Latent Reasoning & Iterative Refinement in Large Language Models

Yufei Gu

December 14, 2025

香港科技大学(广州)
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

xLeaf Lab
Machine Learning Foundations

Before we start: What's wrong with CoT reasoning?

Chain-of-Thought (CoT): A model generates a sequence of intermediate reasoning steps (in tokens) that connect the input to the final answer. However, CoT has limitations in:

- Faithfulness issues: Models don't always say what they think in reasoning chains [1].
- Scalability limits: Long token chains introduce latency, computational overhead, and diminishing returns [2].
- Reversal curse: Forward-only causal generation limited by the next-token prediction paradigm, and cannot refine previously generated tokens [3].

In this paper trip, we will briefly visit:

1. Allocation strategies of compute budget from skipping layers/tokens to refine tokens/features iteratively.
2. Using hidden states as inputs and scaling up the reasoning depth in the latent space.
3. Improving stability with feature-token interpolation.
4. The diffusion paradigm in language modelling and how advanced unmasking/remasking strategies are designed.

Table of Contents

1. Scable Reasoning Depth in LLMs: From Layer Skipping to Token Refinement

In the human thinking process, simple questions can be answered quickly and complex questions require more time for reasoning.

- **Can LLMs use less parameters on some inputs?**
- There exists dozens of studies on Dynamic Depth (e.g., Early Exit, Skip Layer) and Dynamic Width (e.g., Pruning CNN channels or Attention Heads, MoE). Let's review one of them:

Not All Layers of LLMs are Necessary during Inference [2024.3]

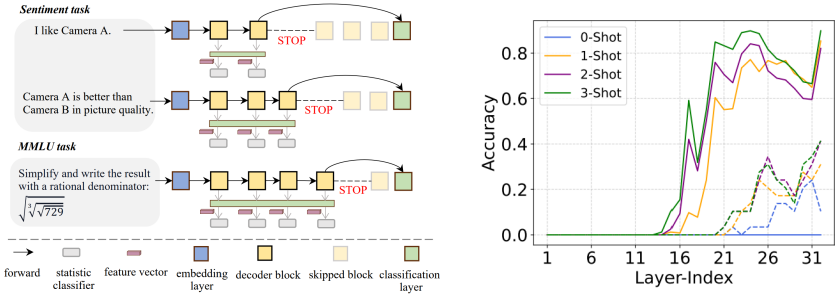


Figure: AdaInfer processes input with inference stopping at different decoding layers [4]. The *solid line* represents accuracies on sentiment analysis and *dashed line* on MMLU with the LLaMA-2-7B model.

- AdaInfer compromises a binary Classifier that predicts whether to stop the inference at each decoding layer.
- Feature selection is performed on hidden states:
  - Confidence Gap: P(top token) - P(second token).
  - Top Probability: P(top token).
  - Cosine Similarities: between current and previous block's attn/mlp/block outputs.
- AdaInfer exhibits an average 17.8% pruning ratio during inference with $< 1\%$ degradation on performance.
- Feature analysis suggests that Gap and Top Prob can serve as universal features for the inference-stopping signal.

Mixture-of-Depths: Dynamically Allocating Compute [arXiv 2024.4]

How can we dynamically allocate the compute budget at token-level?

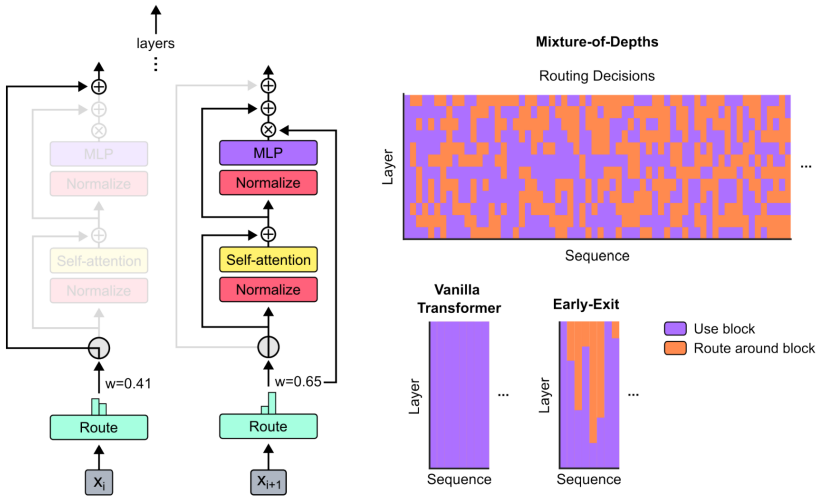


Figure: Mixture-of-Depths (MoD) Transformer. A block-wise router chooses tokens between a standard block's computation or a residual connection [5].

