Graph Signal Processing Meets Mamba2: Adaptive Filter Bank via Delta Modulation

Yehjin Shin [*1] Seojin Kim [*1] Noseong Park [1]

Abstract

State-space models (SSMs) provide efficient alternatives to attention with linear-time recurrence. Mamba2, a recent SSM-based language model, employs selective input gating and a multi-head structure for parallel computation and strong performance. However, its multi-head recurrence operates independently without structured utilization or analysis. In this work, we introduce **H**ierarchical **AD**aptive filter bank for **E**fficient **S**SMs (*HADES*), a GSP-inspired framework that reinterprets Mamba2 as an adaptive filter bank on a line graph. *HADES* features two filter types: shared filters for global low-pass behavior and expert filters for local high-pass behavior, achieved via structured bias on parameter Δ. *HADES* matches baseline models, including Mamba2, on key benchmark tasks while using only **58.9%** of the original parameters.

1. Introduction

Transformer architectures dominate sequence modeling in tasks like text generation and machine translation, but their quadratic complexity has driven the search for more efficient alternatives (Gu et al., 2022; Yang et al., 2024b; Peng et al., 2023; Sun et al., 2024). Mamba (Gu & Dao, 2023) and Mamba2 (Dao & Gu, 2024) have shown that continuous-time SSMs can match or exceed transformer performance.

Despite this success, Mamba2's internal structure, particularly its multi-head recurrence, remains under-explored. Prior studies have focused on enhancing long-context performance through delta modulation (Ben-Kish et al., 2025; Azizi et al., 2025; Ye et al., 2025). Another study (Wang et al., 2025) identified issues like recency bias and information bottlenecks in SSMs and proposed polarization as a solution. However, it remains unclear how individual heads contribute to the model's representation or if they exhibit complementary dynamics.

To address this issue, we reinterpret Mamba2 within the framework of Graph Signal Processing (GSP). Specifically, we model the input sequence as a signal on a line graph, where tokens serve as nodes and their temporal connections form edges. In this view, each recurrent head in Mamba2 functions as a graph filter applied to this signal. This perspective naturally leads to a filter bank interpretation, where individual heads can be understood as specialized filters, each capturing distinct spectral characteristics of the input.

Building on this formulation, we further propose a hierarchical filter bank model architecture, *HADES*, which allows adaptive and efficient information flow. *HADES* organizes filters into two functional categories: (1) *shared filters*, which perform globally consistent filtering across the sequence, and (2) *expert filters*, which adapt their filtering behavior on a per-token basis.

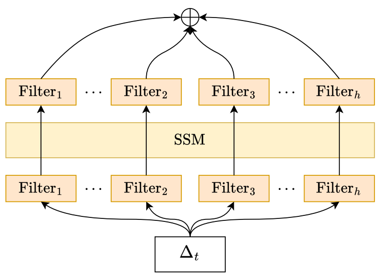
HADES demonstrates competitive performance across a diverse set of benchmarks, including eight zero-shot commonsense reasoning tasks, two language modeling tasks, and long-context retrieval, while utilizing only 58.9% of the parameters compared to Mamba2. For interpretability, we further examine *HADES* through spectrum analysis, demonstrating how our filter bank approach affects the model's internal dynamics. By integrating principles from GSP into sequence modeling, our method offers a scalable, hierarchical, and transparent filtering mechanism within SSMs.

2. Background

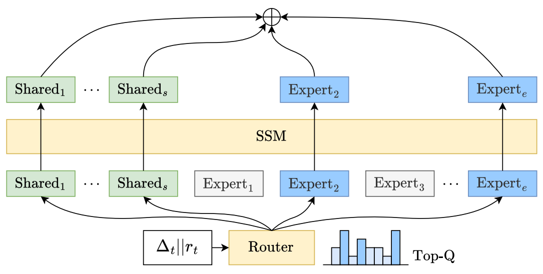
Our method, *HADES*, is based on a reinterpretation of structured sequence models from the perspective of Graph Signal Processing (GSP). We view the multi-head state-space model (SSM) as a learnable graph filter bank, where each head captures distinct frequency-selective dynamics. This section outlines the necessary background to support this perspective. Basic concepts and terminology for SSMs and GSP are in Appendices A and B, forming the basis of the notation adopted in this paper.

[*]Equal contribution [1]KAIST, South Korea. Correspondence to: Noseong Park <noseong@kaist.ac.kr>.

The 3^{rd} Workshop on Efficient Systems for Foundation Models at the 41^{st} International Conference on Machine Learning, Vancouver, Canada. Copyright 2025 by the author(s).



(a) Mamba2



(b) *HADES*

Figure 1: Architectural Comparison between Mamba2 and *HADES*. Mamba2 applies all filters uniformly to every input token, whereas *HADES* employs a routing mechanism that selects and activates filters conditioned on r_t and Δ_t.

SSMs as Graph Filters A one-dimensional token sequence can be naturally represented as a signal defined on a line graph (i.e., a linearly connected graph), where each token corresponds to a node and edges connect adjacent tokens in the sequence. This perspective enables the application of GSP tools to sequential data. In particular, the line graph admits a natural notion of convolution, where filtering operations over token sequences can be interpreted as graph convolutions. This provides a principled foundation for analyzing state-space models from a GSP perspective. Specifically, the S4 model (cf. Eq. 10) can be viewed as a LTI system operating on a line graph, where its kernel acts as the convolutional filter. This interpretation allows the SSM to be expressed as a graph convolution over the input sequence, offering a unified framework that bridges sequence modeling and GSP framework in Eq. 1:

$$\mathbf{y} = \mathbf{x} * \mathbf{K} = \sum_{k=0}^{K} \underbrace{(\mathbf{C}\mathbf{A}^k\mathbf{B})}_{h_k} \mathbf{S}^k\mathbf{x} \tag{1}$$

In contrast, Mamba (cf. Eq. 11) can be interpreted as a linear time-varying (LTV) system operating on a line graph. Unlike an LTI system, which applies the same filter across all nodes, Mamba applies distinct, input-dependent filters at each node, enabling more flexible and adaptive sequence modeling. This formulation can be written as:

$$\mathbf{y}_t = \sum_{k=0}^{K} \underbrace{(\mathbf{C}_t\mathbf{A}_{t:t-k}\mathbf{B}_{t-k})}_{h_k^{(t)}} \mathbf{S}^k\mathbf{x}, \tag{2}$$

where $\mathbf{A}_{t:t-k} = \prod_{t-k}^{t} \mathbf{A}_i$ means cumulative product of A_t from shift start index for k hops.

