Polymorphic Universal Transformer

Yilong Chen[1,2], **Zitian Gao**[3], **Yihao Xiao**[3], **Jason Klein Liu**[3], **Xinyu Yang**[3], **Yifan Luo**[3], **Tingwen Liu**[1,2,†], **Haoming Luo**[3], **Zhengmao Ye**[3], **Ran Tao**[3‡], **Bryan Dai**[3]

[1] Institute of Information Engineering, Chinese Academy of Sciences

[2] School of Cyber Security, University of Chinese Academy of Sciences

[3] IQuest Research

chenyilong.616@gmail.com

Abstract

Although the Universal Transformer (UT) mitigates the diminishing returns of standard LLM scaling by decoupling parameter count from depth, it remains constrained by linear computational costs and rigid weight-sharing mechanisms. These limitations lead to severe *functional homogeneity*, which subsequently induces over-smoothing, representation rank collapse, and degraded reasoning performance. In this work, we present the first systematic study of *Compute Distribution Skew*, identifying it as the primary driver of extrapolation failure. This is a pathological phenomenon in ultra-deep recurrent Transformers characterized by a disproportionate distribution of contributions across recurrent steps, resulting in distinct functional states during prefix and suffix processing phases. To address this challenge, we propose the **Polymorphic Transformer**, which aims to achieve functional polymorphism and depth sparsity within a shared-parameter framework. By integrating conditional sparse subspaces, SiLU Attention, and an uncertainty-aware depth scheduler, our architecture mitigates power-method collapse and effectively decouples logical depth from computational cost. Experiments demonstrate that our model significantly enhances representation rank and robustness, achieving complex reasoning performance comparable to baseline while reducing computation by 64.7%.

1 Introduction

The precipitous rise of Large Language Models (LLMs) has been driven primarily by a steadfast adherence to Scaling Laws, in which performance gains are inextricably linked to the exponential expansion of parameter counts and training corpora (Kaplan et al., 2020). However, this scaling trajectory faces critical bottlenecks: hardware limitations, notably the memory wall, limit the deployment efficiency of large-scale architectures (Gholami et al., 2024), while the depletion of high-quality corpora suggests that naive scaling is approaching saturation (Villalobos et al., 2022; Gao et al., 2025b). Consequently, further progress can no longer rely on indiscriminate parameter growth. The central challenge is to extend reasoning and generalization without a commensurate increase in model size, motivating a shift from extensive scaling—adding more parameters—to intensive scaling—extracting deeper computation from existing weights.

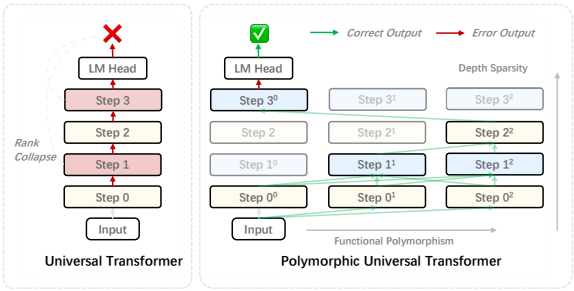

Figure 1: The Universal Transformer can cause rank collapse in depth computation when repeatedly calling the same set of parameters. The Polymorphic Universal Transformer encourages the model to sample sub-states of parameters in each computation and update the sparse samples as the depth increases.

In this pursuit, the **Universal Transformer** (UT) (Dehghani et al., 2018) has re-emerged as an attractive candidate for parameter-efficient scaling. By recursively applying a single set of shared parameters over multiple time steps, UTs decouple model size from logical depth, theoretically enabling the network infinite adequate depth and Turing-completeness, making it suitable for complex iterative reasoning. However, despite these conceptual advantages, recursive architectures have struggled to displace standard deep Transformers. They face a persistent "depth paradox": in prac-

[†] Corresponding author. [‡] Project lead.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 39001–39013

tice, UTs rarely match the performance of non-recursive baselines with equivalent FLOPs (Tay et al., 2023; Saunshi et al., 2025). This performance gap is further exacerbated by a linear $O(L)$ growth in computational cost and KV cache requirements (Pope et al., 2023), where each additional recursive step incurs a uniform latency penalty, rendering deep recurrence practically intractable for latency-sensitive deployment.

The scalability failure of current recursive paradigms arises from intrinsic structural degeneracy, rather than optimization artifacts. First, mathematically, the repeated application of identical linear operators approximates the Power Method, forcing internal representations to converge rapidly toward their principal eigenvectors, systematically eroding the model's expressive diversity as recursion depth increases. Second, a *Compute Distribution Skew* exists: while shallow steps are pivotal for constructing global topology, deep steps often contribute marginally to residual refinement yet consume identical computational resources, resulting in waste of FLOPs.

To address these homogeneous bottlenecks, we propose the **Polymorphic Transformer**. Rather than enforcing strict isomorphism where the operator f_t is identical to f_{t+1}, we introduce **Functional Polymorphism** to induce functional diversity within shared weights. Specifically, by integrating *Conditional Sparse Subspaces*, the model dynamically activates distinct parameter submanifolds across recursion depths, modulating the operator's spectral properties and escaping the Power Method's convergence trap without introducing additional parameters. To further mitigate rank collapse, we replace linear projections with nonlinear *SiLU Attention*, injecting macroscopic nonlinearity directly into the token-mixing process. Finally, to correct compute skew, we introduce **Depth Sparsity**, an entropy-guided heterogeneous scheduler that allocates increased computation to high-uncertainty tokens while allowing low-entropy tokens to exit early, thereby aligning computational expenditure with the non-uniform distribution of information density.

Our comprehensive empirical evaluation confirms that the Polymorphic Transformer resolves the tension between parameter efficiency and expressive power. Across a suite of logic-intensive benchmarks, our architecture significantly outperforms both standard Universal Transformers and depth-matched non-recursive baselines. Furthermore, heterogeneous depth scheduling reduces latency without loss of accuracy by aligning adequate sparse computation with depth. These findings suggest a scalable paradigm where static parameters support dynamic evolution.

2 Preliminaries

We formally define the architectural frameworks, establishing notation for the standard causal Transformer and its recursive generalization, the Universal Transformer. This formulation distinguishes spatial layer stacking from temporal recurrence, framing the spectral analysis in Section 4.

