Delayed Attention Training Improves Length Generalization in Transformer–RNN Hybrids

Buu Phan[1]* **Reza Ebrahimi** [2] **Sanjay Haresh** [2] **Roland Memisevic**[2]
[1]University of Toronto [2]Qualcomm AI Research [†]
truong.phan@mail.utoronto.ca, {ebrahimi,sanjayh,rmemisev}@qti.qualcomm.com

Abstract

We study length generalization in sequence models on a composite problem involving both state tracking and associative recall. Prior work finds that recurrent networks handle state tracking well but struggle with recall, whereas Transformers excel at recall yet fail to extend state-tracking capabilities to longer sequences. Motivated by the complementary strengths of these architectures, we construct hybrid models integrating recurrent and attention-based components, and train them on the combined task to evaluate whether both capabilities can be preserved. Our results reveal that, in such hybrids, the Transformer component tends to exploit shortcut solutions, leading to poor length generalization. We identify this shortcut reliance as a key obstacle and propose a simple yet effective training strategy—delaying the training of the attention layers—that mitigates this effect and significantly improves length generalization performance. Our experiments show that this approach enables hybrid models to achieve near-perfect accuracy ($> 90\%$) on hybrid sequences three times longer than those used during training.

1 Introduction

Transformers play a central role in modern language models [3, 10]. Part of this success has been attributed to their ability to perform a kind of *content-based retrieval*, making them especially effective for associative recall and related reasoning tasks prevalent in language modeling [1]. On the other hand, transformers are known to struggle with tasks that require *state tracking* and to length-generalize in the absence of carefully designed task-specific features or reasoning formats [5, 12]. The reason is that the lack of an inductive bias towards recurrent (step-by-step) processing allows transformers to rely on shortcut heuristics [6] (e.g., solving modulo addition without maintaining sequence-level state), leading to brittle length generalization behavior. In contrast, recurrent models such as LSTMs [8] naturally maintain state across long sequences and generalize well on state-tracking tasks, but their fixed-capacity memory makes content-based retrieval challenging for these models.

Contributions. Given the importance of length generalization for general reasoning, we investigate whether a hybrid architecture can effectively combine the complementary strengths of recurrent and attention-based models. To this end, we design a synthetic task that jointly evaluates a model's ability to perform both *state tracking* and *selective recall*. Our experiments reveal that transformers are prone to adopting shortcut solutions, even within hybrid architectures, thereby undermining their ability to generalize over longer sequences. Nevertheless, we find that by delaying the training of the attention component, we can effectively prevent such shortcuts and enable the network to preserve state-tracking capabilities while still benefiting from the recall advantages of attention.

2 Methods and Results

Experimental Setup. We design each training instance as a sequence consisting of key–value pairs, followed by two query markers that require multiple different forms of reasoning. Each input

---

[*]Work done during internship at Qualcomm AI Research

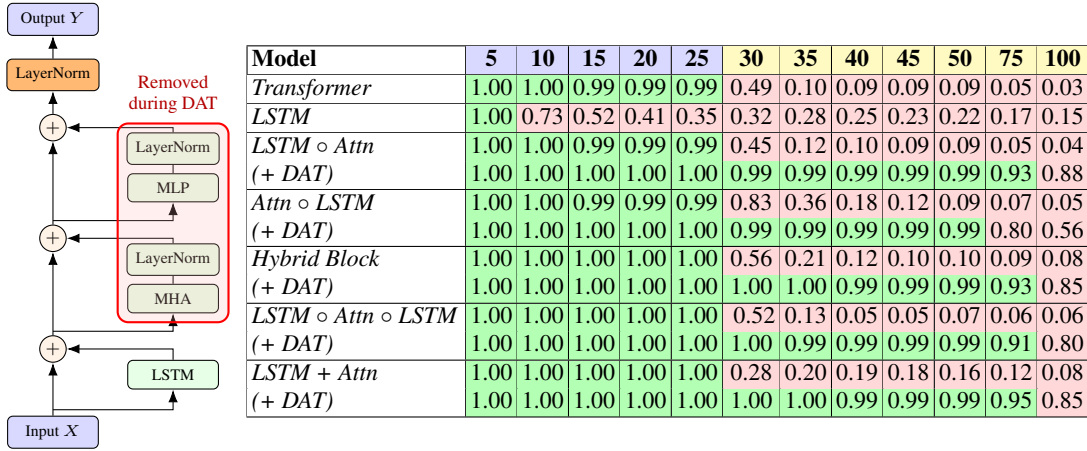[†]Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

Work presented at What Can('t) Transformers Do? Workshop at NeurIPS 2025.

Figure 1: Left: Hybrid block diagram with a single attention layer, i.e. the red box, which is replicated ×4 in experiments. Right: Accuracy across sequence lengths, evaluated on both in-distribution (≤ 25) and out-of-distribution (> 25) cases. Accuracy with $> 90\%$ is highlighted in green color.

