Activation-Aware BitDelta

Michael Peng*, Emily Zhou*, Shrika Eddula*, Jessie Lee*, Sophia Chen*
*Equal contribution

Introduction

Large language models have revolutionized natural language processing across academia, industry, and real-world applications.

We aim to achieve the following:

- Comprehensive evaluation of activation-aware training on perplexity and inference latency
- Evaluation of the impact of activation-aware techniques on different architectures under diverse domain constraints
- Analysis of memory-efficient training strategies to facilitate large-scale LLM training

Related Works

Model Efficiency

- LlaMa series [1] introduced a group of foundational models optimized for efficiency and open accessibility
  - Inspiration for numerous derivatives such as Vicuna [2] and Mistral [3]
- Quantization [4] and gradient checkpointing [5] techniques have reduced memory requirements
  - Allows deployment of LLMs in resource-constrained environments

Model Calibration

- Calibration has been shown to be important for machine learning
  - Techniques such as temperature scaling are amongst the many calibration techniques [6]
- Activation-aware training improves calibration without sacrificing the model's accuracy or efficiency.

Domain-Specific Fine-Tuning

- Fine-tuning LLMs for specific domains has proven to be highly effective for tasks such as medical question answering [7] and financial analysis [8]
  - Instruction tuning and few-shot learning have greatly enhanced domain adaptation [9][10]

References

[1] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[2] Vicuna Team. Vicuna: An open-source chatbot based on llama models. *GitHub Repository*, 2023. Available at https://github.com/lm-sys/Vicuna.

[3] Mistral AI. Mistral 7b: A next-generation dense language model. *arXiv preprint arXiv:2310.06801*, 2023.

[4] Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022

[5] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. In *Advances in Neural Information Processing Systems*, 2016.

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

[7] Yan Li et al. Chatgpt in medicine: The next step in healthcare transformation. *Journal of Medical Internet Research*, 2023.

[8] Shawn Wu et al. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2304.03279*, 2023.

[9] Long Ouyang, Jeffrey Wu, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

[10] Jason Wei et al. Finetuned language models are few-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Methodologies