Multi-Head SSMs as Filter Banks Mamba2 employs multiple parameterized state-space recurrences, one per head, formulated as:

$$h_t^{(i)} = \mathbf{A}_t^{(i)} h_{t-1}^{(i)} + \mathbf{B}_t^{(i)} x_t, \quad y_t^{(i)} = \mathbf{C}_t^{(i)} h_t^{(i)}, \tag{3}$$

where $i \in [M]$ indexes the heads. This structure can be interpreted as a filter bank, with each head i acting as a distinct filter applied to the input signal x_t.

$$\begin{aligned} \mathbf{y}_t^{(i)} &= \mathbf{\Phi}\left(\left\{\mathbf{y}_t^{(i)}\right\}_{i=1}^{M}\right) \\ &= \mathbf{\Phi}\left(\left\{\sum_{k=0}^{K}(\mathbf{C}_t^{(i)}\mathbf{A}_{t:t-k}^{(i)}\mathbf{B}_{t-k}^{(i)})\mathbf{S}^k\mathbf{x}\right\}_{i=1}^{M}\right), \end{aligned} \tag{4}$$

where $\mathbf{A}_i^{(j)} \in \mathbb{R}^{N\times N}$, $\mathbf{B}_i^{(j)} \in \mathbb{R}^{N\times 1}$, $\mathbf{C}_i^{(j)} \in \mathbb{R}^{1\times N}$ are parameters of SSM equations and M denotes the number of filters. Likewise, we can interpret multi-head architectures into graph filter bank. Detailed parameterizations of Mamba2 are deferred to Appendix A.

Although Mamba2's head-specific recurrence parameters and time-varying coefficients enable diverse temporal and spectral responses, it lacks explicit structural constraints or functional differentiation among heads. This leads to an unstructured, static filter bank that may struggle to capture both global and local dynamics. To overcome this, we propose a structured, adaptive filter bank design that promotes functional diversity across heads, enhancing the model's ability to capture both global and local sequence patterns.

3. Proposed Method

3.1. *HADES*: Hierarchical ADaptive filter bank for Effieicnt SSMs

From the perspective of node-adaptive filtering, a key challenge lies in how to effectively select and combine diverse filters. To enhance the structural expressivity of Mamba2 without compromising its efficiency, we propose *HADES*, an adaptive filter bank architecture based on GSP principles. Our approach decomposes the multi-head structure into two complementary components: shared filters and expert filters. A router is employed to select the Top-Q expert filters, where the expert scores are computed based on the spectral residual and the characteristics of the input sequence.

Fig. 1 illustrates how our method functions as a filter bank. Fig. 1(a) shows the general Mamba2 architecture, where all filters are always utilized regardless of context. In contrast, the proposed method in Fig. 1(b), selects a subset of filters to be used at each timestep t. Among them, the shared filters are always applied, independent of the router's selection. In our method, from M filters of the general Mamba2 architecture, H filters are selected at each timestep. These H filters are composed of S shared filters and E expert filters, which are dynamically chosen based on the routing mechanism. The final output is computed as a weighted linear combination of the selected filter outputs.

Expert Filters To enable token-level adaptivity, we introduce a router that assigns a subset of expert filters to each token based on its frequency characteristics. Specifically, for each token at time step t, we compute the spectral residual as $r_t = x_t - \mu_t$, where μ_t is a running mean across the sequence. The base delta parameter $\Delta_{t,\text{base}}$ is concatenated with the residual r_t, and the resulting vector is passed through a linear projection to compute selection scores s_t for the expert filters:

$$s_t = f_e([\Delta_{t,\text{base}} \,\|\, r_t]), \quad r_t = x_t - \mu_t, \tag{5}$$

where f_e is a function that computes expert selection scores based on both the base $\Delta_{t,\text{base}}$ and the token's spectral residual r_t, and $[\cdot \,\|\, \cdot]$ denotes vector concatenation. The resulting score vector $s_t \in \mathbb{R}^E$ contains a scalar score for each of the E expert filters. The Top-Q filters with the highest scores are then selected and applied to the token. While expert filters are not explicitly assigned to specific frequency bands, their distinct Δ configurations induce varied update dynamics, implicitly shaping their responses based on the token's frequency characteristics.

The residual r_t is used not only for expert selection, but also for modulating the delta value itself. Specifically, it introduces a frequency-sensitive bias that adjusts Δ in accordance with token-level frequency characteristics, spectral bias:

$$\Delta_{t,\mathit{HADES}} = \text{Softplus}(\Delta_{t,\text{base}} + \gamma \cdot f_b([\Delta_{t,\text{base}} \,\|\, r_t])) \tag{6}$$

where f_b is a function that generates a content-aware adjustment to $\Delta_{t,\text{base}}$ based on the token's residual r_t, and γ is a scaling hyperparameter that controls the strength of residual-based modulation. We use a single-layer linear projection for f_e and f_b in our implementation.

Shared Filters Shared filters are always applied without additional bias, relying solely on the base $\Delta_{t,\text{base}}$ and targeting globally smooth components across the sequence. Unlike expert filters, they don't incorporate per-token modulation, naturally preserving low-frequency patterns while attenuating high-frequency variations. This design, akin to fixed low-pass filters in GCN models (Dong et al., 2021) and structure-preserving averaging (Wu et al., 2022), ensures spectral stability and mitigates over-adaptation.

