

# Sparse Growing Transformer: Training-Time Sparse Depth Allocation via Progressive Attention Looping

Yao Chen<sup>1,2\*</sup>, Yilong Chen<sup>1,2\*</sup>, Yinqi Yang<sup>3‡</sup>, Junyuan Shang<sup>3</sup>, Zhenyu Zhang<sup>3</sup>,  
Zefeng Zhang<sup>1,2</sup>, Shuaiyi Nie<sup>1,2</sup>, Shuohuan Wang<sup>3</sup>,  
Yu Sun<sup>3</sup>, Hua Wu<sup>3</sup>, HaiFeng Wang<sup>3</sup>, Tingwen Liu<sup>1,2†</sup>

<sup>1</sup> Institute of Information Engineering, Chinese Academy of Sciences

<sup>2</sup> School of Cyber Security, University of Chinese Academy of Sciences

<sup>3</sup> Baidu Inc.

{chenyao2023, chenylong, liutingwen}@iie.ac.cn

## Abstract

Existing approaches to increasing the effective depth of Transformers predominantly rely on parameter reuse, extending computation through recursive execution. Under this paradigm, the network structure remains static along the training timeline, and additional computational depth is uniformly assigned to entire blocks at the parameter level. This rigidity across training time and parameter space leads to substantial computational redundancy during training. In contrast, we argue that depth allocation during training should not be a static preset, but rather a progressively growing structural process. Our systematic analysis reveals a deep-to-shallow maturation trajectory across layers, where high-entropy attention heads play a crucial role in semantic integration. Motivated by this observation, we introduce the Sparse Growing Transformer (SGT). SGT is a training-time sparse depth allocation framework that progressively extends recurrence from deeper to shallower layers via targeted attention looping on informative heads. This mechanism induces structural sparsity by selectively increasing depth only for a small subset of parameters as training evolves. Extensive experiments across multiple parameter scales demonstrate that SGT consistently outperforms training-time static block-level looping baselines under comparable settings, while reducing the additional training FLOPs overhead from approximately 16–20% to only 1–3% relative to a standard Transformer backbone.

## 1 Introduction

Large language models (LLMs) have demonstrated exceptional knowledge integration and reasoning capabilities (OpenAI, 2023; Zhu et al., 2024). This

\* Equal contribution. † Corresponding author. ‡ Project lead. Our code is available at <https://github.com/YaoChen0203/Sparse-Growing-Transformer>.

success is fundamentally underpinned by the scalability of the Transformer architecture, which supports substantial expansion in both model width and, more crucially, network depth (Vaswani et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022). For instance, leading models such as the LLaMA 3 and Qwen 3 series exhibit considerable depth across different parameter scales, ranging roughly from 30 to 120 layers (Team, 2024, 2025). Such depth is essential as it enables the learning of high-level hierarchical representations and is indispensable for the complex abstraction and compositional reasoning required for challenging tasks (Tenney et al., 2019; Chen and Zou, 2024). However, this increasing depth comes with a cost: stacking more layers inevitably expands the parameter count and memory footprint, leading to greater GPU memory consumption and deployment latency (Hsieh et al., 2023).

To decouple model depth from parameter scale, recent studies have attempted to introduce recurrence into Transformer architectures, extending the effective computational depth to enable multi-step latent reasoning without inflating parameters (Fan et al., 2024; Chen et al., 2025c; Geiping et al., 2025). However, existing recursive architectures adopt a training-time static topology and uniformly apply block-level reuse at the parameter level, lacking fine-grained differentiation of parameter roles (see Appendix A for a detailed related work discussion). This rigidity across training time and parameter space leads to a substantial and unnecessary increase in training computational cost. Such coarse-grained recurrence contrasts with biological neural circuits, where autaptic self-connections emerge selectively within specific neuronal populations as evolutionarily and developmentally specialized microcircuit motifs, rather than being uniformly instantiated across all neurons (Bacci et al., 2003; Jiang et al., 2015; Pan et al., 2025). Guided by this perspective, we introduce a new paradigm

of *Training-Time Structural Sparsity*, in which model depth progressively self-grows during training through selective allocation of additional computation to fine-grained subsets of parameters.

To instantiate this paradigm during training, we require a reliable signal to guide structural growth. Recent studies have revealed a clear division of labor within Transformers: Self-Attention Modules drive dynamic in-context reasoning, while Feed-Forward Networks predominantly encode static statistical regularities (Geva et al., 2021; Olsson et al., 2022; Chen et al., 2025a). Attention entropy, a metric quantifying the reasoning uncertainty and information content of attention heads, has been shown to correlate closely with contextual integration capabilities (Zhang et al., 2025b). Naturally, attention entropy can serve as a critical indicator. We further investigate the functional characteristics and training dynamics of attention entropy. Our analysis reveals two key findings: (1) Functionally, high-entropy heads act as pivotal hubs for semantic integration rather than noise; (2) Dynamically, layers follow a *deep-to-shallow* maturation trajectory, where deeper layers differentiate earlier in training while shallower layers evolve more gradually. Guided by these findings, we propose the **Sparse Growing Transformer (SGT)**, a self-growing depth architecture via progressive attention looping that realizes training-time structural sparsity. Specifically, SGT establishes a progressive evolution aligned with the maturation trajectory: recurrence is first activated in early-differentiating deeper layers and then gradually extended toward shallower ones, while within active layers, recursive looping is selectively applied to high-entropy heads to concentrate computation on critical semantic units.

Our contributions are summarized as follows: **1)** We conduct a systematic analysis of attention entropy from both functional and dynamic perspectives. Our findings show that high-entropy heads serve as pivotal hubs for semantic integration, and that layers follow a depth-dependent evolutionary trajectory characterized by a *deep-to-shallow* maturation pattern. **2)** We formalize the paradigm of *Training-Time Structural Sparsity* and instantiate it through the **Sparse Growing Transformer (SGT)**, which progressively allocates recurrent computation to high-entropy heads via fine-grained structural growth. We further provide a theoretical analysis demonstrating that concentrating recurrence on high-entropy components accelerates

convergence. **3)** Extensive experiments across multiple parameter scales demonstrate that SGT consistently outperforms training-time static block-level looping baselines under comparable settings, while reducing the additional training FLOPs overhead from approximately 16–20% to only 1–3% relative to a standard Transformer backbone.

## 2 Preliminaries

To facilitate the understanding of the empirical observations in Section 3 and the formalization of our proposed method in Section 4, we establish the notation for the standard attention mechanism and introduce the definition of attention entropy.

**Multi-Head Attention.** We formulate the input hidden states as  $H \in \mathbb{R}^{N \times d}$ , where  $N$  is the sequence length and  $d$  is the model dimension. We focus on the Multi-Head Attention (MHA) module, which processes  $H$  through multiple parallel attention heads. For the  $i$ -th head, the input is projected into queries  $Q^{(i)}$ , keys  $K^{(i)}$ , and values  $V^{(i)}$  using parameter matrices  $W_Q^{(i)}, W_K^{(i)}, W_V^{(i)} \in \mathbb{R}^{d \times d_h}$ , where  $d_h$  is the head dimension. The attention matrix  $A^{(i)} \in \mathbb{R}^{N \times N}$  is computed as:

$$A^{(i)} = \text{softmax} \left( \frac{Q^{(i)}(K^{(i)})^\top}{\sqrt{d_h}} \right). \quad (1)$$

The output of a single head is then given by projecting the weighted values:

$$\text{Attn}^{(i)}(H) = (A^{(i)} H W_V^{(i)}) W_O^{(i)}, \quad (2)$$

where  $W_O^{(i)} \in \mathbb{R}^{d_h \times d}$  represents the output projection matrix of the  $i$ -th head. The outputs from all parallel heads are summed and then added to the input  $H$  via a residual connection to produce the post-attention hidden state  $H_{\text{post}}$ :

$$H_{\text{post}} = H + \sum_{i=1}^{N_{\text{head}}} \text{Attn}^{(i)}(H), \quad (3)$$

where  $N_{\text{head}}$  denotes the number of attention heads per layer. This resulting representation  $H_{\text{post}}$  is subsequently processed by the Feed-Forward Network.

**Attention Entropy.** Attention entropy serves as a metric for quantifying the concentration and uncertainty of information flow. Considering computational efficiency and the fact that the final token aggregates information from the entire preceding context (Appendix B), we focus on the attention

entropy of the final token. To eliminate the bias introduced by sequence lengths, we utilize the *length-normalized attention entropy*. Formally, for the  $i$ -th head, this is defined as:

$$\mathcal{E}^{(i)} = -\frac{1}{\log N} \sum_{j=1}^N A_{N,j}^{(i)} \log A_{N,j}^{(i)}, \quad (4)$$

where  $\log N$  serves as the length normalization term, and  $A_{N,j}^{(i)}$  denotes the attention weight in the  $i$ -th head assigned by the  $N$ -th token to the  $j$ -th input token, with the resulting metric  $\mathcal{E}^{(i)}$  lying in the interval  $[0, 1]$ . Based on this, we further define the *layer-wise attention entropy* to capture the uncertainty of the  $l$ -th layer. This is computed by averaging the normalized entropies across all  $N_{head}$  heads within the  $l$ -th layer:

$$\bar{\mathcal{E}}^{(l)} = \frac{1}{N_{head}} \sum_{i=1}^{N_{head}} \mathcal{E}^{(l,i)}. \quad (5)$$

In the following section, we use  $\mathcal{E}^{(i)}$  and  $\bar{\mathcal{E}}^{(l)}$  as the analytical lens to investigate attention entropy, providing the insights for the design of SGT.

### 3 Observations

By visualizing attention entropy across representative open-source models, we observe that *attention entropy is closely correlated with the model architecture*. Specifically, this correlation manifests as a universal *three-phase* progression at the layer level (Figure 7) and highly consistent head-level patterns across varying parameter scales with identical layer-head configurations (Figure 8), establishing attention entropy as a stable architectural attribute. Building upon this foundation, we further investigate the functional characteristics and training dynamics of attention entropy.

**Observation 1. High-entropy heads serve as information-dense hubs critical for global reasoning.** We identified heads in the Qwen3-0.6B model that consistently exhibit high or low entropy across 11 diverse tasks, and then examined these groups (details in Appendix C.3.1). Qualitatively, low-entropy heads predominantly exhibit attention sink behavior, focusing on local tokens. In contrast, high-entropy heads distribute attention selectively across globally distributed tokens rather than uniformly, capturing critical evidence for reasoning (Figure 1, 12 and 13). Quantitatively, head masking experiments demonstrate that disabling these

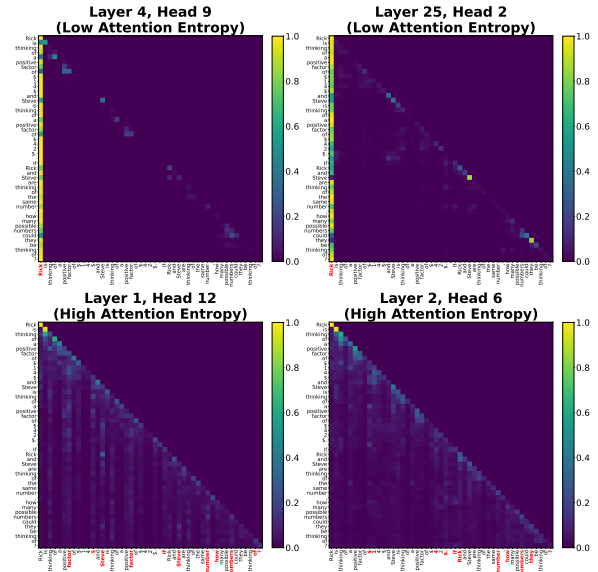


Figure 1: Visualization of attention heatmap in low- and high-entropy heads from Qwen3-0.6B. Along the horizontal axis, tokens highlighted in red denote the subset that receives the top 50% of the attention from the final query position (details in Appendix C.3.2).

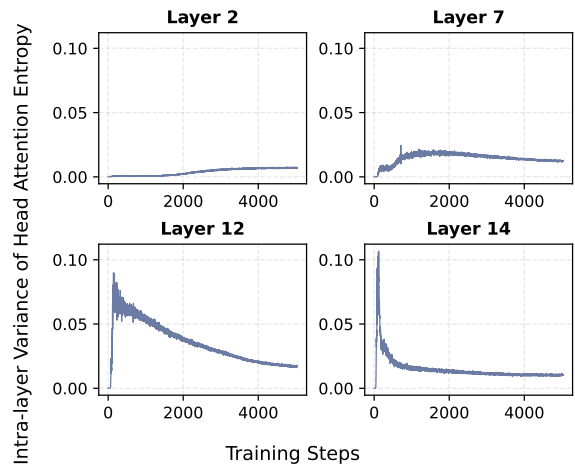


Figure 2: Evolution of Intra-layer Variance in Head-wise Attention Entropy Across Training Steps for Models Pretrained from Scratch (we visualize four representative layers; see Figure 14b for detailed results)

high-entropy heads results in the most substantial performance degradation (Table 7). Based on these findings, we select high-entropy heads as the targets for allocating additional computational depth to facilitate iterative *inner thinking*.

