, $\mathbf{A}^{(\ell)}$ is the attention matrix on layer ℓ, $V^{(\ell)} : \mathbb{R}^{dH} \to \mathbb{R}^{dh}$ is a linear function that maps the token embeddings to the values vector, Norm is a normalization function, and Linear is a conventional multilayer perceptron (MLP) function. Then, we can compute the Influence of token k, $\mathbf{E}_k^{(\ell)}$, as follows:

$$\begin{aligned}
\mathrm{I}_{\mathcal{U}}(E_k^{(\ell)}) &= \mathrm{I}_{\mathcal{U}}\left(\text{MLP}\left(\text{Norm}(\mathbf{E}^{(\ell-1)} + \text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)}))_k\right)\right) \\
&= \mathrm{I}_{\mathcal{U}}\left((\mathbf{E}^{(\ell-1)} + \text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)}))_k\right) \\
&= \frac{\mathrm{I}_{\mathcal{U}}(E_k^{(\ell-1)}) \cdot \|E_k^{(\ell-1)}\| + \mathrm{I}_{\mathcal{U}}(\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k) \cdot \|\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k\|}{\|E_k^{(\ell-1)}\| + \|\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k\|} \\
&= \frac{\mathrm{I}_{\mathcal{U}}(E_k^{(\ell-1)})\, r_k^{(\ell-1)} + \mathrm{I}_{\mathcal{U}}(\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k)}{1 + r_k^{(\ell-1)}},
\end{aligned} \tag{8}$$

where $r_k^{(\ell-1)} := \frac{\|E_k^{(\ell-1)}\|}{\|\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k\|}$.

Influence is computed recursively over layers, *i.e.*, when we compute the Influence on layer ℓ, we have already computed the Influence on layers $1, \ldots, \ell - 1$. This means that $\mathrm{I}_{\mathcal{U}}(E_k^{(\ell-1)})$ is already computed, while we still need to compute $\mathrm{I}_{\mathcal{U}}(\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k)$. Developing equation 7:

$$\text{Attention}^{(\ell)}(\mathbf{E}^{(\ell-1)})_k = \sum_{i=1}^{s} \mathbf{A}_{k,i}^{(\ell)} E_i^{(\ell-1)},$$

$$\mathrm{I}_{\mathcal{U}}(\text{Attention}^{(\ell)}(E^{(\ell-1)})_k) = \frac{\sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell)} \|E_i^{(\ell-1)}\|\, \mathrm{I}_{\mathcal{U}}(E_i^{(\ell-1)})}{\sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell)} \|E_i^{(\ell-1)}\|}.$$

Then, if we approximate the norm of the embeddings $\|E_i^{(\ell-1)}\|$ with a constant, we obtain a simplified expression

$$\mathrm{I}_{\mathcal{U}}(\text{Attention}^{(\ell)}(E^{(\ell-1)})_k) = \sum_{i=1}^{k} \mathbf{A}_{k,i}^{(\ell)} \mathrm{I}_{\mathcal{U}}(E_i^{(\ell-1)}).$$

With this approximation, equation 8 becomes equal to:

$$\mathrm{I}_{\mathcal{U}}(E_k^{(\ell)}) = \frac{\mathrm{I}_{\mathcal{U}}(E_k^{(\ell-1)})}{1 + r_k^{(\ell-1)}} r_k^{(\ell-1)} + \frac{\sum_{i=1}^{s} \mathbf{A}_{k,i}^{(\ell)} \, \mathrm{I}_{\mathcal{U}}(E_i^{(\ell-1)})}{1 + r_k^{(\ell-1)}}. \tag{9}$$

C Additional experiments

C.1 Evaluating the quality of generated summaries

In addition to verifying that the LLM summary is in French in Section 3, we have also evaluated the quality of the outputs using BERTScore [25], calculated in comparison to target summaries generated by a Llama 3 70B model [2].

To highlight the pertinence of BERTScore, in evaluating the quality of the summaries, we show in Fig. 4 the distribution of the observed BERTScore conditioned to the generated text being in French or not. We observe that the distribution for texts generated in French is shifted to the right compared to those not in French, indicating that BERTScore is a suitable metric for assessing the quality of generated texts.