3.2. Training Loss Terms

To ensure effective learning of the adaptive filter bank, it is crucial that the model utilizes a diverse set of filters rather than overfitting to a subset. We introduce a dual loss mechanism that encourages balanced filter utilization during training. The final training objective combines two auxiliary components:

$$\mathcal{L} = \mathcal{L}_\text{task} + \lambda_1 \cdot \underbrace{\frac{\text{Var}(s_t)}{(\mathbb{E}[s_t])^2 + \epsilon}}_{\mathcal{L}_\text{balance}} + \lambda_2 \cdot \underbrace{\mathbb{E}_{i,j}\left[(\langle \hat{\mathbf{y}}_i, \hat{\mathbf{y}}_j \rangle - \delta_{ij})^2\right]}_{\mathcal{L}_\text{diversity}}, \tag{7}$$

where $\mathcal{L}_\text{task}$ is the primary task loss (cross-entropy loss for language modeling), $s_t = f_e([\Delta_{t,\text{base}} \,\|\, r_t])$ is the vector of selection scores for the E experts at time step t, and ϵ is a small constant for numerical stability. $\hat{\mathbf{y}}_i$ denotes the ℓ_2-normalized output of the i-th expert filter, $\delta_{ij} = 1$ if $i = j$, 0 otherwise, and λ_1, λ_2 are hyperparameters controlling the strength of the selection and diversity losses respectively. $\mathcal{L}_\text{balance}$ ensures balanced utilization of all expert filters by penalizing high variance in selection scores across experts. Specifically, it minimizes the squared coefficient of variation of the selection scores, encouraging uniform distribution among experts. $\mathcal{L}_\text{diversity}$ promotes functional diversity among the expert filters by minimizing the similarity between their normalized outputs. This dual loss mechanism ensures that the model effectively learns a diverse set of filters, each specialized for different aspects of the input sequence.

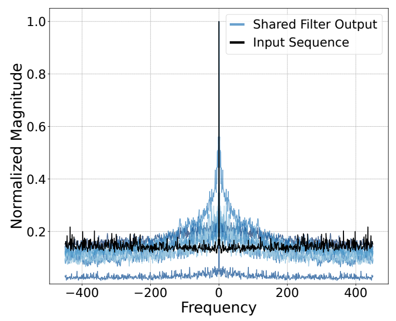
4. Empirical Studies

Setup Our experiments encompass a comprehensive comparison of recent state-of-the-art architectures. We evaluate against the following baselines: Linear Transformer (Katharopoulos et al., 2020), RetNet (Sun et al., 2024), Mamba (Gu & Dao, 2023), Mamba2 (Dao & Gu, 2024), and DeltaNet (Yang et al., 2024b). We trained all models using approximately 200B tokens from the Pile dataset (Gao et al., 2020). Detailed settings for training and evaluation are deferred to Appendix C.

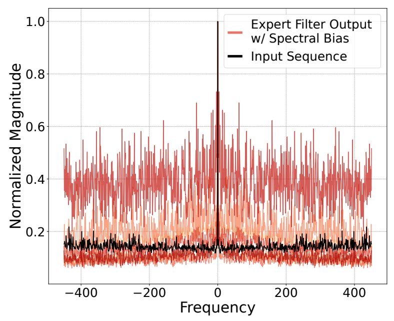
Language Modeling and Commonsense Reasoning In Table 1, we present the performance of each model across multiple benchmarks, including language modeling perplexity and zero-shot accuracy on commonsense reasoning benchmarks for models with 370M parameters. Even with 58.92% of parameters (218M), *HADES* demonstrates comparable performance to baselines including Linear Transformer, RetNet, Mamba, Mamba2, and DeltaNet.

Table 1: Performance comparison on language modeling and zero-shot common-sense reasoning. The best results are in **bold**, and the second-best results are underlined. Avg. denotes the average of (normalized) accuracies over 8 tasks.

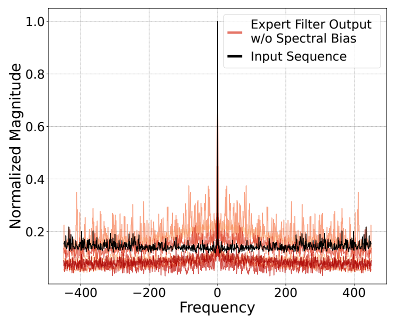
Model	Wiki. ppl ↓	LMB. ppl ↓	LMB. acc ↑	BoolQ acc ↑	Hella. acc_n ↑	Wino. acc ↑	ARC-e acc ↑	ARC-c acc ↑	PIQA acc ↑	OBQA. acc_n ↑	Avg. 8 tasks ↑
Linear Transformer	45.43	73.93	24.06	61.50	28.20	51.30	42.05	21.76	60.55	27.60	39.63
RetNet	34.12	29.46	35.36	55.57	31.31	51.70	44.49	23.46	62.40	28.00	41.54
DeltaNet	33.25	26.82	35.75	54.07	31.40	49.96	44.11	22.18	63.60	**29.60**	41.96
Mamba1	47.51	85.53	22.43	**62.17**	28.71	50.67	42.09	22.35	60.72	26.60	39.73
Mamba2	**31.34**	24.38	36.46	53.88	32.62	50.83	**45.29**	**24.15**	63.44	26.40	41.63
HADES (Ours)	31.48	**21.74**	**39.24**	58.84	**32.82**	**52.64**	45.03	22.01	**63.93**	28.80	**42.91**



(a) Shared Filter Outputs

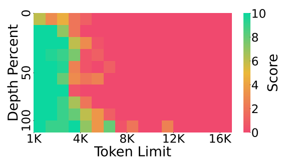


(b) Expert Filter Outputs with Spectral Bias

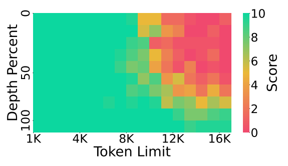


(c) Expert Filter Outputs without Spectral Bias

Figure 2: Frequency spectrum analysis of filter outputs from 13th layer.



(a) Mamba2



(b) *HADES*

Figure 3: Passkey retrieval result of Mamba2 and *HADES*

Long Range Retrieval To evaluate the long-range memory capacity, we adopt the passkey retrieval task, where a key-value pair is planted at various depth in a long sequence and queried at the end. In Fig. 3, the results show that our model significantly outperforms Mamba2, demonstrating the effectiveness of our GSP-inspired adaptive filtering in retaining distant dependencies.