The Mixture-of-Depths Strategy:

- Limit the number of tokens in a sequence that can participate in a block's computation.
- Use a per-block router to emit a scalar weight for each token; Identify the top-k weights to select tokens.
- Since some tokens are routed around the block, the total FLOP footprint is reduced compared to vanilla/MOE transformers.
- A 220M MoD model slightly outperforms the isoFLOP optimal 220M baseline but steps 66% faster [5].
- Expert or Top-k token routing schemes chooses over- or under-process some tokens or load-imbalance.

Mixture-of-Recursions: Dynamic Recursive Depths for Adaptive Token-Level Computation [NIPS 2025]

What if some tokens are routed recursively around parameter blocks?

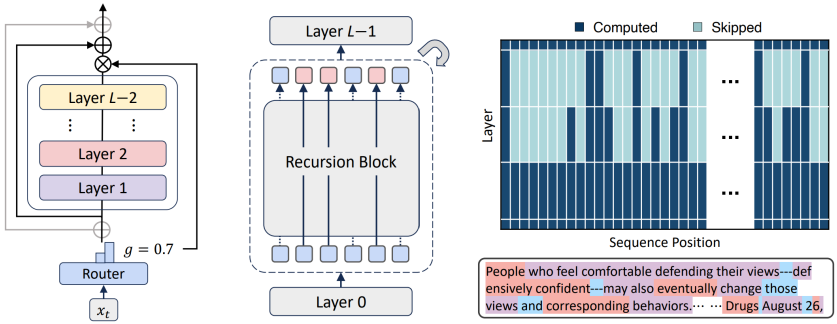
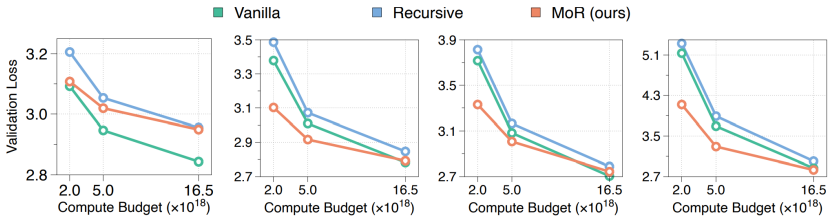


Figure: Mixture-of-Recursions (MoR) Transformer. A router decides whether each token should pass a fixed stack of layers or exit, where right-below shows the number of recursion steps in different colors [6].

The Mixture-of-Recursions Strategy:

- Assign token-specific recursion steps to adaptively concentrate computation on more challenging tokens.
- Expert- or Token-choice routing chooses reasoning depth at each recursion step or at the outset for each token.
- Unify efficiency paradigms: parameter sharing, token-level adaptive thinking depth, and memmory KV caching.



(a) 135M-based model (b) 360M-based model (c) 730M-based model (d) 1.7B-based model

Figure: MoR consistently outperforms recursive baselines on validation loss, and matches vanilla transformers at larger scales with $\sim 1/3$ parameters [6].

Think-at-Hard: Selective Latent Iterations [arXiv 2025.11]

Over 85% of next-tokens are correctly predicted at the first iteration.
$\Rightarrow$ *Deeper iterations should only refine these 'Hard' tokens.*

Figure: (a) Uniform latent iteration can fix wrong predictions but also correct ones (b) Next-token prediction accuracy on fine-tuned Qwen3-1.7B [7].

Think-at-Hard (TaH) Strategy:

- Backbone Model Design: LoRA adapter only for iterations $d > 1$, and MLP iteration decider for routing each token.

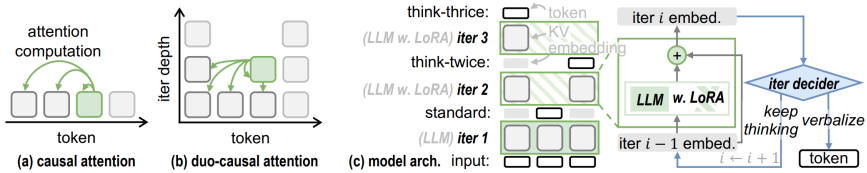


Figure: Think-at-Hard (TaH) uses LoRA at deeper iterations to shift from next-token prediction to hard-token refinement [7].

- **Duo-causal Attention:** Extend attention to two dimensions: previous positions and shallower iteration depths.
- **Oracle Iteration Policy:** A token is classified as easy if the reference (base) model correctly predicts it in one forward pass.
- Prune one model layer to match the baseline parameter count.
- Two-stage Training: Backbone supervision + Decider imitation under frozen backbone.