2.1 Standard Transformer Architecture

Given an input token sequence $\mathbf{x} = (x_1, \ldots, x_n)$, we define an embedding function ϕ that maps discrete tokens to d-dimensional continuous representations, and an unembedding function ψ that maps final hidden states to output predictions. Let

$$H_0 = \phi(\mathbf{x}) \in \mathbb{R}^{n \times d} \tag{1}$$

denote the initial hidden representation. A Transformer layer $\mathcal{T}_\theta$ composes Multi-Head Self-Attention (MHSA) and Feed-Forward Networks (FFN) with residual connections:

$$\begin{aligned} \mathcal{T}_\theta(H) &= H + \text{FFN}(\text{LN}(H' + H)), \\ H' &= \text{MHSA}(\text{LN}(H)). \end{aligned} \tag{2}$$

A standard model $\mathcal{M}_{\text{std}}$ of depth L stacks layers with distinct parameters, where the forward pass is the composition:

$$\mathcal{M}_{\text{std}}(\mathbf{x}) = \psi \circ \mathcal{T}_{\theta_L} \circ \cdots \circ \mathcal{T}_{\theta_1} \circ \phi(\mathbf{x}). \tag{3}$$

2.2 Universal Transformer

The Universyinal Transformer decouples logical depth from model size by iterating a shared layer $\mathcal{T}_\theta$. Let $H_0 = \phi(\mathbf{x})$. To distinguish iteration steps, a depth embedding $P(t)$ is injected at each step $t \in \{1, \ldots, T\}$:

$$H_t = \mathcal{T}_\theta(H_{t-1} + P(t)), \text{ yielding } \mathcal{M}_{\text{UT}}(\mathbf{x}) = \psi(H_T). \tag{4}$$

Unlike the heterogeneous composition of standard Transformers, this recurrence mirrors the Power Method. In a linearized regime, repeated application of $\mathcal{T}_\theta$ drives H_t toward the operator's dominant eigenspace, inducing the rank collapse phenomenon analyzed in Section 3.

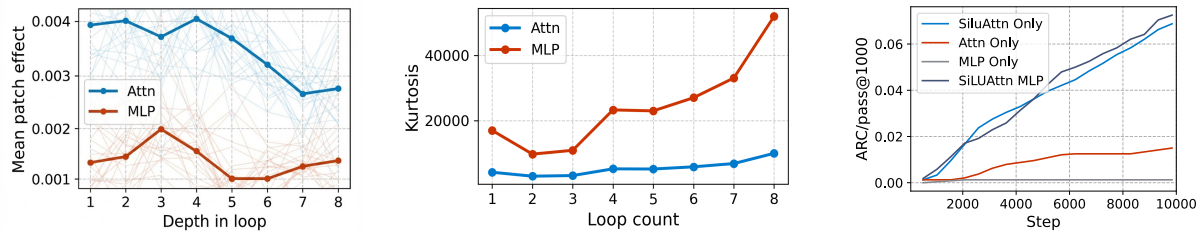

Figure 2: **Left:** Attention vs. MLP causal importance across recurrence depth. **Middle:** Progressive sparsification with loop count. **Right:** Ablation of the effects of different UT model components in ARG-AGI experiments

3 Observation

3.1 Analysis of Universal Transformers

To quantify functional dynamics, we employ a targeted causal tracing protocol on the *ouro-1.4B* model (see Appendix B). The marginal importance, or Patch Effect (PE), of recurrent step r is defined as:

$$\mathrm{PE}(r) = \frac{\mathrm{logit}_{\mathrm{clean}} - \mathrm{logit}_{\mathrm{patched}}(r)}{|\mathrm{logit}_{\mathrm{clean}}|}. \quad (5)$$

Marginal Utility Collapse. Empirical results show that causal utility is strongly front-loaded: early recurrent depths contribute most, while later depths yield diminishing returns. This supports *functional homogeneity* under naive recurrence, where repeatedly applying an identical operator drives representations toward a stable subspace, making additional steps increasingly redundant.

Attention Dominates MLP in Causal Contribution. Module-level patching reveals a consistent gap between components: Attention exhibits substantially larger mean PE than the MLP across depths (Figure 2 Left). This indicates that the effective computation in deep recurrence is primarily mediated by attention-based token mixing, while the MLP plays a comparatively weaker refinement role. Importantly, both components decay with depth, implying that deeper loops do not reliably introduce new causal influence even for the dominant Attention pathway.

3.2 Progressive Sparsification

We further track stepwise statistics of hidden states to characterize how representations evolve across loops (methodology in Appendix B).

Sparsity Amplification. As loop count increases, activations become progressively sparser (Figure 2 Middle). Concretely, the representation distribution concentrates more mass near zero (e.g., a higher zero ratio and/or increased kurtosis), suggesting that more units become inactive while a small subset carries the remaining signal. This trend appears in both Attention and MLP pathways, indicating that sparsification is an emergent property of repeated application of the shared operator rather than a module-specific artifact.

Functional Saturation. Taken together, the joint picture suggests a tight coupling between diminishing causal utility and progressive sparsification: deeper recurrence increasingly suppresses diverse feature channels, pushing computation into a narrow set of active directions. Consequently, additional loops tend to reinforce a low-dimensional regime instead of expanding representational capacity, limiting algorithmic generalization at depth.

3.3 The Critical Role of Nonlinearity

To isolate sources of expressive power, we conduct a component-wise ablation study on the ARC-AGI 1 benchmark (experimental setup in Appendix C, result demonstrated in Figure 2 Right).

Monotonic Performance Degradation. Table 4 demonstrates a strict decline in reasoning efficacy as nonlinear capacity diminishes. The full model achieves 53.75% pass@1. Weakening MLP activations significantly reduces performance to 29.75%, while removing Attention Softmax induces catastrophic collapse to 2.00%. This confirms high sensitivity to operator strength at each recurrent step.

Nonlinearity and Recursive Depth. We posit that nonlinear mapping richness, rather than parameter count, drives performance in loop-based reasoning. Strong operators are essential to prevent the collapse of iterative steps into near-linear transformations, sustaining the *logical depth* required to synthesize complex abstractions.