Filter Spectrum Analysis We analyzed the spectral characteristics of our model's filters using Fourier transform on layer 13 outputs from a sample sentence in the Pile dataset (Fig.2). Shared filters (Fig.2(a)) exhibited consistent low-pass behavior, emphasizing low-frequency components, which aligns with their role in capturing global information. In contrast, expert filters (Fig. 2(b)) showed a mix of low- and high-frequency responses, reflecting their adaptive specialization for localized details. This distinction confirms our model's design: shared filters provide a stable context, while expert filters adapt to fine-grained variations.

Effect of Spectral Bias We analyzed the impact of spectral bias on expert filters. Fig. 2(c) shows that without spectral bias (Δ_t), the filter outputs primarily capture low-frequency information. In contrast, Fig. 2(b) reveals that applying spectral bias ($\Delta_{t,HADES}$) shifts the frequency distribution upward, enhancing the model's sensitivity to high-frequency details. This effect is further quantified in Fig. 4 (cf. Appendix E), where the log-scale histogram of the difference between $\Delta_{t,\mathrm{HADES}}$ and Δ_t across 38,000 tokens shows predominantly positive values, indicating increased delta values for high-frequency capture. Occasionally, negative values reduce delta, preserving global context. This aligns with filter response in Fig. 2(b), where high frequency behavior is emphasized. This adaptive mechanism allows *HADES* to flexibly balance local and global information.

5. Conclusion

In this work, we present *HADES*, a hierarchical adaptive filtering architecture for SSMs that bridges SSMs and GSP. By reinterpreting Mamba2 as a GSP-inspired filter bank, we separate shared and expert filters using delta modulation and spectral residual bias. This design enables efficient, frequency-adaptive filtering, achieving strong performance in language modeling, commonsense reasoning, and long-context tasks while using only 58.9% of the parameters of baseline models like Mamba2.

Acknowledgments

This work was partly supported by the Institute for Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korean government (MSIT) (No. RS-2022-II220113, Developing a Sustainable Collaborative Multi-modal Lifelong Learning Framework, 50%; No. RS-2024-00457882, AI Research Hub Project, 20%), and by Samsung Electronics Co., Ltd. (No. G01240136, KAIST Semiconductor Research Fund (2nd), 30%).

Impact Statement

We introduce *HADES*, a lightweight and interpretable state-space model that achieves strong performance with fewer parameters, reducing energy use and improving transparency. While offering efficiency and insight, *HADES*—like all LMs—may reflect biases in data. We urge responsible use, monitoring, and research to enhance fairness.

References

Azizi, S., Kundu, S., Sadeghi, M. E., and Pedram, M. Mambaextend: A training-free approach to improve long context extension of mamba. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LgzRo1RpLS.

Ben-Kish, A., Zimerman, I., Abu-Hussein, S., Cohen, N., Globerson, A., Wolf, L., and Giryes, R. Decimamba: Exploring the length extrapolation potential of mamba. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=iWSl5Zyjjw.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., and Jia, J. Longlora: Efficient fine-tuning of long-context large language models. In *The International Conference on Learning Representations (ICLR)*, 2024.

Choi, J., Wi, H., Kim, J., Shin, Y., Lee, K., Trask, N., and Park, N. Graph convolutions enrich the self-attention in transformers! *Advances in Neural Information Processing Systems*, 37:52891–52936, 2024.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Dao, T. and Gu, A. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024.

Dong, Y., Ding, K., Jalaian, B., Ji, S., and Li, J. Adagnn: Graph neural networks with adaptive frequency response filter. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 392–401, 2021.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Gu, A., Goel, K., and Re, C. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Guo, X., Wang, Y., Du, T., and Wang, Y. Contranorm: A contrastive learning perspective on oversmoothing and beyond. In *The Eleventh International Conference on Learning Representations*, 2023.

Jin, P., Zhu, B., Yuan, L., and Yan, S. Moh: Multi-head attention as mixture-of-head attention, 2024. URL https://arxiv.org/abs/2410.11842.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast autoregressive transformers with linear attention. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 13–18 Jul 2020.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C. A., Manning, C. D., Re, C., Acosta-Navas, D., Hudson, D. A., Zelikman, E., Durmus, E., Ladhak, F., Rong, F., Ren, H., Yao, H., WANG, J., Santhanam, K., Orr, L., Zheng, L., Yuksekgonul, M., Suzgun, M., Kim, N., Guha, N., Chatterji, N. S., Khattab, O., Henderson, P., Huang, Q., Chi, R. A., Xie, S. M., Santurkar, S., Ganguli, S., Hashimoto, T., Icard, T., Zhang, T., Chaudhary, V., Wang, W., Li, X., Mai, Y., Zhang, Y., and Koreeda, Y. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=iO4LZibEqW. Featured Certification, Expert Certification.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, N. Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Erk, K. and Smith, N. A. (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Derczynski, L., Du, X., Grella, M., Gv, K., He, X., Hou, H., Kazienko, P., Kocon, J., Kong, J., Koptyra, B., Lau, H., Lin, J., Mantri, K. S. I., Mom, F., Saito, A., Song, G., Tang, X., Wind, J., Woźniak, S., Zhang, Z., Zhou, Q., Zhu, J., and Zhu, R.-J. RWKV: Reinventing RNNs for the transformer era. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 14048–14077, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.936. URL https://aclanthology.org/2023.findings-emnlp.936/.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/1907.10641.

Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., and Wei, F. Retentive network: A successor to transformer for large language models, 2024. URL https://openreview.net/forum?id=UU9Icwbhin.

Wang, P., Zheng, W., Chen, T., and Wang, Z. Anti-oversmoothing in deep vision transformers via the fourier domain analysis: From theory to practice. In *International Conference on Learning Representations*, 2022.

Wang, P., Cai, R., Wang, Y., Zhu, J., Srivastava, P., Wang, Z., and Li, P. Understanding and mitigating bottlenecks of state space models through the lens of recency and oversmoothing. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=pymXpl4qvi.

