DIAGNOSING FP4 INFERENCE: A LAYER-WISE AND BLOCK-WISE SENSITIVITY ANALYSIS OF NVFP4 AND MXFP4

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantization addresses the high resource demand for large language models (LLMs) by alleviating memory pressure and bandwidth congestion and providing significantly scaled compute power with a tolerable impact on accuracy. Four-bit floating point (FP4), the lowest-precision format that preserves essential numerical properties such as exponent and sign, has begun to be adopted in cutting-edge architectures, including Blackwell and AMD CDNA, to support LLM quantization and reduce deployment costs. Although aggressive quantization can yield efficiency gains, the quantization sensitivity of within-transformer layers and whether these sensitivities generalize across existing FP4 formats and model scales remain underexplored. To elucidate quantization sensitivity, this study conducts a systematic analysis of two FP4 formats, MXFP4 and NVFP4, across three Qwen2.5 model scales (0.5B, 7B, and 14B), using controlled component-wise and block-wise isolation methodologies. We observe that MLP up- and down-projection layers consistently dominate in terms of sensitivity, while gate and attention projections are moderately and substantially less sensitive to FP4 quantization, respectively. We further find that sensitivity does not universally localize to the final blocks, but early blocks can be highly sensitive, particularly under MXFP4. Our results provide a diagnostic characterization of the inference behavior of FP4 across components, depths, and FP4 formats.

1 INTRODUCTION

During the past decade, large language models (LLMs) have reshaped both the AI research paradigm and industrial practices (Maslej et al., 2025), and empirical scaling laws (Kaplan et al., 2020) demonstrate that scaling up the size of the model can lead to further performance improvements. However, deploying larger-scale models–trained over prolonged runs across thousands of server-class accelerators–substantially increases the costs and energy consumption required to power LLM-based applications. Quantization (Jacob et al., 2017; Dettmers et al., 2022b), which narrows the numerical precision, has emerged as a promising technique to alleviate pressure on memory and bandwidth, scale the logical computational capability (e.g., TFLOPs), and thus reduce the cost of model deployment. Compared with traditional full- or half-precision formats (e.g., 32-bit and 16-bit), modern accelerators now support lower-precision representations, including 8-bit Dettmers et al. (2022b), 6-bit, and even 4-bit formats (NVIDIA, 2024a; AMD, 2024a), enabling the practical deployment of aggressive quantization techniques.

In particular, FP4 quantization has seen a growing adoption, with formats such as MXFP4 (AMD, 2024a) and NVFP4 (NVIDIA, 2024a) increasingly used in both the development and deployment of LLMs (Liu et al., 2023; NVIDIA et al., 2025; Salvator, 2025). Prior work has established that quantization error in transformers is highly non-uniform across layers and components (Fan et al., 2019; Frantar et al., 2022; Lin et al., 2024), and that rare activation outliers dominate low-bit quantization error (Dettmers et al., 2022a; Xiao et al., 2023), motivating component-aware handling. Additionally, previous studies on layer-wise importance (Xiong et al., 2020; Zhang et al., 2024; Nepal et al., 2025; Skean et al.) often emphasize that later layers play a dominant role in shaping model outputs and task performance. However, to our knowledge, no comprehensive component- and block-wise sensitivity analysis exists specifically for FP4 formats, leaving open questions about which architectural elements are most affected and how sensitivity varies across model scales. To address this gap,

we analyze the quantization sensitivity of the MXFP4 and NVFP4 formats across three Qwen2.5 model scales (0.5B, 7B, and 14B), using controlled component- and block-wise isolation methodologies.

This paper makes the following **contributions**:

- It presents an analysis indicating that up- and down-projections in the MLP layer consistently form the most sensitive tier, and this trend remains stable across FP4 formats and model scales.
- It observes, across blocks, that the FP4 sensitivity is not necessarily confined to the final blocks; for the 0.5B model under MXFP4, earlier blocks exhibit strong sensitivity, challenging the common assumption that the last blocks dominate.
- It reveals that extreme activation outliers are consistent with the high sensitivity of the down projection, but do not fully account for FP4 sensitivity, as the up projection is comparably sensitive despite lower outlier ratios.
- It demonstrates that the model scale affects the magnitude of sensitivity but not the relative sensitivity between components.

2 Background and Motivation

Background: Large language models (LLMs) impose substantial memory bandwidth and compute demands, making low-precision inference a central technique for efficient deployment. Quantization formats such as FP16 and FP8 have demonstrated strong accuracy and efficiency trade-offs (Micikevicius et al., 2018; 2022) and are now widely supported in production inference and training pipelines Reddi et al. (2019). More recently, proposals have emerged for inference in ultra-low-precision regimes, particularly with variants of FP4 that preserve sign and exponent while further reducing storage and bandwidth demands , and accelerator vendors such as NVIDIA and AMD have introduced both native FP4 tensor-operation support and specialized FP4 formats (e.g., NVFP4 (NVIDIA, 2024a) and MXFP4 (AMD, 2024a)), along with hardware-accelerated scaling mechanisms, making FP4 inference increasingly practical rather than purely algorithmic (AMD, 2024b). As a result, FP4 has transitioned from a research concept to a deployable precision target for large-scale LLM inference.

Motivation: Although FP4 inference is now supported by modern accelerators, it is increasingly evident that there is no universal recipe for applying FP4 across different models, formats, and applications, as aggressive quantization can significantly degrade inference quality. Prior work in low-bit quantization has shown that quantization behavior is highly sensitive to activation distributions and data characteristics, motivating activation-aware and data-dependent strategies rather than uniform precision assignments (Lin et al., 2024; Xiao et al., 2023). In parallel, different FP4 formats employ distinct scaling mechanisms and calibration assumptions, leading to format-specific considerations even for the same model architecture (Abecassis et al., 2025). Hence, FP4 sensitivity is jointly influenced by model architecture, format preference, and data distribution, and heuristics that are effective in one setting may fail in another. Consequently, a principled FP4 deployment requires a detailed analysis of FP4 sensitivity within transformer layers and across blocks to avoid accuracy degradation. This motivates a controlled component- and block-wise sensitivity analysis to diagnose FP4 failure modes and to provide a foundation for adapting FP4 recipes to specific models, formats, and deployment scenarios.