With continuation threshold 0.9 and $\sim 6\%$ tokens iterated twice, TaH delivers $4.0 - 5.4\%$ accuracy gains on math benchmarks.

2. Latent Reasoning: Looping & Continuity

Chain-of-Thought, but in latent space, than vocabulary space.

Training LLMs to Reason in a Continuous Latent Space [2024.12] [COLM 2025]

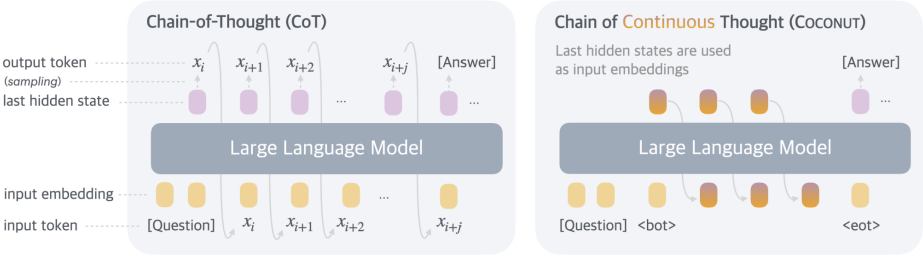


Figure: Chain of Continuous Thought (Coconut) regards the last hidden state as a representation of the reasoning state and directly uses it as the next input embedding [8].

- Include special tokens <bot> and <eot> to mark the beginning and end of latent thoughts.
- SFT on regular CoT data in the initial stage; Then at the k-th stage, replace the first k reasoning steps with $k \times c$ continuous thoughts for hyperparameter c.
- Compute normal negative log-likelihood loss on answers (mask on questions and latent thoughts).

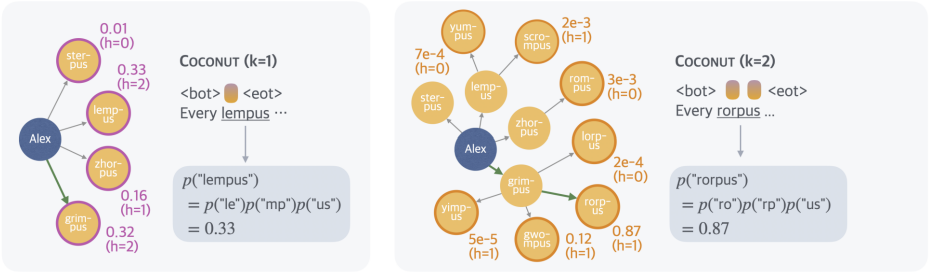


Figure: Coconut searches the latent space with higher reasoning steps [8].

Reasoning with Latent Thoughts: On the Power of Looped Transformers [ICLR 2025]

Universally use hidden states as inputs? We have a Looped Model!

Looped Models: Multi-layer model with weight sharing.

- In comparison of 24-layer 1B model (GPT-2-style) to k-layer model looped $24/k$ times, looped models are much better for reasoning primitives with fewer parameters.
- Looped models have an inductive bias towards good reasoning despite having worse perplexity and memorization to an iso-flop non-looped model.
- Accuracies for all task groups increases with more loops/depths, though also have diminishing returns.
- Looping-inspired regularization can leverage this inductive bias towards better reasoning.

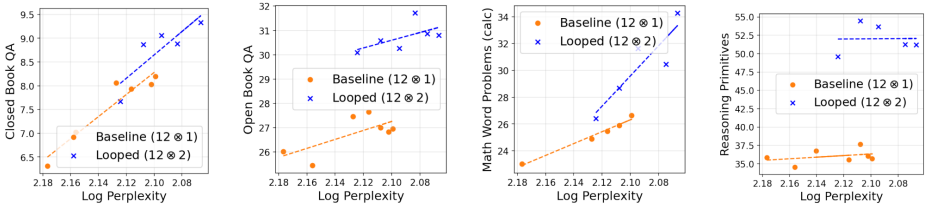


Figure: Comparison of a 12-layer baseline model and a looped model [9].

Looped Model plays scaling in reasoning: Compute more, Answer more.

Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach [NIPS 2025]

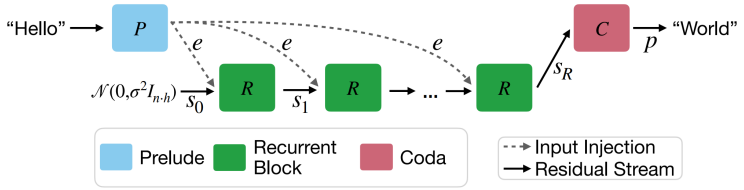


Figure: The recurrent depth architecture, where each block consists of a number of sub-layers [10]. For the same log probability, looped models tend to perform better on reasoning-intensive tasks.