**Observation 2. Attention entropy reveals a deep-to-shallow maturation trajectory: deeper layers differentiate early in training, while shallower layers evolve more gradually.** To investigate the dynamic properties of attention entropy, we train

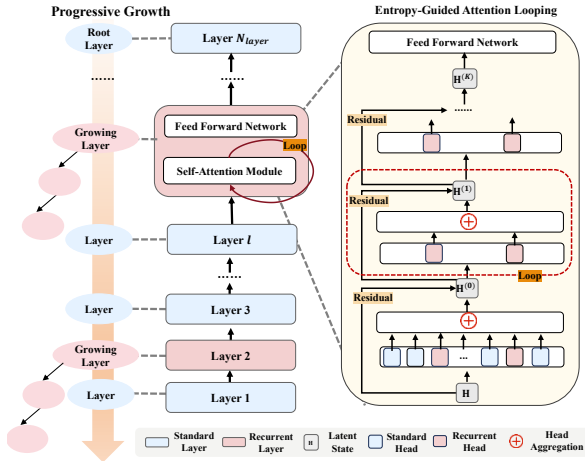


Figure 3: The Sparse Growing Transformer (SGT) architecture. It implements Training-Phase Structural Sparsity via a deep-to-shallow progressive growth schedule (Left) and selective high-entropy head looping (Right).

a 573M LLaMA model from scratch and monitor head-wise attention entropy across all layers (Figure 14a). We observe a general downward trend in entropy accompanied by significant discrepancy, which we quantify using intra-layer entropy variance (Figure 2 and Figure 14b). Specifically, for intra-layer entropy variance, deep layers exhibit a rapid surge followed by a decline, whereas shallow layers show a steady increase. Interpreting such variance as a measure of functional differentiation (Observation 1), we find that deep-layer heads differentiate earlier, while shallow-layer heads evolve over a longer horizon. These depth-dependent evolutionary patterns serve as the primary motivation for our progressive growth strategy.

## 4 Method

Building upon our above observations, we propose the Sparse Growing Transformer (SGT), a self-growing depth architecture via progressive attention looping (Figure 3). Specifically, SGT progressively grows fine-grained recurrent micro-circuits during training, selectively allocating additional depth to high-value parameters within a block rather than uniformly reapplying entire layers. This growth unfolds along the *deep-to-shallow* maturation trajectory and concentrates computation on a subset of functional components, thereby inducing training-time structural sparsity and reducing substantial pretraining overhead. This section details the SGT mechanism composed of *Entropy-Guided Attention Looping* and *Progressive Growth Training*.

### 4.1 Entropy-Guided Attention Looping

Guided by our Observation 1 that high-entropy heads serve as information-dense hubs critical for global reasoning, we identify them as the optimal candidates for allocating additional computational depth, rather than indiscriminately applying recurrence to entire blocks. Formally, following the notation in Section 2, we denote the post-attention hidden state  $H_{\text{post}}$  as  $H^{(0)}$ , which serves as the initial state for the looping mechanism. Subsequently, we apply a selective head-level recursive loop defined by an efficient *allocation operator*  $\mathcal{A}$ , which maps the layer’s attention entropy distribution to a specific set of active heads  $\mathcal{S}$  for optimized information flow. Let  $\mathcal{S}$  be the set of active heads designated for looping, and let  $k$  denote the  $k$ -th loop step. For step  $k = 1, \dots, K$ , the recursive update is formulated as:

$$H^{(k)} = H^{(k-1)} + \sum_{i \in \mathcal{S}} \text{Attn}^{(i)}(H^{(k-1)}). \quad (6)$$

Specifically, for a given layer, we compute the length-normalized attention entropy  $\mathcal{E}^{(i)}$  (Eq. 4) for each head  $i$ . The allocation operator  $\mathcal{A}$  then identifies the top- $h$  heads exhibiting the highest entropy to form the active recursive set  $\mathcal{S}$ :

$$\mathcal{S} = \mathcal{A}(\{\mathcal{E}^{(i)}\}) = \arg \text{top}_h \left( \{\mathcal{E}^{(i)}\}_{i=1}^{N_{\text{head}}} \right). \quad (7)$$

Upon completion of the grown depth  $K$ , the final state  $H^{(K)}$  is passed to the Feed-Forward Network. Notably, we provide a theoretical justification in Appendix E, demonstrating that applying recursive processing to high-entropy heads accelerates convergence toward the quasi-stationary state in recursive dynamics. By concentrating computational resources exclusively on these high-uncertainty units, this mechanism enables the SGT to form fine-grained recurrent micro-circuits. The emergence and expansion of these structures along the training process are governed by a *Progressive Growth Training* strategy.

### 4.2 Progressive Growth Training

To operationalize this *Progressive Growth Training* strategy, we design a progressive growth schedule that aligns structural evolution with the model’s intrinsic maturation trajectory (Observation 2). By activating recurrent micro-circuits gradually over the training timeline rather than from the outset,

this design induces temporal sparsity during training and substantially reduces pretraining computational overhead.

This progressive growth mechanism is formalized through a growth operator  $\mathcal{O}$  that updates the looping configuration along training time. At training step  $t$ , the architectural state is defined as

$$\Theta_t := \{\mathcal{L}_t, l_t^*, \{K_l\}_{l \in \mathcal{L}_t}\}, \quad (8)$$

where  $\mathcal{L}_t$  is the set of active looping layers,  $l_t^*$  denotes the currently growing layer,  $K_l$  is the current loop depth for layer  $l \in \mathcal{L}_t$ . The growth operator  $\mathcal{O}$  updates  $\Theta_t$  based on the attention entropy  $\bar{\mathcal{E}}$

$$\Theta_{t+\Delta t} = \mathcal{O}(t, \bar{\mathcal{E}}; \Theta_t). \quad (9)$$

Initially, during the *Warm-up Phase* ( $t < t_{\text{start}}$ ), the model trains as a standard Transformer without structural growth. The warm-up period allows attention entropy to stabilize, providing a reliable signal for subsequent structural decisions. Upon entering the *Growing Phase* ( $t \geq t_{\text{start}}$ ), the growth operator  $\mathcal{O}$  periodically updates the architectural state  $\Theta_t$  every  $\Delta t$  steps. At each growth step, all layers are ranked to form a candidate pool  $\mathcal{C}$  according to their mean attention entropy

$$\mathcal{C} = \arg \operatorname{top}_L \left( \{\bar{\mathcal{E}}^{(l)}\} \right). \quad (10)$$

Following the *deep-to-shallow* maturation trajectory (Observation 2), structural expansion proceeds in an ordered manner. If the currently growing layer  $l^*$  remains among the candidate pool  $\mathcal{C}$  and has not reached the maximum depth  $K_{\text{max}}$ , its loop depth is updated as  $K_{l^*} \leftarrow K_{l^*} + 1$ . Otherwise, provided that the number of active looping layers has not yet reached  $L$ , a new layer is activated. The newly selected layer is the deepest eligible candidate not yet in  $\mathcal{L}_t$ , subject to the constraint that it lies deeper than the shallowest active looping layer:

$$l^* = \max\{l \mid l \in \mathcal{C}, l \notin \mathcal{L}_t, l < \min(\mathcal{L}_t)\}. \quad (11)$$

This rule enforces a monotonic deep-to-shallow expansion, anchoring growth in earlier-stabilized deeper layers before extending toward shallower ones, preserving training stability during structural expansion.

Once the target configuration is reached, the training enters the *Fixed Phase*, in which the architecture is frozen to allow full parameter convergence. The complete training procedure is pro-

vided in **Algorithm 1**. Through this staged evolution, computational depth becomes a progressively expanding yet temporally sparse structure, enabling SGT to gradually form fine-grained recurrent micro-circuits aligned with intrinsic entropy dynamics.

## 5 Experiments

### 5.1 Setup

**Data.** C4 (Raffel et al., 2020) is a large-scale, cleaned web text corpus widely used for language model pretraining. We train all model variants on a 20B-tokens subset of C4 and evaluate perplexity on a validation set constructed from the same corpus.

**Training.** All experiments are conducted using the OLMo open-source training framework (Groeneveld et al., 2024). We adopt the AdamW optimizer with a learning rate of  $6.0 \times 10^{-4}$  applied to all parameters. Models are trained with a sequence length of 4096 and a global batch size of 1024 for 5035 training steps. Training is performed on 8 NVIDIA A100 GPUs with 40 GB of memory each. All model scales (Table 8) adopt the LLaMA-style architecture and are pre-trained from scratch under the same training setup to ensure fair comparison.

**Evaluation.** Our experimental evaluation assesses model capabilities across a diverse set of Reasoning & Knowledge tasks. Specifically, We report accuracy on ARC-Easy (ARC-E) (Clark et al., 2018), SocialIQA (SIQA) (Welbl et al., 2017a), OpenBookQA (OBQA) (Mihaylov et al., 2018), SCIQ (Welbl et al., 2017b), WinoGrande (WG) (Sakaguchi et al., 2021), BasicArithmetic (Basic Arith) (Brown et al., 2020), CommonsenseQA (CSQA) (Talmor et al., 2019), HellaSwag (Hella.) (Zellers et al., 2019), and the four subtasks of MMLU (Hendrycks et al., 2021) (STEM, Social Sciences, Humanities, and Other).

**Baseline.** We compare the following methods across varying model scales. 1) *Vanilla*: The standard Transformer architecture without any recurrence mechanism, serving as the reference baseline. 2) *Block Loop*: A baseline implementing block-level recurrence by iteratively reusing entire Transformer blocks. 3) *Sparse Growing Transformer (SGT)*: Our proposed self-growing depth architecture utilizing entropy-guided attention looping. We configure it with  $h = 2, L = 3$  and evaluate three variants with  $K_{\text{max}} \in \{1, 2, 3\}$ . To ensure a fair

Model	FLOPs	Reasoning & Knowledge ( $\uparrow$ )								
		ARC-E	WG	SIQA	Hella.	OBQA	CSQA	BA	MMLU	Avg
<b>Vanilla (275M)</b>	42.83	44.56	49.57	41.30	30.89	27.40	29.24	22.70	24.80	33.81
<b>Block Loop</b>	51.50 (+20.24%)	44.74	50.36	41.10	<b>31.85</b>	27.60	<b>29.48</b>	21.80	25.38	34.04
<b>SGT(K=1)</b>	43.46 (1.47%)	45.09	51.46	41.40	31.21	<b>28.00</b>	29.16	24.07	25.19	34.48
<b>SGT(K=2)</b>	43.88 (+2.45%)	45.79	<b>52.33</b>	<b>42.38</b>	31.41	26.80	28.75	<b>24.47</b>	25.45	<b>34.64</b>
<b>SGT(K=3)</b>	44.21 (+3.22%)	<b>46.32</b>	50.91	41.81	31.43	27.40	28.83	22.37	<b>26.16</b>	34.40
<b>Vanilla (573M)</b>	87.41	47.01	50.35	41.86	35.24	<b>29.00</b>	29.24	24.03	26.37	35.39
<b>Block Loop</b>	101.76 (+16.42%)	48.07	50.19	42.27	<b>36.11</b>	28.00	30.30	24.90	26.68	35.82
<b>SGT(K=1)</b>	88.15 (+0.85%)	45.61	49.64	42.22	35.36	<b>29.00</b>	<b>31.29</b>	24.93	<b>27.42</b>	35.68
<b>SGT(K=2)</b>	88.60 (+1.36%)	47.37	50.20	<b>42.47</b>	35.42	28.88	28.50	24.80	27.10	35.59
<b>SGT(K=3)</b>	88.96 (+1.77%)	<b>50.00</b>	<b>53.43</b>	42.22	35.25	27.40	29.89	<b>26.07</b>	27.03	<b>36.41</b>
<b>Vanilla (1.2B)</b>	175.76	50.00	50.19	43.39	<b>41.02</b>	30.00	31.69	<b>26.60</b>	<b>27.76</b>	37.58
<b>Block Loop</b>	205.65 (+17.01%)	50.17	51.38	42.42	41.01	31.00	31.28	25.83	27.01	37.51
<b>SGT(K=1)</b>	177.06 (+0.74%)	50.88	51.78	<b>43.50</b>	40.30	30.00	31.36	25.43	27.65	37.61
<b>SGT(K=2)</b>	177.85 (+1.19%)	<b>51.05</b>	<b>52.49</b>	43.14	40.62	31.00	32.11	25.33	27.52	<b>37.91</b>
<b>SGT(K=3)</b>	178.48 (+1.55%)	49.30	51.54	43.04	40.51	<b>31.40</b>	<b>32.27</b>	25.80	26.92	37.60

Table 1: Main experimental results. FLOPs denote the total training compute, measured in units of  $10^{18}$ . The best results at each model scale are shown in **bold**.  $K$  denotes  $K_{\max}$ , the upper bound on the loop depth for the selected loop layers.

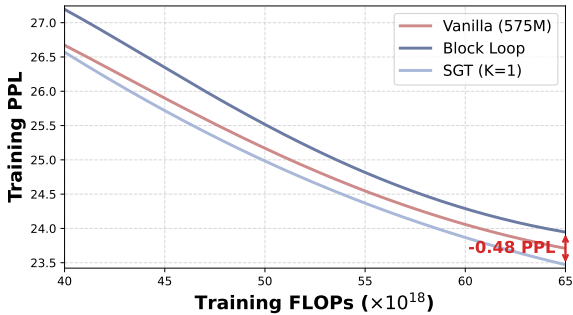


Figure 4: Comparative convergence trajectories of training perplexity (PPL) versus cumulative training FLOPs for the 573M model scale.

comparison, for *Block Loop*, we select the three layers exhibiting the highest entropy from a vanilla model pre-trained for  $t_{\text{start}}$  steps as the fixed looping layers (more details in Appendix D).

## 5.2 Main Results

**Improvements on Reasoning Tasks.** Notably, SGT achieves particularly pronounced improvements on reasoning-intensive benchmarks. On tasks such as WG, ARC-E, and CSQA, which require complex relational integration and multi-step inference, SGT demonstrates substantial gains of 3.24, 1.93, and 0.99 points, respectively, over the Block Loop baseline at the 573M scale. Crucially, SGT achieves these superior results with signif-

icantly higher efficiency: while Block Loop incurs a heavy +16.42% computational overhead, our method ( $K = 3$ ) surpasses Block Loop’s average performance by 0.59 points while increasing FLOPs by only +1.77%, which is approximately an order of magnitude less additional cost. These results corroborate that high-entropy heads play a pivotal role in semantic dependency construction, and that selectively deepening their computation enables more thorough inner thinking without the burden of redundant parameters.

