ASA: Adaptive Subquadratic Attention with Dynamic Complexity Adjustment

Xinyuan Liu
Northwestern University
xinyuanliu2024@u.northwestern.edu

Abstract

This paper introduces Adaptive Subquadratic Attention (ASA) with Dynamic Complexity Adjustment and Information Retrieval, a novel attention mechanism that improves computational efficiency while preserving model expressiveness by restructuring the query, key, and value (QKV) representations through linear projections and Softmax operations. ASA reduces attention complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nd^2 + ndm)$, where $m < d$, enabling more scalable sequence modeling by avoiding dense pairwise token interactions. Specifically, Q and K - initially in $\mathbb{R}^{n\times d}$ - are independently projected into a lower-dimensional space $\mathbb{R}^{n\times m}$ using separate linear transformations, minimizing the cost of downstream operations. Softmax is then applied independently to the compressed Q and K to produce probability-like representations, retaining nonlinear attention behavior while avoiding the quadratic cost of pairwise similarity. Finally, ASA performs a two-stage attention process: the transformed key is first used to summarize the value matrix into a compact representation, which is subsequently weighted by the transformed query to produce the final output - enabling efficient and structured information retrieval. Unlike standard Softmax attention, ASA avoids the quadratic computation of pairwise token interactions and decouples Q/K interactions. Compared to linear attention, ASA preserves richer contextual representations by retaining the nonlinear selectivity of Softmax and maintaining probabilistic weighting. This structure enables both efficient attention computation and selective information retrieval, making ASA a compelling trade-off between speed and expressiveness. Empirical results show that ASA consistently outperforms leading Softmax- and linear-based attention mechanisms—including Transformer++, Mamba, and GLA—across a range of NLP tasks such as machine translation, question answering, and text summarization.

1 Introduction

In recent years, Transformer-based architectures have revolutionized Natural Language Processing (NLP), driving significant advancements in tasks such as machine translation, question answering, and text summarization. Central to these models is the attention mechanism, particularly the Softmax-based attention, which enables the model to capture rich contextual dependencies by computing pairwise interactions between all input tokens. However, this comes at a cost: the computational complexity of the attention mechanism grows quadratically with the input sequence length, i.e., $\mathcal{O}(n^2d)$, where n is the sequence length and d is the token dimension. For instance, in widely-used models like BERT Devlin *et al.* [2019] and GPT Series Brown *et al.* [2020], processing long sequences becomes computationally prohibitive due to this scaling issue.

To address this limitation, researchers have proposed linear attention mechanisms such as Linformer Wang *et al.* [2020], Performer Choromanski *et al.* [2021], and Mamba Gu and Dao [2024], which reduce the computational complexity to $\mathcal{O}(nd)$. These approaches approximate the attention computation using techniques such as low-rank factorization, kernel-based projections, or selective state spaces (e.g., S4), thereby enabling efficient processing of longer sequences. While linear attention mechanisms offer substantial computational savings, they often come at the cost of reduced expressive capacity—resulting in degraded performance on tasks that require modeling long-range dependencies, nuanced contextual relationships, or complex linguistic structures Fan *et al.* [2025].

In addition, hybrid attention mechanisms have emerged to combine the efficiency of linear attention with the expressiveness of traditional Softmax-based attention. For example, Longformer Beltagy *et al.* [2020] introduces a mix of local windowed attention and global tokens, allowing selective long-range interactions while maintaining linear complexity. Similarly, BigBird Zaheer *et al.* [2020] constructs a sparse attention pattern through a combination of sliding windows, global tokens, and random connections, which is shown to retain the expressive power of full attention under certain theoretical conditions. Beyond static patterns, some hybrid approaches introduce dynamic adaptability into the attention mechanism itself. For instance, GSA Zhang *et al.* [2024] blends Transformer and RNN characteristics by building on

Gated Linear Attention and introducing bounded-memory control. This enables adaptive forgetting and context-aware memory access, which helps retain long-term dependencies during recurrent inference—effectively preserving expressiveness while maintaining efficiency.

This paper introduces *Adaptive Subquadratic Attention (ASA)*, a novel attention mechanism that bridges the high expressiveness of Softmax attention with the computational efficiency of linear attention. ASA achieves this by applying separate linear projections and Softmax operations to the query, key, and value components, enabling a two-stage attention process that reduces complexity while retaining rich contextual interactions. This design allows ASA to selectively focus on relevant information through structured transformations, effectively balancing performance, scalability, and efficiency. Our contributions can be summarized as follows:

1. ASA reduces attention complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nd^2 + ndm)$ by combining low-dimensional projection with a two-stage attention process that avoids explicit pairwise query-key interactions.

2. Despite its reduced complexity, ASA preserves strong model expressiveness by leveraging Softmax-based weighting over compressed representations. This structured design maintains the model's ability to capture rich contextual dependencies and retrieve salient information effectively.