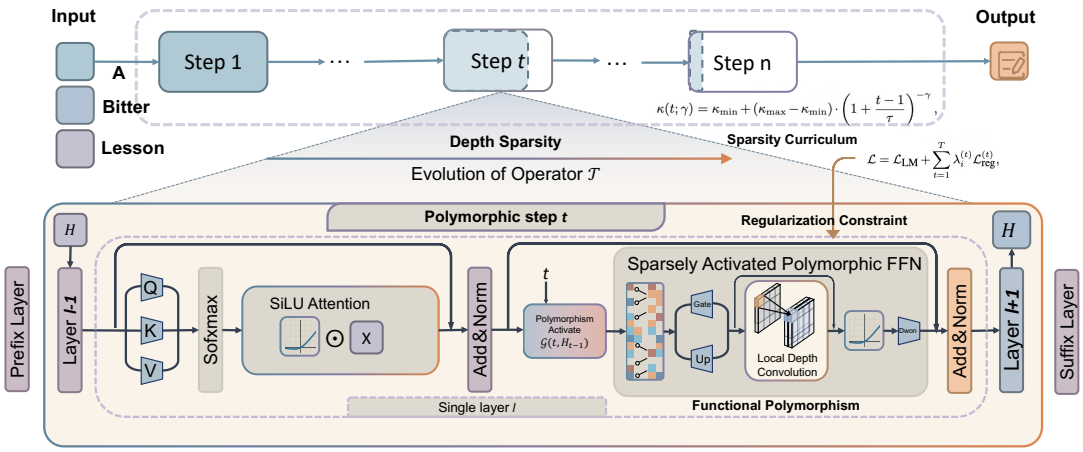

Figure 3: **Overview of the Polymorphic Universal Transformer (PUT) architecture.** At each recurrent step, PUT samples and activates sparse, conditional subspaces of the shared parameter set, enabling dynamic functional diversity across depth. This polymorphic design prevents rank collapse and over-smoothing by ensuring that different computation sub-states are used as depth increases. PUT maintains high-rank, expressive representations and adaptively allocates compute, supporting deeper and more robust reasoning without increasing parameter count.

4 The Polymorphic Transformer

Standard UT suffers from *spectral degeneracy* (rank collapse under repeated shared operators) and *computational homogeneity* (uniform recursion across tokens). To address this, we introduce the **Polymorphic Transformer**, which reinterprets recursion as a structured traversal over an implicit operator family. This is achieved via two core mechanisms: **Functional Polymorphism** (dynamic subspaces, see§4.1) and **Depth Sparsity** (adaptive capacity, see §4.3).

4.1 Functional Polymorphism via Conditional Sparse Subspaces

Standard UT updates $H_t = \mathcal{T}_\theta(H_{t-1})$ behave like a Power Method, compressing states into a low-rank subspace. To preserve expressivity without extra parameters, we induce step-dependent functional variation. We replace the static FFN with a sparsely activated polymorphic module conditioned on recursion depth t and state H_{t-1}:

$$\begin{aligned} Z_{\text{poly}} &= \sigma_{\text{poly}}(t, H_{t-1})\big(H_{t-1}W_{\text{up}} + \mathcal{P}(t)\big), \\ \text{FFN}_{\text{Poly}}(H_{t-1}) &= Z_{\text{poly}}W_{\text{down}}, \end{aligned} \tag{6}$$

where $\mathcal{P}(t)$ is a learnable step embedding. This allows distinct functional realizations under shared parameters, mitigating spectral collapse. Further analysis are provided in Appendix F.

4.2 Nonlinearity Enhancement

We augment the architecture with two non-linear mechanisms to further preserve rank and capture local structure.

SiLU Attention. To disrupt spectral contraction in the attention mechanism, we apply a post-projection nonlinearity:

$$\hat{O} = \text{SiLU}(O) \odot O, \text{ where } O = \text{Attn}(Q, K, V). \tag{7}$$

This parameter-free operation maintains convex token mixing yet alters contraction properties, empirically preserving higher-rank dynamics across recursive steps.

Local Depth Convolution. We insert a causal depthwise convolution within the FFN to enhance local autoregressive modeling (Gao et al., 2025a):

$$U = \text{Dropout}\Big(\text{SiLU}\big(\tilde{U} + \text{Conv1d}_{\text{dw}}(\tilde{U}; \mathcal{K})\big)\Big), \tag{8}$$

where $\tilde{U}$ denotes the FFN intermediate activations and $\mathcal{K}$ the Conv1d kernel size. This adds local inductive bias within each expert's token sequence, while remaining orthogonal to global attention. Implementation details are in Appendix F.

4.3 Sparsity via Adaptive Regularization

Polymorphic ReLU Gating. We implement depth-adaptive capacity allocation using continuous ReLU gating instead of discrete Top-K routing. The polymorphic projection becomes

$$\begin{aligned} \sigma_{\text{poly}}(t, H_{t-1}) &= \text{ReLU}\big(H_{t-1}W_{\text{gate}} + \mathcal{P}(t)\big), \\ \text{FFN}_{\text{Poly}}(H_{t-1}) &= \Big(\sigma_{\text{poly}}(t, H_{t-1}) \odot \big(H_{t-1}W_{\text{up}}\big)\Big)W_{\text{down}} \end{aligned} \tag{9}$$

where sparsity emerges naturally from ReLU thresholding in the gating space.

Compute Distribution Skew. We impose a power-law schedule $\kappa(t)$ to reduce capacity at deeper layers, reflecting a shift from feature expansion to refinement:

$$\kappa(t;\gamma) = \kappa_{\min} + (\kappa_{\max} - \kappa_{\min}) \cdot \left(1 + \frac{t-1}{\tau}\right)^{-\gamma}, \ \gamma > 0. \tag{10}$$

Adaptive Regularization. To enforce the target sparsity, we augment the training objective with a step-wise ℓ_1 penalty on gating activations:

$$\mathcal{L} = \mathcal{L}_{\text{LM}} + \sum_{t=1}^{T} \lambda_i^{(t)} \mathcal{L}_{\text{reg}}^{(t)}, \quad \mathcal{L}_{\text{reg}}^{(t)} = \mathbb{E}_{b,l}\left[\|R_t^{(b,l)}\|_1\right], \tag{11}$$

where $\mathbb{E}_{b,l}[\cdot]$ denotes the empirical expectation over batch and sequence positions. The coefficient $\lambda^{(t)}$ is adjusted online to match the empirical sparsity with $\kappa(t)$:

$$\lambda_{i+1}^{(t)} = \lambda_i^{(t)} \cdot \alpha^{\text{sign}\left(\mathcal{S}_i^{(t)} - (1-\kappa(t))\right)}. \tag{12}$$

This feedback loop stabilizes training and tightly couples computation to the prescribed depth-dependent budget. Full derivations are included in Appendix F.