To measure the impact of GUIDE on the quality of the LLM outputs, we have evaluated the winning rate by comparing the quality of the texts generated with and without GUIDE in terms of BERTScore. Specifically, for each pair of texts generated in French $(t_{i,\Delta}, t_{i,\text{raw}})$ by GUIDE and the unmodified (raw) model, we have determined which text had a higher BERTScore. Table 1 shows that for a small enough choice of Δ, the quality of the output is not highly affected, with winning rates of 50.5% for $\Delta = 0.5$ and $\Delta = 1$ and 49% for $\Delta = 2$. These results indicate that GUIDE maintains the model's capability to generate semantically correct text. However, as mentioned in Sec. 2, larger values of Δ, e.g., $\Delta = 5$, results in poor outputs (see also Appendix E).

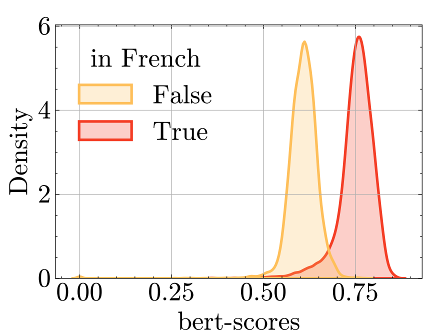


Figure 4: Distribution of Bert scores conditioned to the generated text being in French or not.

Table 1: Winning rate of text generated in French for GUIDE versus the baseline model (Mistral 7B).

Δ	Winning Rate
0.5	50.5%
1	50.5%
2	49%
5	38.5%

C.2 Summarization in French with another model

We have conducted the summarization study presented in Section 3 also using the Gemma 2 - 2B Instruct model [20]. Our results indicate that, even with smaller models, GUIDE can still improve the accuracy of generating text in French, increasing it from 43.3% to 59.8% without a significant loss in quality (see Figure 5 and Table 2). Similar to our findings with the Mistral model, using uppercase letters does not lead to a significant improvement in performance.

Additionally, our results show that setting $\Delta = 5$ reduces the accuracy to zero.

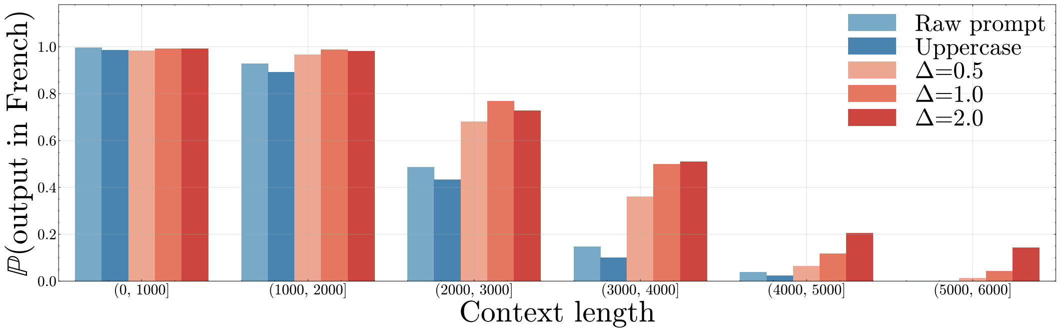


Figure 5: Probability of outputting a summary in French - Gemma 2 - 2B

Table 2: Winning rate of text generated in French for GUIDE versus the baseline model (Gemma 2-2B).

Δ	Winning Rate
0.5	53.5%
1	46.7%
2	45.1%

C.3 Further experiments showcasing the benefits of GUIDE

C.3.1 A needle in a haystack

To evaluate the impact of our approach on the model ability to retain information, we have conducted the Needle in a Haystack test using using Mistral 7B Instruct[9]. This test involves embedding specific information at a particular location within a text and then asking a question related to that information at the end of the text. Our hypothesis is that by adding extra attention to this text, the model's outputs would improve, as the final representation should be more closely aligned with the information tokens.

We have followed the methodology outlined by [10]. Specifically, we have inserted specific information, referred to as the "needle" at variable positions within a given text. After this insertion, we have asked a question to the LLM related to the inserted information (see the complete prompt on C.5).