3. Extensive experiments on NLP benchmarks demonstrate that ASA consistently outperforms state-of-the-art attention mechanisms across a range of tasks, including machine translation, question answering, and commonsense reasoning, particularly excelling in scenarios requiring high recall and long-range dependency modeling.

The remainder of this paper is organized as follows. Section 2 details the ASA framework, and Section 3 presents experimental results. Finally, Section 4 concludes the paper.

2 Method

2.1 Review on Softmax and Linear Attention

Softmax Attention

Softmax attention, introduced in the Transformer architecture Vaswani *et al.* [2017], computes attention scores based on pairwise interactions between tokens. It operates on three learnable variables: the query $Q \in \mathbb{R}^{n\times d}$, key $K \in \mathbb{R}^{n\times d}$, and value $V \in \mathbb{R}^{n\times d}$, where n represents the sequence length and d is the embedding dimension.

The attention output $A \in \mathbb{R}^{n\times d}$ is computed as: $A = \text{softmax}\left(QK^\top/\sqrt{d}\right)V$, where $1/\sqrt{d}$ is a scaling factor introduced to prevent large dot-product values from dominating the Softmax operation, and $\text{softmax}(\cdot)$ is the Softmax function that ensures the attention scores sum to 1 across the sequence.

Softmax attention excels at capturing global dependencies due to its pairwise computation, making it highly effective for tasks like machine translation, text summarization, and question answering. However, its computational complexity is $\mathcal{O}(n^2d)$ due to the matrix multiplication $QK^\top$, which scales quadratically with the sequence length n. This high complexity limits its scalability for very long sequences, as it requires significant memory and computation.

Linear Attention

Linear attention addresses the computational inefficiency of Softmax attention by reformulating the attention mechanism to scale linearly with sequence length. The key idea is to approximate the dot-product computation using kernel functions, reducing the computational complexity from quadratic to linear.

The attention output in linear attention is computed as:

$$A = \phi(Q)\left(\phi(K)^\top V\right),$$

where $\phi(\cdot)$ is a kernel function and the term $\phi(K)^\top V \in \mathbb{R}^{d\times d}$ is precomputed, enabling efficient computation.

The computational complexity of linear attention is $\mathcal{O}(nd)$, which significantly reduces the memory and computation requirements for long sequences. This efficiency makes linear attention well-suited for tasks like document modeling, speech processing, and genomic data analysis. However, the approximation used in linear attention can struggle to model nuanced global dependencies, as the kernel-based reformulation sacrifices some of the exact pairwise interactions. This trade-off may impact performance in tasks requiring deep contextual understanding, where precise global relationships are essential.

2.2 Adaptive Subquadratic Attention

In this section, we describe the **Adaptive Subquadratic Attention (ASA)** mechanism. It is motivated by the need to retain the expressive power of Softmax attention while avoiding its quadratic complexity. Unlike linear attention, which sacrifices pairwise expressiveness for speed, ASA maintains nonlinear selectivity through a structured and efficient two-stage process. Unlike Softmax-based attention, this design enables ASA to operate with a total complexity of $\mathcal{O}(nd^2 + ndm)$, significantly lower than the $\mathcal{O}(n^2d)$ cost of traditional attention mechanisms.

ASA consists of the following steps: (1) *low-dimensional projection*, where the query and key representations are independently projected into a lower-dimensional space $\mathbb{R}^{n\times m}$ using learned linear transformations, with $m < d$; (2) *independent Softmax-based weighting*, where the projected query and key are transformed into probability-like representations through row-wise Softmax operations; (3) *two-stage attention*, where the transformed key is used to summarize the value matrix, and the transformed query then retrieves an intermediate output based on this summary; and (4) *output projection*, which integrates the outputs from multiple attention heads into a single representation. This step does not change the output dimension, but ensures it matches the model's hidden size. This design enables ASA to operate with a total complexity of $\mathcal{O}(nd^2 + ndm)$, significantly lower than the $\mathcal{O}(n^2d)$ cost of traditional attention mechanisms.

As shown in Figure 1, ASA follows three steps to reduce computational complexity while maintaining contextual expressiveness.