4.4 Discussion: The High-Attention Regime

Defining the Attention Density Ratio as $\mathcal{D}_{\text{attn}}(t) = \frac{\text{FLOPs}_{\text{attn}}}{\text{FLOPs}_{\text{total}}(t)}$, the monotonic decay of FFN capacity ($\kappa(t)$) implies $\partial_t \mathcal{D}_{\text{attn}} > 0$. This enforces a structural transition: early steps prioritize local FFN synthesis, while deeper steps focus on global attention-based refinement. (see Appendix D for detailed implications).

4.5 Unified Formulation

We define the **Polymorphic Universal Transformer (PUT)** update rule, which incorporates three additional improvements over the baseline: (1) **Decoupled Boundaries**. Prefix and suffix layers use independent parameters, separating semantic encoding/decoding from recurrent latent reasoning and maximizing representational capacity. (2) **Continuous Additive Residuals**. To counter rank collapse and ensure stable propagation, we impose explicit residual connections at each step. (3) **Global Load Balancing.** We adopt an auxiliary-free batch-level strategy to avoid expert saturation and maintain efficient budget utilization (Appendix E). The polymorphic core consists of $\text{Attn}_{\text{Poly}}$ (SiLU-gated attention, Eq. 7) and FFN_{Poly} (sparsely activated module, Eq. 9), enhanced with local convolution (Eq. 8).

5 Experiments

5.1 Baselines

We instantiate models following LLaMA2 (Touvron et al., 2023) at 137M, 156M, and 500M scales. All variants share a training budget to isolate the effects of depth computation (detailed configuration in Appendix G). Recurrent models are evaluated at $2\times$, $3\times$, and $4\times$ the depth of Vanilla. (1) **Vanilla Transformer.** Standard non-recursive causal baseline. (2) **Standard UT.** Weight-sharing baseline iterating a recurrent body with time-step encodings. Computation is uniform across tokens without depth adaptation. (3) **UTMOE.** A Universal Transformer variant replacing dense FFNs with MoE layers to decouple parameter count from active compute, serving as a sparse recurrent baseline. (4) **PUT.** Retains the UT structure but integrates functional polymorphism (Section 4) and uncertainty-aware depth sparsity (Section 4.3). Unlike UT, it induces step-wise diversity and routes only high-uncertainty tokens to deeper steps.

5.2 Result

Overall Performance and Depth Scaling. Table 1 reports that PUT achieves the strongest average score under matched compute, consistently surpassing standard UT across both Loop $\times 2$ and Loop $\times 4$ regimes while remaining competitive with depth-increased dense baselines. Importantly, improvements persist as the number of thinking steps increases: Figure 6 shows a monotonic accuracy gain with additional steps for PUT, whereas UT exhibits weaker scaling and earlier saturation. These trends support our central mechanism: functional polymorphism sustains step-wise operator diversity, enabling effective depth scaling without degrading representations. Additional scaling results are summarized in Table 4.

Training Stability and Data Efficiency. Figure 4 (Left, Middle) compares loss and evaluation perplexity during 50B-token pretraining. PUT converges more smoothly and reaches lower perplexity than UT and the vanilla baseline at similar compute. This behavior aligns with the design in Section 4: SiLU-based non-linear enhancement and local depth convolution reduce iterative contraction, while polymorphic gating varies the active subspace across steps to mitigate oversmoothing. These components improve data efficiency by achieving better perplexity at the same

Model-Params	FLOPs	PPL↓	Commonsense & Reading Comprehension						Continued		LM	Avg.
			SciQ	PIQA	WG	ARC-E	ARC-C	Hella.	LogiQA	BoolQ	Lam.	
Vanilla-Deep	1.00×	25.03	52.5	63.3	50.9	41.8	21.6	29.3	19.7	59.6	36.1	41.6
Depth ×2-197M	1.44×	**22.20**	53.0	63.7	51.8	42.1	**25.1**	**31.5**	20.7	54.3	**41.4**	42.6
Loop ×2												
UT -Deep	1.44×	24.29	**54.0**	63.7	49.3	41.0	21.2	30.4	20.4	50.5	37.0	40.8
UTMoE ×2 -A138M	1.44×	23.50	53.8	63.9	51.5	41.9	22.0	30.8	21.5	57.5	38.2	42.3
PUT ×2 -A138M	1.29×	**22.90**	53.2	**64.1**	52.0	42.0	22.5	31.0	**22.1**	58.0	38.5	**42.6**
Loop ×4												
UT-Deep	2.31×	23.57	52.5	63.0	**53.0**	42.1	21.9	30.5	17.7	54.9	39.2	41.6
UTMoE ×2-A138M	2.31×	22.80	53.5	63.5	52.8	42.3	**23.0**	31.2	20.5	59.0	40.1	42.8
PUT ×2-A138M	2.09×	**22.31**	**53.7**	**63.8**	52.9	**42.8**	22.8	**31.4**	**21.8**	**60.2**	**40.5**	**43.3**
Vanilla -Wide	1.00×	26.31	52.3	63.7	49.9	41.7	24.1	29.4	23.2	53.2	34.6	41.3
Depth-210M	1.34×	**22.42**	53.5	64.2	50.4	42.2	24.6	31.2	18.0	57.1	40.3	42.4
Loop ×2												
UT -Wide	1.34×	25.03	51.7	63.4	50.0	41.1	21.4	30.6	21.7	58.4	35.5	41.5
UTMoE ×2-A156M	1.34×	23.81	**54.0**	64.0	50.1	41.7	22.4	30.8	**22.7**	58.6	37.7	42.4
PUT ×2-A156M	1.11×	**22.87**	52.2	**64.6**	50.4	41.2	21.9	31.7	21.9	58.0	37.4	42.7
Loop ×4												
UT-Wide	2.03×	24.29	52.2	64.4	49.3	42.0	22.6	30.1	22.6	59.1	39.7	42.4
UTMoE ×2-A156M	2.03×	22.65	52.9	63.9	51.4	42.2	22.6	**31.8**	21.0	60.1	39.8	42.9
PUT ×2-A156M	1.57×	**19.69**	53.5	63.2	**52.5**	**42.7**	**22.1**	**31.8**	21.2	**60.7**	**41.4**	**43.2**

Table 1: Comprehensively evaluate the basic capabilities of models with different activated parameters. In particular, PUT ×4-180M represents a model with 180M total parameters using PUT to think total 4 steps.