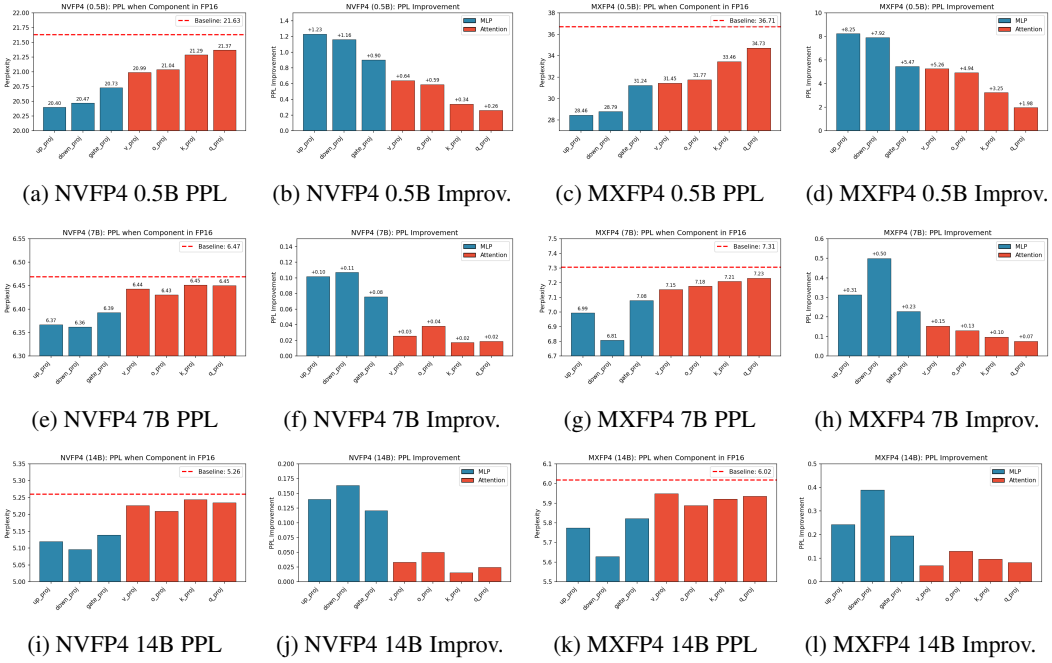
3 Experimental Design

We adopt a controlled isolation methodology that quantizes individual components and blocks while keeping all other factors fixed. Under this setting, we formulate four hypotheses:

1. **Mechanism:** Component sensitivity to FP4 is primarily determined by activation tail-heaviness (e.g., Max/P99) rather than by whether the component belongs to MLP or attention.
2. **Isolation/Ordering:** When components are isolated and quantized one at a time, their sensitivity ranking remains consistent across experiments.
3. **Model Scale:** Increasing model size (0.5B $\rightarrow$ 7B $\rightarrow$ 14B) mainly affects the magnitude of sensitivity, not the relative ordering of the sensitive components.
4. **Block Localization:** FP4 sensitivity does not always peak in the final blocks; early-block sensitivity can emerge, particularly under MXFP4.

We experiment with three different scales of Qwen2.5 models: 0.5B (24 layers), 7B (28 layers), and 14B (48 layers) to assess scale generalization. Across FP4 formats, we compare MXFP4 (E2M1, 32-element blocks, shared 8-bit exponent (Aralimatti, 2025)) and NVFP4 (dynamic scaling, 16-element blocks, 4-bit scales, max calibration algorithm (NVIDIA, 2024a)). Experiments are conducted on RTX 5090 (NVIDIA, 2024b) for smaller models and RTX 6000 Pro (NVIDIA, 2024c) for larger models. Perplexity is measured on WikiText-2 (Merity et al., 2016) using 256 calibration samples.

We use on-the-fly quantization such that weights are stored in FP16 and quantized to FP4 during inference, then dequantized back for computation. For component sensitivity, we quantize one of all seven projection types (Query, Key, Value, Output, Gate, Up, Down) to FP4 at a time, and keep remaining six of them in FP16 across all layers. For block sensitivity, we keep one specific block's component in FP16 while the remaining six component types plus other blocks of the same type are quantized to FP4.



(a) NVFP4 0.5B PPL (b) NVFP4 0.5B Improv. (c) MXFP4 0.5B PPL (d) MXFP4 0.5B Improv.

(e) NVFP4 7B PPL (f) NVFP4 7B Improv. (g) MXFP4 7B PPL (h) MXFP4 7B Improv.

(i) NVFP4 14B PPL (j) NVFP4 14B Improv. (k) MXFP4 14B PPL (l) MXFP4 14B Improv.

Figure 1: Component sensitivity comparison across three model scales. Rows: 0.5B, 7B, 14B. Blue = MLP, Red = Attention. MLP projections (down and up) consistently form the most sensitive tier across all scales and formats.

4 Results

Component Sensitivity. Figure 1 shows that FP4 sensitivity is highly non-uniform across transformer components. Across all model scales and both FP4 formats, the quantization sensitivity is dominated by MLP projections, with up- and down-projections consistently forming the most sensitive tier, while gate and attention projections are moderately and substantially less sensitive, respectively. Although larger models become more sensitive to quantization—and MXFP4 shows higher sensitivity than NVFP4, the relative tiering of components remains stable across all settings. Tables 2, 3, and 4 in the Appendix provide detailed numerical results regarding the sensitivity for 0.5B, 7B, and 14B model scales, respectively.

Key Takeaway #1: MLP projections are highly sensitive and consistently behave the most sensitive tier to FP4 quantization, whereas gate and attention projections are moderately and substantially less sensitive, respectively.

Block Sensitivity. We analyze the FP4 sensitivity across transformer depth using block-wise isolation and observe that the sensitivity is structured rather than uniformly concentrated in the final blocks. While later blocks are often sensitive, particularly for MLP projections, early-block sensitivity can also be substantial, most clearly in the 0.5B model and under MXFP4. At larger scale, sen-

sitivity is more concentrated towards later blocks, although early blocks can still exhibit non-trivial effects; notably, for 14B under MXFP4, isolating early blocks can even yield negative improvement. Figure 2 shows the block-wise perplexity when quantizing down projection across all three model scales.

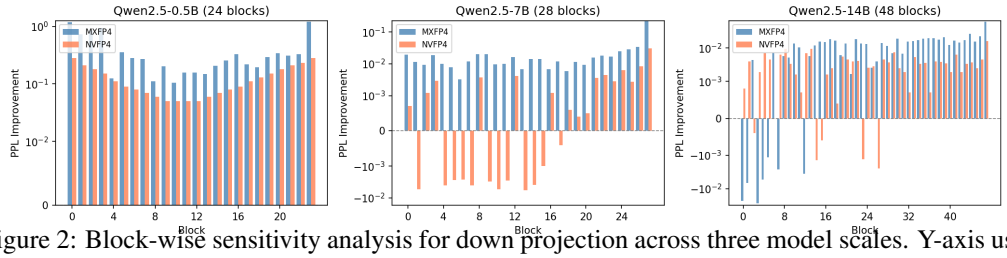


Figure 2: Block-wise sensitivity analysis for down projection across three model scales. Y-axis uses symlog scale. Positive values indicate PPL improvement when keeping that block in FP16.

Key Takeaway #2: FP4 sensitivity does not always peak in the final blocks and can vary across depths depending on the model network configuration.

Activation Outlier Analysis. To understand the component sensitivity ordering, we analyze activation statistics as reported in Table 1. Down projection exhibits extreme outlier behavior with Max/P99.9 ratios 10-100× larger than other components, consistent with its high FP4 sensitivity. However, up projection shows comparable sensitivity despite much lower outlier ratios, indicating that outliers alone do not fully explain the FP4 sensitivity.

	Qwen2.5-0.5B			Qwen2.5-7B			Qwen2.5-14B		
Component	P99.9	Max	Ratio	P99.9	Max	Ratio	P99.9	Max	Ratio
`down_proj`	2.8	253	**159**	4.3	310	**80**	4.2	300	**334**
`up_proj`	11.2	58	5.5	9.9	49	5.5	5.1	42	10
`gate_proj`	11.2	58	5.5	9.9	49	5.5	5.1	46	11
`v_proj`	13.0	36	2.9	7.0	30	4.4	4.8	23	4.8
`o_proj`	2.4	6	2.6	2.4	9	3.4	1.8	6	3.6
`k_proj`	13.0	36	2.9	7.0	29	4.3	4.8	24	4.9
`q_proj`	13.0	36	2.9	7.0	29	4.4	4.8	22	4.7

Table 1: Activation statistics by component across all model scales. All values are averaged across layers. The Max/P99.9 ratio is computed as the mean of per-layer ratios (not the ratio of averaged Max to averaged P99.9), which better captures outlier severity across depth. Down projection, as highlighted, consistently shows substantially worse ratios than other components.