Wu, Z., Pan, S., Long, G., Jiang, J., and Zhang, C. Beyond low-pass filtering: Graph convolutional networks with automatic filtering. *IEEE Transactions on Knowledge and Data Engineering*, 35(7):6687–6697, 2022.

Yang, S. and Zhang, Y. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, January 2024. URL https://github.com/fla-org/flash-linear-attention.

Yang, S., Wang, B., Shen, Y., Panda, R., and Kim, Y. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning*, 2024a. URL https://openreview.net/forum?id=ia5XvxFUJT.

Yang, S., Wang, B., Zhang, Y., Shen, Y., and Kim, Y. Parallelizing linear transformers with the delta rule over sequence length, 2024b. URL https://arxiv.org/abs/2406.06484.

Ye, Z., Xia, K., Fu, Y., Dong, X., Hong, J., Yuan, X., Diao, S., Kautz, J., Molchanov, P., and Lin, Y. C. Longmamba: Enhancing mamba's long-context capabilities via training-free receptive field enlargement. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=fMbLszVO1H.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In Korhonen, A., Traum, D., and Màrquez, L. (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Zhang, B. and Sennrich, R. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhang, X., Shen, Y., Huang, Z., Zhou, J., Rong, W., and Xiong, Z. Mixture of attention heads: Selecting attention heads per token. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

A. State Space Models (SSMs) and Mamba

State Space Models Structured state space models represent a new category of sequence models in deep learning, drawing connections to RNNs, CNNs, and traditional state space models. These models are motivated by a specific continuous system that processes a one-dimensional input sequence $x \in \mathbb{R}^T$ into an output sequence $y \in \mathbb{R}^T$ via an implicit latent state $h \in \mathbb{R}^{T \times N}$. Eq. 8 is a fundamental representation of organized SSMs.

$$\begin{aligned} h'(t) &= \bar{\mathbf{A}}h(t-1) + \bar{\mathbf{B}}x(t) \\ y(t) &= \mathbf{C}h(t) + \mathbf{D}x(t) \end{aligned} \tag{8}$$

$$\begin{aligned} h_t &= \mathbf{A}h_{t-1} + \mathbf{B}x_t \\ y_t &= \mathbf{C}h_t + \mathbf{D}x_t \end{aligned} \tag{9}$$

where $\mathbf{A}_t \in \mathbb{R}^{N \times N}$, $\mathbf{B}_t \in \mathbb{R}^{N \times 1}$, $\mathbf{C}_t \in \mathbb{R}^{1 \times N}$. This continuous SSMs in Eq. 8 are discretized to Eq. 9 through fixed formulas: $\mathbf{A} = f_A(\Delta, \bar{\mathbf{A}})$, $\mathbf{B} = f_B(\Delta, \bar{\mathbf{B}})$. For the remainder of this paper, we will omit the parameter $\mathbf{D}$ for exposition (or equivalently, assume $\mathbf{D}$ = 0) because the term $\mathbf{D}x_t$ can be viewed as a skip connection and is easy to compute.

$$\begin{aligned} \mathbf{K} &= [\mathbf{CB}, \mathbf{CAB}, \dots, \mathbf{CA}^k\mathbf{B}] \\ y &= x * \mathbf{K} \end{aligned} \tag{10}$$

In S4 (Gu et al., 2022), the authors refer to this formulation as linear time-invariant (LTI), meaning the system parameters $\mathbf{A}, \mathbf{B}, \mathbf{C}$ do not change over time. The resulting sequence model can be computed either as a linear recurrence or as a global convolution using the kernel $\mathbf{K}$ in Eq. 10.

Using definitions from (Dao & Gu, 2024), we describe Mamba's internal dynamics. Each vector is designated as a row vector. Assuming that $U = [u_1, u_2, ..., u_T]^T \in \mathbb{R}^{T \times d}$, that is, $u_i \in \mathbb{R}^d$, is a discrete time sequence of T tokens, the inner equation for the t-th token of each head of the Mamba layer can be understood as follows:

$$\begin{aligned} h_t &= \mathbf{A}_t h_{t-1} + \mathbf{B}_t x_t \in \mathbb{R}^{N \times P}, \quad y_t = \mathbf{C}_t h_t \in \mathbb{R}^P \\ o_t &= W_o(\mathrm{Norm}(y_t \odot W_z u_t)) \in \mathbb{R}^d \end{aligned} \tag{11}$$

where t is current time step, $x_t, y_t \in \mathbb{R}^P$ are projected input representation and output hidden representations of t-th token respectively, Norm denotes RMS normalization (Zhang & Sennrich, 2019), $W_z \in \mathbb{R}^{P \times d}$, $W_o \in \mathbb{R}^{d \times P}$ are trainable parameters. Especially, in Mamba2, $\mathbf{A}_t$ is scalar-identity matrix, i.e. $\mathbf{A}_t = a_t I$. We denote d for hidden representation dimension, N for state size, P for dimension of each head, T for sequence length.

$$\Delta_{t,\text{base}} = W_\Delta u_t + b_\Delta \in \mathbb{R}, \quad \Delta_t = \mathrm{Softplus}(\Delta_{t,\text{base}}) \in \mathbb{R} \tag{12}$$

By Δ, Mamba implements input-dependent selection mechanism. Δ decides the discretization step size in Mamba, which is used to formulate SSM parameters $\mathbf{A}_t, \mathbf{B}_t$. Detailed parameterization of $\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t, x_t$ are deferred to Appendix A.