Model Selection

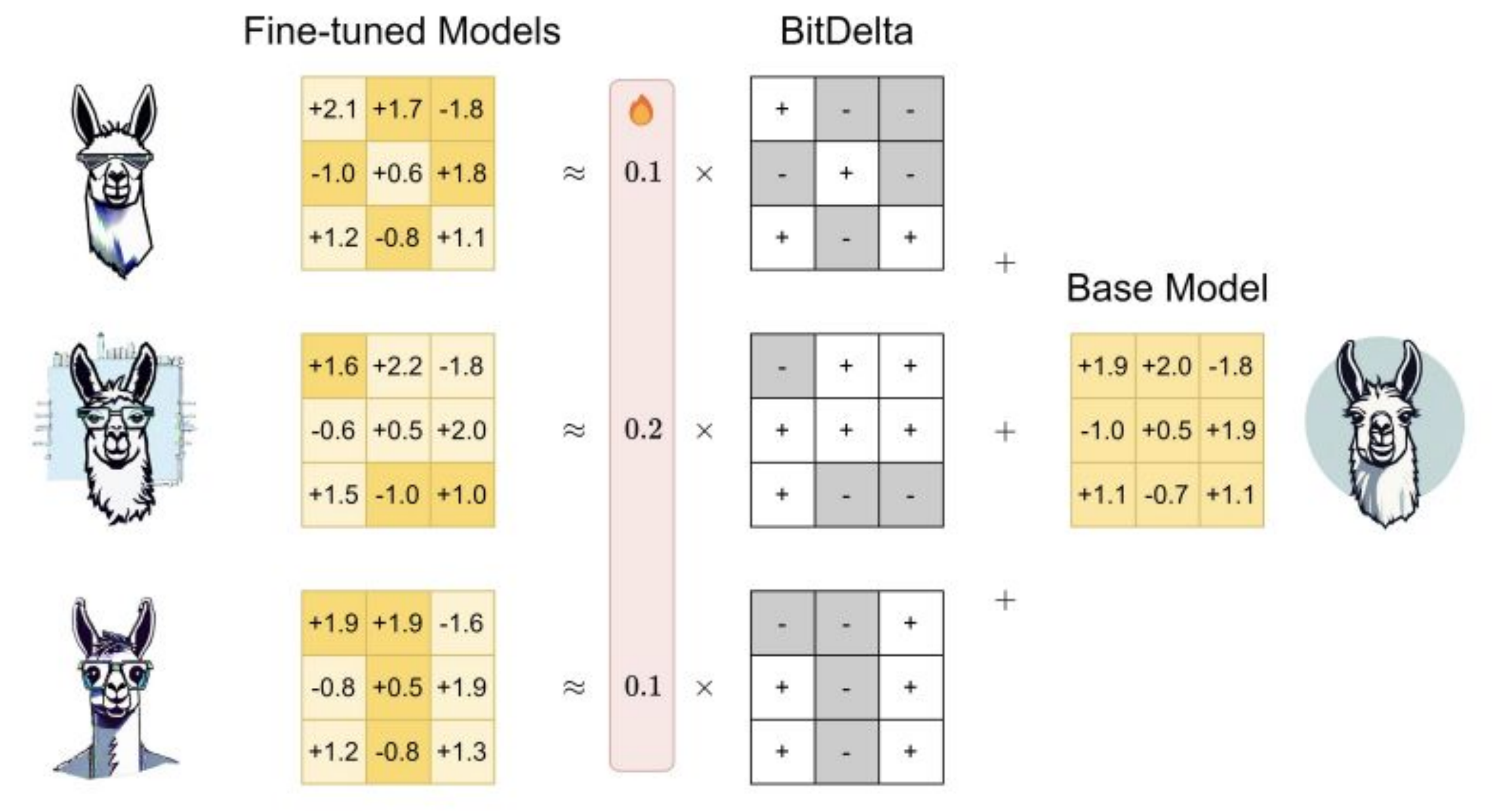
- Mistral Pairs: Base model *Mistral-7B-v0.1* paired with its fine-tuned variants *Mistral-7B-Instruct-v0.1* and *Zephyr-7B-beta.*
- LLaMA-2 Pairs: Base model *LLaMA-2-7B-hf* paired with *Vicuna-7B-v1.5* and *LLaMA-2-7B-chat-hf.*

Training Configuration

- **Baseline Training:** Conventional training without activation awareness, serving as the baseline.
- **Activation-Aware Training**: incorporates activation-aware loss functions and calibration specific adjustments. This configuration leverages activation statistics during training to enhance model calibration.

Evaluation Metrics

- Perplexity, Wall clock time(s)



$$\#\text{params} \times \#\text{models} \times 16\text{bits} \quad \Rightarrow \quad \#\text{params} \times (\#\text{models} \times 1\text{bit} + 16\text{bits})$$

Experiments

Datasets

Evaluations are performed on a set of benchmark datasets representing diverse domains:

- *FremyCompany/AGCT-Dataset* (Medical Domain)
- *bigcode/the_stack* (Code Domain)
- *atrost/financial_phrasebank* (Financial Domain)
- *wikitext-2-raw-v1* (General Domain)

Experimental Setup

- All experiments are conducted on 2 NVIDIA A100 GPUs with 40GB
- Memory allocation is optimized using PyTorch's *max_split_size_mb* configuration, and training scripts are executed with transformers

Analysis Pipeline

- Comparing perplexity scores between standard and activation-aware training for each domain
- Measuring latency improvements and memory usage for activation-aware models
- Visualizing results to highlight trade-offs between computational efficiency and accuracy

Results

Figure 1 compares end-to-end training times across batch sizes for standard and activation-aware training. Activation-aware training is batch-size-independent, showcasing its efficiency by leveraging activation statistics without additional training.