Key Takeaway #3: Extreme activation outliers in down projection, which receives post-activation values (e.g., after SiLU/SwiGLU gating (Shazeer, 2020)), are consistent with its high sensitivity, but up projection exhibits comparable sensitivity despite much lower outlier ratios, suggesting that extreme outliers alone do not fully account for FP4 sensitivity.

5 Conclusion and Discussion

We presented a controlled component-wise and block-wise analysis of FP4 quantization in transformer models across multiple formats and scales. Our results show that FP4 sensitivity is dominated by MLP projections with a stable tiering across formats and model sizes, while model scale primarily affects sensitivity magnitude. At the block level, sensitivity is structured across depth and not exclusively concentrated in the final blocks, with early-block effects emerging in some configurations. Together, these findings provide a diagnostic view of FP4 inference behavior and motivate component- and depth-aware approaches to low-precision deployment. Looking ahead, this analysis can be extended to additional model families and larger scales, as well as to settings that employ native FP4 computation kernels. Future work should also evaluate FP4 sensitivity on diverse downstream tasks beyond perplexity on WikiText, such as reasoning, coding, and instruction following benchmarks, to better understand how component level quantization effects translate to task specific performance degradation.

REFERENCES

Felix Abecassis, Anjulie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, et al. Pretraining large language models with nvfp4. *arXiv preprint arXiv:2509.25149*, 2025.

AMD. Mxfp4 and mxfp6 quantization on amd instinct gpus. https://rocm.blogs.amd.com/software-tools-optimization/mxfp4-mxfp6-quantization/README.html, 2024a.

AMD. Precision support reference. https://rocm.docs.amd.com/en/latest/reference/precision-support.html, 2024b.

Rakshit Aralimatti. What's mxfp4? the 4-bit secret powering openai's gpt-oss models on modest hardware, August 2025. URL https://huggingface.co/blog/RakshitAralimatti/learn-ai-with-me. Hugging Face community article, published August 8, 2025.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332, 2022a.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale, 2022b. URL https://arxiv.org/abs/2208.07339.

Vage Egiazarian, Roberto L Castro, Denis Kuznedelev, Andrei Panferov, Eldar Kurtic, Shubhra Pandit, Alexandre Marques, Mark Kurtz, Saleh Ashkboos, Torsten Hoefler, et al. Bridging the gap between promise and performance for microscaling fp4 quantization. *arXiv preprint arXiv:2509.23202*, 2025.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Alejandro Hernández-Cano, Dhia Garbaya, Imanol Schlag, and Martin Jaggi. Towards fully fp8 gemm llm training at scale. *arXiv preprint arXiv:2505.20524*, 2025.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference, 2017. URL https://arxiv.org/abs/1712.05877.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

Shih-yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. Llm-fp4: 4-bit floating-point quantized transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 592–605. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.39. URL http://dx.doi.org/10.18653/v1/2023.emnlp-main.39.

Wenyuan Liu, Haoqian Meng, Yilun Luo, Peng Zhang, and Xindian Ma. Micromix: Efficient mixed-precision quantization with microscaling formats for large language models. *arXiv preprint arXiv:2508.02343*, 2025.

Nestor Maslej, Loredana Fattorini, Raymond Perrault, Yolanda Gil, Vanessa Parli, Njenga Kariuki, Emily Capstick, Anka Reuel, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Juan Carlos Niebles, Yoav Shoham, Russell Wald, Toby Walsh, Armin Hamrah, Lapo Santarlasci, Julia Betts Lotufo, Alexandra Rome, Andrew Shi, and Sukrut Oak. Artificial intelligence index report 2025, 2025. URL https://arxiv.org/abs/2504.07139.

Haoqian Meng, Yilun Luo, Yafei Zhao, Wenyuan Liu, Peng Zhang, and Xindian Ma. Arcquant: Boosting nvfp4 quantization with augmented residual channels for llms. *arXiv preprint arXiv:2601.07475*, 2026.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training, 2018. URL https://arxiv.org/abs/1710.03740.

Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, Naveen Mellempudi, Stuart Oberman, Mohammad Shoeybi, Michael Siu, and Hao Wu. Fp8 formats for deep learning, 2022. URL https://arxiv.org/abs/2209.05433.

Aadim Nepal, Safal Shrestha, Anubhav Shrestha, Minwu Kim, Jalal Naghiyev, Ravid Shwartz-Ziv, and Keith Ross. Layer importance for mathematical reasoning is forged in pre-training and invariant after post-training. *arXiv preprint arXiv:2506.22638*, 2025.

NVIDIA. Introducing nvfp4 for efficient and accurate low precision inference. https://developer.nvidia.com/blog/introducing-nvfp4-for-efficient-and-accurate-low-precision-inference, 2024a.

NVIDIA. NVIDIA RTX Blackwell GPU Architecture, 2024b. URL https://images.nvidia.com/aem-dam/Solutions/geforce/blackwell/nvidia-rtx-blackwell-gpu-architecture.pdf. Whitepaper on NVIDIA RTX Blackwell GPU architecture, 57 pages.

NVIDIA. NVIDIA RTX PRO 6000 Blackwell Workstation Edition, 2024c. URL https://www.nvidia.com/content/dam/en-zz/Solutions/data-center/rtx-pro-6000-blackwell-workstation-edition/workstation-blackwell-rtx-pro-6000-workstation-edition-nvidia-us-3519208-web.pdf. Datasheet for RTX PRO 6000 Blackwell Workstation Edition (NVIDIA), available at:.

NVIDIA, Felix Abecassis, Anjulie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, Ian Buck, Bryan Catanzaro, Jinhang Choi, Mike Chrzanowski, Eric Chung, Victor Cui, Steve Dai, Bita Darvish Rouhani, Carlo del Mundo, Deena Donia, Burc Eryilmaz, Henry Estela, Abhinav Goel, Oleg Goncharov, Yugi Guvvala, Robert Hesse, Russell Hewett, Herbert Hum, Ujval Kapasi, Brucek Khailany, Mikail Khona, Nick Knight, Alex Kondratenko, Ronny Krashinsky, Ben Lanir, Simon Layton, Michael Lightstone, Daniel Lo, Paulius Micikevicius, Asit Mishra, Tim Moon, Deepak Narayanan, Chao Ni, Abhijit Paithankar, Satish Pasumarthi, Ankit Patel, Mostofa Patwary, Ashwin Poojary, Gargi Prasad, Sweta Priyadarshi, Yigong Qin, Xiaowei Ren, Oleg Rybakov, Charbel Sakr, Sanjeev Satheesh, Stas Sergienko, Pasha Shamis, Kirthi Shankar, Nishant Sharma, Mohammad Shoeybi, Michael Siu, Misha Smelyanskiy, Darko Stosic, Dusan Stosic, Bor-Yiing Su, Frank Sun, Nima Tajbakhsh, Shelby Thomas, Przemek Tredak, Evgeny Tsykunov, Gandhi Vaithilingam, Aditya Vavre, Rangharajan Venkatesan, Roger Waleffe, Qiyu Wan, Hexin Wang, Mengdi Wang, Lizzie Wei, Hao Wu, Evan Wu, Keith Wyss, Ning Xu, Jinze Xue, Charlene Yang, Yujia Zhai, Ruoxi Zhang, Jingyang Zhu, and Zhongbo Zhu. Pretraining large language models with nvfp4, 2025. URL https://arxiv.org/abs/2509.25149.