Full Mamba2 Architecture Given an input sequence $U = [u_1, u_2, ..., u_T]^\top \in \mathrm{R}^{T \times d}$, a Mamba2 block with d channels is built on top of the S6 layer via the following formula, generating output sequence $O = [o_1, o_2, ..., o_T]^\top \in \mathbb{R}^{T \times d}$:

$$h_t = \mathbf{A}_t h_{t-1} + \mathbf{B}_t x_t \in \mathbb{R}^{N \times P}, \quad y_t = \mathbf{C}_t h_t \in \mathbb{R}^P \tag{13}$$

$$o_t = W_o(\mathrm{Norm}(y_t \odot W_z u_t)) \in \mathbb{R}^d \tag{14}$$

where $W_x, W_z \in \mathrm{R}^{d \times P}$, $W_o \in \mathbb{R}^{P \times d}$ are trainable parameters. Each Mamba2 block consists of M heads, so that $M \times P = d$, which are computed in parallel, the result of which is summed together. We can specify how each matrices are created for each head:

$$\begin{aligned} \bar{\mathbf{A}}_t &= a_t \mathbf{I} \in \mathbb{R}^{N \times N} \\ a_t &= \exp(-\Delta_t \exp(A)) \in \mathbb{R} \\ \mathbf{B}_t &= \Delta_t \bar{\mathbf{B}}_t \in \mathbb{R}^{N \times 1} \\ \bar{\mathbf{B}}_t &= \sigma(\mathrm{Conv}(W_B u_t)) \in \mathbb{R}^{N \times 1} \end{aligned} \tag{15}$$

$$\begin{aligned} \mathbf{C}_t &= \sigma(\mathrm{Conv}(W_C u_t))^\top \in \mathbb{R}^{1 \times N} \\ \Delta_{t,\text{base}} &= W_\Delta u_t + b_\Delta \in \mathbb{R} \\ \Delta_t &= \mathrm{Softplus}(\Delta_{t,\text{base}}) \in \mathbb{R} \\ x_t &= \sigma(\mathrm{Conv}(W_x u_t)) \in \mathbb{R}^{P \times 1} \end{aligned} \tag{16}$$

where $W_B, W_C \in \mathbb{R}^{N\times d}$, $W_\Delta \in \mathbb{R}^{1\times d}$. σ denotes SiLU activation function and $\mathrm{Conv}(\cdot)$ denotes a channel-wise one-dimensional convolution. By Δ, Mamba2 implements input-dependent selection mechanism. A_t performs as decay-ratio as it is cumulatively multiplied. DeciMamba (Ben-Kish et al., 2025) elaborates the condition of a_t. For computational stability, $\Delta > 0$ and $A < 0$ is guaranteed in original implementation. Therefore, we can conclude $a_t \in (0, 1)$.

Using this Mamba2 block, we can derive layer-wise Mamba2 architecture with L layers as below. For initial input, input sequence is $I = [i_0, i_1, ..., i_T] \in \mathbb{R}^T$ where $i_t \in [V]$ and we have $U^{(l-1)} = [u_1^{(l-1)}, u_2^{(l-1)}, ...u_T^{(l-1)}]$ as input sequence for the l-th layer. $O^{(l)} = [o_1^{(l)}, o_2^{(l)}, ...o_T^{(l)}]$ serves as output sequence of l-th Mamba2 layer $\mathrm{Mamba}^{(l)}$, V denotes vocab size and $P \in \mathbb{R}^{T\times V}$ denotes final logits.

$$U^{(0)} = \mathrm{Embedding}_{in}(I) \in \mathbb{R}^{T\times d} \tag{17}$$

$$O^{(l)} = \mathrm{Mamba}^{(l)}(\mathrm{Norm}[U^{(l-1)}]) \in \mathbb{R}^{T\times d} \tag{18}$$

$$P = \mathrm{Embedding}_{out}(\mathrm{Norm}([O^{(L)}]) \in \mathbb{R}^{T\times V} \tag{19}$$

Here, the output of the l-th layer is used as the input for the $l+1$-th layer, i.e. $O^{(l)} = U^{(l)}$.

B. Graph Signal Processing (GSP)

Graph Signals and Filtering Graph Signal Processing (GSP) provides tools for analyzing and processing data defined over graph structures. In GSP, a signal is defined as a vector $\mathbf{x} \in \mathbb{R}^N$, where each element is associated with a node in a graph of N nodes. One of the core operations in GSP is *graph filtering*, which can be viewed as a form of graph convolution. This operation emphasizes or suppresses specific frequency components of the signal based on the graph topology. Given a shift operator $\mathbf{S} \in \mathbb{R}^{N\times N}$—typically chosen as the adjacency matrix or the (normalized) graph Laplacian—a linear graph filter $\mathbf{G}$ is often defined as a polynomial in $\mathbf{S}$:

$$\mathbf{y} = \mathbf{G}\mathbf{x} = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}, \tag{20}$$

where $\mathbf{x}$ is the input graph signal, h_k are the filter coefficients (also called filter taps), and K is the filter order. This convolution operation aggregates information from neighboring nodes up to K hops away, as determined by powers of the shift operator. This filtering can also be interpreted as a linear time-invariant (LTI) system on graphs, where the filter coefficients h_k determine the system's impulse response under the graph structure. This system-theoretic view enables a conceptual connection to structured sequence models such as SSMs, which we explore in the following sections.

Graph Filter Banks A graph filter bank applies multiple filters to a graph signal and combines their outputs to form a unified representation. Given a graph signal $\mathbf{x} \in \mathbb{R}^N$ and a graph shift operator $\mathbf{S} \in \mathbb{R}^{N\times N}$, the filter bank output can be expressed as:

$$\mathbf{y} = \mathbf{\Phi}\left(\left\{\mathbf{y}^{(i)}\right\}_{i=1}^{M}\right) = \mathbf{\Phi}\left(\left\{\sum_{k=0}^{K} h_k^{(i)} \mathbf{S}^k \mathbf{x}\right\}_{i=1}^{M}\right), \tag{21}$$

where $h_k^{(i)}$ are the coefficients of the i-th filter, K is the filter order, M is the number of filters in the bank, and $\Phi(\cdot)$ denotes the aggregation function over the filter outputs (e.g., concatenation, summation, or projection). This general form enables the system to capture diverse frequency characteristics of the graph signal through multiple learned filters.

Our method adopts this perspective to reinterpret the multi-head SSM as a learnable graph filter bank, where each head corresponds to a distinct frequency response. While our model does not explicitly compute the graph spectrum, this filter bank perspective serves as a conceptual tool for understanding the role of the learned dynamic filters.