**Training Efficiency and Convergence.** We evaluate training efficiency by comparing the convergence trajectories of different strategies relative to their computational consumption. As illustrated in Figure 4, when aligned by equivalent training FLOPs, SGT consistently maintains a lower training perplexity compared to both the Vanilla and Block Loop baselines. Specifically, at a cumulative compute of  $65 \times 10^{18}$  FLOPs, SGT achieves a PPL that is 0.48 points lower than the Vanilla baseline. In contrast, the Block Loop method exhibits higher perplexity than Vanilla at the same FLOP budget, indicating computational inefficiency. This demonstrates that by strategically allocating depth to high-entropy components, SGT not only improves final performance but also significantly accelerates convergence efficiency per unit of compute.

### 5.3 Ablation Studies

#### 5.3.1 Impact of Loop Components

To evaluate the specific contributions of distinct model components to iterative refinement, we conduct ablation experiments on the 573M model by applying loop operations at varying depths (Layer 1, 7, and 15). We compare four configurations: Block Loop, All Head Loop, High-Entropy Head Loop (top-2 highest), and Low-Entropy Head Loop (top-2 lowest). In all configurations, the loop operation is restricted to a single iteration on the designated layer to isolate component effects. Further details are provided in Appendix D.2.

**Targeted allocation of computational depth to high-entropy components yields the optimal efficiency-performance ratio.** As shown in Table 2, the High-Entropy Head Loop consistently achieves the most significant PPL reduction across all tested depths while incurring the smallest increase in FLOPs (+0.38%). Specifically, within the attention mechanism, looping the top-2 high-entropy heads significantly outperforms both looping all heads and looping low-entropy heads. These empirical findings are further corroborated by our theoretical analysis in Appendix E, which establishes that recursive dynamics on high-entropy matrices exhibit faster convergence due to their superior token-mixing properties.

**Block-level recurrence incurs computational redundancy due to a lack of fine-grained parameter discrimination.** Our results indicate that attention-level looping consistently outperforms or matches block-level looping regardless of layer depth. This empirical finding supports our hypothesis that block-level approaches lack deep insights into the granular roles of distinct parameters. By treating the Transformer block as a monolithic unit, these methods fail to isolate the components truly effective for iterative refinement. Indiscriminately applying recurrence to the entire block results in significant computational redundancy without yielding proportional performance gains.

#### 5.3.2 Impact of Growth Direction

**Performance Comparison.** Guided by Observation 2, which establishes that deeper layers differentiate and stabilize earlier, we design a Deep-to-Shallow (D2S) progressive growth strategy (Section 4.2). To validate this design choice, we compare our proposed D2S approach against a reversed

Loop Component	FLOPs ( $\times 10^{18}$ )	Perplexity $\downarrow$
Vanilla	87.41	23.973
<b>Layer 2</b>		
Block Loop	92.20 (+5.48%)	23.560 (-0.413)
Attention Loop	89.01 (+1.83%)	23.503 (-0.470)
High-Ent. Loop	87.74 (+0.38%)	<b>23.452</b> (-0.521)
Low-Ent. Loop	87.74 (+0.38%)	23.502 (-0.471)
<b>Layer 8</b>		
Block Loop	92.20 (+5.48%)	23.836 (-0.137)
Attention Loop	89.01 (+1.83%)	23.537 (-0.436)
High-Ent. Loop	87.74 (+0.38%)	<b>23.487</b> (-0.486)
Low-Ent. Loop	87.74 (+0.38%)	23.505 (-0.468)
<b>Layer 16</b>		
Block Loop	92.20 (+5.48%)	23.711 (-0.262)
Attention Loop	89.01 (+1.83%)	23.711 (-0.262)
High-Ent. Loop	87.74 (+0.38%)	<b>23.503</b> (-0.470)
Low-Ent. Loop	87.74 (+0.38%)	23.797 (-0.176)

Table 2: Ablation study of different loop components at varying layer depths. We report total training FLOPs (in units of  $10^{18}$ ) and perplexity (PPL) on the validation set, where lower is better. High-Ent. Loop denotes looping the top-2 highest-entropy heads, while Low-Ent. Loop denotes looping the top-2 lowest-entropy heads.

Method	CSQA	SIQA	SCIQ	MMLU <sub>STEM</sub>	Avg
Vanilla	29.24	41.86	71.00	26.28	42.10
S2D	30.39	41.40	71.80	27.40	42.75
D2S (Ours)	<b>31.29</b>	<b>42.22</b>	<b>72.00</b>	<b>28.46</b>	<b>43.49</b>

Table 3: Ablation study on growth direction. Experiments are conducted on the 573M model with SGT ( $K = 1$ ) configuration. **D2S** denotes deep-to-shallow growth, while **S2D** represents shallow-to-deep growth.

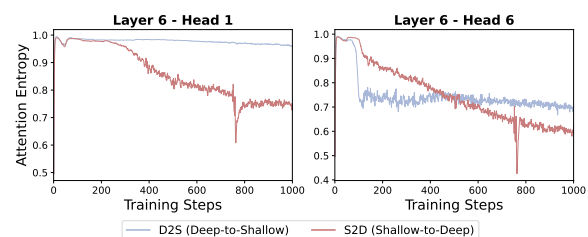


Figure 5: Attention entropy dynamics of two heads in Layer 6 during training, including the warm-up phase and layer selection phase, for models trained with D2S and S2D strategies (more visualizations in Figure 15).

Shallow-to-Deep (S2D) variant on the 573M model with SGT ( $K = 1$ ) configuration. The only modification in the S2D variant is the reversal of the layer selection order during the progressive growth

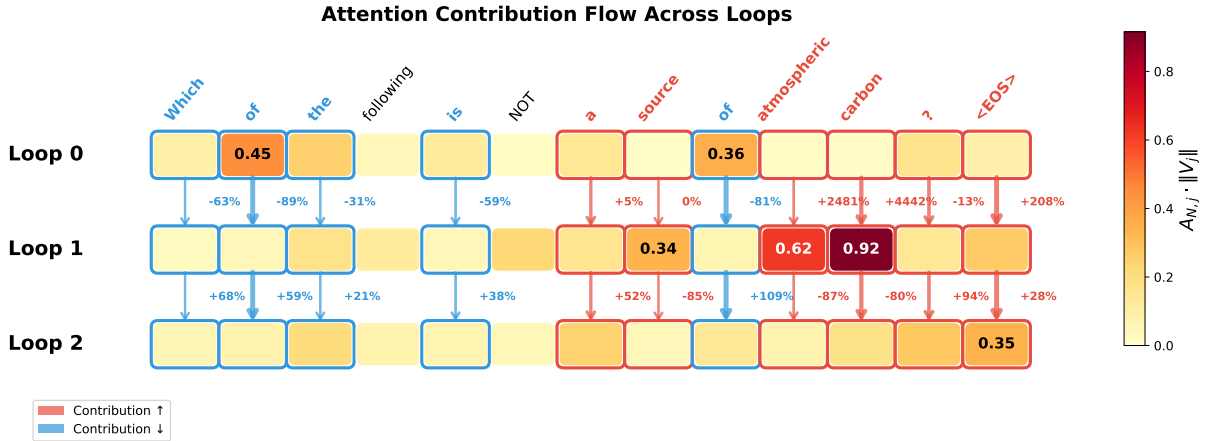


Figure 6: Attention contribution flow across loops in a high-entropy head (Layer 2, Head 15). Each cell shows the weighted attention contribution  $C_j = A_{N,j} \cdot \|V_j\|_2$  from the final token to position  $j$ , with color intensity indicating magnitude. Red and blue borders mark tokens with the largest contribution increase and decrease from Loop 0 to Loop 2, respectively. Arrows with percentage labels denote inter-loop changes. The visualization shows attention progressively shifting from syntactic elements toward task-critical and answer-relevant tokens.

Setting	Vanilla	Block Loop	High-Ent. Loop
1024×1	24.21	24.18 (−0.03)	24.16 (−0.05)
1024×2	57.38	58.80 (+1.42)	56.62 (−0.76)
1024×3	116.94	119.48 (+2.54)	111.66 (−5.28)
1024×4	180.70	175.05 (−5.65)	171.90 (−8.80)

Table 4: Long Context Extrapolation Results (PPL). *Setting* denotes the sequence length for extrapolation evaluation. The configurations for High-Ent. Loop and Block Loop follow the *Layer 2 settings* in Table 2.

phase. As presented in Table 3, the D2S strategy consistently outperforms the S2D variant across all evaluation metrics. This empirical evidence validates our hypothesis that initiating structural growth from naturally stable deeper layers is crucial for maximizing the performance of the SGT.

**Stability Analysis.** To further investigate the underlying causes of this performance disparity, we analyze the attention entropy dynamics during the dynamic layer selection phase. We observe that the S2D strategy exhibits frequent entropy spikes, defined as abrupt fluctuations across multiple attention heads, indicating severe training instability. For instance, Layer 6 of the S2D model displays the most pronounced volatility (Figure 5 and 15). In contrast, our D2S strategy maintains smooth entropy dynamics throughout the selection phase. This confirms that aligning the growth schedule with the model’s intrinsic maturation trajectory ensures training stability, providing a robust founda-

tion for progressive structural expansion.

## 5.4 Analysis

### Allocating computational depth to high-entropy heads enhances long-context generalization.

To further investigate generalization capabilities, we conduct long-context extrapolation experiments. We train models on sequences of length 1024 and evaluate them on progressively longer sequences (2048, 3072, and 4096 tokens), representing 2×, 3×, and 4× extrapolation relative to training length. Crucially, these evaluations are conducted directly on extended sequences without utilizing any positional embedding extrapolation techniques (e.g., YARN (Peng et al., 2024), base-scaling), providing a strict test of the model’s inherent extrapolation ability. As shown in Table 4, the High-Ent. Loop consistently outperforms the Vanilla baseline across all extrapolation settings, reducing PPL by 0.76, 5.28, and 8.80 points at 2×, 3×, and 4× extrapolation, respectively. In comparison, the Block Loop exhibits degradation at moderate extrapolation levels (2× and 3×), suggesting that indiscriminate block-level recurrence introduces noise that interferes with length generalization.

### High-Entropy Looping Enables Progressive Semantic Focusing.

To qualitatively analyze the latent *Inner Thinking* mechanism within high-entropy head looping, we visualize the attention dynamics of a head (Layer 2, Head 15) selected from SGT with  $K = 2$  loops. We quantify informa-

tion flow using Weighted Attention Contribution  $C_j = A_{N,j} \cdot \|V_j\|_2$ , where  $A_{N,j}$  denotes the attention weight from the final token to position  $j$ , and  $\|V_j\|_2$  is the value vector norm. This metric captures each token’s actual contribution to the output representation. Both Figure 6 and Figure 11 reveal a consistent refinement process where the SGT iteratively focuses its semantic lens. Initially, Loop 0 exhibits scattered attention dispersed across both task-critical content and general syntactic elements. The first recurrence (Loop 1) triggers a sharp semantic filtering effect, drastically amplifying focus on pivotal tokens, such as *risk* (+337%) and *individual* (+1080%) in the medical context, and *atmospheric* (+2481%) and *carbon* (+4442%) in the scientific context, while simultaneously suppressing broad syntactic words like *as*, *of*, and *the*. By the second recurrence (Loop 2), the attention further converges toward answer-relevant positions to finalize the semantic anchoring. In this stage, key tokens like *carbon* maintain high contributions while the model further eliminates residual attentional biases from the surrounding context. The transition from scattered attention to precise semantic anchoring demonstrates that SGT’s recursive mechanism enables *Inner Thinking* by iteratively correcting initial attentional biases and progressively refining semantic dependencies.

## 6 Conclusion

We establish attention entropy as a proxy for where added depth is valuable and uncover depth-dependent training dynamics. SGT then progressively loops high-entropy heads, which is theoretically shown to speed up convergence and empirically delivers stronger efficiency–performance and length generalization with interpretable inner thinking at minimal extra FLOPs.

## Limitations

The primary limitation of this work lies in the scale of the pre-training experiments. Due to computational resource constraints, our validation of SGT was conducted on models up to 1.2B parameters. While our analysis reveals consistent entropy patterns across the studied scales, confirming the scalability of the SGT framework to substantially larger foundation models remains an important direction for future work. In addition, we did not perform a fine-grained exploration of hyperparameter design, which may further improve the effi-

ciency–performance trade-off.

## Ethics Statement

Our work studies architectural modifications for large language models using publicly available benchmarks. No personal or sensitive data is used. The proposed method does not introduce new deployment risks beyond those commonly associated with language models.

## Acknowledgments

We would like to thank the anonymous reviewers, the meta-reviewer, as well as the area chairs and program chairs for their valuable comments and efforts. This work is supported by the National Natural Science Foundation of China (Grant No.62572465) and the Beijing Nova Program (Grant No.20250484895).