Model	5	10	15	20	25	30	35	40	45	50	75	100
Transformer	1.00	1.00	0.99	0.99	0.99	0.49	0.10	0.09	0.09	0.09	0.05	0.03
LSTM	1.00	0.73	0.52	0.41	0.35	0.32	0.28	0.25	0.23	0.22	0.17	0.15
LSTM ∘ Attn	1.00	1.00	0.99	0.99	0.99	0.45	0.12	0.10	0.09	0.09	0.05	0.04
(+ DAT)	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.99	0.93	0.88
Attn ∘ LSTM	1.00	1.00	0.99	0.99	0.99	0.83	0.36	0.18	0.12	0.09	0.07	0.05
(+ DAT)	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.99	0.80	0.56
Hybrid Block	1.00	1.00	1.00	1.00	1.00	0.56	0.21	0.12	0.10	0.10	0.09	0.08
(+ DAT)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.93	0.85
LSTM ∘ Attn ∘ LSTM	1.00	1.00	1.00	1.00	1.00	0.52	0.13	0.05	0.05	0.07	0.06	0.06
(+ DAT)	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.91	0.80
LSTM + Attn	1.00	1.00	1.00	1.00	1.00	0.28	0.20	0.19	0.18	0.16	0.12	0.08
(+ DAT)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.95	0.85

sequence takes the form: $\langle\text{bos}\rangle\, k_1 v_1\, k_2 v_2 \cdots k_n v_n\, \langle\text{modulo}\rangle\, m\, \langle\text{recall}\rangle\, k_j v_j$, where k_i and v_i are discrete tokens representing keys and values, respectively. The sequence begins with a special start symbol $\langle\text{bos}\rangle$. The segment $\langle\text{modulo}\rangle$ queries the model for the sum of all values modulo 10, and the segment $\langle\text{recall}\rangle$ queries the model for the value associated with a randomly chosen key k_j from the sequence. A prediction is considered correct only if the model successfully answers both tasks.

Architectures. We fix the embedding dimension to $d = 256$ for all models and adopt RoPE [9] as the default positional encoding method. We evaluate two baselines—Transformer-only and LSTM-only—along with four hybrid variants composed of the same components. Details of each hybrid are provided in Appendix A. Figure 1 (left) illustrates one such design, referred to as the *Hybrid Block*, where an LSTM block is integrated into the residual architecture of a Post-LayerNorm Transformer.

Delayed Attention Training (DAT). In the initial phase of DAT, we deactivate the attention modules, effectively disabling their contribution while the recurrent pathway learns to handle state tracking. After a fixed number of epochs, we enable gradient flow through the attention layers, allowing them to train jointly with the recurrent module. This encourages the model to first establish a robust state-tracking mechanism before leveraging attention for content-based retrieval. Figure 1 (left) shows how we deactivate the attention function in the hybrid block during early phase.

Results. Figure 1 (right) compares the performance of mentioned architectures in the proposed task. LSTM underperforms even in-distribution due to its restricted memory capacity in handling the recall component. Transformers performs relatively well in-distribution but fail to perform state-tracking and consequently do not length generalize. Hybrid architectures also achieve high accuracy in-distribution, but their ability to generalize degrades with increasing sequence length, as the incorporation of attention tends to encourage shortcut learning. This effect is most clearly observed when training solely on the modulo-10 task (without the recall component) (see Appendix B): while an LSTM model exhibits perfect length generalization, augmenting it with an additional attention layer immediately forfeits this capability and results in rapid performance degradation. This is somewhat counterintuitive, since one might naturally expect the attention mechanism to merely propagate the hidden states to the final projection head without impairing generalization. Importantly, DAT mitigates this failure and demonstrate robust length generalization across different hybrid architectures. Finally, we note that the accuracy drop at sequence length 100 is due to the reduced sharpness of the attention softmax[11], evidenced in Table 1 in Appendix B.

3 Conclusion

In this work, we studied the challenge of length generalization on tasks that require both recall and state tracking. We showed that while LSTMs generalize well on arithmetic state-tracking tasks and Transformers excel at selective recall, naïve hybrid architectures, which combine the two, fail, since the attention mechanism allows a model to learn shortcuts. To address this, we proposed *Delayed Attention Training*, a simple strategy that prevents shortcut learning and enables reliable length generalization in hybrid models. Our experiments demonstrate that this approach recovers the complementary strengths of recurrence and attention. Looking forward, extending the framework to large-scale language modeling and designing specialized modules or training curricula may further improve generalization in real-world settings.

References

[1] Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=LY3ukUANko`.