C. Experimental Setup

C.1. Training Details

We adopt all baseline implementations from `flash-linear-attention` (Yang & Zhang, 2024). For fair comparison, all models are trained under identical conditions with 370M parameters exculding readout head on 200B tokens from the

Pile dataset (Gao et al., 2020). We use the AdamW optimizer with a peak learning rate of 48e-4, weight decay of 0.1, $\beta \in [0.9, 0.95]$ following Mamba2, and gradient clipping of 1.0. The learning rate follows a cosine annealing schedule with a warm-up phase of 375M tokens and a batch size of 2^{22} tokens (# sequences $\times$ sequence length) and the number of training steps as 47,042 (# tokens / # tokens in one batch) steps. All models employ the GPT-NeoX tokenizer with a vocabulary size of 50,277. For sequence modeling, we set the training length to 2K tokens. Our experiments were conducted on a computing server equipped with an AMD EPYC 9654 CPU (2 sockets, 192 cores, 384 threads, 1.5–3.7 GHz, L3 cache 768 MiB) and four NVIDIA A100 80GB PCIe GPUs with CUDA version 12.4. For our model, we used hyperparameter set of $H = 16$, $S = 8$, $\lambda_1 = 1e-3$, $\lambda_2 = 1e-3$, $\gamma = 25e-2$. Our implementation is publicly available at `https://github.com/yehjin-shin/HADES`.

C.2. Evaluation

Following prior works (Gu & Dao, 2023; Yang et al., 2024a), we evaluate our method against five baseline models across two evaluation categories: WikiText (Wiki.) perplexity and zero-shot commonsense reasoning tasks. The commonsense tasks include LAMBADA (LMB.; (Paperno et al., 2016)), PIQA (Bisk et al., 2019), HellaSwag (Hella.; (Zellers et al., 2019)), WinoGrande (Wino.; (Sakaguchi et al., 2019)), ARC-easy (ARC-e) and ARC-challenge (ARC-c) (Clark et al., 2018), BoolQ (Clark et al., 2019), and OpenbookQA (OBQA.; (Mihaylov et al., 2018)).

We measure perplexity (ppl) on WikiText and LAMBADA, normalized accuracy(acc_n) on HellaSwag and ARC-challenge, and standard accuracy (acc) on the remaining tasks (as normalized accuracy provides higher scores for most models on these tasks). Avg. denotes the averaged result of the accuracies and normalized accuracies of eight tasks together. All evaluations are conducted using `lm-evaluation-harness` (Liang et al., 2023). We provide details of the evaluation tasks below.

- WikiText (Merity et al., 2017): A dataset consisting of high-quality, clean text extracted from Wikipedia articles, commonly used to evaluate language modeling tasks by measuring a model's ability to predict and generate coherent and fluent text.
- LAMBADA (Paperno et al., 2016): A text completion task that measures a model's ability to predict the final word of a passage, requiring comprehension of the context, commonsense reasoning, as well as the ability to generate text coherently.
- PIQA (Bisk et al., 2019): A physical commonsense reasoning task focused on selecting the most plausible solution to everyday scenarios.
- HellaSwag (Zellers et al., 2019): A multiple-choice task that evaluates a model's ability to select the most coherent continuation of a given situation based on commonsense and narrative reasoning.
- WinoGrande (Sakaguchi et al., 2019): An expanded version of the Winograd Schema Challenge: a pronoun resolution task designed to test commonsense reasoning by identifying which noun a pronoun refers to in a given sentence.
- OpenbookQA (Mihaylov et al., 2018): A multiple-choice question answering task designed to test a model's understanding of elementary-level science facts and its ability to apply this knowledge to novel scenarios requiring reasoning and inference.
- ARC-easy (Clark et al., 2018): A subset of the AI2 Reasoning Challenge focusing on questions that require basic scientific and commonsense knowledge.
- ARC-challenge (Clark et al., 2018): A more difficult subset of the AI2 Reasoning Challenge that tests advanced reasoning and deep understanding of scientific and commonsense knowledge.
- BoolQ (Clark et al., 2019): A yes/no question answering dataset with 15,942 examples, derived from Google search queries, paired with Wikipedia passages.

Passkey Retrieval For the passkey retrieval task, we adopt the task formulation from (Chen et al., 2024). The evaluation is conducted across context lengths from 1K to 16K, with the target digit hidden at depths of 0% to 100% with the gap of 10% of each of these sequences. Assuming that each correct retrieval receives a score of 1 and each incorrect retrieval receives a score of 0, we compute the retrieval score as count out of 10, across all the depths overall context lengths. We did not apply any fine-tuning with longer sequences. We structure the prompt for the passkey retrieval task into four distinct components: task description, passkey, query, and dummy text.

D. Ablation Studies

In this subsection, we report the result of ablation studies. We test variations of our model with same evaluation setting in Appendix C.2. In Table 2, our ablation studies demonstrate both the robustness and tunability of our model. We first ablate on two auxiliary losses: $\mathcal{L}_{\text{diversity}}$ and $\mathcal{L}_{\text{balance}}$. As shown in Table 2, the best performance is achieved when both losses are applied together. Interestingly, the absence of $\mathcal{L}_{\text{balance}}$ results in a performance drop, which can be attributed to the filter selection becoming overly concentrated on a few filters. This imbalance hinders the training of expert filters that are rarely selected, preventing them from effectively learning the underlying dynamics when they are eventually chosen. We also evaluate the impact of different filter configurations. Table 2 shows that the best performance is achieved when both filter types are combined. Using only shared filters outperforms using only expert filters, as shared filters consistently capture global low-frequency information, while expert filters adaptively capturing low and high frequency information. This complementary behavior makes their combination essential for optimal performance.

Table 2: Full result for Ablation Studies. Avg. denotes the averaged result of the accuracies and normalized accuracies of eight tasks together.