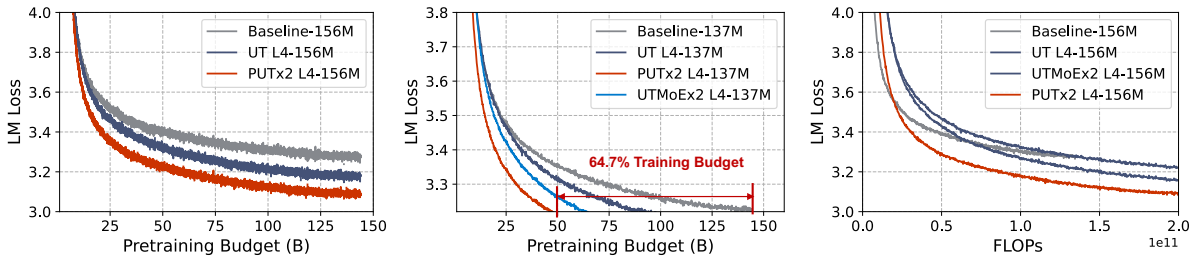

Figure 4: **Left:** Loss curves for 180M-models pre-trained on 50B tokens. **Middle:** Eval Perplexity curves for 180M-models pre-trained on 50B tokens. **Right:** Eval Perplexity for 240M-models with Training FLOPs.

training budget.

Compute Allocation and Efficiency. Figure 4 (Right) shows that PUT dominates the perplexity–FLOPs frontier across selection strategies. The gain is amplified at higher loop counts, indicating that the model translates extra recursion into measurable accuracy rather than redundant computation. The learned step allocation is visualized in Figure 5 (Middle, Right): step encoding weights concentrate on earlier steps while preserving nontrivial mass at later steps, and router weights remain well-distributed across steps. This pattern matches the objective of depth sparsity in Section 4.3, where computation is concentrated when refinement is most beneficial.

Depth Scaling Behavior. Figure 6 compares average accuracy under increasing thinking steps after 50B-token pretraining. Standard UT shows limited depth scalability: performance improves slightly at shallow depth and quickly saturates as recursion increases, consistent with spectral degeneracy under shared operators. UTMoE delays this saturation by introducing conditional computation, but the gains remain sublinear. In contrast, PUT exhibits a monotonic and sustained improvement as depth increases, with no visible saturation up to eight steps. The widening gap at larger depths indicates that additional computation is effectively utilized rather than becoming redundant. These results validate that functional polymorphism preserves step-wise operator diversity, enabling meaningful refinement at greater recursion depth.

5.3 Ablation Studies

We ablate the modules introduced in Section 4 on PUT ×4-156M after 150B-token pretraining. Table 2 reports perplexity at matched compute, to-

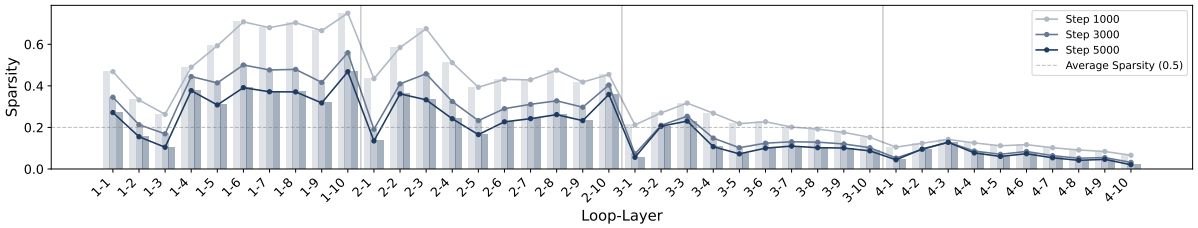

Figure 5: ReLU sparsity variation across different loops and layers. The x-axis represents loop-layer combinations (e.g., '1-5' denotes layer 5 of loop 1), and the y-axis shows sparsity values. The three curves correspond to training steps 1000, 3000, and 5000, with the horizontal dashed line indicating the average sparsity of 0.5.

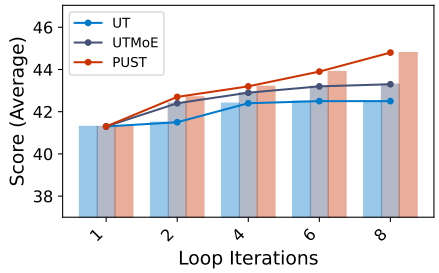

Figure 6: Average accuracy after training 50B tokens for the PUT and Standard UT models (180M, 240M, 460M) under different thinking step configurations.

gether with the dense and UT baselines. We additionally track spectral proxies and stepwise causal contribution. More details in Appendix H.

Functional polymorphism. Removing functional polymorphism yields the largest drop, increasing perplexity from 19.69 to 23.89. This confirms that step-invariant recurrence quickly becomes redundant under shared weights, while conditional activation sustains step-wise operator variation and preserves late-step contribution. In our diagnostics, rank- and entropy-related measures deteriorate most sharply in this setting, indicating accelerated homogenization.

SiLU Attention. Disabling SiLU Attention causes a smaller but consistent regression to 19.92. This supports the role of post-attention nonlinearity in counteracting contraction from repeated convex token mixing, helping maintain non-trivial dynamics in later steps. We observe reduced deep-step contribution without SiLU, consistent with increased over-smoothing.

Local depth convolution. Removing local depth convolution increases perplexity to 20.33. The effect is moderate yet stable, suggesting that the causal depthwise convolution provides an orthogonal locality bias that complements global attention and supports refinement under recurrence.

Configuration	FLOPs	Perplexity ↓
PUT ×2-A156M (Loop×4)	1.57×	19.69
w/o Functional Polymorphism	1.57×	23.89 (+1.58)
w/o SiLU Attention	1.56×	19.92 (+0.23)
w/o Local Depth Convolution	1.56×	20.33 (+0.64)
w/o Depth Sparsity	1.57×	21.85 (+2.16)
PUT Router: ReLU	1.57×	19.69 (-)
PUT Router: Top-K	1.55×	19.84 (+0.15)
PUT Loss: Auxloss	1.57×	20.06 (+0.37)
UT-156M (Loop×4)	1.57×	24.29 (+4.59)
Vanilla-156M	1.00×	25.03 (+5.34)

Table 2: Ablation study on PUT ×2-A156M (Loop×4).Due to gradient distortion caused by depth, PUT uses the Auxfree method with an 1e-2 update rate.