[2] Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, ZIJIA CHEN, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Celine Lin, Jan Kautz, and Pavlo Molchanov. Hymba: A hybrid-head architecture for small language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=A1ztozypga`.

[3] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

[4] M Reza Ebrahimi and Roland Memisevic. Revisiting bi-linear state transitions in recurrent neural networks. *arXiv preprint arXiv:2505.21749*, 2025.

[5] MohammadReza Ebrahimi, Sunny Panchal, and Roland Memisevic. Your context is not an array: Unveiling random access limitations in transformers. In *First Conference on Language Modeling*, 2024.

[6] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[7] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[9] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[11] P Velickovic, C Perivolaropoulos, F Barbero, and R Pascanu. softmax is not enough (for sharp out-ofdistribution), 2024. *URL https://arxiv. org/abs/2410.01104*, 2025.

[12] Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024.

Model	5	10	15	20	25	30	35	40	45	50	75	100
Transformer	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.90	0.80
LSTM	1.00	0.98	0.95	0.89	0.83	0.76	0.70	0.64	0.59	0.53	0.37	0.28
LSTM ∘ Attn	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.99	0.90	0.76
Attn ∘ LSTM	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.96	0.89
Hybrid Block	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.98	0.92
LSTM ∘ Attn ∘ LSTM	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.97	0.90
LSTM + Attn	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.92	0.82

Table 1: Results on the Recall-Only setting.

Model	5	10	15	20	25	30	35	40	45	50	75	100
Transformer	1.00	1.00	1.00	0.99	0.98	0.51	0.11	0.10	0.10	0.10	0.10	0.10
LSTM	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
LSTM ∘ Attn	1.00	1.00	1.00	1.00	1.00	0.20	0.10	0.10	0.10	0.10	0.10	0.10
Attn ∘ LSTM	1.00	1.00	1.00	1.00	1.00	0.63	0.10	0.10	0.10	0.10	0.10	0.10
Hybrid Block	1.00	1.00	1.00	1.00	1.00	0.45	0.22	0.13	0.10	0.10	0.10	0.10
LSTM ∘ Attn ∘ LSTM	1.00	1.00	1.00	1.00	1.00	0.73	0.16	0.08	0.09	0.10	0.10	0.10
LSTM + Attn	1.00	1.00	1.00	1.00	0.99	0.11	0.11	0.10	0.10	0.10	0.10	0.10

Table 2: Results on the Modulo-Only setting.

A Hybrid Architectures

Training Setup. We use Adam optimizer with the learning rate of 10^{-4} for the first 500 epoches and 10^{-5} afterwards for all end-to-end setup. For DAT setup, we train the LSTM component for 1500 epoches with learning rate of 10^{-4}. During attention training phase, we use the learning rate of 10^{-4} for 500 epoches before switching to 10^{-5}. Each epoch consists of 2^{14} randomly generated samples.

Architectures. We describe the hybrid architectures in details as follows. Unless otherwise stated, we use 4 block in the transformer components with post-layernorm setup.

(1) Attention-on-LSTM (Attn ∘ LSTM). An LSTM encodes the sequence, and its hidden states are fed to a self-attention (MHA+MLP) block: $H = \text{LSTM}(X)$, $Z = \text{Attn}(H)$.

(2) LSTM-on-Attention (LSTM ∘ Attn). A self-attention block first produces contextualized features that are then processed by an LSTM: $H = \text{Attn}(X)$, $Z = \text{LSTM}(H)$.

(3) Parallel branch-and-add (LSTM+Attn). LSTM and self-attention process the input in parallel and their outputs are combined by an elementwise sum: $H_{\text{lstm}} = \text{LSTM}(X)$, $H_{\text{attn}} = \text{Attn}(X)$, $Z = H_{\text{lstm}} + H_{\text{attn}}$.

(4) Hybrid block (Fig.1). A composite module that integrates LSTM, self-attention, and MLP within a single residual block as depicted in Figure 1 (Left).

(5) Sandwich Attention (LSTM ∘ Attn ∘ LSTM). The sequence is encoded by an LSTM, passed through a self-attention block (MHA+MLP), and then processed by another LSTM: $H = \text{LSTM}(X)$, $Z = \text{Attn}(H)$, $Y = \text{LSTM}(Z)$.

Remarks. Beyond LSTMs, there exist several alternative sequence models based on state-space formulations, such as Mamba [7] and its hybrid variants, e.g., Hymba [2]. Since state-space models are known to lack effective state-tracking capabilities [4], we exclude them from our hybrid design, expecting the conclusions drawn for transformers in Appendix B to extend naturally to these models.

B Additional Results

We report results on the performance of each model under single-task settings, where models are trained and evaluated solely on either the recall or modulo task in Table 1 and 2. The results confirm that the transformer component enables all hybrid models to excel at the recall task but leads to poor length generalization on the state-tracking task.