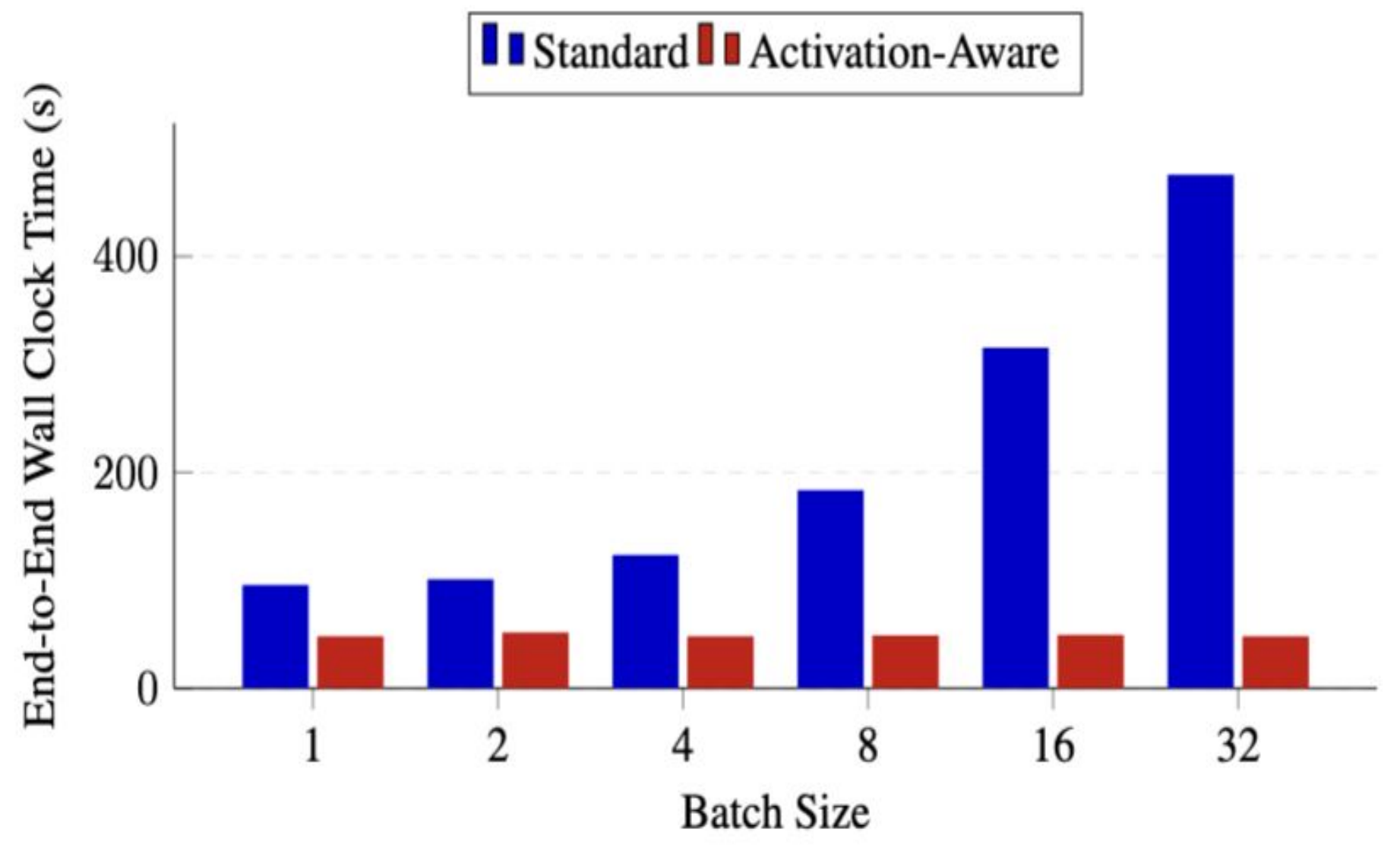


Figure 1: End-to-end wall clock time comparison between standard and activation-aware approaches (using train_examples=250)

Figures 2–5 show consistent perplexity improvements with activation-aware models across code, health, finance, and wikitext domains. The largest gains appear in structured domains like finance and code, with robust performance across all calibration sizes.