Depth sparsity. Turning off depth sparsity both increases the compute and degrades quality: FLOPs rise from 3.29 to 4.70, while perplexity increases to 21.85. This aligns with our compute-skew diagnosis: uniform recursion wastes steps on easy tokens without improving uncertainty resolution. Depth sparsity yields a better trade-off.

Routing and stabilization variants. Under identical FLOPs, Top-K routing is slightly worse than the ReLU router baseline, and replacing our stabilization with auxiliary loss further degrades perplexity to 20.06. These results indicate that continuous gating is sufficient for stable conditional computation in our setting, while auxiliary load balancing is not required.

5.4 Analysis

Transferable Depth Sparsity. Figure 6 shows that **Depth Sparsity** functions as an active depth–width exchange mechanism rather than a static constraint. By extending the recursive horizon T, PUT front-loads computation at early steps, yielding an effective FFN width at t=1 that exceeds the physical parameter scale and maximizes initial feature

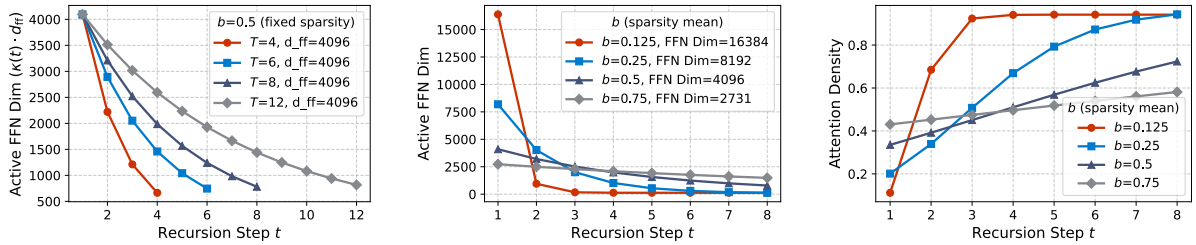

Figure 7: **Sparsity and Attention Dynamics.** Left/Center: Power-law decay of active FFN capacity () under varying depths and budgets. Right: Complementary rise in attention density, illustrating the shift from capacity-heavy expansion to context-heavy refinement.

expansion. As depth increases, local FFN capacity is progressively reduced and traded for temporal refinement, which is reflected by the monotonic increase in **Attention Density**. This transition indicates a smooth shift from high-capacity local processing to dense integration of global context. Overall, depth sparsity enables PUT to convert temporal redundancy into a variable-width architecture, jointly optimizing early expressivity and late-stage contextual modeling.

Depth Sparsity as an Efficient Compute–Performance Trade-off. Table 3 analyzes the sparsity curriculum by varying the decay parameters γ and τ. Slower decay schedules retain more active tokens in later steps, increasing FLOPs while consistently improving perplexity, whereas aggressive decay rapidly suppresses late computation and degrades performance. The default configuration achieves a favorable balance between perplexity and FLOPs with approximately 50% sparsity. The monotonic relationship between sparsity and performance confirms that recurrent steps remain beneficial only for a subset of tokens, validating depth sparsity as a principled mechanism for allocating compute to regions with the highest marginal gain.

Nonlinearity Placement and Conv-SiLU Design. Table 5 disentangles the effects of attention nonlinearity and Conv-SiLU under a fixed FLOPs budget. Removing all attention nonlinearity significantly degrades perplexity, confirming its central role in disrupting spectral contraction. Among partial variants, applying nonlinearity to the value and output projections yields the most substantial gains, outperforming query- or key-only designs. Conv-SiLU benefits from a lightweight local bias: small causal kernels and mild expansion ratios match or exceed larger configurations at comparable compute cost, whereas additional dropout provides no

Power-Law Decay (γ, τ)	FLOPs	PPL ↓	Sparsity (%)
$\gamma = 1.0,\ \tau = 4$ *(default)*	1.57×	19.69	50.0
$\gamma = 0.5,\ \tau = 4$	2.38×	19.07	76.5
$\gamma = 2.0,\ \tau = 4$	1.26×	21.68	40.2
$\gamma = 1.0,\ \tau = 2$	1.46×	20.63	46.5
$\gamma = 1.0,\ \tau = 8$	2.29×	19.32	73.0

Table 3: Ablation of Sparsity Curriculum (Power-Law Decay) hyperparameters γ and τ. Varying the schedule affects sparsity, FLOPs, and PPL.

benefit. These results indicate that nonlinearity is most effective when applied to value-space transformations, and that shallow local convolutions suffice to enhance functional diversity.

Layer-wise Evolution of Depth Sparsity. Figure 5 further shows that ReLU-induced sparsity follows a stable depth-wise structure. Early loops remain relatively dense, while later loops progressively suppress a larger fraction of channels, concentrating computation in early refinement stages. This pattern remains consistent across training, indicating convergence to a stable allocation strategy rather than oscillation or collapse. Together, these observations confirm that depth sparsity emerges as an organized, depth-dependent schedule that enables deeper recurrence without proportional increases in compute.

6 Conclusion

We propose PUT, a dynamic architecture that redefines recursive computation. By enforcing **Functional Polymorphism** through conditional sparse subspaces and SiLU attention, PUT effectively breaks the **Power Method** convergence trap inherent in standard weight sharing. Our uncertainty-aware depth scheduling further decouples logical depth from computational cost, allocating resources to critical tokens. Experiments demonstrate that PUT significantly improves representa-

tion rank and reasoning robustness compared to Standard UT, validating our theoretical premise that structural diversity is key to scalable recursion.

7 Limitations

Despite the promising improvements brought by the Polymorphic Universal Transformer, our approach still faces several limitations. First, the adaptive routing and sparsity scheduling mechanisms introduce additional complexity and computational overhead, especially during training, which may limit scalability in extremely large models or real-time applications. Second, the effectiveness of our method depends on careful tuning of hyperparameters such as sparsity decay rates and gating thresholds, which may not generalize optimally across different tasks or domains. Future work is needed to address these challenges and further improve the robustness and practicality of polymorphic recurrent architectures. This manuscript was prepared with the assistance of AI tools for language polishing and structural refinement; all scientific content and conclusions are solely the responsibility of the authors.