Houwen Peng, Kan Wu, Yixuan Wei, Guoshuai Zhao, Yuxiang Yang, Ze Liu, Yifan Xiong, Ziyue Yang, Bolin Ni, Jingcheng Hu, et al. Fp8-lm: Training fp8 large language models. *arXiv preprint arXiv:2310.18313*, 2023.

Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, Ramesh Chukka, Cody Coleman, Sam Davis, Pan Deng, Greg Diamos, Jared Duke, Dave Fick, J. Scott Gardner, Itay Hubara, Sachin Idgunji, Thomas B. Jablin, Jeff Jiao, Tom St. John, Pankaj Kanwar, David Lee, Jeffery Liao, Anton Lokhmotov, Francisco Massa, Peng Meng, Paulius Micikevicius, Colin Osborne, Gennady Pekhimenko, Arun Tejusve Raghunath Rajan, Dilip Sequeira, Ashish Sirasao, Fei Sun, Hanlin Tang, Michael Thomson, Frank Wei, Ephrem Wu, Lingjie Xu, Koichi Yamada, Bing Yu, George Yuan, Aaron Zhong, Peizhao Zhang, and Yuchen Zhou. Mlperf inference benchmark, 2019.

Dave Salvator. Mlperf training v5.1: First verified fp4 training results using nvidia blackwell gpus. https://blogs.nvidia.com/blog/mlperf-training-benchmark-blackwell-ultra/, 2025. Accessed: 2026-02-02.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models, 2025. *URL https://arxiv.org/abs/2502.02013*, 1.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Albert Tseng, Tao Yu, and Youngsuk Park. Training llms with mxfp4. *arXiv preprint arXiv:2502.20586*, 2025.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

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

Jintao Zhang, Jia Wei, Pengle Zhang, Xiaoming Xu, Haofeng Huang, Haoxu Wang, Kai Jiang, Jun Zhu, and Jianfei Chen. Sageattention3: Microscaling fp4 attention for inference and an exploration of 8-bit training. *arXiv preprint arXiv:2505.11594*, 2025.

Yang Zhang, Yanfei Dong, and Kenji Kawaguchi. Investigating layer importance in large language models. *arXiv preprint arXiv:2409.14381*, 2024.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems*, 6:196–209, 2024.

A Related Work

Prior work on efficient large language model deployment has primarily focused on higher-precision regimes, including FP16 and FP8. Foundational model families such as LLaMA-2 and LLaMA-3 are trained and deployed in FP16, establishing strong baselines for accuracy and scaling behavior

Touvron et al. (2023); Grattafiori et al. (2024). More recent work explores FP8 training and inference, showing that carefully designed scaling, normalization, and GEMM kernels can enable stable low-precision execution at scale Peng et al. (2023); Hernández-Cano et al. (2025). These approaches largely treat quantization as a uniform transformation across model components and layers, without examining how sensitivity varies within the transformer architecture.

Recent studies have pushed further toward ultra-low-precision FP4 formats to maximize efficiency. Prior work proposes microscaling and mixed-precision techniques for FP4 inference and training, including NVFP4 and MXFP4-based approaches, residual channels, and hybrid precision strategies Egiazarian et al. (2025); Tseng et al. (2025); Abecassis et al. (2025); Zhao et al. (2024); Meng et al. (2026); Liu et al. (2025). More recently, SageAttention3 investigates microscaling FP4 specifically for attention inference and explores interactions with 8-bit training Zhang et al. (2025), highlighting that different architectural components may exhibit distinct FP4 sensitivity. In contrast to these method-driven efforts, our work provides a diagnostic component-wise and block-wise analysis of FP4 sensitivity, characterizing how quantization effects distribute across components and depth rather than proposing a new quantization scheme.

B Component Sensitivity Tables

Table 2: Component sensitivity under NVFP4 and MXFP4 for Qwen2.5-0.5B. PPL improvement measures sensitivity (higher = keeping that component in FP16 helps more).

	NVFP4		**MXFP4**	
Component	PPL	Improvement	PPL	Improvement
Baseline	21.63	—	36.71	—
up_proj	20.40	+1.23	28.46	+8.25
down_proj	20.47	+1.16	28.79	+7.92
gate_proj	20.73	+0.90	31.24	+5.47
v_proj	20.99	+0.64	31.45	+5.26
o_proj	21.04	+0.59	31.77	+4.94
k_proj	21.29	+0.34	33.46	+3.25
q_proj	21.37	+0.26	34.73	+1.98

Table 3: Component sensitivity under NVFP4 and MXFP4 for Qwen2.5-7B.

	NVFP4		**MXFP4**	
Component	PPL	Improvement	PPL	Improvement
Baseline	6.47	—	7.31	—
down_proj	6.36	+0.11	6.81	+0.50
up_proj	6.37	+0.10	6.99	+0.31
gate_proj	6.39	+0.08	7.08	+0.23
o_proj	6.43	+0.04	7.18	+0.13
v_proj	6.44	+0.03	7.15	+0.15
k_proj	6.45	+0.02	7.21	+0.10
q_proj	6.45	+0.02	7.23	+0.07

Table 4: Component sensitivity under NVFP4 and MXFP4 for Qwen2.5-14B.

	NVFP4		MXFP4	
Component	PPL	Improvement	PPL	Improvement
Baseline	5.26	—	6.02	—
`down_proj`	5.10	+0.16	5.63	+0.39
`up_proj`	5.12	+0.14	5.77	+0.24
`gate_proj`	5.14	+0.12	5.82	+0.20
`o_proj`	5.21	+0.05	5.89	+0.13
`v_proj`	5.23	+0.03	5.95	+0.07
`k_proj`	5.24	+0.02	5.92	+0.10
`q_proj`	5.24	+0.02	5.94	+0.08

C Block Sensitivity Analysis (Qwen2.5-0.5B)

This section presents detailed block sensitivity analysis for the Qwen2.5-0.5B model (24 blocks). Each component shows two figures: (1) raw perplexity values, and (2) percentage change from baseline. Negative percentages indicate improvement (lower PPL), with 0% representing the baseline.

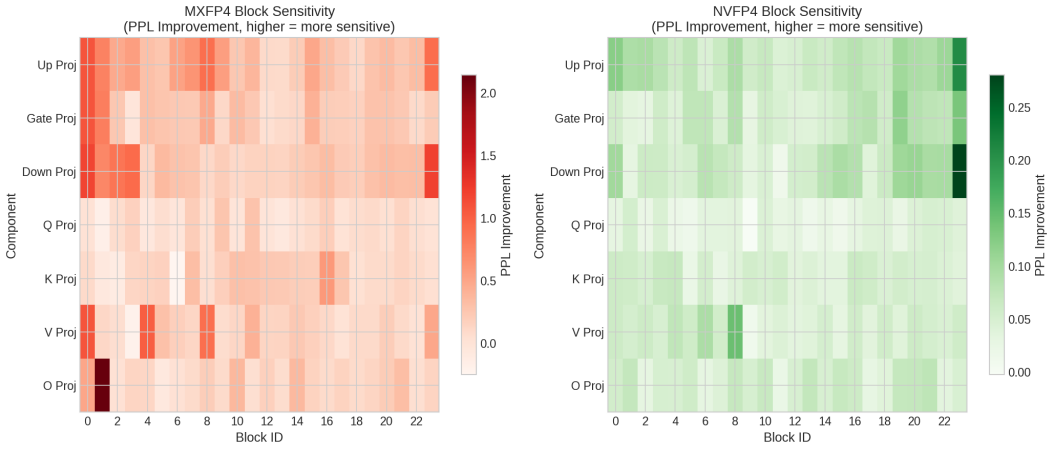


Figure 3: Block sensitivity heatmaps showing PPL improvement when each block's component is kept in FP16. Left: MXFP4 (scale 0–2.0). Right: NVFP4 (scale 0–0.28). Note the 7× scale difference and different spatial patterns.