To conduct this experiment, we have sampled 200 texts from the OpenWebText [6] dataset, selecting 25 texts for each context window of size 500, ranging from 0 to 4000 tokens. For each text, the needle was inserted at 10 different quantiles (10%, 20%, ..., 100%). We placed the needle immediately after a period ('.') to maintain the semantic integrity of the text.

Figure 6 shows the probability of outputting the correct phrase over the context length and the position of the needle, respectively. The Mistral model demonstrates stable performance across varying context lengths and needle positions within this window.

As expected, the addition of Δ to the needle tokens consistently enhances performance from 89.0% to 95.9%, with optimal values of Δ around 1. We can also note that, on average, the LLM is more effective at retrieving information when it is located at the beginning or the end of the text. This is in accordance with previous results [11, 10].

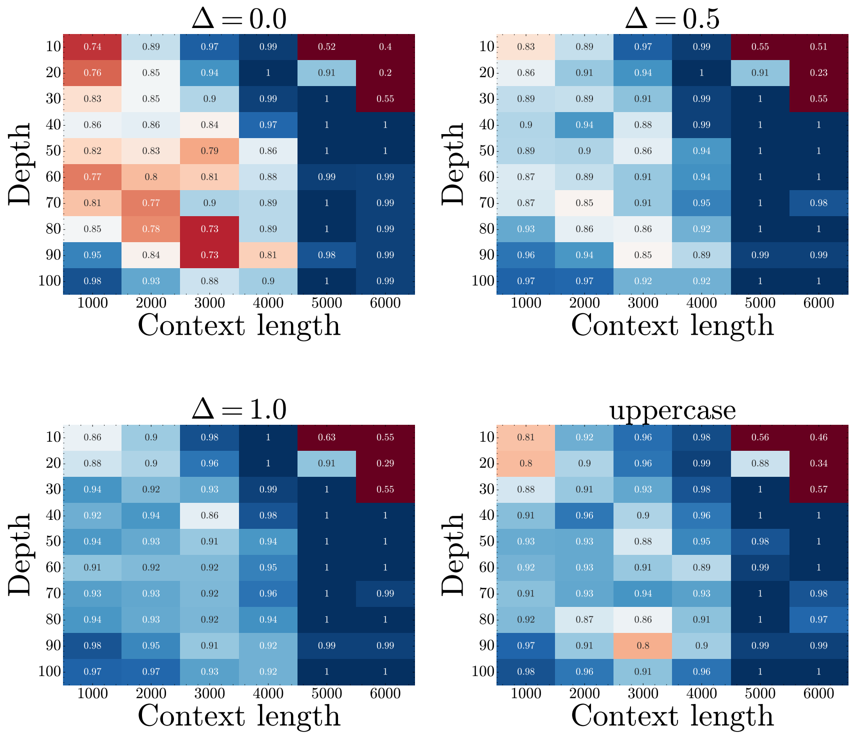


Figure 6: Heatmap of scores in a needle in haystack test.

C.3.2 JSON generation

To assess the efficiency of GUIDE in generating outputs in a specified format, we have conducted experiments focused on JSON generation. For our inputs, we have used texts from books written between 1510 and 1699, sourced from the BL Books dataset [12]. We have prompted the model to extract and generate key information about each book in a predetermined JSON format, as detailed in C.5.

We have randomly selected 300 books from the BL Books dataset and divided each text into context length windows of 500 tokens, ranging from 0 to 4000 tokens. These text segments were then incorporated into our template, where the Mistral model was expected to generate a JSON output that precisely followed the specified format.

We have inputted special attention into the tokens of `Your response should follow exactly this template` and we have then evaluated the Jaccard index between the keys of the generated JSON and the schema.

We observed that the optimal value for Δ is approximately 3, resulting in an average score improvement of 30% compared to the raw model (Figure 7).

We also note that in almost every generation the scores were 0 or 100%. This indicates that most of the time, the generated output was either a perfect match to the requested schema or not in JSON format at all.