## References

- Alberto Bacci, John R Huguenard, and David A Prince. 2003. Functional autaptic neurotransmission in fast-spiking interneurons: a novel form of feedback inhibition in the neocortex. *Journal of Neuroscience*, 23(3):859–866.
- Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoung Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron C. Courville, and Se-Young Yun. 2025. [Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation](#). *CoRR*, abs/2507.10524.
- Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2019. Deep equilibrium models. *Advances in neural information processing systems*, 32.
- Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating llms on uncontaminated math competitions. *arXiv preprint arXiv:2505.23281*.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. [Language models are few-shot learners](#). *Preprint*, arXiv:2005.14165.
- Lei Chen, Joan Bruna, and Alberto Bietti. 2025a. [Distributional associations vs in-context reasoning: A study of feed-forward and attention layers](#). *Preprint*, arXiv:2406.03068.
- Mark Chen. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

- Xingwu Chen and Difan Zou. 2024. What can transformer learn with varying depth? case studies on sequence learning tasks. In *International Conference on Machine Learning*, pages 7972–8001. PMLR.
- Yao Chen, Jiawei Sheng, Wenyuan Zhang, and Tingwen Liu. 2025b. Improving reasoning capabilities in small models through mixture-of-layers distillation with stepwise attention on key information. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 4952–4971. Association for Computational Linguistics.
- Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. 2025c. Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28241–28259, Vienna, Austria. Association for Computational Linguistics.
- Minsik Choi, Hyegang Son, Changhoon Kim, and Young Geun Kim. 2025. Entropy meets importance: A unified head importance-entropy score for stable and efficient transformer pruning. *arXiv preprint arXiv:2510.13832*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Róbert Csordás, Piotr Piękos, Kazuki Irie, and Jürgen Schmidhuber. 2024. Switchhead: Accelerating transformers with mixture-of-experts attention. *Advances in Neural Information Processing Systems*, 37:74411–74438.
- Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502.
- Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. 2024. Looped transformers for length generalization. *arXiv preprint arXiv:2409.15647*.
- Yuchen Fu, Zifeng Cheng, Zhiwei Jiang, Zhonghui Wang, Yafeng Yin, Zhengliang Li, and Qing Gu. 2025a. Token prepending: A training-free approach for eliciting better sentence embeddings from LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3168–3181, Vienna, Austria. Association for Computational Linguistics.
- Zihao Fu, Ming Liao, Chris Russell, and Zhenguang G Cai. 2025b. Cast: Compositional analysis via spectral tracking for understanding transformer layer functions. *arXiv preprint arXiv:2510.14262*.
- Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley Osher, and Wotao Yin. 2021. Jfb: Jacobian-free backpropagation for implicit networks. *Preprint*, arXiv:2103.12803.
- Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*.
- Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495.
- Zixuan Gong, Jiaye Teng, and Yong Liu. 2025. What makes looped transformers perform better than non-recursive ones (provably). *arXiv preprint arXiv:2510.10089*.
- Dirk Groeneveld, Iz Beltagy, Evan Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, and 24 others. 2024. Olmo: Accelerating the science of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15789–15809. Association for Computational Linguistics.
- Xiangzhao Hao, Zefeng Zhang, Zhenyu Zhang, Linhao Yu, Yao Chen, Yiqian Zhang, Haiyun Guo, Shuohuan Wang, and Yu Sun. 2026. Clear: Unlocking generative potential for degraded image understanding in unified multimodal models. *arXiv preprint arXiv:2604.04780*.
- Tao He, Hao Li, Jingchang Chen, Runxuan Liu, Yixin Cao, Lizi Liao, Zihao Zheng, Zheng Chu, Jiafeng Liang, Ming Liu, and Bing Qin. 2025. Breaking the reasoning barrier a survey on LLM complex reasoning through the lens of self-evolution. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 7377–7417, Vienna, Austria. Association for Computational Linguistics.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. [Training compute-optimal large language models](#). *CoRR*, abs/2203.15556.
- Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekish, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What’s the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.
- Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017.
- Fanding Huang, Guanbo Huang, Xiao Fan, Yi He, Xiao Liang, Xiao Chen, Qinting Jiang, Faisal Nadeem Khan, Jingyan Jiang, and Zhi Wang. 2026. [Semantic-space exploration and exploitation in rlvr for llm reasoning](#). *Preprint*, arXiv:2509.23808.
- Hongru Ji, Yuyin Fan, Meng Zhao, Xianghua Li, Lianwei Wu, and Chao Gao. 2026. [Stride-ed: A strategy-grounded stepwise reasoning framework for empathetic dialogue systems](#). *Preprint*, arXiv:2604.07100.
- Man Jiang, Mingpo Yang, Luping Yin, Xiaohui Zhang, and Yousheng Shu. 2015. Developmental reduction of asynchronous gaba release from neocortical fast-spiking neurons. *Cerebral Cortex*, 25(1):258–270.
- Peng Jin, Bo Zhu, Li Yuan, and Shuicheng Yan. 2024. Moh: Multi-head attention as mixture-of-head attention. *arXiv preprint arXiv:2410.11842*.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. [TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.
- 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. 2019. [Natural questions: A benchmark for question answering research](#). *Transactions of the Association for Computational Linguistics*, 7:452–466.
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let’s verify step by step. *arXiv preprint arXiv:2305.20050*.
- Huan Ma, Jiadong Pan, Jing Liu, Yan Chen, Joey Tianyi Zhou, Guangyu Wang, Qinghua Hu, Hua Wu, Changqing Zhang, and Haifeng Wang. 2025. Semantic energy: Detecting llm hallucination beyond entropy. *arXiv preprint arXiv:2508.14496*.
- William Merrill and Ashish Sabharwal. 2024. A little depth goes a long way: The expressive power of log-depth transformers. In *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*.
- Meta. 2024. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. <https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/>.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.
- Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.
- Thiziri Nait Saada, Alireza Naderi, and Jared Tanner. 2025. [Mind the gap: a spectral analysis of rank collapse and signal propagation in attention layers](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 45561–45587. PMLR.
- Kei-Sing Ng and Qingchen Wang. 2024. [Loop neural networks for parameter sharing](#). *Preprint*, arXiv:2409.14199.
- Shuaiyi Nie, Siyu Ding, Wenyuan Zhang, Linhao Yu, Tianmeng Yang, Yao Chen, Tingwen Liu, Weichong Yin, Yu Sun, and Hua Wu. 2026. [ATTNPO: attention-guided process supervision for efficient reasoning](#). *CoRR*, abs/2602.09953.
- Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. [In-context learning and induction heads](#). *CoRR*, abs/2209.11895.

- OpenAI. 2023. [GPT-4 technical report](#). *CoRR*, abs/2303.08774.
- OpenAI. 2024. GPT-4o Mini Technical Specifications and Model Card. <https://platform.openai.com/docs/models/gpt-4o-mini>.
- Xingru Pan, Ruoqing Pan, Quansheng He, Wei Ke, and Yousheng Shu. 2025. Functional autapses selectively form in a subpopulation of subicular pyramidal cells in hippocampal formation. *Journal of Neuroscience*, 45(27).
- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. [Yarn: Efficient context window extension of large language models](#). In *International Conference on Representation Learning*, volume 2024, pages 31932–31951.
- Jackson Petty, Sjoerd Steenkiste, Ishita Dasgupta, Fei Sha, Dan Garrette, and Tal Linzen. 2024. The impact of depth on compositional generalization in transformer language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7239–7252.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Sampurna Roy, Ayan Sar, Anurag Kaushish, Kanav Gupta, Tanupriya Choudhury, and Abhijit Kumar. 2025. Dynamic reasoning chains through depth-specialized mixture-of-experts in transformer architectures. *arXiv preprint arXiv:2509.20577*.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavata, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64:99–106.
- Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations*.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. [CommonsenseQA: A question answering challenge targeting commonsense knowledge](#). 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)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yixuan Tang and Yi Yang. 2024. Pooling and attention: What are effective designs for llm-based embedding models? *arXiv preprint arXiv:2409.02727*.
- Llama Team. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Qwen Team. 2025. [Qwen3 technical report](#). *CoRR*, abs/2505.09388.
- Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Chengbing Wang, Yang Zhang, Wenjie Wang, Xiaoyan Zhao, Fuli Feng, Xiangnan He, and Tat-Seng Chua. 2025. Think-while-generating: On-the-fly reasoning for personalized long-form generation. *arXiv preprint arXiv:2512.06690*.
- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916.
- Tongxi Wang. 2026. Fbs: Modeling native parallel reading inside a transformer. *arXiv preprint arXiv:2601.21708*.
- Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017a. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*.
- Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017b. [Crowdsourcing multiple choice science questions](#). *Preprint*, arXiv:1707.06209.
- Chao Xue and Ziyuan Gao. 2025. Structcoh: Structured contrastive learning for context-aware text semantic matching. In *Pacific Rim International Conference on Artificial Intelligence*, pages 300–315. Springer.
- Chao Xue, Di Liang, Sirui Wang, Jing Zhang, and Wei Wu. 2023. Dual path modeling for semantic matching by perceiving subtle conflicts. In *ICASSP 2023-IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, Haibo Shi, Shuang Liang, Minlong Peng, and Flora D. Salim. 2026a. [Reason only when needed: Efficient generative reward modeling via model-internal uncertainty](#). *Preprint*, arXiv:2604.10072.
- Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, Haibo Shi, Shuang Liang, Minlong Peng, and Flora D. Salim. 2026b. [Why supervised fine-tuning fails to learn: A systematic study](#)

- of incomplete learning in large language models. *Preprint*, arXiv:2604.10079.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. 2025. [HELMET: How to evaluate long-context models effectively and thoroughly](#). In *The Thirteenth International Conference on Learning Representations*.
- Lin hao Yu, Tianmeng Yang, Siyu Ding, Renren Jin, Naibin Gu, Xiangzhao Hao, Shuaiyi Nie, Deyi Xiong, Weichong Yin, Yu Sun, and 1 others. 2026. [Knowrl: Boosting llm reasoning via reinforcement learning with minimal-sufficient knowledge guidance](#). *arXiv preprint arXiv:2604.12627*.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. [Hellaswag: Can a machine really finish your sentence?](#) *arXiv preprint arXiv:1905.07830*.
- Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. 2023. [Stabilizing transformer training by preventing attention entropy collapse](#). In *International Conference on Machine Learning*, pages 40770–40803. PMLR.
- Xinglang Zhang, Yunyao Zhang, ZeLiang Chen, Junqing Yu, Wei Yang, and Zikai Song. 2026a. [Logical phase transitions: Understanding collapse in llm logical reasoning](#). *Preprint*, arXiv:2601.02902.
- Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. [∞bench: Extending long context evaluation beyond 100k tokens](#). *CoRR*, abs/2402.13718.
- Yunyao Zhang, Xinglang Zhang, Junxi Sheng, Wenbing Li, Junqing Yu, Yi-Ping Phoebe Chen, Wei Yang, and Zikai Song. 2026b. [Semantic-aware logical reasoning via a semiotic framework](#). *Preprint*, arXiv:2509.24765.
- Zefeng Zhang, Xiangzhao Hao, Hengzhu Tang, Zhenyu Zhang, Jiawei Sheng, Xiaodong Li, Zhenyang Li, Li Gao, Daiting Shi, Dawei Yin, and 1 others. 2025a. [Cooper: A unified model for cooperative perception and reasoning in spatial intelligence](#). *arXiv preprint arXiv:2512.04563*.
- Zhisong Zhang, Yan Wang, Xinting Huang, Tianqing Fang, Hongming Zhang, Chenlong Deng, Shuaiyi Li, and Dong Yu. 2025b. [Attention entropy is a key factor: An analysis of parallel context encoding with full-attention-based pre-trained language models](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9840–9855.
- Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, and 14 others. 2025. [Scaling latent reasoning via looped language models](#). *CoRR*, abs/2510.25741.
- Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. [A survey on model compression for large language models](#). *Transactions of the Association for Computational Linguistics*, 12:1556–1577.

## A Related Work

**Depth Expansion and Recurrence.** Increasing effective depth has been widely recognized as a key factor in enhancing Transformer reasoning capacity (Petty et al., 2024; Chen and Zou, 2024; Merrill and Sabharwal, 2024; Saunshi et al., 2025; Roy et al., 2025; Xue and Gao, 2025; Zhang et al., 2026b; Xue et al., 2026b; Zhang et al., 2026a). On the recurrence side, recent studies introduce recurrence to expand effective depth without increasing parameter count, showing improvements in implicit reasoning, input-length generalization, and training stability (Ng and Wang, 2024; Fan et al., 2024; Geiping et al., 2025; Gong et al., 2025; Wang et al., 2025). These approaches primarily adopt block-level recursion, uniformly reapplying entire layers or layer groups to expand depth. While such recursive strategies achieve parameter efficiency by reusing existing weights, they introduce additional training-time computational overhead due to indiscriminate depth allocation across parameters, and leave unexplored which components within a block may benefit most from deeper recurrence.

**Paradigms of Sparsity.** A prominent line of work improves efficiency through input-dependent sparsity, where computation is selectively allocated based on token characteristics (Csordás et al., 2024; Jin et al., 2024; Wang, 2026). Mixture-of-Experts (MoE) architectures and head-level routing methods such as SwitchHead (Csordás et al., 2024) and Mixture of Heads (Jin et al., 2024) introduce input-dependent sparsity by routing tokens to distinct expert FFN parameters or selectively activating attention heads conditioned on token characteristics. Similarly, recent inner-thinking dynamically selects which tokens receive additional computational steps (Chen et al., 2025c), while structured stepwise reasoning frameworks decompose complex tasks into explicit multi-stage cognitive processes (Ji et al., 2026; Zhang et al., 2025a; Hao et al., 2026; Yu et al., 2026; Xue et al., 2026a). These methods, therefore, introduce heterogeneous execution paths across samples, as computation varies with input-dependent routing decisions. In contrast, our work introduces sparsity along a different axis. Rather than dynamically selecting tokens or experts, we focus on parameter-level depth allocation and training-time structural sparsity. Specifically, SGT progressively grows fine-grained recurrent micro-circuits during training, selectively allocating additional depth to high-value

parameters within a block. This growth unfolds along the training timeline and concentrates computation on a subset of functional components, thereby reducing substantial pretraining overhead. Since these approaches introduce sparsity along different dimensions, SGT is orthogonal to input-dependent routing methods and can be naturally combined with them to further improve efficiency.