C.1 MLP Components

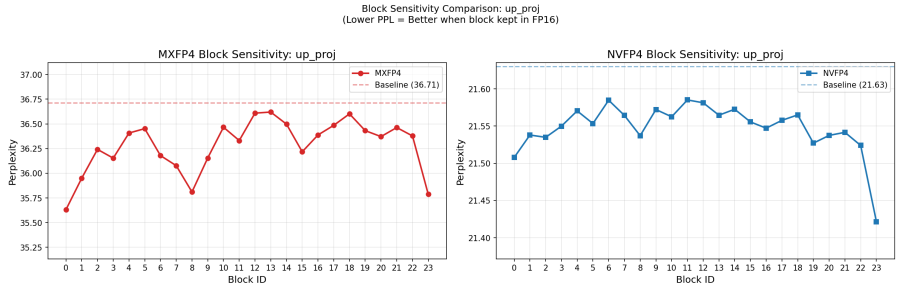


Figure 4: Block sensitivity for `up_proj`. MXFP4 shows strong early-block sensitivity (blocks 0, 8, 23), while NVFP4 peaks at block 23.

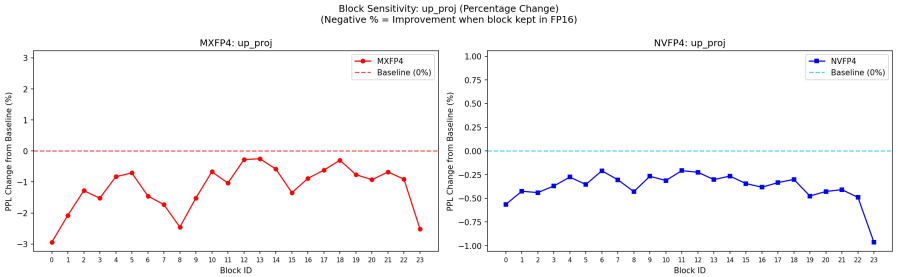


Figure 5: Percentage change from baseline for `up_proj`. Block 23 shows -3.3% (MXFP4) and -1.3% (NVFP4) improvement.

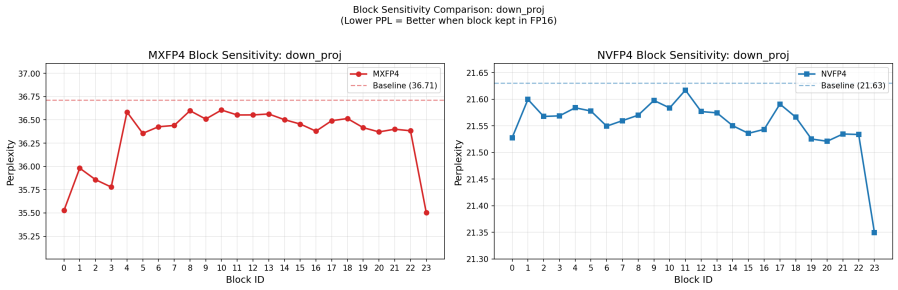


Figure 6: Block sensitivity for `down_proj` (0.5B only). Both formats show block 23 as highly sensitive, but MXFP4 also shows strong sensitivity in blocks 0–3.

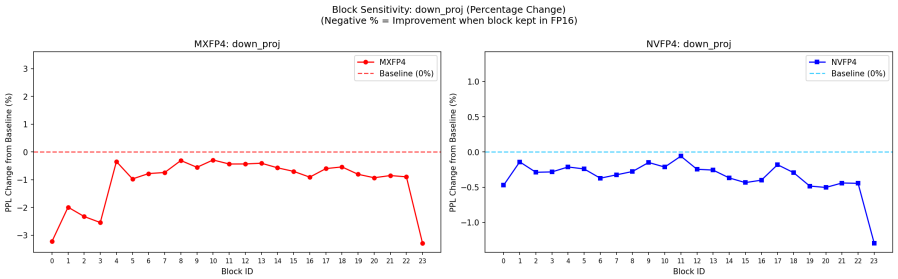


Figure 7: Percentage change from baseline for `down_proj`. Block 23 shows -3.3% (MXFP4) and -1.3% (NVFP4) improvement.

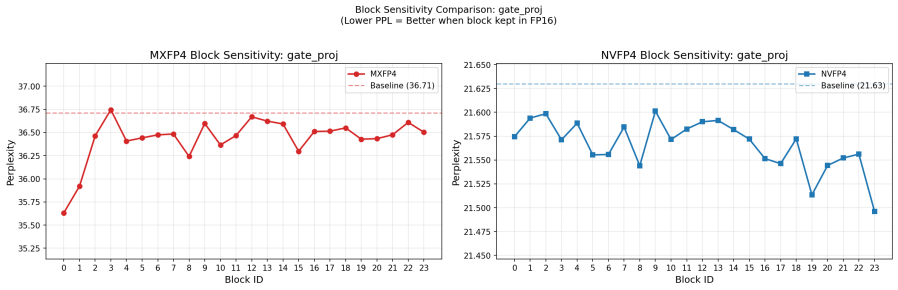


Figure 8: Block sensitivity for `gate_proj`. MXFP4 exhibits pronounced early-block dominance (blocks 0–2), while NVFP4 shows late-block sensitivity peaking at block 23.

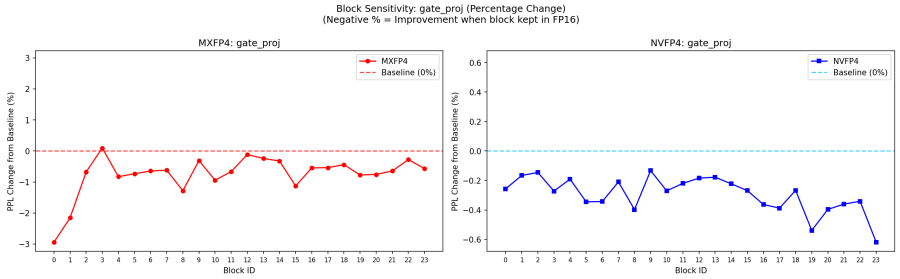


Figure 9: Percentage change from baseline for `gate_proj`.

C.2 Attention Components

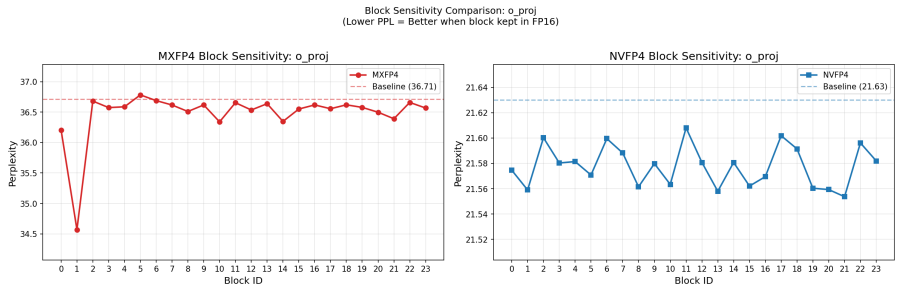


Figure 10: Block sensitivity for `o_proj`. MXFP4 shows extreme sensitivity in block 1 (+2.14 PPL improvement), a unique pattern not seen in other components.