Ethical Considerations

Our work adheres to ethical AI principles through three key aspects: 1) All experiments use publicly available datasets with proper anonymization, 2) The enhanced parameter efficiency reduces environmental impact from model training/inference, and 3) Our architecture-agnostic approach promotes accessible performance improvements without proprietary dependencies. We acknowledge potential risks of enhanced reasoning capabilities being misapplied, and recommend implementing output verification mechanisms when deploying PUT-based systems. Our work is committed to advancing accessible and efficient NLP technologies, fostering a more inclusive and automated future for AI.

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).

References

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Valentino Braitenberg. 1986. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, Cambridge, MA.

Yanxi Chen, Xuchen Pan, Yaliang Li, Bolin Ding, and Jingren Zhou. 2024. Ee-llm: Large-scale training and inference of early-exit large language models with 3d parallelism.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

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*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2018. Universal transformers. *arXiv preprint arXiv:1807.03819*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *Preprint*, arXiv:2311.01460.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. 2024. Layerskip: Enabling early exit inference and self-speculative decoding. page 12622–12642.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270.

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

2023. A framework for few-shot language model evaluation.

Zitian Gao, Lynx Chen, Yihao Xiao, He Xing, Ran Tao, Haoming Luo, Joey Zhou, and Bryan Dai. 2025a. Universal reasoning model. *Preprint*, arXiv:2512.14693.

Zitian Gao, Haoming Luo, Lynx Chen, Jason Klein Liu, Ran Tao, Joey Zhou, and Bryan Dai. 2025b. What makes diffusion language models super data learners? *Preprint*, arXiv:2510.04071.

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. *Preprint*, arXiv:2502.05171.

Felix Alexander Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12:2451–2471.

Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W Mahoney, and Kurt Keutzer. 2024. Ai and memory wall. *IEEE Micro*, 44(3):33–39.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *Preprint*, arXiv:2412.06769.

A Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Benoît Savary, Charles Bamford, Devendra Singh Chaplot, Daniele de la Casas, Emily Bressand Hanna, François Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

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*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *Preprint*, arXiv:1909.11942.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. 2024. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Yongyu Mu, Yuzhang Wu, Yuchun Fan, Chenglong Wang, Hengyu Li, Qiaozhi He, Murun Yang, Tong Xiao, and Jingbo Zhu. 2024. Cross-layer attention sharing for large language models. *Preprint*, arXiv:2408.01890.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. 2025. Olmoe: Open mixture-of-experts language models. *Preprint*, arXiv:2409.02060.

Kei-Sing Ng and Qingchen Wang. 2024. Loop neural networks for parameter sharing. *Preprint*, arXiv:2409.14199.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Mario Neumann, Rodolphe Jenatton, António Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 34, pages 8583–8595.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. *arXiv preprint arXiv:2502.17416*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, et al. 2024. Dolma: An open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 15725–15788.

Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Tran, Dani Yogatama, and Donald Metzler. 2023. Scaling laws vs model architectures: How does inductive bias influence scaling? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12342–12364.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint*.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2022. Will we run out of data? limits of llm scaling based on human-generated data. *arXiv preprint arXiv:2211.04325*.

Jingcun Wang, Yu-Guang Chen, Ing-Chao Lin, Bing Li, and Grace Li Zhang. 2024. Basis sharing: Cross-layer parameter sharing for large language model compression. *Preprint*, arXiv:2410.03765.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text, NUT@EMNLP 2017, Copenhagen, Denmark, September 7, 2017*, pages 94–106. Association for Computational Linguistics.

F. Xue, Z. Zheng, Y. Fu, J. Ni, and W. Zhou. 2024. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics.

Jun Zhang, Desen Meng, Ji Qi, Zhenpeng Huang, Tao Wu, and Limin Wang. 2024. p-mod: Building mixture-of-depths mllms via progressive ratio decay. *Preprint*, arXiv:2412.04449.

Yutian Zhou, Tao Lei, Henry Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. 2022. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*.

A Appendix

A Related Work

Recurrent Computation The concept of recurrence in machine learning traces back to foundational works on neural computation (Braitenberg, 1986) and LSTM networks (Gers et al., 2000). Modern extensions integrate recurrence into transformers through depth recurrence (Dehghani et al., 2019; Lan et al., 2020; Ng and Wang, 2024). Recent works have rediscovered this idea for implicit reasoning (Deng et al., 2023; Hao et al., 2024) and test-time scaling (Geiping et al., 2025). In contrast, PUT establishes a general-purpose recursive reasoning framework within individual layers and designs the Residual Thinking Cnnection (RTC) for enhanced capability.

Dynamic Computation Allocation Dynamic Computation Allocation, like MoE, reduces computational overhead by activating only a subset of networks (Fedus et al., 2022; Riquelme et al., 2021; Zhou et al., 2022; Jiang et al., 2024; Xue et al., 2024). Some works focus on elastic computation in depth, such as early exit (Elhoushi et al., 2024; Chen et al., 2024), parameter sharing (Mu et al., 2024; Wang et al., 2024), or using token-routing for dynamic layer skipping (Zhang et al., 2024). Inspired by these works, ITT designs an elastic deep thinking architecture with Adaptive Token Routing (ATR) for efficient and adaptive computational resources allocation.

B Experimental Protocols

Causal Tracing. We evaluate a model trained with recurrent steps. A three-stage patching procedure (clean, corrupted, patched) isolates the contribution of each recurrent step to the final prediction, as measured by the PE metric.

Statistical Monitoring. To quantify information density, we compute the -norm, empirical entropy, and effective matrix rank of activations at each step. We simultaneously track the proportion of non-zero elements and gradient norms for Attention and MLP weights to measure functional activity.

C Ablation Protocols

To isolate the sources of expressive power within the Universal Reasoning Model, we conduct a systematic ablation study on the ARC-AGI 1 benchmark. We progressively simplify or remove key nonlinear architectural components—specifically replacing SiLU with simpler activations (e.g., ReLU), removing Short Convolutions, and ablating the Attention Softmax. All models are evaluated using pass@n metrics under an identical recurrent setup (8 inner loop steps, adaptive outer loop), allowing us to gauge how nonlinearity influences the efficacy of iterative computation.