**Attention Entropy.** Attention entropy, defined as the Shannon entropy of attention weight distributions, provides a quantitative measure of how concentrated or dispersed a model’s attention is across input tokens. Zhai et al. (2023) demonstrates that fluctuations in average attention entropy correlate with training stability, while Zhang et al. (2025b) shows its influence on contextual modeling during reasoning. However, these analyses typically operate at a coarse, model-wide granularity. In the context of efficiency, HIES (Choi et al., 2025) combines gradient sensitivity with attention entropy to identify heads for pruning, reporting stable performance when removing high-entropy heads. Nevertheless, its validation relies solely on simple sentiment classification. For reasoning tasks characterized by complex dependencies, we posit that high-entropy heads may not be mere noise; instead, they may capture essential but widely distributed contextual signals (Xue et al., 2023; He et al., 2025; Ma et al., 2025) required for global reasoning.

## B Attention Entropy Computation and Rationale

We utilize the attention distribution of the last token to compute the entropy metric for the following reasons. First, the final query token is the position where the model integrates all preceding contextual information to produce the next-token distribution. Since next-token prediction is the core training objective of autoregressive LLMs, the last token of the input sequence inherently encodes all preceding information (Muennighoff, 2022; Wang et al., 2024; Tang and Yang, 2024; Fu et al., 2025a; Nie et al., 2026), making its attention pattern the most semantically meaningful for characterizing how context is aggregated. Second, prior work shows that in decoder-only Transformers, attention entropy tends to increase smoothly and monotonically along the sequence without abrupt fluctuations at intermediate positions (Zhang et al., 2025b), making the entropy of the final token a stable summary of the model’s contextual uncertainty at the end of decod-

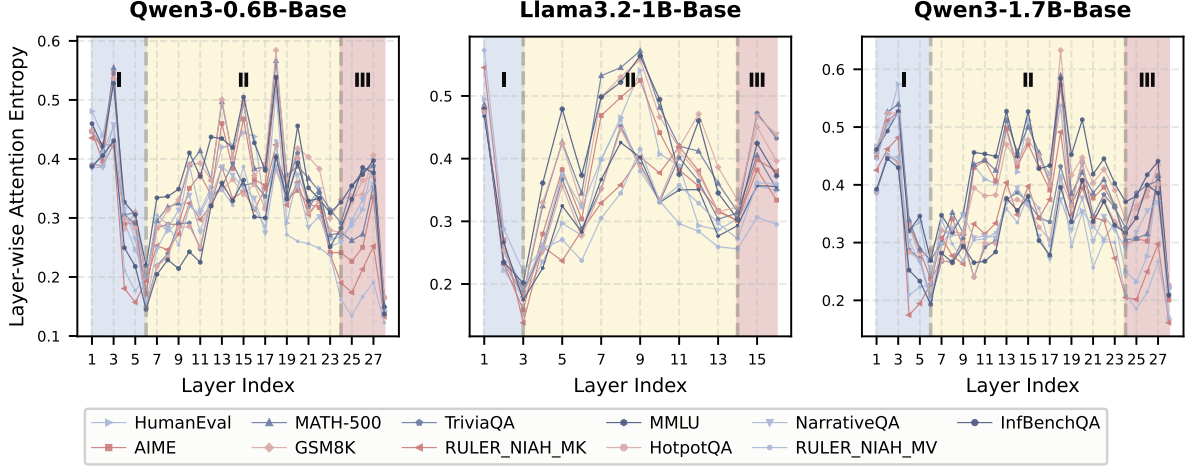


Figure 7: Layer-wise attention entropy across 11 diverse tasks for three open-source models (details in Appendix C.1 and C.2.1).

### Algorithm 1 Progressive Growth Training

**Require:** Model with  $N_{\text{layer}}$  layers, training steps  $T$ , growth start  $t_{\text{start}}$ , window interval  $\Delta t$ , target number of looping layers  $L$ , the upper bound on the loop depth for the selected layers  $K_{\text{max}}$ , the top- $L$  candidate layers per ranking forming the candidate set  $\mathcal{C}$ , and the top- $h$  high-entropy heads forming  $\mathcal{S}_l$  for looping layer  $l$ .

- 1: Initialize  $\mathcal{L} \leftarrow \emptyset$  (set of layers with active looping)
- 2: Initialize  $K_l \leftarrow 0$  for all layers  $l$  (current loop depth for layer  $l$ )
- 3: Initialize  $l^* \leftarrow \text{None}$  (currently growing layer)
- 4: **for**  $t = 1$  to  $T$  **do**
- 5:     // Forward pass with entropy-guided looping (Eq. 6)
- 6:     **for**  $l = 1$  to  $N_{\text{layer}}$  **do**
- 7:          $H^{(0)} \leftarrow H + \sum_i \text{Attn}^{(i)}(H)$
- 8:         **if**  $l \in \mathcal{L}$  **then**
- 9:             **for**  $k = 1$  to  $K_l$  **do**
- 10:                  $H^{(k)} \leftarrow H^{(k-1)} + \sum_{i \in \mathcal{S}_l} \text{Attn}^{(i)}(H^{(k-1)})$
- 11:             **end for**
- 12:         **end if**
- 13:          $H \leftarrow \text{FFN}(H^{(K_l)})$
- 14:     **end for**
- 15:     // Progressive growth (Selection Phase)
- 16:     **if**  $t \geq t_{\text{start}}$  and  $(t - t_{\text{start}}) \bmod \Delta t = 0$  **then**
- 17:          $\mathcal{C} \leftarrow \arg \text{top}_L(\{\mathcal{E}^{(l)}\}_{l=1}^{N_{\text{layer}}})$
- 18:         **if**  $l^* \neq \text{None}$  and  $l^* \in \mathcal{C}$  and  $K_{l^*} < K_{\text{max}}$  **then**
- 19:              $K_{l^*} \leftarrow K_{l^*} + 1$
- 20:         **else if**  $|\mathcal{C}| < L$  **then**
- 21:              $l^* \leftarrow \max\{l \mid l \in \mathcal{C}, l \notin \mathcal{L}, l < \min(\mathcal{L})\}$
- 22:              $\mathcal{S}_{l^*} \leftarrow \arg \text{top}_h(\{\mathcal{E}^{(i)}\})$
- 23:              $\mathcal{L} \leftarrow \mathcal{L} \cup \{l^*\}$
- 24:              $K_{l^*} \leftarrow 1$
- 25:         **end if**
- 26:     **end if**
- 27: **end for**

ing. Third, this approach significantly reduces computational overhead. By avoiding full-sequence aggregation, we ensure the computational efficiency required for the effective implementation of the

Benchmarks	Avg. Input Length
<b>Mathematical Reasoning</b>	
AIME (Balunović et al., 2025)	160 tokens
MATH-500 (Lightman et al., 2023)	89 tokens
GSM8K (Cobbe et al., 2021)	78 tokens
<b>General Knowledge QA</b>	
MMLU (Hendrycks et al., 2020)	117 tokens
<b>Code Reasoning</b>	
HumanEval (Chen, 2021)	165 tokens
<b>Long-Context Retrieval</b>	
RULER_NIAH_MK (Hsieh et al., 2024)	3051 tokens
RULER_NIAH_MV (Hsieh et al., 2024)	3133 tokens
<b>Long-Context Multi-Hop QA</b>	
NarrativeQA (Kwiatkowski et al., 2019)	3473 tokens
InfBenchQA (Zhang et al., 2024)	3760 tokens
TriviaQA (Joshi et al., 2017)	3308 tokens
HotpotQA (Yang et al., 2018)	3287 tokens

Table 5: Overview of the 11 diverse reasoning benchmarks, categorized by task family, used for analyzing the model’s attention entropy. The tasks span various domains and sequence lengths.

entropy metric in our proposed method.

## C Observational Study Details

### C.1 Experimental Setup

We select three representative open-source models: Qwen3-0.6B-Base, Qwen3.1-1.7B-Base (Team, 2025), and LLaMA3.2-1B-Base (Meta, 2024).

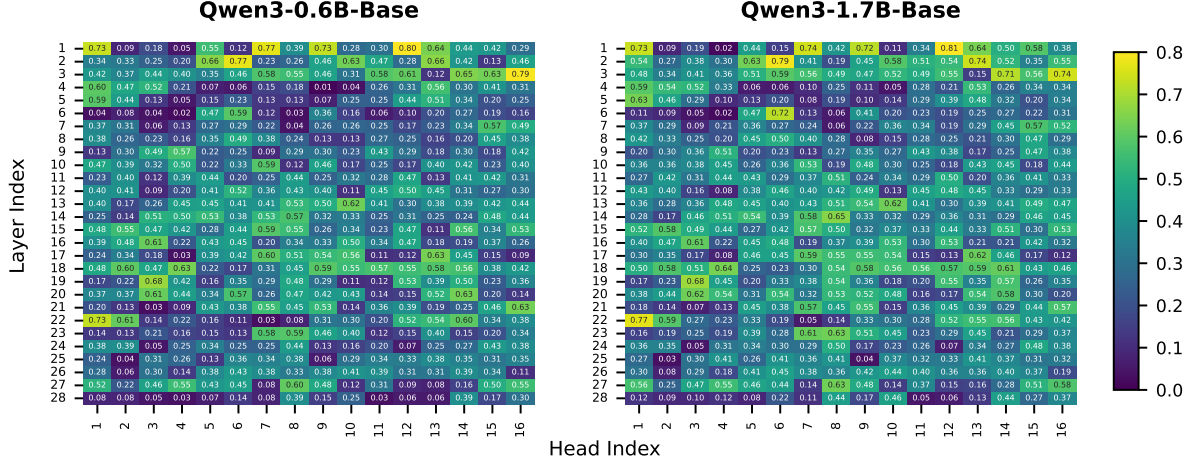


Figure 8: Layer-head attention entropy pattern of Qwen3-0.6B and Qwen3-1.7B. The x-axis denotes the head index, and the y-axis denotes the layer index (details in Appendix C.1 and C.2.2).

Model	layers	heads	hidden size	intermediate size
Qwen3-0.6B-Base	28	16	1024	3072
LLaMA3.2-1B-Base	16	32	2048	8192
Qwen3-1.7B-Base	28	16	2048	6144

Table 6: Architectural hyperparameters of three representative open-source models used in our observational study of Section 3. *heads* denotes the number of query heads.

These models span diverse parameter scales, and their architectural configurations are detailed in Table 6. To minimize task-specific bias, we evaluated the models across 11 diverse reasoning benchmarks. These tasks cover a broad range of domains and sequence lengths, including Mathematical Reasoning, General Knowledge QA, Code Reasoning, Long-Context Retrieval, and Long-Context Multi-Hop QA (see Table 5). Among these, all long-context tasks are drawn from HELMET (Yen et al., 2025), a comprehensive benchmark for long-text reasoning ability. For each task, we randomly sample 100 instances; for datasets with many fine-grained subcategories, such as MMLU and MATH-500, we sample uniformly across all subtypes to reduce category bias.

## C.2 Intrinsic Structural Patterns of Attention Entropy

### C.2.1 Layer-wise Attention Entropy

We visualize the layer-wise attention entropy, as defined in Equation 5, across 11 diverse tasks on three representative open-source models. The results

(Figure 7) reveal a highly consistent and cross-task stable three-phase (I–II–III) entropy progression across all models: Phase I (shallow layers) shows a sharp entropy decrease; Phase II (middle layers) exhibits a gradual rise followed by a steady decline; Phase III (deep layers) presents a renewed entropy increase before a final decline before to-ken generation. This robust and task-independent *three-phase* progression demonstrates that attention entropy serves as an intrinsic architectural characteristic. Notably, similar depth-dependent functional phases have previously been reported in interpretability studies (Tenney et al., 2019; Dai et al., 2022; Fu et al., 2025b; Chen et al., 2025b; Huang et al., 2026), but primarily through indirect diagnostics such as probing classifiers, neuron-level interventions, and spectral analyses of representation spaces. In contrast, layer-wise attention entropy provides a direct measurement of attention uncertainty and contextual focus, offering more task-agnostic and model-agnostic evidence for such depth-dependent functional specialization.

### C.2.2 Layer-head Attention Entropy Pattern

We visualize the layer-head attention entropy pattern, based on the mean attention entropy computed over all tasks. As shown in Figure 8, the two Qwen3 models with different parameter scales exhibit strikingly similar layer-head entropy patterns, despite substantial differences in their hidden sizes and intermediate sizes. Since the models share the same number of layers and attention heads (Table 6), this consistency indicates that attention-entropy patterns are closely related to architectural

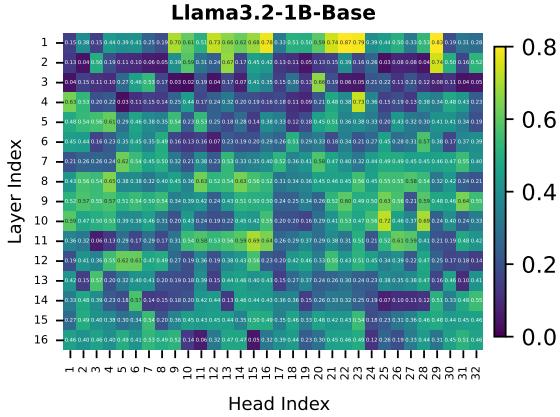


Figure 9: Layer-head attention entropy pattern of LLaMA3.2-1B-Base. The x-axis denotes the head index, and the y-axis denotes the layer index (details in Appendix C.1 and C.2.2).

configuration. We also visualize the layer-head attention entropy patterns of LLaMA3.2-1B, as shown in Figure 9. Compared with the Qwen3 models, LLaMA3.2-1B exhibits a different layer-head entropy pattern.