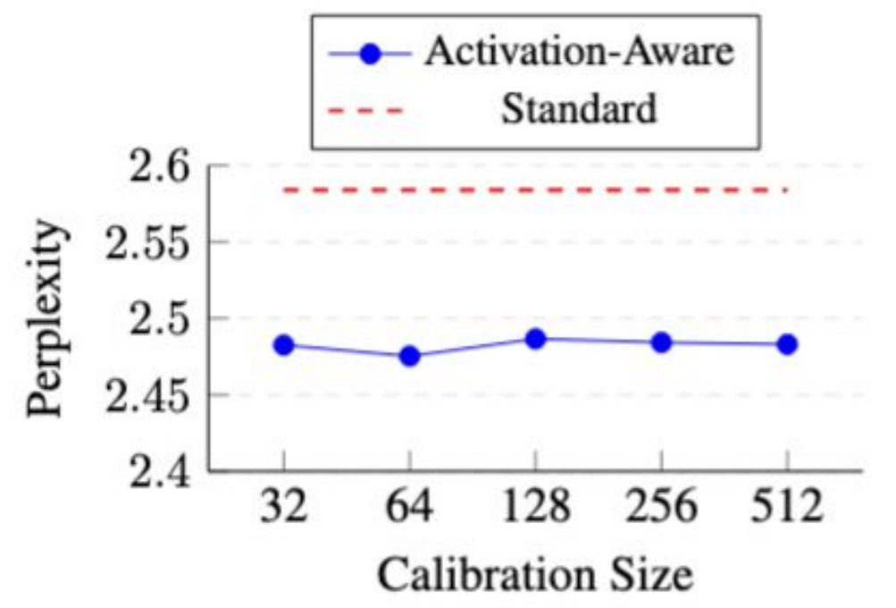


Figure 2: Perplexity comparison for `stack` domain.

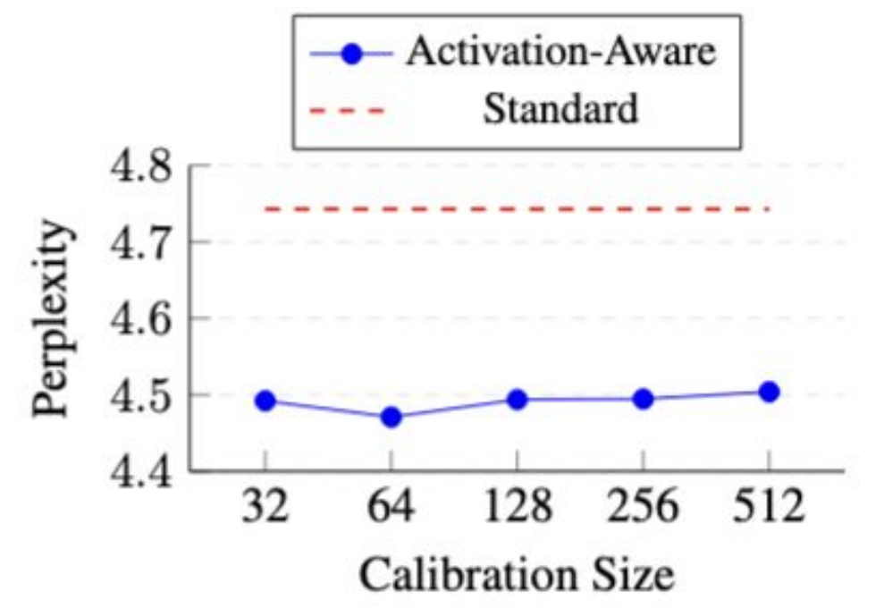


Figure 3: Perplexity comparison for `health` domain.