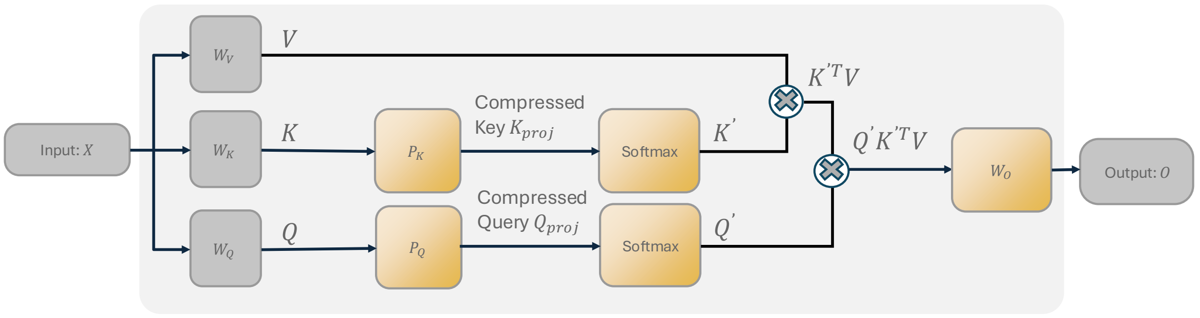

Figure 1: **Overall framework of ASA.** Starting from the input X, the model first computes query, key, and value (QKV) representations using the standard attention framework. To reduce downstreaming computational complexity and extract salient information, ASA applies low-dimensional linear projections to compress the QK representations. Independent Softmax transformations are then applied to the projected query and key, allowing the model to retain its ability to capture rich contextual dependencies. The transformed key is used to summarize the value matrix into a compact representation, which is then selectively weighted by the transformed query to produce the final output. This final step effectively retrieves the most relevant information, making ASA both efficient and expressive.

Step 1: Low-Dimensional Projection. We first compute the standard query, key, and value matrices using learned weight parameters:

$$Q = xW_Q, \quad K = xW_K, \quad V = xW_V,$$

where $x \in \mathbb{R}^{n\times d}$ is the input sequence, and $W_Q, W_K, W_V \in \mathbb{R}^{d\times d}$. This results in $Q, K, V \in \mathbb{R}^{n\times d}$. The complexity of each transformation is $\mathcal{O}(nd^2)$. To reduce dimensionality, we apply learned projection matrices $P_Q,\ P_K \in \mathbb{R}^{d\times m}$ with $m < d$:

$$Q_{\text{proj}} = QP_Q, \quad K_{\text{proj}} = KP_K,$$

resulting in compressed representations $Q_{\text{proj}}, K_{\text{proj}} \in \mathbb{R}^{n\times m}$. This step incurs a cost of $\mathcal{O}(ndm)$.

Step 2: Independent Softmax-Based Weighting. We then apply Softmax transformations independently to the projected query and key:

$$Q' = \text{softmax}(Q_{\text{proj}}), \quad K' = \text{softmax}(K_{\text{proj}}),$$

where the Softmax is applied row-wise. The resulting $Q', K' \in \mathbb{R}^{n\times m}$ act as probability-like representations that determine attention focus across compressed dimensions. The cost of this step is $\mathcal{O}(nm)$, which is negligible compared to matrix multiplications.

Step 3: Two-Stage Attention Process. In the first stage, we use the transformed key to summarize the value matrix:

$$V' = K'^{\top} V, \quad V' \in \mathbb{R}^{m\times d},$$

with a cost of $\mathcal{O}(ndm)$. This produces a compact representation V' that aggregates important value information. In the second stage, we compute the output by weighting V' with the transformed query:

$$O_{\text{inter}} = Q'V', \quad O_{\text{inter}} \in \mathbb{R}^{n\times d},$$

which also costs $\mathcal{O}(ndm)$.

Step 4: Output Projection. As in standard multi-head attention, we apply a final learned projection $W_O \in \mathbb{R}^{d\times d}$ to integrate the output:

$$O = O_{\text{inter}} W_O, \quad O \in \mathbb{R}^{n\times d},$$

which brings the attended representation back into the original feature space. This step has a complexity of $\mathcal{O}(nd^2)$ and serves to (i) fuse information across attention heads (in the multi-head setup) and (ii) allow for additional flexibility in transforming the final attended sequence. Note that this output projection is a standard component of the Transformer architecture and is commonly applied after concatenating outputs from multiple heads.

Overall Complexity. Assuming $m < d$ and $d \leq n$, the total complexity is:

$$\mathcal{O}(nd^2) + \mathcal{O}(ndm),$$

which is significantly lower than the standard attention complexity of $\mathcal{O}(n^2 d)$. For small values of m, the cost approaches $\mathcal{O}(nd^2)$, offering substantial computational savings.