### C.3 Functional Dichotomy of High- and Low-Entropy Attention Heads

#### C.3.1 Entropy-based Head Selection

We identify high- and low-entropy heads in the Qwen3-0.6B model based on their attention entropy across 11 diverse tasks from Table 5. Heads whose entropy remains consistently above 0.5 across 11 tasks are classified as high-entropy heads, whereas those consistently below 0.06 across the same tasks are classified as low-entropy heads. As shown in Table 7, we use the notation L<sub>x</sub>-H<sub>y</sub> to denote the head at layer *x* and head index *y* (1-indexed). The identified high-entropy heads are L1-H1, L1-H7, L1-H12, and L2-H6, while the low-entropy heads are L4-H9, L25-H2, and L28-H4. We analyze the functional roles of these two groups from both qualitative and quantitative perspectives.

#### C.3.2 Qualitative Analysis

We visualize attention patterns on representative samples from three benchmarks: GSM8K (Figure 1), MATH-500 (Figure 12), and MMLU (Figure 13), with consistent qualitative distinctions observed between high- and low-entropy heads across all datasets. Along the horizontal axis, we highlight in red the tokens that receive the top 50% of attention weights from the final query position. While the top 50% tokens of low-entropy heads remain

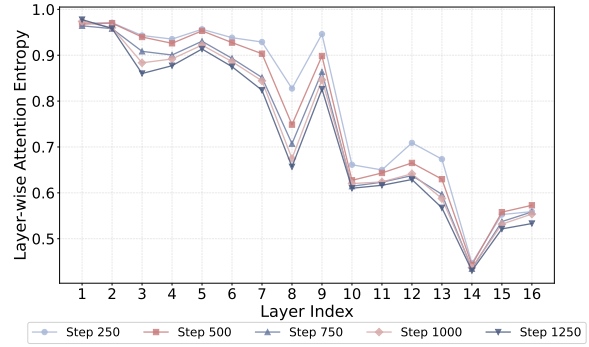


Figure 10: Layer-wise Attention Entropy at Different Training Steps. The horizontal axis represents the model’s layer index. Each line shows the entropy distribution recorded at 250-step intervals during the early stages of training (details in Appendix C.4).

confined to a narrow local neighborhood near the diagonal, high-entropy heads selectively attend to tokens that are directly relevant for reasoning, indicating their role in capturing globally meaningful dependencies.

#### C.3.3 Quantitative Analysis

We evaluate how masking different groups of attention heads affects model performance on the Qwen3-0.6B model across five tasks from Table 5, with all outputs assessed using GPT-4o-mini (OpenAI, 2024) as a unified judge model. To control for randomness, we construct four random-head baselines sampled from the remaining heads after excluding both high- and low-entropy heads: two groups match the high-entropy set in terms of layer and head count, and the other two match the low-entropy set. As shown in Table 7, masking high-entropy heads results in the greatest performance degradation relative to the original model across almost all tasks, with performance collapsing much more severely on reasoning datasets than on long-context retrieval tasks.

#### C.4 Training Dynamics of Attention Entropy

We pre-trained a 575M parameter model from scratch based on the LLaMA architecture with Multi-Head Attention (MHA). The model comprises 16 layers with 16 attention heads per layer; detailed configurations are provided in Table 8. The pre-training was conducted on a corpus of 20B tokens for a total of 5,035 steps. To investigate the dynamic properties of the model, we tracked three key metrics throughout the training process: the layer-wise average attention entropy (Equation 5), the individual attention entropy of all heads within

Mask Settings	Original Model	High-Entropy Heads	Low-Entropy Heads	Random Heads (Various Seeds)			
				Set 1	Set 2	Set 3	Set 4
Masked Heads	-	L1-H1, L1-H7 L1-H12, L2-H6	L4-H9, L25-H2 L28-H4	L4-H1, L4-H5 L4-H12, L21-H8	L2-H5, L2-H6 L2-H8, L10-H4	L1-H9, L4-H8 L21-H8	L2-H5, L10-H15 L23-H12
MATH-500	27.0	5.0 ↓ 81.5%	25.0 ↓ 7.4%	20.0 ↓ 25.9%	24.0 ↓ 11.1%	25.0 ↓ 7.4%	23.0 ↓ 14.8%
GSM8K	32.0	7.0 ↓ 78.1%	29.0 ↓ 9.4%	37.0 ↑ 15.6%	32.0 ±0.0%	32.0 ±0.0%	27.0 ↓ 15.6%
MMLU	50.5	18.0 ↓ 64.4%	46.0 ↓ 8.9%	45.5 ↓ 9.9%	47.0 ↓ 6.9%	46.0 ↓ 8.9%	50.5 ±0.0%
NarrativeQA	13.7	11.0 ↓ 19.7%	14.2 ↑ 3.7%	13.6 ↓ 0.7%	12.2 ↓ 11.0%	12.6 ↓ 8.0%	14.5 ↑ 5.8%
NIAH_MK	81.0	69.0 ↓ 14.8%	80.0 ↓ 1.2%	83.0 ↑ 2.5%	72.0 ↓ 11.1%	74.0 ↓ 8.6%	84.0 ↑ 3.7%

Table 7: Impact of head masking on Qwen3-0.6B-Base performance across reasoning and long-context tasks. The *Original Model* column represents the unmasked model baseline. *Mask Settings* compare the removal of High-Entropy heads against Low-Entropy and four groups of Random head baselines. The specific indices of removed heads are detailed in the *Masked Heads* row, denoted as  $L_x-H_y$  (Layer  $x$ , Head  $y$ , 1-indexed). Cells report absolute performance scores and the relative percentage change compared to the *Original Model* (details in Appendix C.3.1 and C.3.3).

Model Setting	OLMo-275M	OLMo-573M	OLMo-1.2B
<i>hidden size</i>	768	1024	1536
<i>intermediate size</i>	6144	8192	12288
<i>attention heads</i>	12	16	16
<i>layers</i>	12	16	16

Table 8: Detailed architecture configurations for the three model scales (275M, 573M, and 1.2B) pre-trained from scratch in our main experiments. We pre-train models from scratch at three scales using the OLMo framework. All models adopt the LLaMA architecture with Multi-Head Attention (MHA).

each layer (Equation 4), and the intra-layer variance of head attention entropy.

Regarding the layer-level dynamics, we observe that during the early training phase, attention entropy decreases universally, while the relative ordering of layers remains largely stable with only minor fluctuations (Figure 10).

Regarding the intra-layer variance, we observe distinct behaviors across depths: deep layers exhibit a rapid surge followed by a gradual decline, while shallow layers show a steady, continuous increase. Interpreting attention entropy as a proxy for head functionality (Observation 1), we consider the intra-layer entropy variance as a measure of the degree of functional differentiation among heads. A higher variance indicates more significant divergence in head roles. Based on this interpretation, the observed patterns reveal that deep layers undergo rapid functional differentiation during the

early stages of training, whereas shallow layers differentiate through a slower, more stable process. Notably, we observe a clear depth-dependent trend: the deeper the layer, the earlier it achieves functional differentiation. In particular, Layer 1 presents a notable exception: it maintains near-zero entropy variance throughout the entire training process, with the attention entropy of all heads consistently approaching 1 (Figure 14). Given this lack of functional differentiation, we exclude Layer 1 from the candidate pool for our dynamic layer selection strategy during the progressive growth training phase.

## D Experimental Details and Supplementary Analysis

### D.1 Details of Main Experiments

The *Block Loop* baseline follows a widely adopted block-level depth reuse design in prior looped-depth studies (Zhu et al., 2025; Chen et al., 2025c; Bae et al., 2025). Its configuration is deliberately designed to ensure strong and competitive performance rather than serve as a weak reference point. In particular, the selection of looped layers is guided by attention entropy rather than random assignment, following the same principle used in our proposed method. This ensures that recurrence is applied to the most information-dense layers, maintaining methodological consistency and enabling a fair comparison between block-level and head-level depth allocation strategies.

Our primary objective is to validate the feasi-

Model	ARC-E	WG	SIQA
Vanilla	44.56	49.57	41.30
Block Loop	44.74	50.36	41.10
SGT ( $t_{\text{start}} = \Delta t = 500$ )	45.26	50.91	41.56
SGT ( $t_{\text{start}} = \Delta t = 250$ )	45.09	51.46	41.40
SGT ( $t_{\text{start}} = \Delta t = 100$ )	45.44	51.62	41.97

Table 9: Analysis of growth schedule parameters ( $t_{\text{start}}$  and  $\Delta t$ ) for the 275M model.

bility of the *training-time sparsity paradigm*. We therefore do not attempt to reproduce the full complexity of existing token-routing loop architectures (Chen et al., 2025c), as discussed in Related Work, such methods explore a data-dependent execution paradigm that is orthogonal to our structural design focus. For the selective head-level looping in SGT, we set the number of high-entropy heads to  $h = 2$ , which represents an extremely small fraction of the total attention heads within a layer. This deliberately sparse configuration allows us to evaluate whether substantial reductions in training FLOPs, achieved by looping only a minimal subset of high-entropy heads, can match the performance of full block-level looping. All results are reported from single runs, which is standard practice in large-scale pretraining due to computational constraints.

## D.2 Details of Ablation Experiments

To isolate individual variables and minimize confounding effects in our ablation and analysis experiments, we adopted a single-layer configuration. This design enables a clean assessment of the computational contributions of different block components across varying loop depths. For the high-entropy head loop, we used a two-stage selection procedure: in the first window, we identified ten high-entropy heads; in the second window, we selected two heads from this set. The resulting structure was then frozen for the remainder of training. The low-entropy head loop follows the same protocol, with the only difference being that low-entropy heads are selected.

## D.3 Analysis of the Growth Schedule Parameters

In our main experiments, we configure the progressive growth schedule with  $t_{\text{start}} = 250$  and  $\Delta t = 250$ . To further examine the sensitivity of SGT to the growth schedule, we conduct additional experiments by varying the temporal parameters

Batch	Model	Prefill TPS	Decode TPS
1	Vanilla (275M)	14381.0	127.9
	SGT (275M)	13805.7	122.7
	Block Loop (275M)	11504.8	102.3
	Vanilla (573M)	10969.2	100.5
	SGT (573M)	10718.0	98.2
	Block Loop (573M)	9237.2	84.6
2	Vanilla (275M)	27646.2	245.2
	SGT (275M)	26540.4	235.3
	Block Loop (275M)	22117.0	196.1
	Vanilla (573M)	3997.6	194.9
	SGT (573M)	3906.1	190.4
	Block Loop (573M)	3366.4	164.1
4	Vanilla (275M)	98181.9	928.5
	SGT (275M)	94254.6	891.4
	Block Loop (275M)	78545.5	742.8
	Vanilla (573M)	41193.5	384.6
	SGT (573M)	40250.1	375.8
	Block Loop (573M)	34689.2	323.9

Table 10: Inference throughput comparison (tokens/s) under different batch sizes. Prefill throughput measures tokens processed during context encoding (KV cache construction), while decode throughput measures autoregressive token generation speed.

$t_{\text{start}}$  and  $\Delta t$ . As shown in Table 9, SGT consistently achieves stable and robust performance improvements across different configurations.

## D.4 Analysis of Inference Throughput

During inference, SGT assigns additional recurrence only to a small subset of attention heads identified during training as high-entropy components. Unlike data-dependent dynamic routing architectures that perform token-level or sample-level conditional execution, SGT maintains a static inference graph in which all samples within a batch follow the same computation path. This design preserves dense and regular execution, enabling full GPU parallelism without introducing branch divergence or dynamic control overhead.

We evaluate inference throughput on a single NVIDIA A800 GPU using BF16 precision, with a maximum prompt length of 128 and a generation length of 128. To isolate architectural effects, all methods are implemented natively without specialized operator-level optimizations. As shown in Table 10, although parameter reuse inevitably introduces moderate latency compared to the vanilla Transformer, SGT consistently achieves

higher throughput than the Block Loop across all batch sizes and model scales.

## E Theoretical Analysis of High-Entropy Recursive Convergence

In this section, we provide a theoretical justification for why high-entropy heads exhibit superior convergence properties compared to low-entropy heads in recursive attention mechanisms. Our analysis connects the entropy of attention matrices to the spectral properties of the recursive dynamics, offering theoretical insight into the experimental results. We retain the notation from Section 3 and Section 4 for consistency, while defining new variables as they appear.

### E.1 Convergence Objective and Equilibrium Analysis

While standard Transformers stack distinct layers to refine token representations progressively, our recursive approach performs this refinement within a shared parameter space. To analyze the efficiency of this refinement, we view the recursive dynamics through the lens of implicit depth models.

**Quasi-Stationary State.** Following the framework of (Bai et al., 2019; Fung et al., 2021), we analyze convergence toward a *quasi-stationary* state  $H^*$  where the recursive update becomes sufficiently small:

$$\|\mathcal{F}(H^*)\|_F \leq \delta, \quad (12)$$

for some small tolerance  $\delta > 0$ . This formulation allows us to analyze finite-depth recursive dynamics without requiring exact convergence.

**Convergence Error Analysis.** We define the error at iteration  $k$  as  $\epsilon^{(k)} = H^{(k)} - H^*$ . Our theoretical goal is to establish that after a finite number of recursive steps  $K$ , high-entropy head selection achieves a smaller error bound  $\|\epsilon^{(K)}\|_F$  compared to low-entropy selections, thereby more efficiently approximating the quasi-equilibrium.