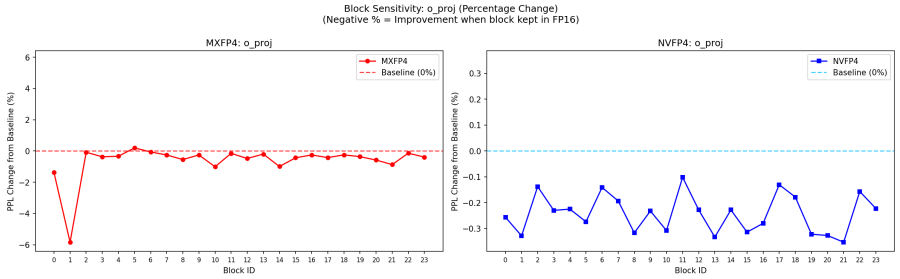


Figure 11: Percentage change from baseline for `o_proj`. Block 1 shows -5.8% improvement for MXFP4.

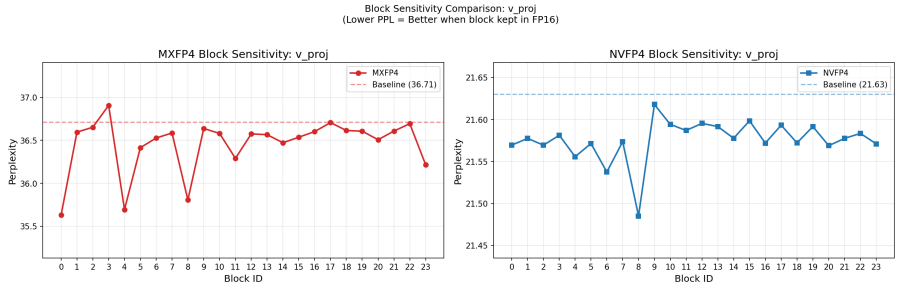


Figure 12: Block sensitivity for `v_proj`. MXFP4 shows early-block dominance, while NVFP4 exhibits relatively flat sensitivity with a slight peak at block 8.

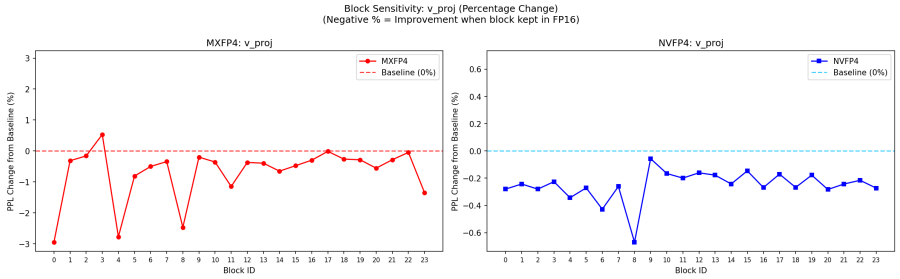


Figure 13: Percentage change from baseline for `v_proj`.

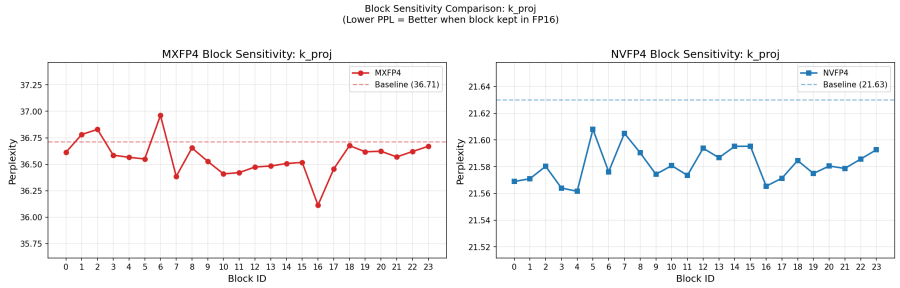


Figure 14: Block sensitivity for `k_proj`. Both formats show relatively low and uniform sensitivity, consistent with `k_proj` being the second-least sensitive component.

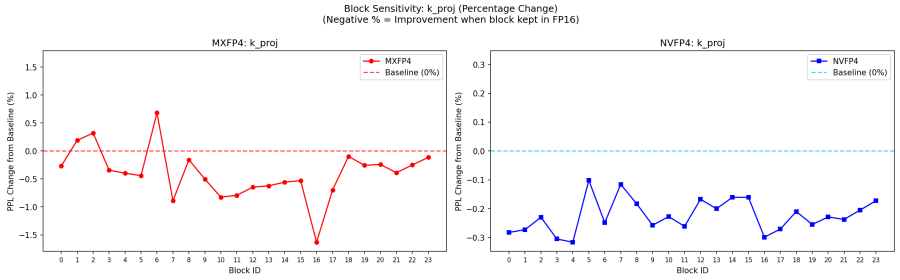


Figure 15: Percentage change from baseline for `k_proj`.

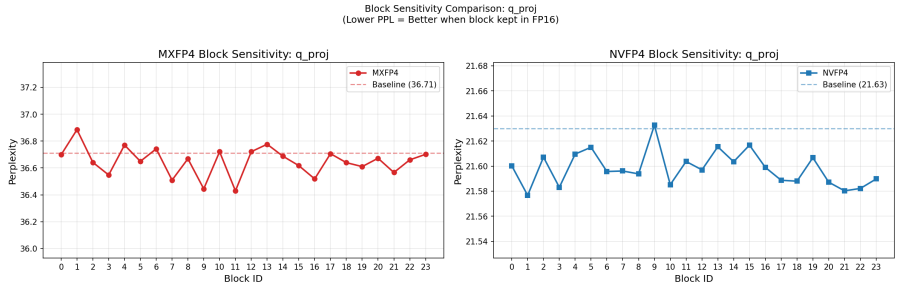


Figure 16: Block sensitivity for `q_proj`. The least sensitive component overall, showing minimal variation across blocks for both formats.

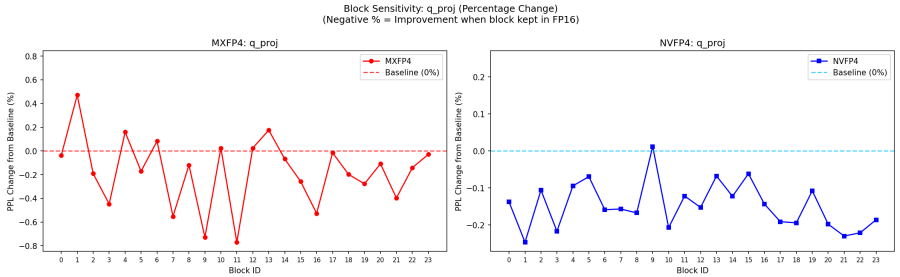


Figure 17: Percentage change from baseline for `q_proj`.

D Block Sensitivity Analysis (Qwen2.5-7B)

This section presents detailed block sensitivity analysis for the Qwen2.5-7B model (28 blocks). Each component shows two figures: (1) raw perplexity values, and (2) percentage change from baseline. Negative percentages indicate improvement (lower PPL), with 0% representing the baseline.