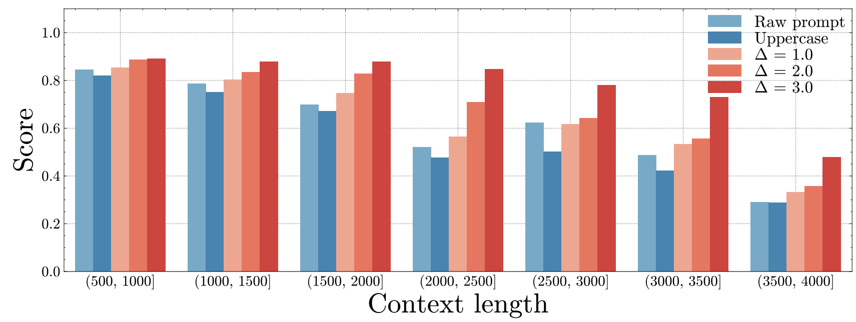


Figure 7: Jaccard Index vs Context length for the JSON generation experiment

C.4 Influence as a predictor of instruction following

For each of the experiments in Section C, we have evaluated the relationship between the *Influence* metric and the probability of obtaining correct outputs. To achieve this, we have calculated both the ROC AUC score and the correlation between the importance of instruction tokens and the last token in the sequence. Then, we have compared these results through non-gradient metrics, such as Attention Rollout and raw attention scores. Our hypothesis is that Influence has a strong positive correlation with the probability of correct outputs.

Given that the ROC AUC is a classification metric, it was necessary to binarize our scores. In the French summarization experiment, we have done this by assigning a score of 1 to texts in French and 0 to those in other languages. In the "needle in a haystack" experiment, a score of 1 was given to prompts that successfully identified the needle information, while those that did not were assigned a 0. Similarly, for the JSON generation experiment, outputs that adhered to the JSON format were assigned a 1, and those that did not were assigned a 0.

Table 3 shows the correlation and ROC AUC of each metric to correct output. We note that, attention rollout shows a negative correlation and an AUC below 0.5 in two out of three experiments. This observation supports our initial hypothesis that attention rollout may not accurately reflect the model focus. Also, as expected, we also see that raw attention has a random behavior in two of three setups, with ROC AUC scores around 0.5.

The stronger positive correlation and ROC AUC between *Influence* and the likelihood of following instructions supports our hypothesis that our metric better quantifies the attention flow in a Transformer than other existing non-gradient metrics.

Table 3: AUC and correlation between the different metrics and the probability that the LLM generates a correct output.

	Metric	ROC AUC	Correlation
Summarizing in french	Influence	0.74	0.72
	Attention rollout	0.24	-0.35
	Raw attention	0.58	0.13
A needle in a haystack	Influence	0.62	0.12
	Attention rollout	0.55	0.10
	Raw attention	0.48	-0.03
JSON generation	Influence	0.63	0.23
	Attention rollout	0.31	-0.29
	Raw attention	0.64	0.23

C.5 Prompts used in experiments

Summarization in French

Summarize in French

{context}

A needle in a haystack

<question>
Your objective is to answer the following question based on the context:

{question}

Don't give information outside the document or repeat our findings
</question>

{context with needle}

<question>
Your objective is to answer the following question based on the context:

{question}

Don't give information outside the document or repeat our findings
</question>

JSON generation

```
You are an assistant designed to provide information in JSON format.
I will give you a story, and you need to extract and return specific details from the story.
Do not output anything else than the JSON.
Your response should follow exactly this template:

<schema>
{
   "title": "title of the story (string)",
   "genre": string,
   "characters":
      [
         {
            "name": string,
            "description": string. If not available set it to none
         }
      ] (one dict per character),
   "author": "the author of the story. If not available, set it to None",
   "summary": "a brief summary of the story. Do not write more than 50 words",
   "date": "when the story was released (string)",
   "scenery": "where the story takes place (string)",
}

</schema>

{content}
```

D Supervised Finetuning hyperparameters

In our supervised fine-tuning experiments, we leveraged LoRa techniques [7], setting the sequence length to 8192 using sample packing and block-attention to prevent cross-sample contamination. We configured the LoRa rank to 64 and set the alpha parameter to 16. For regularization, we applied a dropout rate of 0.05. To maximize the adapter's expressiveness, our LoRa implementation targeted all modules. The batch size was set to 128k tokens, with a maximum learning rate of 1e-4, following a cosine scheduler with a 10-step warm-up. Training was conducted for 2 epochs, tracking the number of tokens processed at each step.