Recurrent Unrolling:

- Layers are organized as recurrent blocks which accept a latent vector (initialized with a random state) as input. .
- Inject the data in every recurrence step with the latent vector to increase token correlation and stability.
- Randomly sample iteration counts during training, and backpropagate only the last k iterations of the recurrent blocks.

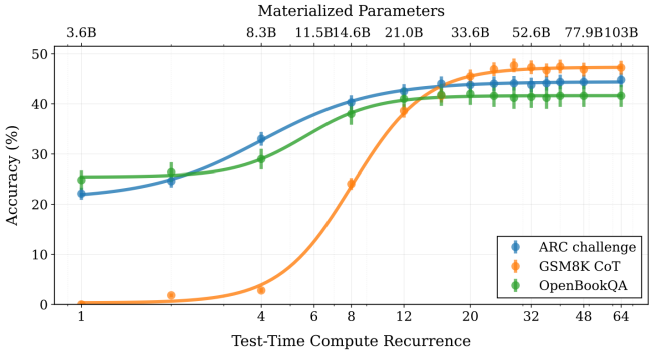


Figure: A 3.5B recurrent depth model can iteratre longer to use more compute and improve its performance in latent space at test time [10].

- **Pro:** Do not require specialized CoT data.
- **Con:** Pretraining is required (800B Tokens).

Hybrid Latent Reasoning via Reinforcement Learning [NIPS 2025 Spotlight]

Let's unify latents and tokens, and always try RL in reasoning tasks!

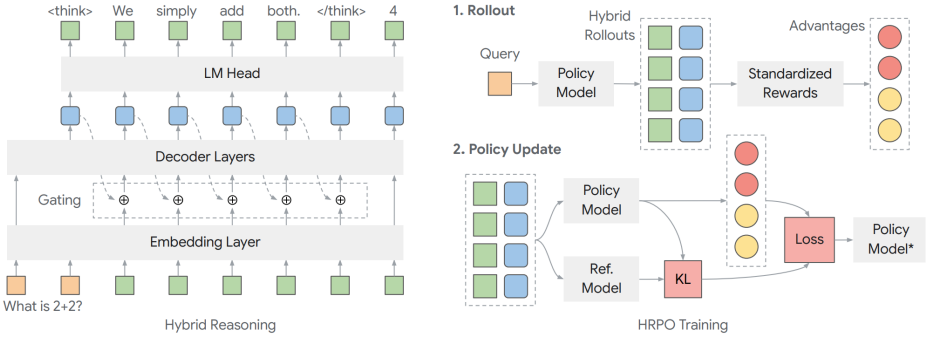


Figure: Hybrid latent reasoning with gating (left) and hybrid reasoning policy optimization (right) [11].

Gated Hybrid Reasoning:

- Project final hidden states back into the embedding space.
- Use normalized and weighted output probabilities as latent features.
- Gated sigmoid function to control the interpolated embeddings' blending, to balance between semantic continuity and noise level.
- The final answer is still generated via standard decoding.

Hybrid Reasoning Policy Optimization (HRPO):

- Use hybrid rollouts per input query and compute advantages by standardizing rewards within the group.
- Hybrid trajectories with higher rewards have higher advantages.
- During training, the hidden ratio of latents increases steadily, and completion lengths decreases.

Pros and Cons: Outperforming prior latent reasoning and distilled CoT methods in both knowledge- and reasoning- tasks, but introduces additional computation overhead.

3. Diffusion Language Models: Implicit Recursion

Does the Recurrent Unrolling method remind you of diffusion model?

- Auto-regressive modeling or the next-token prediction paradigm are not implicitly / perfectly designed for nonlinear reasoning or learning bi-directional causal relationships.
- Diffusion transformers have achieved success on complex visual tasks already with bi-directional dependencies.

So we introduce next ...