Algorithm 1 ASA Algorithm

```
1: Require: Input sequence $x \in \mathbb{R}^{n\times d}$, projection matrices $P_Q, P_K \in \mathbb{R}^{d\times m}$, transformation matrices $W_Q, W_K, W_V, W_O \in \mathbb{R}^{d\times d}$, and dimensions $n, d, m$ where $m < d$.
2: Initialization: Compute query, key, and value: $Q = xW_Q, K = xW_K, V = xW_V$.
3: for $t = 1$ to $T$ do
4:     $Q'_t \leftarrow \text{softmax}(Q_t P_Q)$
       // Update query matrix with projection and softmax
5:     $K'_t \leftarrow \text{softmax}(K_t P_K)$
       // Update key matrix with projection and softmax
6:     $V'_t \leftarrow K'_t V_t$
       // Compute compact value
7:     $O_{\text{inter},t} \leftarrow Q'_t V'_t$
       // Compute intermediate output
8:     $O_t \leftarrow O_{\text{inter},t} W_O$
       // Attention Output
9: end for
10: Return $O_T \in \mathbb{R}^{n\times d}$.
```

ASA reduces the computational overhead of traditional Softmax attention by compressing query, key representations into a lower-dimensional space using linear projections with dimension $m < d$. This allows the model to focus on the most informative aspects of each component while avoiding dense pairwise token interactions. By applying independent Softmax operations to the projected query and key, ASA maintains the nonlinearity necessary for expressive contextual modeling. The resulting probability-like representations are used in a two-stage retrieval process: first summarizing the value matrix, then selectively extracting relevant information with the transformed query. This design preserves the strengths of Softmax-based attention while reducing the overall complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nd^2 + ndm)$. The full attention computation procedure is detailed in Algorithm 1.

3 Experiments

3.1 Experimental Setup

In our experiments, we assess the effectiveness of ASA within the language modeling setting. We compare its performance against two categories of models: (i) a strong Transformer baseline incorporating state-of-the-art architectural enhancements, and (ii) recent attention mechanisms optimized for linear-time complexity. Owing to limited computational resources, all baseline results are taken directly from Zhang *et al.* [2024]. Accordingly, we follow their experimental setup, including data splits, optimizer configurations, and evaluation metrics.

Baselines

We benchmark ASA against four reference models: Transformer++ Touvron *et al.* [2023], RetNet Sun *et al.* [2023], Mamba Gu and Dao [2024], and GLA Zhang *et al.* [2024]. Transformer++ is built upon the LLaMA architecture and incorporates Rotary Positional Embeddings Su *et al.* [2023], SWiGLU Shazeer [2020], and RMSNorm Zhang and Sennrich [2019]. For a fair comparison, we replace RetNet's original feed-forward network (FFN) with SWiGLU. Mamba is evaluated using its official publicly available implementation. To ensure fairness, we strictly follow the experimental protocol reported by Zhang *et al.* [2024] and include all relevant baselines without modification.

Training details

Following the training protocol of Zhang *et al.* [2024], we train ASA at the model scale of 1.3B parameters. All models are trained on the SlimPajama dataset Soboleva *et al.* [2023], tokenized using the Mistral tokenizer Jiang *et al.* [2023]. While the full dataset contains 627 billion tokens, we use a 100 billion subset for our experiments.

For optimization, we use AdamW Loshchilov and Hutter [2019] with a maximum learning rate of 3e-4, and adopt a cosine learning rate schedule. The 1.3B model is trained on the full 100 billion tokens with a batch size of 2 million tokens. The warmup period is set to 1 billion tokens for the 1.3B model. In addition, the initial and final learning rates are set to 3e-5. We also apply a weight decay of 0.01 during training.

The 1.3B model adopts a standard decoder-only Transformer architecture with 24 layers, a hidden size of 2048, and 4 attention heads. Each layer uses the Swish activation function and a feedforward network expansion ratio of 4, resulting in an intermediate MLP size of 8192. We apply layer normalization with an ϵ value of 1e-6 across all layers. This architecture design balances model expressiveness and training stability, and aligns with recent high-performing models in the literature.

Evaluation Metrics

We evaluate the models through two tasks: commonsense reasoning tasks and recall-intensive tasks.

In addition to reporting perplexity (PPL) on WikiText (Wiki.), we evaluate ASA on a diverse suite of tasks centered on common-sense reasoning and question answering, following the evaluation protocol of Zhang *et al.* [2024]. The benchmark includes LAMBADA (LMB.) Paperno *et al.* [2016], PiQA Bisk *et al.* [2019], HellaSwag (Hella.) Zellers *et al.* [2019], WinoGrande (Wino.) Sakaguchi *et al.* [2019], as well as ARC-easy (ARC-e) and ARC-challenge (ARC-c) Clark *et al.* [2018]. Regarding to common-sense reasoning tasks, our evaluation metrics include: perplexity on Wiki. and LMB.; standard accuracy on LMB., PiQA, Wino., and ARC-e; and accuracy normalized by sequence length for Hella., and ARC-c. For perplexity, the lower the better. For standard accuracy and accuracy normalized by sequence length, the higher the better.