D Implications of the High-Attention Regime

This emergent computational regime aligns naturally with the requirements of multi-step reasoning. In early recurrence stages, where $\mathcal{D}_{\text{attn}}$ is low, the model allocates most of its capacity to FFNs, facilitating knowledge retrieval, feature synthesis, and local pattern processing.

As recursion deepens and $\mathcal{D}_{\text{attn}}$ increases, computation shifts toward the attention mechanism. At this stage, the dominant objective is no longer feature expansion but global consistency: resolving long-range dependencies, maintaining coherence across iterative updates, and suppressing semantic drift. By explicitly constraining FFN capacity in deeper steps via the Sparsity Curriculum, we induce a *global refinement phase* in which attention heads become the primary computational pathway. This design mitigates the instability and representation collapse commonly observed in deeply recursive transformers, while preserving expressive power where it is most needed.

E Global Load Balancing Strategy

To resolve the deep-layer load-balancing problem common in recurrent models, we employ a **global, batch-level strategy** that utilizes auxiliary-free bias correction. During training, the regularization loss for depth sparsity is adaptively scaled based on the batch-wise utilization of recurrence steps and experts. This mechanism allows the model to self-organize its compute allocation, preventing pathological saturation or starvation of experts, and ensuring efficient use of the active compute budget at scale.

F Additional Details of the Polymorphic Transformer

G Baseline Configurations

To provide a fair and controlled comparison, we instantiate models at three parameter scales: 180M,

Model-Params	FLOPs	PPL↓	Commonsense & Reading Comprehension						Continued		LM	Avg.
			SciQ	PIQA	WG	ARC-E	ARC-C	Hella.	LogiQA	BoolQ	Lam.	
Vanilla-500M	1.00×	19.45	68.5	64.2	47.5	48.0	25.4	38.2	24.1	54.6	38.5	45.4
Depth ×2-1.09B	1.69×	17.10	73.2	66.5	49.1	51.5	28.8	41.5	26.5	57.2	42.8	48.6
Loop ×4												
UT -500M	3.07×	18.20	70.8	65.5	48.0	49.5	26.5	39.8	25.5	55.8	40.2	46.8
UTMoE ×8 -A500M	3.07×	17.55	72.5	66.8	49.3	51.0	28.2	41.0	26.8	57.5	42.0	48.3
PUT ×8 -A500M	2.87×	16.90	74.1	67.5	49.8	52.8	29.1	42.1	27.5	58.8	43.5	49.5
Loop ×8												
UT-500M	5.83×	17.65	72.0	66.0	48.8	50.8	27.5	40.5	26.2	56.5	41.5	47.7
UTMoE ×8-A500M	5.83×	16.85	74.5	67.8	50.1	52.5	29.5	42.5	27.8	59.0	44.0	49.7
PUT ×8-A500M	5.49×	**16.27**	**76.0**	**68.8**	**50.7**	**54.0**	**30.5**	**43.3**	**28.5**	**59.9**	**45.2**	**50.8**

Table 4: Comprehensively evaluate the basic capabilities of models with different activated parameters. In particular, PUT ×4-180M represents a model with 180M total parameters using PUT to think total 4 steps.

Nonlinearity Configuration	Perplexity ↓
PUT ×2-A156M (Loop×4)	**10.42**
Location of SiLU in Attention	
SiGLU on Q only	10.96 (+0.54)
SiLU on K only	10.91 (+0.49)
SiLU on V only	10.51 (+0.09)
SiLU on O only	**10.42**
Conv-SiLU Hyperparameters	
Kernel size $k = 1$	10.56 (+0.14)
Kernel size $k = 2$ (default)	**10.42**
Kernel size $k = 3$	10.48 (+0.06)

Table 5: Ablation of nonlinearity and Conv-SiLU hyperparameters.

240M, and 520M, with hidden dimensions of 1024, 1536, and 2048 respectively. Unless otherwise specified, all variants at a given parameter scale share the same initialization, optimizer, training corpus, and total token budget, enabling us to attribute performance differences to the depth computation scheme rather than data or parameter count.

H Experiments Details

Training. All models are trained using Megatron (Shoeybi et al., 2019) on H200 GPUs, with a sequence length of 4096 and a global batch size of 256. We adopt a cosine learning rate decay schedule, starting from a peak learning rate of 5×10^{-4} and decaying to 10% of the initial value, with a warmup over the first 5% of total tokens. Model weights are initialized with standard deviation $\sqrt{2/(5d)}$, where d is the hidden size. Mixture-of-experts models use an auxiliary loss coefficient of 10^{-3}. Each model is trained with a data-to-parameter ratio of 200, on the same data volume, from random initialization.

Model Setting	L.2-Small	L.2-Middle	L.2-Large
hidden size	1024	1536	2048
intermediate size	2560	2560	4096
attention heads	32	32	32
num kv heads	32	16	32
layers	8	8	8
# Params	162M	230M	466M

Table 6: Detailed configuration, activation parameters, and total parameters of the models included in our study. L.2-162M represents the LLaMA-2 architecture model with 162M total parameters.

Data. To pretrain PUT models and baseline models, we use OLMo 2 Mix 1124, which matches the OLMoE (Muennighoff et al., 2025) pretraining set. It combines DCLM (Li et al., 2024), Dolma 1.7 (Soldaini et al., 2024) subsets (arXiv, OpenWebMath, Algebraic Stack, peS2o, Wikipedia), and StarCoder (Li et al., 2023). The dataset totals 3.4 trillion training tokens, and we sample 2 million tokens for validation.

Evaluation. We employed the lm-evaluation-harness (Gao et al., 2023) to evaluate our models. For common sense and reading comprehension tasks, we report 0-shot accuracy for SciQ (Welbl et al., 2017), PIQA (Bisk et al., 2020), WinoGrande (WG) (Sakaguchi et al., 2020), ARC Easy(ARC-E) (Clark et al., 2018), and 10-shot HellaSwag (Hella.) (Zellers et al., 2019), alongside 25-shot accuracy for ARC Challenge (ARC-C) (Clark et al., 2018). For continued QA and text understanding, we report 0-shot accuracy for LogiQA (Liu et al., 2020), 32-shot BoolQ (Clark et al., 2019), and 0-shot LAMBADA (Lam.) (Paperno et al., 2016). All reported results are calculated with the mean and stderr of multiple experiments.