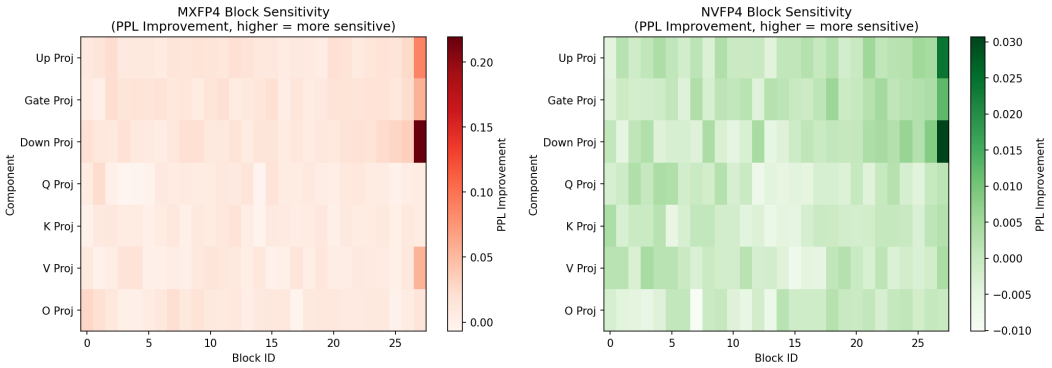


Figure 18: Block sensitivity heatmaps for Qwen2.5-7B showing PPL improvement when each block's component is kept in FP16. Left: MXFP4. Right: NVFP4. Block 27 shows strong sensitivity for MLP components in both formats.

D.1 MLP Components

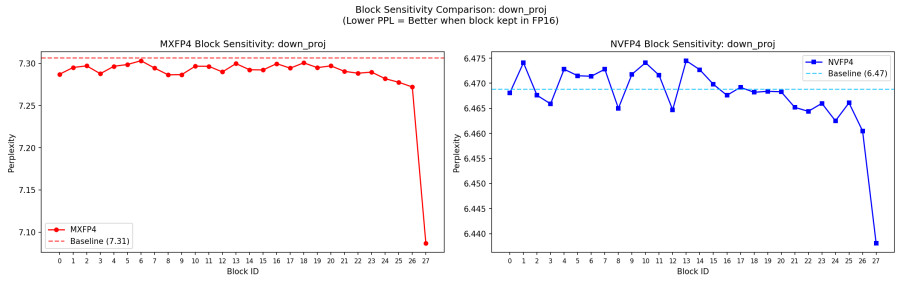


Figure 19: Block sensitivity for `down_proj` (7B). Both formats show block 27 as highly sensitive.

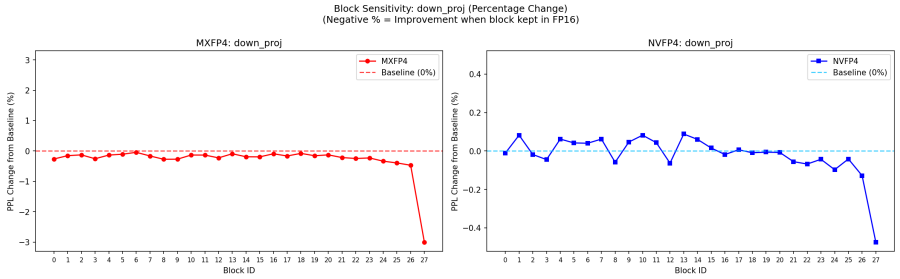


Figure 20: Percentage change from baseline for `down_proj` (7B). Block 27 shows -3.0% (MXFP4) and -0.47% (NVFP4) improvement.

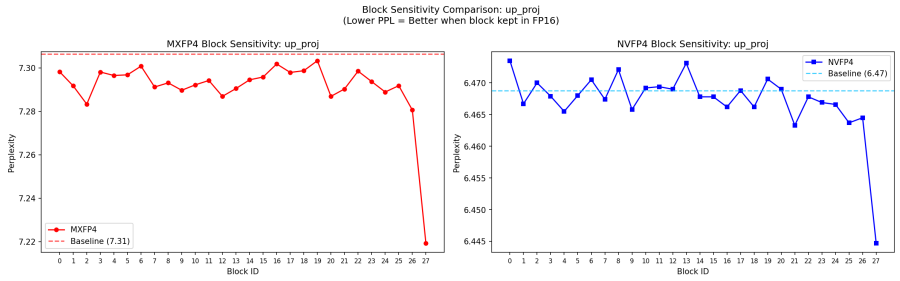


Figure 21: Block sensitivity for `up_proj` (7B). Block 27 is most sensitive for both formats.

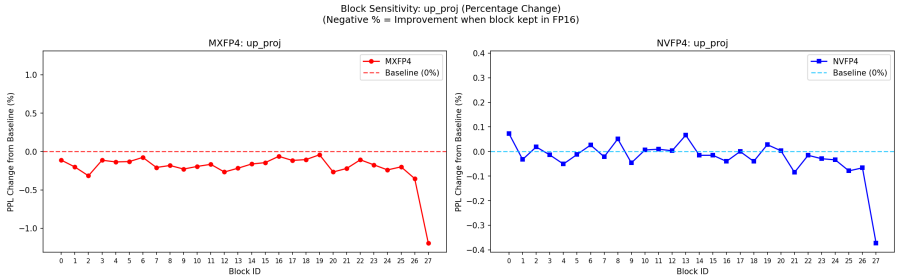


Figure 22: Percentage change from baseline for `up_proj` (7B). Block 27 shows -1.2% (MXFP4) and -0.37% (NVFP4) improvement.

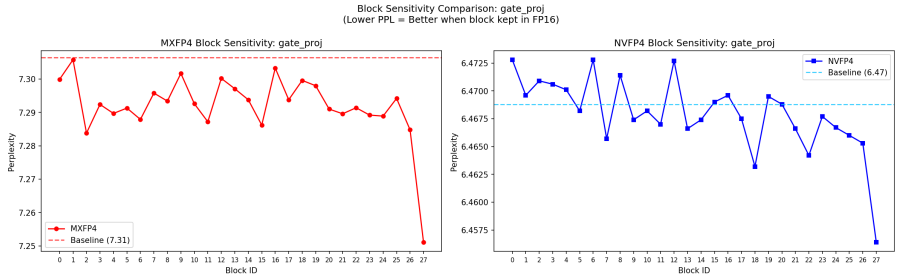


Figure 23: Block sensitivity for `gate_proj` (7B). Block 27 shows highest sensitivity.

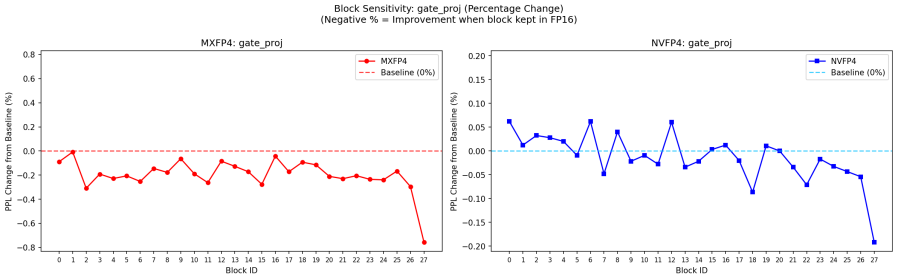


Figure 24: Percentage change from baseline for `gate_proj` (7B). Block 27 shows -0.76% (MXFP4) and -0.19% (NVFP4) improvement.