Regarding to recall-intensive tasks, We evaluate the zero-shot in-context learning performance on recall-intensive tasks, as used in Zhang *et al.* [2024]. Specifically, we assess information retrieval on FDA Wu *et al.* [2021] and SWDE Hao *et al.* [2011], which are designed to evaluate retrieval from in-context passages scraped from HTML/PDFs. We also evaluate question answering on SQuAD Rajpurkar *et al.* [2018], NQ Kwiatkowski *et al.* [2019], TriviaQA Joshi *et al.* [2017], and Drop Dua *et al.* [2019], where models must ground their answers in in-context documents. For recall-intensive tasks such as FDA, SWDE, SQuAD, NQ, TriviaQA, and DROP, we report accuracy, following the protocol in Zhang *et al.* [2024], where exact match or F1 scores are normalized into a unified accuracy metric for consistent cross-task comparison.

All evaluations are conducted using the LM Evaluation Harness Gao *et al.* [2024].

3.2 Evaluation Results on Commonsense Reasoning Tasks

The experimental results on common-sense reasoning and question answering tasks are presented in Table 1. ASA consistently achieves the best performance across the model scale of 1.3B parameters, demonstrating superior efficiency and reasoning capability.

At the 1.3B scale, ASA achieves the lowest perplexity on both Wiki. (12.5) and LMB. (10.4), with a significant margin over the next-best models. In terms of task accuracy, ASA sets a new state-of-the-art among the compared methods, achieving 54.2 on LMB., 78.4 on PIQA, 64.5 on Wino. and 66.2 on ARC-e. Notably, ASA also delivers the highest scores on Hella. (59.4) and ARC-c (35.6), with its average accuracy reaching 59.7%, significantly surpassing all baselines. These improvements highlight ASA's strong scalability and ability to model complex long-range dependencies effectively.

Overall, the results confirm that ASA consistently outperforms both traditional Transformer-based models and recent linear attention variants. Its ability to retain Softmax-level expressiveness while reducing attention complexity allows it to excel across both language modeling and reasoning-intensive benchmarks.

3.3 Evaluation Results on Recall-Intensive Tasks

The zero-shot in-context learning results on recall-intensive tasks are shown in Table 2. ASA achieves the best overall performance at the 1.3B scale, demonstrating strong retrieval and grounding capabilities across a diverse set of benchmarks.

Specifically, ASA attains the highest scores on both FDA (52.6) and SWDE (38.5), outperforming prior models by a substantial margin on these HTML/PDF-based information retrieval tasks. On question answering benchmarks, ASA consistently leads with 49.1 on SQuAD, 36.7 on NQ, 75.2 on TriviaQA, and 28.8 on Drop, significantly surpassing all subquadratic baselines. Its average score reaches 46.8, marking a notable improvement over existing linear and structured attention methods.

These results highlight ASA's ability to combine efficient subquadratic attention with high expressiveness, making it particularly effective for tasks requiring precise retrieval and reasoning over in-context documents.

3.4 Ablation Studies

We conduct a small-scale ablation study on the 340M-parameter ASA model trained on 7B tokens to investigate the impact of two key design components: (i) the use of the Softmax transformation, and (ii) the effect of varying head dimensions. In addition, we benchmark ASA's computational efficiency across a range of sequence lengths, observing substantial speedups over FlashAttention-2 at longer contexts. Results are summarized in Table 3 and Table 4.

1) Effect of Softmax Kernels. To examine the necessity of using Softmax transformations on both the projected query and key, we evaluate ASA variants with either one or no Softmax transformation applied. The results show that removing Softmax entirely leads to a substantial drop in performance (PPL 25.47), while using only one Softmax transformation on Q or K moderately improves results (PPL 20.63 and 17.98 separately), but still lags behind the default setting. This confirms that applying Softmax independently to both query and key projections is critical for maintaining the probabilistic selectivity and contextual sensitivity that ASA relies on.

2) Impact of Head Dimension. We vary the number of attention heads to examine the effect of head dimension on performance. ASA uses 8 heads by default. Increasing the number of heads to 16—which reduces the dimension per head—leads to a rise in perplexity (PPL 15.93), likely due to insufficient capacity within each head. Conversely, reducing the number of heads to 4 results in a slight improvement in performance (PPL 14.32). These results suggest that fewer heads with larger dimensions may offer marginal gains in accuracy. Practical deployment requires a balance between performance and resource efficiency.

Model variants	**Training ppl.**
ASA MODEL (8 heads)	15.69
No Softmax	25.47
One Softmax on Q	20.63
One Softmax on K	17.98
Small head dimension (16 heads)	15.93
Large head dimension (4 head)	14.32

Table 3: **Ablation study results on the 340M ASA model trained for 7B tokens.** We evaluate the model variants via the average perplexity of the last 200 training steps.