### E.2 Spectral Analysis of Convergence

The key insight underlying our analysis is that global context integration is fundamentally a *token-mixing* problem. The challenge in recursive refinement is achieving consensus across spatially distributed tokens—this information diffusion process is the primary bottleneck for convergence. Recent spectral analyses of attention mechanisms

(Nait Saada et al., 2025) have demonstrated that the spectral properties of the attention matrix  $A$  (particularly the distribution of its eigenvalues) dominantly govern signal propagation dynamics through transformer layers. The connectivity structure encoded in  $A$  determines how quickly information flows between tokens, largely independent of the specific feature-space transformations applied by  $W_V$  and  $W_O$ .

**Spectral Dominance Approximation.** Building on this spectral perspective, we model the aggregate effect of selected attention heads through their token-mixing properties. The aggregate recursive update can be approximated by factoring out an effective scaling factor  $\gamma$  that absorbs the contributions of the projection matrices:

$$\mathcal{F}(H^{(k)}) = \sum_{i \in \mathcal{S}} \text{Attn}^{(i)}(H^{(k)}) \approx \gamma \bar{A} H^{(k)}, \quad (13)$$

where  $\bar{A}$  is the effective average attention matrix defined as:

$$\bar{A} = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} A^{(i)}. \quad (14)$$

This factorization separates the token-mixing dynamics (captured by  $\bar{A}$ ) from feature-space transformations (absorbed into  $\gamma$ ), allowing us to focus our spectral analysis on the attention matrix, which governs the fundamental convergence rate.

**Error Dynamics.** We analyze how the error  $\epsilon^{(k)}$  evolves over iterations. Starting from the recursive update  $H^{(k+1)} = H^{(k)} + \mathcal{F}(H^{(k)})$  and the quasi-equilibrium  $H^* \approx H^* + \mathcal{F}(H^*)$ , we obtain:

$$\begin{aligned} \epsilon^{(k+1)} &= H^{(k+1)} - H^* \\ &\approx \epsilon^{(k)} + \mathcal{F}(H^{(k)}) - \mathcal{F}(H^*). \end{aligned} \quad (15)$$

Substituting the spectral approximation from Equation 13:

$$\begin{aligned} \epsilon^{(k+1)} &\approx \epsilon^{(k)} + \gamma \bar{A} H^{(k)} - \gamma \bar{A} H^* \\ &= \epsilon^{(k)} + \gamma \bar{A} (H^{(k)} - H^*) \\ &= (I + \gamma \bar{A}) \epsilon^{(k)}. \end{aligned} \quad (16)$$

To model the effective mixing strength (accounting for LayerNorm stabilization and residual connection balancing), we reparameterize with a normalized coefficient  $\beta = \gamma / (1 + \gamma) \in (0, 1]$ , yielding:

$$\begin{aligned} \epsilon^{(k+1)} &\approx ((1 - \beta)I + \beta \bar{A}) \epsilon^{(k)} \\ &= M \epsilon^{(k)}, \end{aligned} \quad (17)$$

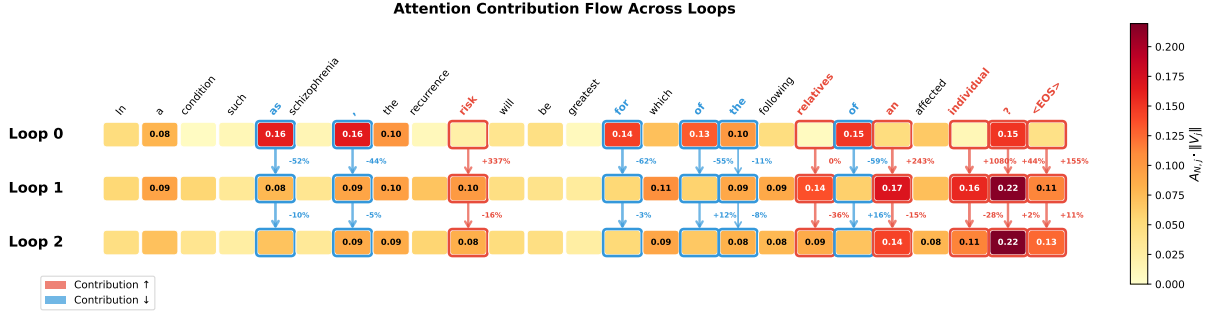


Figure 11: Attention contribution flow across loops in a high-entropy head (Layer 2, Head 15). Each cell shows the weighted attention contribution  $C_j = A_{N,j} \cdot \|V_j\|_2$  from the final token to position  $j$ , with color intensity indicating magnitude. Red and blue borders mark tokens with the largest contribution increase and decrease from Loop 0 to Loop 2, respectively. Arrows with percentage labels denote inter-loop changes. The visualization shows attention progressively shifting from syntactic elements toward task-critical and answer-relevant tokens.

where  $M = (1 - \beta)I + \beta\bar{A}$  is the effective mixing matrix. Since  $\bar{A}$  is row-stochastic (each row sums to 1),  $M$  is also row-stochastic

**Spectral Characterization of Convergence Rate.** The convergence of the error  $\epsilon^{(k)} = H^{(k)} - H^*$  is governed by the spectral properties of the mixing matrix  $M$ . For the error dynamics  $\epsilon^{(k+1)} = M\epsilon^{(k)}$ , we can bound the error norm by analyzing the eigenvalue structure of  $M$ . Since  $M = (1 - \beta)I + \beta\bar{A}$ , the eigenvalues of  $M$  are related to those of  $\bar{A}$  by:

$$\lambda_i(M) = (1 - \beta) + \beta\lambda_i(\bar{A}), \quad i = 1, \dots, N. \quad (18)$$

Due to the causal mask in decoder-only architectures,  $\bar{A}$  is strictly lower triangular, and thus its eigenvalues equal its diagonal entries:  $\lambda_i(\bar{A}) = \bar{A}_{ii}$ . For a fixed number of recursive steps  $K$ , the error after  $K$  iterations can be bounded using the spectral norm:

$$\begin{aligned} \|\epsilon^{(K)}\|_F &\leq \|M^K\|_F \|\epsilon^{(0)}\|_F \\ &\lesssim \left( \sum_{i=1}^N \lambda_i(M)^2 \right)^{K/2} \|\epsilon^{(0)}\|_F \\ &= \left( \sum_{i=1}^N [(1 - \beta) + \beta\bar{A}_{ii}]^2 \right)^{K/2} \|\epsilon^{(0)}\|_F. \end{aligned} \quad (19)$$

Expanding the squared terms and noting that the  $(1 - \beta)^2$  contributes uniformly across all eigenvalues, the variation in convergence behavior is determined by the attention-dependent terms. The dominant factor for comparing different head selec-

tions is the trace:

$$\left( \sum_{i=1}^N \bar{A}_{ii} \right)^K = (\text{Tr}(\bar{A}))^K, \quad (20)$$

where  $\text{Tr}(\bar{A}) = \sum_{i=1}^N \bar{A}_{ii}$  is the trace of the attention matrix. This reveals that minimizing the trace of the attention matrix accelerates convergence. Our theoretical goal is to show that high-entropy head selection achieves this.

### E.3 Connecting Entropy to Trace

We now establish a quantitative connection between attention entropy and the trace  $\text{Tr}(\bar{A})$ .

**Lemma 1** (Entropy-Based Bound on Diagonal Elements). *For the  $i$ -th row of a causal attention matrix  $\bar{A}$  with entropy  $\mathcal{E}_i = -\sum_{j=1}^i \bar{A}_{ij} \log \bar{A}_{ij}$ , the diagonal element satisfies:*

$$\bar{A}_{ii} \leq e^{-\mathcal{E}_i/i}. \quad (21)$$

*Proof Sketch.* Consider the  $i$ -th row as a probability distribution over positions  $\{1, \dots, i\}$ . We seek to maximize  $\bar{A}_{ii}$  subject to the constraints  $\sum_{j=1}^i \bar{A}_{ij} = 1$  and  $-\sum_{j=1}^i \bar{A}_{ij} \log \bar{A}_{ij} = \mathcal{E}_i$ . Applying the method of Lagrange multipliers, the optimal configuration that maximizes the diagonal weight while maintaining fixed entropy concentrates mass on the diagonal and distributes the remaining mass uniformly over other positions. Standard variational calculus yields that the maximum achievable diagonal weight decays exponentially with entropy.

**Bounding the Trace.** Applying Lemma 1 to each row:

$$\text{Tr}(\bar{A}) = \sum_{i=1}^N \bar{A}_{ii} \leq \sum_{i=1}^N e^{-\mathcal{E}_i/i}. \quad (22)$$

By first-order Taylor expansion around the mean attention entropy  $\bar{\mathcal{E}} = \frac{1}{N} \sum_{i=1}^N \mathcal{E}_i$ , the sum can be approximated as:

$$\sum_{i=1}^N e^{-\mathcal{E}_i/i} \approx \sum_{i=1}^N e^{-\bar{\mathcal{E}}/N} = N \cdot e^{-\bar{\mathcal{E}}/N}. \quad (23)$$

Therefore, we obtain:

$$\text{Tr}(\bar{A}) \lesssim N \cdot e^{-\bar{\mathcal{E}}/N}, \quad (24)$$

where the attention entropy  $\bar{\mathcal{E}}$  directly controls the convergence rate: higher entropy yields a tighter bound on  $\text{Tr}(\bar{A})$ , leading to accelerated decay of the error  $\|\epsilon^{(K)}\|_F$  over  $K$  recursive steps. Our theoretical analysis establishes that high entropy of the attention matrix accelerated convergence over finite recursive steps. This provides the theoretical foundation for entropy-based head selection strategies.

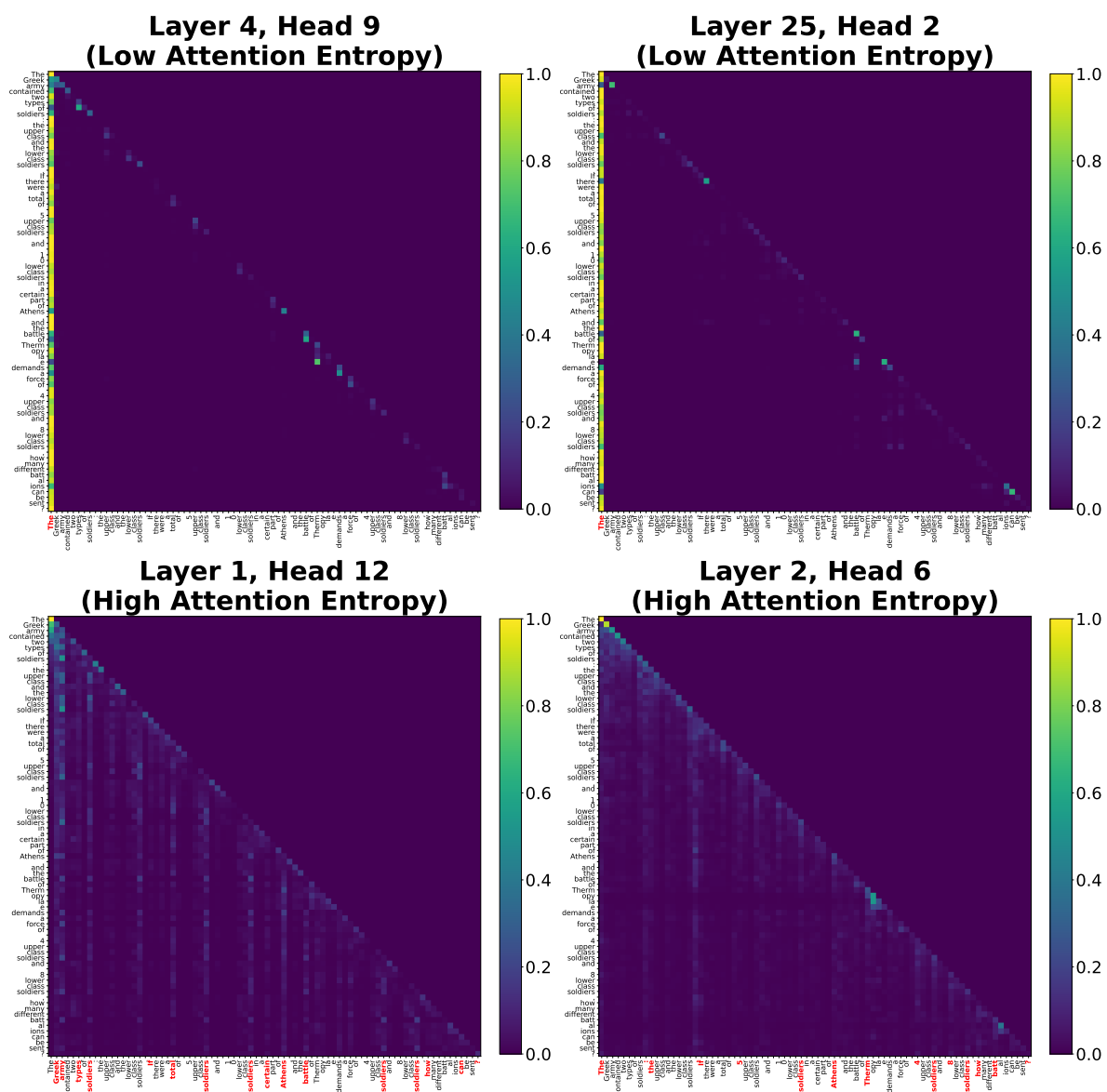


Figure 12: Visualization of attention heatmap in low- and high-entropy heads from Qwen3-0.6B, with a sample from MATH-500. Along the horizontal axis, tokens highlighted in red denote the subset that receives the top 50% of the attention from the final query position (details in Appendix C.3.1 and C.3.2).