D.2 Attention Components

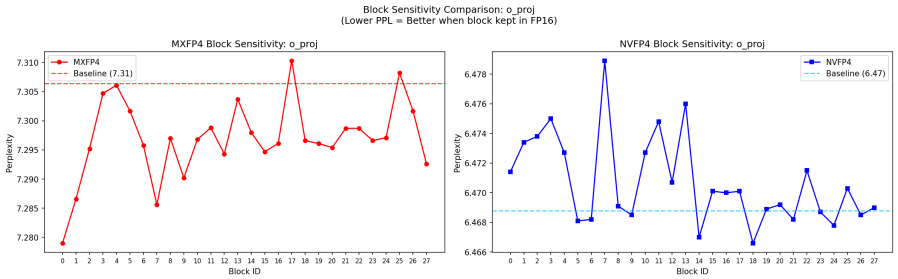


Figure 25: Block sensitivity for `o_proj` (7B). MXFP4 shows early-block sensitivity, while NVFP4 shows relatively flat sensitivity.

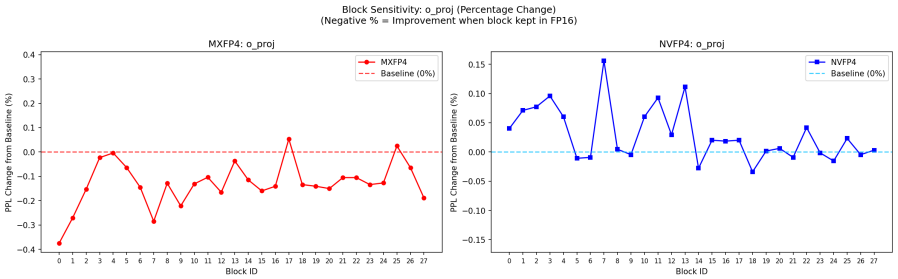


Figure 26: Percentage change from baseline for `o_proj` (7B). MXFP4 shows $\sim$0.3% variation, while NVFP4 shows $<$0.05%.

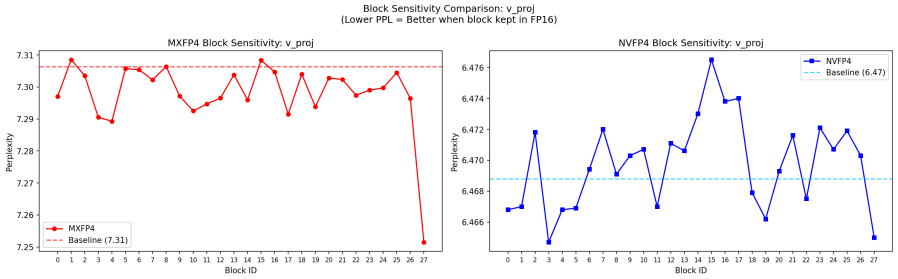


Figure 27: Block sensitivity for `v_proj` (7B). MXFP4 shows block 27 as most sensitive, while NVFP4 exhibits relatively flat sensitivity.

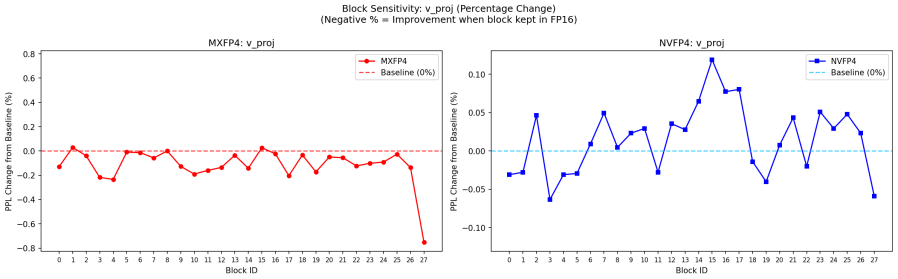


Figure 28: Percentage change from baseline for `v_proj` (7B).

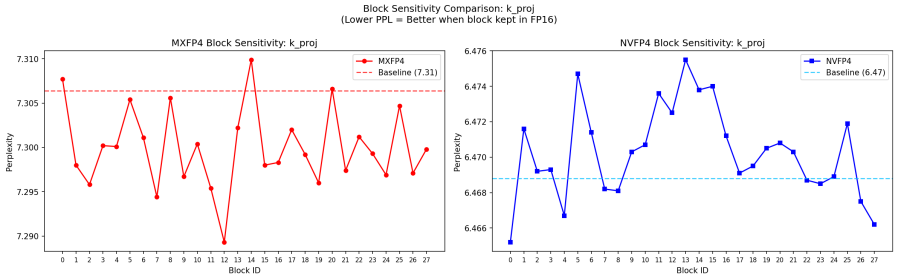


Figure 29: Block sensitivity for `k_proj` (7B). Both formats show relatively low and uniform sensitivity across blocks.

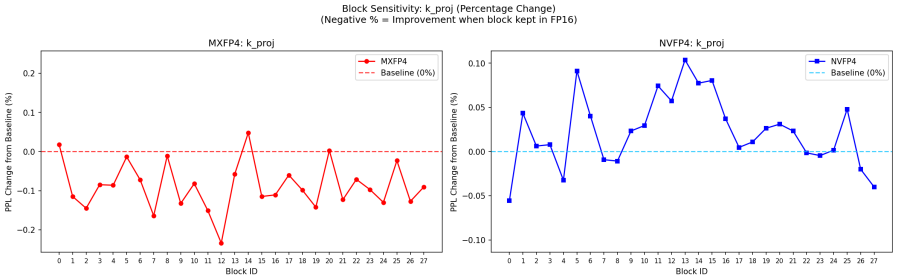


Figure 30: Percentage change from baseline for `k_proj` (7B).

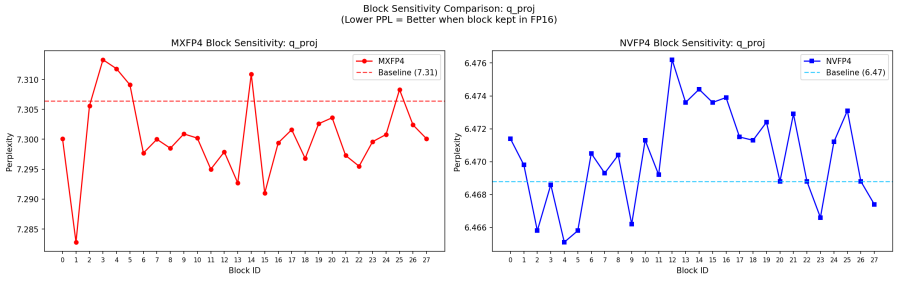


Figure 31: Block sensitivity for `q_proj` (7B). MXFP4 shows block 1 as most sensitive, while NVFP4 shows minimal variation across blocks.

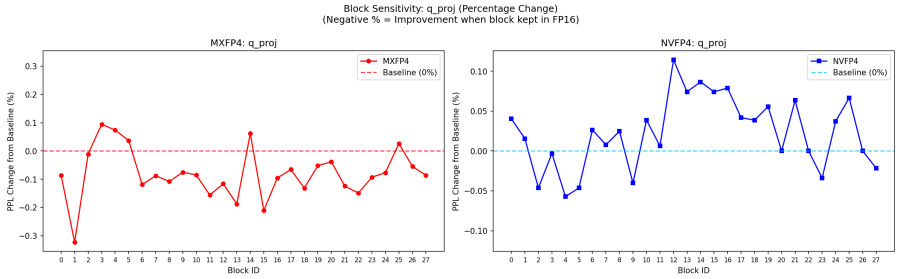


Figure 32: Percentage change from baseline for `q_proj` (7B).