3) Efficiency vs. Sequence Length. To empirically evaluate the computational efficiency of our proposed ASA mechanism, we benchmark its **forward** and **forward + backward** pass runtimes across varying sequence lengths using Triton implementations. All experiments are conducted on a single A100 80G GPU, using float16 precision, batch size 8, and hidden dimension $d = 128$. The results, summarized in Table 4, show that ASA exhibits subquadratic scaling with respect to sequence length.

Crucially, ASA demonstrates clear runtime advantages over FlashAttention-2 at longer sequences. For instance, at $n = 16{,}384$, ASA achieves nearly a $5\times$ speedup in the *forward + backward* pass, and more than a $9\times$ speedup in the *forward* pass alone. These improvements confirm ASA's theoretical complexity reduction of $\mathcal{O}(nd^2+ndm)$ and highlight its suitability for long-context modeling. ASA offers substantial efficiency gains both during training and inference, without compromising model expressiveness.

Sequence Length (n)	**ASA Forward (s)**	**ASA Fwd+Bwd (s)**	**Flash Forward (s)**	**Flash Fwd+Bwd (s)**
128	0.000316	0.001135	0.000083	0.000321
256	0.000230	0.001174	0.000094	0.000808
512	0.000363	0.001659	0.000098	0.000857
1024	0.000301	0.001547	0.000157	0.001161
2048	0.000383	0.001992	0.000341	0.001956
4096	0.000538	0.003726	0.001005	0.004814
8192	0.001211	0.006162	0.003625	0.015024
16384	0.001472	0.011251	0.013710	0.054337

Table 4: Runtime (in seconds) of ASA attention and FlashAttention-2 on different sequence lengths using Triton.

4 Conclusion

In this work, we introduced **Adaptive Subquadratic Attention (ASA)**, a novel attention mechanism that achieves a compelling trade-off between computational efficiency and expressive power. ASA restructures the traditional attention pipeline into a three-step process: (1) low-dimensional linear projection of QKV representations, (2) independent Softmax-based weighting to preserve selectivity, and (3) a two-stage attention composition that enables efficient and structured information retrieval. This design reduces attention complexity to $\mathcal{O}(nd^2 + ndm)$ while preserving the core strengths of Softmax attention, including probabilistic weighting and contextual sensitivity. Empirical results across a range of NLP benchmarks demonstrate that ASA outperforms both Softmax- and linear-based attention mechanisms, highlighting its effectiveness in balancing performance and efficiency. We believe ASA offers a general and practical solution for scaling attention in modern deep learning architectures.

Scale	Model	Wiki. ppl ↓	LMB. ppl ↓	LMB. acc ↑	PIQA acc ↑	Hella. acc_norm ↑	Wino. acc ↑	ARC-e acc ↑	ARC-c acc_norm ↑	Avg. ↑
1.3B Params	Transformer++	17.1	15.3	47.0	70.3	49.3	54.9	54.1	27.1	50.5
100B Tokens	RetNet	17.3	15.4	44.6	71.7	50.3	51.8	57.4	27.9	50.6
	Mamba	17.3	15.4	44.4	71.8	50.3	52.3	57.1	28.2	50.7
	GLA	17.6	15.4	46.4	69.9	49.0	54.0	55.4	27.7	50.4
	ASA	**12.5**	**10.4**	**54.2**	**78.4**	**59.4**	**64.5**	**66.2**	**35.6**	**59.7**

Table 1: **ASA Performance Compared to Transformer++, RetNet, Mamba, and GLA (1.3B Models).** All models are trained on the same subset of the SlimPajama dataset with 100B tokens, using the Mistral tokenizer. Task performance is evaluated in a zero-shot setting. The final column presents the average performance across all benchmarks, using (normalized) accuracy as the evaluation metric.

Model	FDA	SWDE	SQuAD	NQ	TriviaQA	Drop	Avg.
Transformer++	46.0	29.2	41.0	24.8	58.8	21.3	36.9
Mamba	13.9	25.4	33.2	18.5	53.5	21.7	27.7
GLA	26.7	30.6	34.8	21.5	56.0	19.1	31.4
HGRN2	9.9	23.1	32.0	16.4	55.2	19.1	25.9
ASA	**52.6**	**38.5**	**49.1**	**36.7**	**75.2**	**28.8**	**46.8**

Table 2: **ASA Performance on recall-intensive tasks.** We use 1.3B model with 24 layers and a hidden dimension of 2048 and train on 100B tokens.

References

Joshua Ainslie, Santiago Ontañón, Chris Alberti, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Harm Zen, Harm de Vries. Etc: Encoding long and structured inputs in transformers. *arXiv preprint arXiv:2004.08483*, 2020.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Krzysztof Choromanski, Valentin Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, and David Belanger. Rethinking attention with performers. *International Conference on Learning Representations (ICLR)*, 2021.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL-HLT*, 1:4171–4186, 2019.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019.