Methods	Wiki. ppl ↓	LMB. ppl ↓	LMB. acc ↑	PIQA acc ↑	Hella. acc_n ↑	Wino. acc ↑	ARC-e acc ↑	ARC-c acc ↑	BoolQ acc ↑	OBQA. acc_n ↑	Avg. 8 tasks ↑
HADES (Ours)	**31.48**	**21.74**	**39.24**	**63.93**	**32.82**	52.64	**45.03**	22.01	**58.84**	**28.80**	**42.91**
w/o $\mathcal{L}_{\text{balance}}$	34.73	26.84	36.77	62.68	30.96	50.75	43.22	**22.70**	59.27	26.20	41.57
w/o $\mathcal{L}_{\text{diversity}}$	33.83	27.40	36.04	62.46	31.48	51.38	44.87	22.61	59.94	28.40	42.15
Only Shared Filters	34.55	27.64	35.75	62.40	31.39	**52.88**	44.23	24.23	**60.83**	26.00	42.21
Only Expert Filters	36.34	30.12	34.89	61.53	30.30	52.41	44.49	**22.70**	58.29	**28.80**	41.68

E. Effect of Spectral Bias

We further explore the impact of spectral bias on expert filters. Specifically, Fig. 2(c) shows the frequency spectrum of the output generated by using the original delta without spectral bias, Δ_t. In contrast, Fig. 2(b) illustrates the effect of applying spectral bias to expert filters, $\Delta_{t,HADES}$, where a clear upward shift in frequency distribution is observed. This shift indicates that the delta values are tuned to capture higher-frequency details, enabling the model to learn finer-grained information. Fig. 4 presents a log-scale histogram of the difference between $\Delta_{t,\text{HADES}}$ and Δ_t, calculated over 25 randomly sampled sentences from the Pile dataset, totaling approximately 38,000 tokens. Throughout our analysis, we observe that the spectral residual bias is generally positive, which encourages larger Delta values and enables the model to effectively capture high-frequency information. Occasionally, the bias becomes negative, reducing the step size for certain tokens and allowing the model to better capture global context. This adaptive mechanism allows *HADES* to flexibly balance the extraction of local and global information, adjusting to the needs of each token in context.

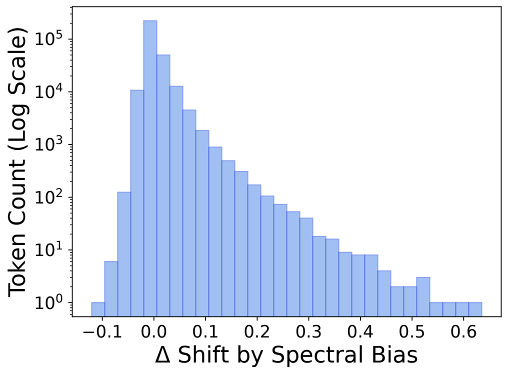


Figure 4: Histogram of $\Delta_{t,HADES} - \Delta_t$

F. Related Works

Graph Signal Processing in Language Modeling Recent work has explored interpreting Transformer architectures through GSP. In this view, the self-attention mechanism functions as a graph filter, where the attention matrix acts as a learned adjacency matrix, with tokens as nodes and attention weights defining edges. GFSA (Choi et al., 2024) explicitly models self-attention as a graph filter on a fully connected graph, while ContraNorm (Guo et al., 2023) treats it as a normalized adjacency matrix, connecting it to Graph Neural Networks. The Anti-Oversmoothing framework (Wang et al., 2022) further characterizes self-attention as a low-pass filter, highlighting its smoothing effect in the spectral domain. However, unlike

Transformers, which rely on a fully connected graph structure, SSMs operate in a fundamentally different way. Their sequential, unidirectional nature is best captured by a line graph, where information flows along a structured path, reflecting their recurrent design. This insight motivates our GSP-based filter bank approach for Mamba2, a recent SSM-based language model.

Adaptive Filtering Recent methods have improved multi-head attention efficiency by dynamically selecting or weighting attention heads. Mixture of Attention Heads (MoA) (Zhang et al., 2022) treats each head as an independent expert, with a router dynamically selecting a subset of K heads per token, enhancing efficiency by focusing on the most relevant heads. Interpreted through a GSP lens, MoA functions as adaptive filtering, where tokens selectively activate the most suitable filters (heads). Building on this, Mixture-of-Heads Attention (MoH) (Jin et al., 2024) further advances this approach by using a router to assign weights to all heads, rather than selecting a subset. This allows each token to receive a weighted combination of all head outputs, offering greater flexibility. Unlike MoA, which treats heads independently, MoH uses a shared set of heads with adaptive weights, providing a more direct form of adaptive filtering where filter weights are continuously adjusted.

Modulation of SSMs Mamba's recursive state update leads to information loss as context length increases, a problem noted in various studies. DeciMamba (Ben-Kish et al., 2025) addresses this by measuring information loss using Effective Receptive Field (ERF) and removing less important tokens with low Δ values. MambaExtend (Azizi et al., 2025) improves on this by offering a training-free scaling method, adjusting Δ values directly to enhance long-context performance. LongMamba (Ye et al., 2025) further refines this by separating global and local channels, using token filtering in global channels to improve memory efficiency and extend the receptive field. Another work tries to emphasize polarization of $\mathbf{A}$, thereby allowing SSMs to capture vanishing influence of earlier tokens in long sequences (Wang et al., 2025).

G. Limitation and Future Works

While this study introduces a novel perspective on Mamba2 by reinterpreting it as a filter bank through the lens of GSP and proposes a new design methodology, there are some limitations. First, our experiments are limited to a single model size due to lack of resource. As a result, we have not yet explored the behavior of our approach across various model scales. In future work, we plan to conduct scaling experiments to better understand the generalizability of our method across different model sizes. Second, although our design is inspired by GSP principles, we have not explicitly enforced spectral properties within the model. Instead, we adopt an implicit design approach, where spectral characteristics are indirectly encouraged with slight modification of biases. Explicitly enforcing spectral properties could lead to overly rigid behavior, which may hinder model performance. Our current approach aims to maintain flexibility while subtly guiding the model toward desirable spectral behavior. For future work, we aim to conduct a theoretical analysis of the advantages of explicit spectral design and explore new methods for biasing and filter selection that directly leverage these properties. Such an investigation could lead to more robust and interpretable state-space models.