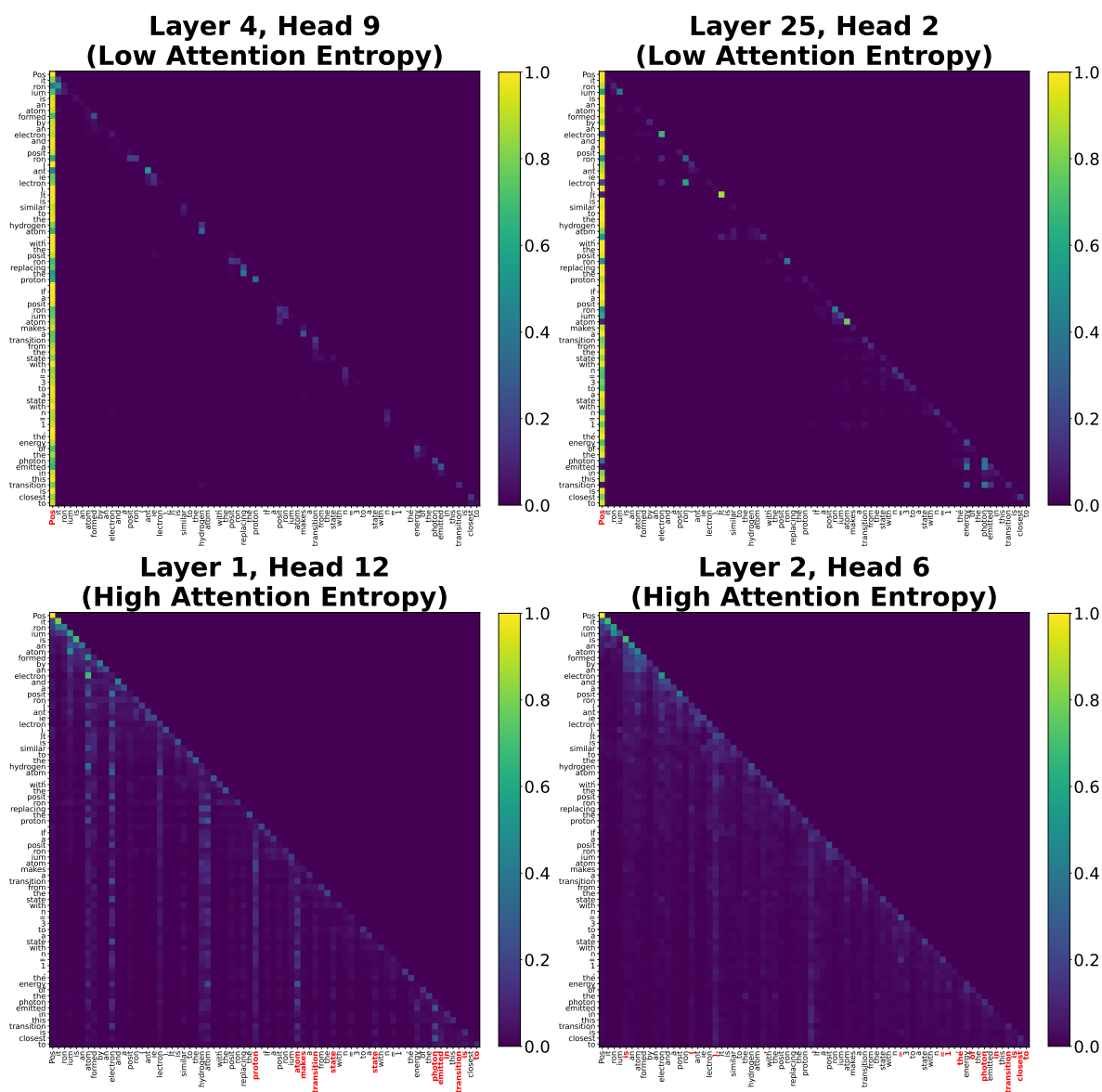
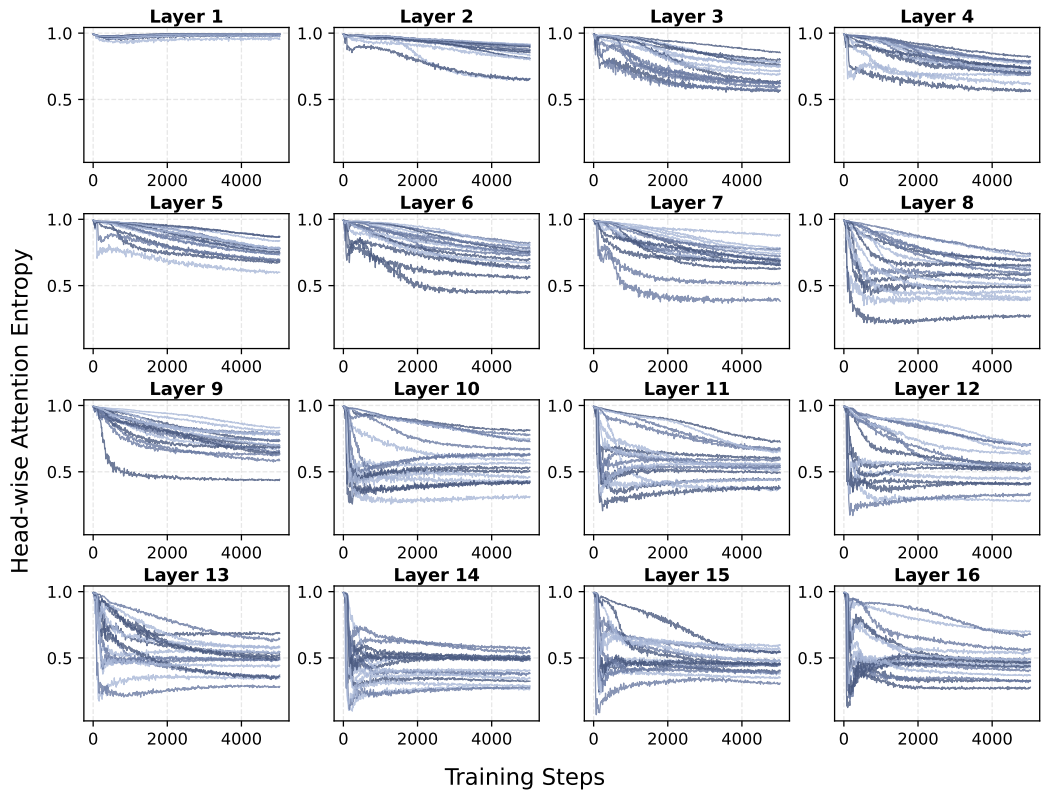
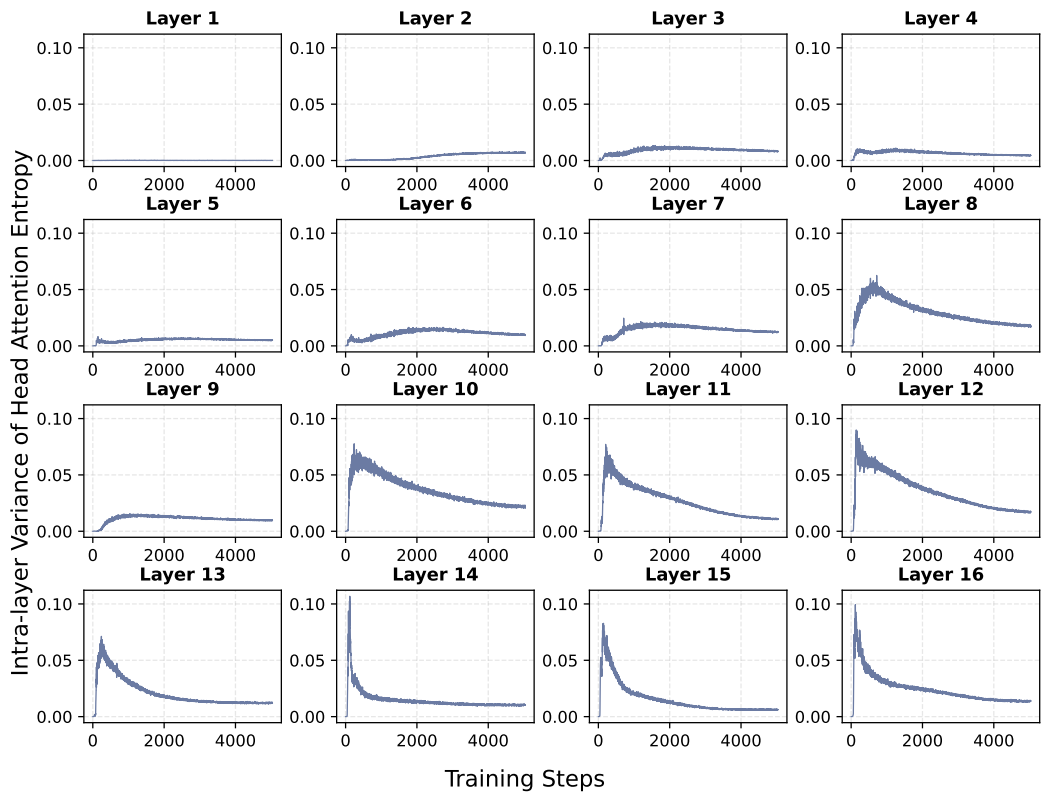


Figure 13: Visualization of attention heatmap in low- and high-entropy heads from Qwen3-0.6B, with a sample from MMLU. Along the horizontal axis, tokens highlighted in red denote the subset that receives the top 50% of the attention from the final query position (details in Appendix C.3.1 and C.3.2).



(a)



(b)

Figure 14: (a) **Head-wise attention entropy trajectories across all layers during training.** Within each layer-specific subplot, individual curves correspond to distinct attention heads. (b) **Intra-layer variance of attention entropy across heads over the course of training (details in Appendix C.4).**

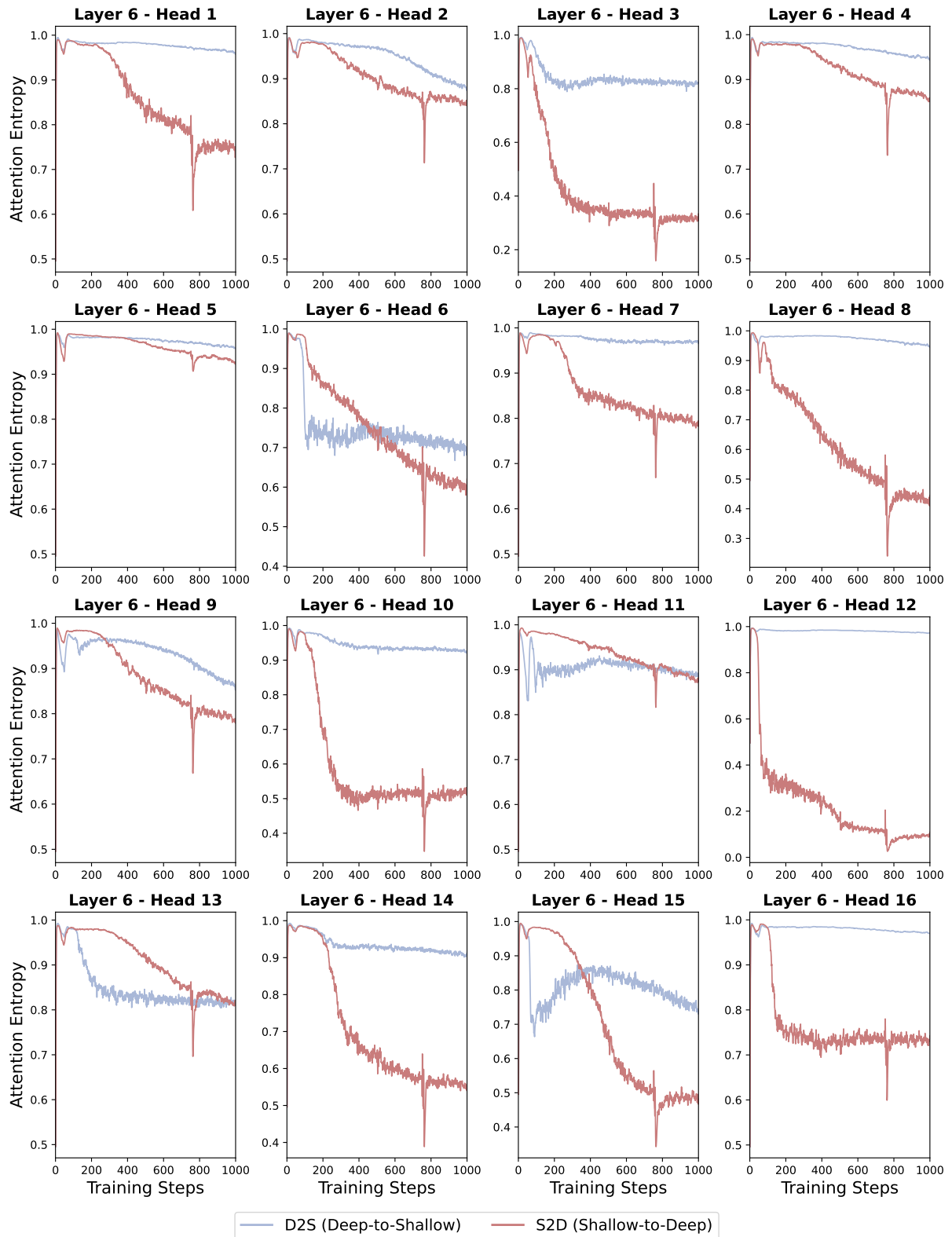


Figure 15: Attention entropy dynamics of Layer 6 heads during training, including the warm-up phase (steps 0–250) and layer selection phase (steps 250–1000), for models trained with D2S and S2D strategies in Table 3.