Qihang Fan, Huaibo Huang, and Ran He. Breaking the low-rank dilemma of linear attention. *arXiv preprint arXiv:2411.07635*, 2025.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *Proceedings of the Conference on Learning Models (COLM)*, 2024.

Dongchen Han, Tianzhu Ye, Yizeng Han, Zhuofan Xia, Siyuan Pan, Pengfei Wan, Shiji Song, and Gao Huang. Agent attention: On the integration of softmax and linear attention. *European Conference on Computer Vision*, 2024.

Qiang Hao, Rui Cai, Yanwei Pang, and Lei Zhang. From one tree to a forest: A unified solution for structured web data extraction. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 775–784, Beijing, China, July 25-29 2011. ACM.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b: A high-performance 7-billion-parameter language model. *arXiv preprint arXiv:2310.06825*, 2023.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, Enwei Jiao, Gengxin Li, Guojun Zhang, Haohai Sun, Houze Dong, Jiadai Zhu, Jiaqi Zhuang, Jiayuan Song, Jin Zhu, Jingtao Han, Jingyang Li, Junbin Xie, Junhao Xu, Junjie Yan, Kaishun Zhang, Kecheng Xiao, Kexi Kang, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Zheng, Linbo Chai, Long Xing, Meizhi Ju, Mingyuan Chi, Mozhi Zhang, Peikai Huang, Pengcheng Niu, Pengfei Li, Pengyu Zhao, Qi Yang, Qidi Xu, Qiexiang Wang, Qin Wang, Qiuhui Li, Ruitao Leng, Shengmin Shi, Shuqi Yu, Sichen Li, Songquan Zhu, Tao Huang, Tianrun Liang, Weigao Sun, Weixuan Sun, Weiyu Cheng, Wenkai Li, Xiangjun Song, Xiao Su, Xiaodong Han, Xinjie Zhang, Xinzhu Hou, Xu Min, Xun Zou, Xuyang Shen, Yan Gong, Yingjie Zhu, Yipeng Zhou, Yiran Zhong, Yongyi Hu, Yuanxiang Fan, Yue Yu, Yufeng Yang, Yuhao Li, Yunan Huang, Yunji Li, Yunpeng Huang, Yunzhi Xu, Yuxin Mao, Zehan Li, Zekang Li, Zewei Tao, Zewen Ying, Zhaoyang Cong, Zhen Qin, Zhenhua Fan, Zhihang Yu, Zhuo Jiang, and Zijia Wu. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025. A technical report from MiniMax. The authors are listed in alphabetical order. We open-sourced our MiniMax-01 at https://github.com/MiniMax-AI.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context, 2016.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad, 2018.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Noam Shazeer. Glu variants improve transformer, 2020.

Zhiyuan Shen, Jiayu Wang, Chaoqun Fang, Yatao Lyu, Yang Zhang, and Nan Duan. Efficient attention: Attention with linear complexities. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021:3531–3539, 2021. Presented at CVPR.

Jerome Sieber, Carmen Amo Alonso, Alexandre Didier, Melanie N. Zeilinger, and Antonio Orvieto. Understanding the differences in foundation models: Attention, state space models, and recurrent neural networks, 2024.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://cerebras.ai/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpaja 2023.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Sinong Wang, Belinda Z Wei, Dale Schuurmans, and Quoc V Le. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Eric Wu, Kevin Wu, Roxana Daneshjou, David Ouyang, Daniel E Ho, and James Zou. How medical ai devices are evaluated: limitations and recommendations from an analysis of fda approvals. *Nature Medicine*, 27(4):582–584, 2021.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2024.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019.

Biao Zhang and Rico Sennrich. Root mean square layer normalization, 2019.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling. *Advances in Neural Information Processing Systems*, 37, 2024.

A Appendix

A.1 Related Work

Softmax Attention

Softmax attention has been the cornerstone of modern Transformer-based architectures since its introduction in the Transformer modelVaswani *et al.* [2017]. It has revolutionized Natural Language Processing (NLP) tasks by enabling models to capture rich contextual dependencies through pairwise interactions between tokens. Models such as BERTDevlin *et al.* [2019], GPT-3Brown *et al.* [2020], and T5Raffel *et al.* [2020] rely heavily on Softmax attention to achieve state-of-the-art performance in tasks like machine translation, question answering, and summarization. Despite its success, the quadratic computational complexity of Softmax attention ($\mathcal{O}(n^2)$) with respect to the sequence length limits its scalability for long sequences, making it computationally expensive for large-scale NLP tasks [Sieber *et al.*, 2024].