E Examples of poor generation of text

This Appendix provides examples of the model failing to follow user instructions.

E.1 Raw generation (without GUIDE)

A common error occurs when the model simply repeats the given prompt. This example has been obtained using multinomial sampling at generation time and without GUIDE ($\Delta = 0$).

Prompt

Summarize in French:
The red Ford Expedition mounted the sidewalk along DeKalb Avenue in Fort Greene, Brooklyn, on Saturday afternoon not once, but twice.
The first time, it narrowly missed two people at the southwest corner of Clermont Avenue, crashing into a parked car, jumping the curb and hitting another vehicle, the police said.
But the second time, pedestrians along the busy Brooklyn thoroughfare were not so lucky: The S.U.V. struck and killed a 9-year-old boy, Lucian Merryweather, who was with his mother on the northeast corner. His 5-year-old brother and a woman who had been in the crosswalk were hurt.
At first glance, the crash seemed likely to attract attention but unlikely to result in criminal charges. The driver, Anthony Byrd, 59, of Clinton Hill, remained at the scene and was not intoxicated, the police said.

Output

The red Ford Expedition mounted the sidewalk along DeKalb Avenue in Fort Greene, Brooklyn, on Saturday afternoon twice. The first time it narrowly missed two people and crashed into a parked car, jumping the curb and hitting another vehicle. However, during the second time, pedestrians were not so lucky as the S.U.V. struck and killed a 9-year-old boy, Lucian Merryweather, who was with his mother on the northeast corner. His 5-year-old brother and a woman who had been in the crosswalk were hurt. Initially, the crash appeared to be unlikely to attract attention and result in criminal charges, as the driver, Anthony Byrd, 59, of Clinton Hill, remained at the scene and was not intoxicated, the police said.

The LLM sometimes generates a very limited number of tokens that, although present in the prompt, are nonsensical when presented alone (and do not fulfil the user task). In this instance, we employ multinomial sampling to generate the text (without GUIDE).

Prompt

Summarize in French:
Vine
Oh, those crazy Sacramento Kings.
On Saturday, head coach George Karl raised some eyebrows by saying guard Seth Curry would only last in the NBA for "a couple years," per Blake Ellington of Sactown Royalty.
According to Ellington, Curry posted (and deleted) this tweet:
Look familiar?
A day later, the 25-year-old started and knocked down four of seven shots from the field en route to a 14-point performance in a 133-111 win over the Dallas Mavericks. During Curry's postgame interview, DeMarcus Cousins—rocking a sweet towel around his head—chimed in.
"You gon' be in the league a long time, boy," the big fella said emphatically. "Remember that. A long time."
Maybe it's a coincidence, and the All-Star was simply supporting a teammate who's been struggling.
But Cousins and Karl's relationship has been volatile ever since the 64-year-old took the Kings job in 2014-15.
Snake emoji, indeed.
Update
Regardless of the reason for Cousins' support, big brother Stephen Curry approved, reacting to the incident with a 100-laden tweet:
[Vine]

Output

Tweet: Look familiar?

E.2 Generation with GUIDE using unreasonably large Δ

We have noted that $\Delta > 5$ very often produces incorrect/nonsensical output. In the following example, we have set $\Delta = 9$ and performed greedy decoding.

Prompt

<!!-> Summarize in French <-!!>
Paris is the capital and largest city of France. With an official estimated population of 2,102,650 residents in January 2023 in an area of more than 105 km2 Paris is the fourth-largest city in the European Union and the 30th most densely populated city in the world in 2022. Since the 17th century, Paris has been one of the world's major centres of finance, diplomacy, commerce, culture, fashion, and gastronomy. For its leading role in the arts and sciences, as well as its early and extensive system of street lighting, in the 19th century, it became known as the City of Light

Output

1. 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century, 19th century,