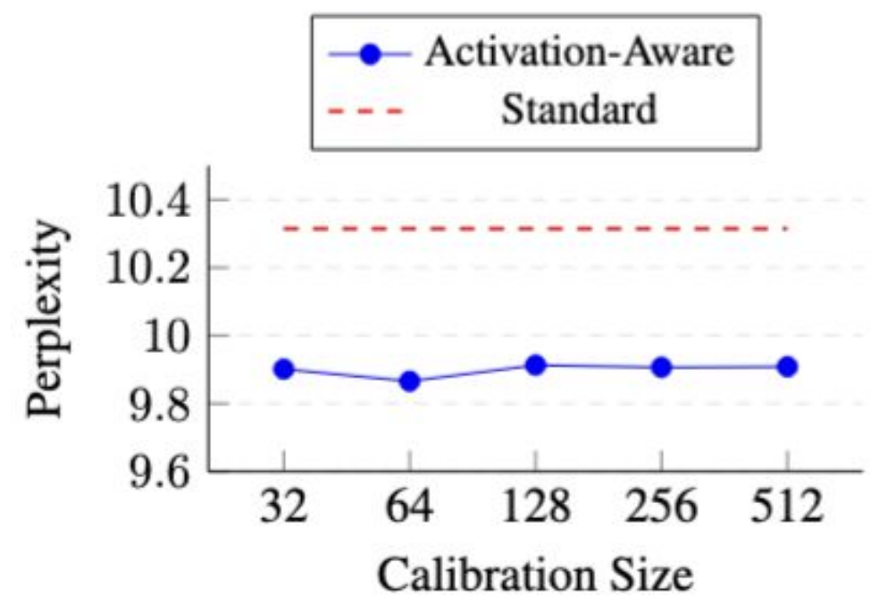


Figure 4: Perplexity comparison for `finance` domain.

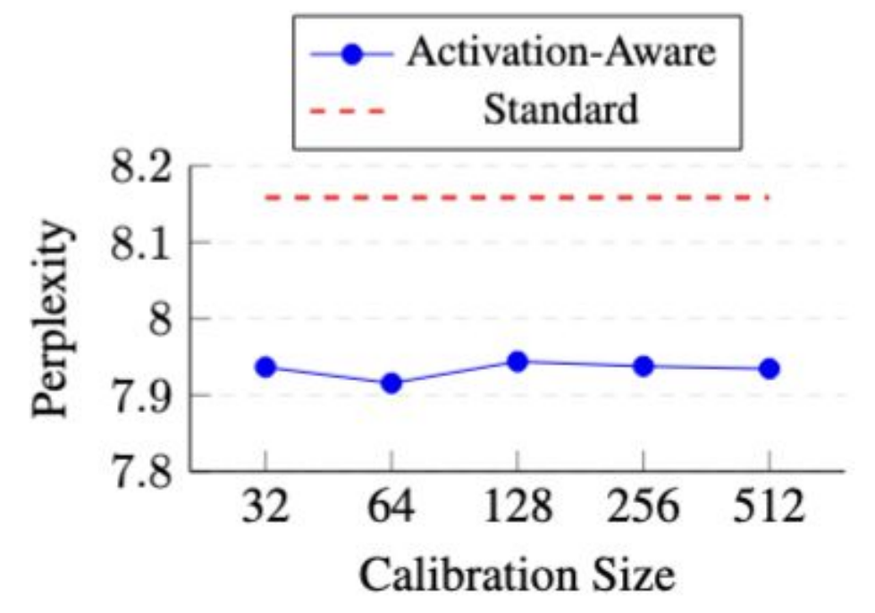


Figure 5: Perplexity comparison for `wikitext` domain.

Model Pair	Activation-Aware	Standard
Mistral-7B-v0.1 / Zephyr-7B-beta	8.878828	7.790850
Mistral-7B-v0.1 / Mistral-7B-Instruct-v0.1	7.864079	8.368696
Llama-2-7B-hf / Vicuna-7B-v1.5	7.581587	7.841721
Llama-2-7B-hf / Llama-2-7B-chat-hf	7.937717	8.158337

Table 1: Perplexity comparison for activation-aware and standard configurations.

Model Pair	Activation-Aware	Standard
Mistral-7B-v0.1 / Zephyr-7B-beta	77.36	150.73
Mistral-7B-v0.1 / Mistral-7B-Instruct-v0.1	92.04	143.06
Llama-2-7B-hf / Vicuna-7B-v1.5	77.32	141.51
Llama-2-7B-hf / Llama-2-7B-chat-hf	67.56	146.27

Table 2: Wall clock time comparison (in seconds) for activation-aware and standard configurations.

- Activation-aware methods consistently achieve lower perplexity and significantly lowered latency: Mistral-7B-v0.1 / Zephyr-7B-beta pair saw over 50% latency reduction with superior perplexity.

Future Work

- Validate scalability of activation-aware techniques by extending to even larger model architectures, e.g. Mistral-Large or LLaMA-3 families
- Incorporating dynamic calibration techniques that adapt activation-aware scales during inference