Linear Attention

To address the computational limitations of Softmax attention, researchers have proposed linear attention mechanisms that scale linearly with sequence length ($\mathcal{O}(n)$). Examples include LinformerWang *et al.* [2020], which approximates the attention matrix using low-rank factorization, PerformerChoromanski *et al.* [2021], which introduces kernel-based projections to approximate Softmax attention efficiently, and MambaGu and Dao [2024], which is an innovative linear-time sequence model that ingeniously combines Selective State Spaces (S4), short convolutions, and gated cross-attention mechanisms. Linear attention mechanisms reduce computational costs and memory usage, making them more suitable for processing long sequences. However, these methods often struggle to capture global dependencies and nuanced contextual relationships, which can affect performance in tasks requiring deep semantic understandingShen *et al.* [2021].

Hybrid Attention

Hybrid attention mechanisms aim to combine the strengths of Softmax and linear attention to balance expressiveness and computational efficiency. Models like LongformerBeltagy *et al.* [2020] and BigBirdZaheer *et al.* [2020] leverage sparse attention patterns, with Longformer combining local and global attention to handle long sequences, and BigBird employing a mix of sliding windows, random connections, and global tokens for efficient long-sequence processing. Similarly, ETCAinslie *et al.* [2020] uses sparse attention to process long inputs effectively. Beyond these, hybrid mechanisms have been integrated more dynamically within attention computation. For instance, Gated Slot Attention (GSA)Zhang *et al.* [2024] addresses recall-intensive tasks by enhancing memory capacity through a two-layer Gated Linear Attention design with adaptive forgetting. Agent AttentionHan *et al.* [2024] introduces additional tokens to aggregate and broadcast information, enabling efficient global context modeling with fewer tokens while unifying the strengths of Softmax and linear attention. Additionally, minimax-01Li *et al.* [2025] incorporates hybrid mechanisms at a structural level, combining lightning attention with Mixture of Experts to scale models to hundreds of billions of parameters, processing contexts up to 4 million tokens during inference. While these approaches offer impressive advancements, they often rely on task-specific designs or extensive hyperparameter tuning, which can limit their generalizability across diverse applications.

B Additional Experimental Details for 340M Model

To further assess the effectiveness of our structured attention design, we conduct additional experiments using a 340M-parameter model. All baseline results are taken directly from Yang *et al.* [2024]. We follow their experimental setup, including data splits, optimizer configurations, and evaluation metrics.

Training is performed using AdamW [Loshchilov and Hutter, 2019] with a maximum learning rate of 3×10^{-4} and a cosine decay schedule. The warmup phase lasts for 0.5 billion tokens, and the initial and final learning rates are set to 3×10^{-5}. A total of 15 billion tokens are used for training with a batch size of 0.5 million tokens. We apply a weight decay of 0.01 and gradient clipping with a threshold of 1.0. The 340M model adopts a standard decoder-only Transformer architecture with 24 layers, a hidden size of 1024, and 4 attention heads. Each layer employs the Swish activation function and a feedforward network expansion ratio of 4. Layer normalization is applied with $\epsilon = 10^{-6}$.

Experimental results on common-sense reasoning and question answering tasks are summarized in Table 5. ASA achieves the lowest perplexity on both Wiki (24.73) and LMB (35.34), outperforming all baselines including Transformer++, ResNet, Mamba, and GLA. On downstream tasks, ASA also demonstrates strong performance, achieving the highest standard accuracy on LMB (38.5), PIQA (68.6), Wino (58.4), and ARC-e (54.2). It also surpasses all baselines on Hella (40.3) and ARC-c (29.4) in terms of normalized accuracy.

These results further highlight ASA's ability to provide not only better language modeling perplexity but also improved generalization in reasoning tasks, validating the advantages of its structured attention mechanism.

		Wiki.	**LMB.**	**LMB.**	**PIQA**	**Hella.**	**Wino.**	**ARC-e**	**ARC-c**	**Avg.**
Scale	**Model**	ppl ↓	ppl ↓	acc ↑	acc ↑	acc_norm ↑	acc ↑	acc ↑	acc_norm ↑	↑
340M Params	Transformer++	28.39	42.69	31.0	63.3	34.0	50.4	44.5	24.2	41.2
15B Tokens	RetNet	32.33	49.19	28.6	63.5	33.5	52.5	44.5	23.4	41.0
	Mamba	28.39	39.66	30.6	65.0	35.4	50.1	46.3	23.6	41.8
	GLA	28.65	43.35	30.3	64.8	34.5	51.4	45.1	22.7	41.5
	ASA	**24.73**	**35.34**	**38.5**	**68.6**	**40.3**	**58.4**	**54.2**	**29.4**	**48.2**

Table 5: **ASA Performance Compared to Transformer++, RetNet, Mamba, and GLA.** All models are trained on the same subset of the SlimPajama dataset, utilizing the Mistral tokenizer. The 340M is trained on 15 billion tokens, respectively. Task performance is evaluated in a zero-shot setting. The final column presents the average performance across all benchmarks, using (normalized) accuracy as the evaluation metric.