LLaDA: Large Language Diffusion Models [NIPS 2025 Oral]

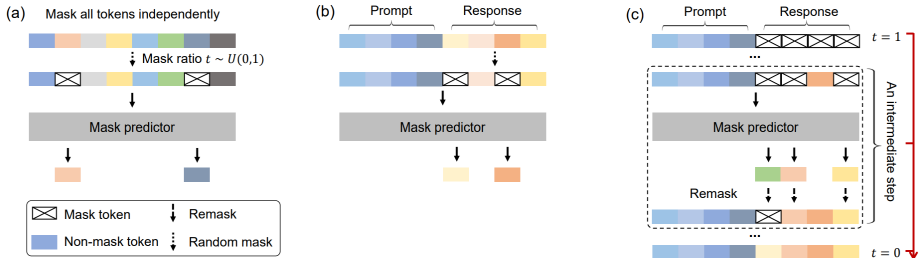


Figure: Overview of the (a) Pretraining, (b) SFT, and (c) Sampling paradigm of LLaDA, a transformer-based mask predictor that predicts all masked tokens simultaneously [12].

- Pretraining: Mask each token independently with the same probability $t \in [0, 1]$ and sample them with a forward process; Compute cross-entropy loss only on the masked tokens.
- Supervised Fine-Tuning: Leave the prompt unchanged and only mask the tokens in the response; Perform sampling-training.
- Inference/Sampling: At each step from time $t \in (0, 1]$ to $s \in [0, t)$, predict all masked tokens and remask $\frac{s}{t}$ tokens with lowest confidence tokens; Timesteps are uniformly distributed.

Another impactful Diffusion Large Language Model is Dream [13]!

LLaDOU: Reinforcing the Diffusion Chain of Lateral Thought with DLMs [NIPS 2025]

Given LLaDA's flexible token prediction paradigm, how can the response generation strategy be further strengthened, or perform implicit CoT?

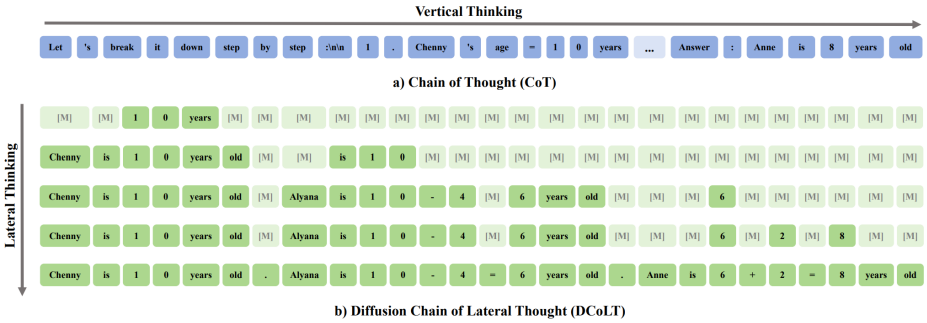


Figure: DCoLT (Diffusion Chain of Lateral Thought) generates tokens at chosen positions non-linearly, not following the auto-regressive order [14].

LLaDOU (Ordered Unmasking) / **DCoLT** Training:

- Define unmasking policy that selects top-K tokens at each step.
- Train unmasking + token prediction policy to optimize final reward.

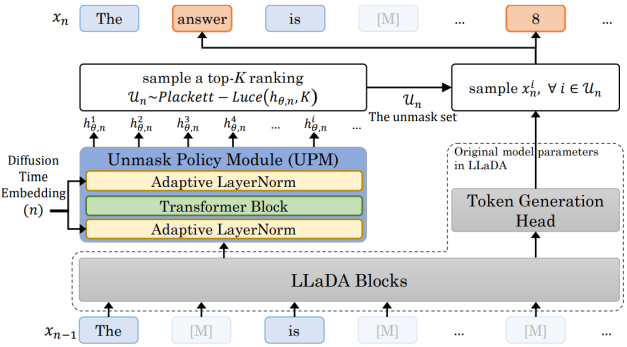


Figure: The LLaDOU architecture, which first predicts the token set to unmask, then samples those tokens by LLaDA [14].

A recent study [15] by Apple introduced another intriguing approach!

Remasking Discrete Diffusion Models with Inference-Time Scaling [NIPS 2025]

For modern MDMs, when a token is generated, it cannot be updated again, even when it introduces an error. ⇒ Why not remask it?

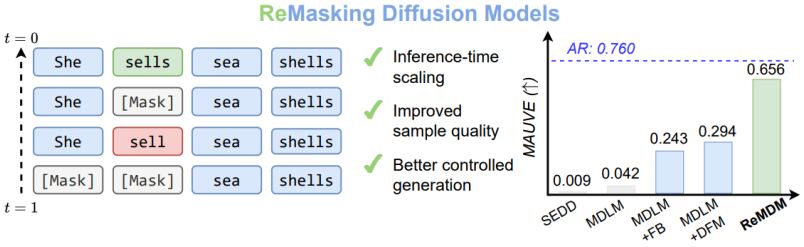


Figure: Remasking Diffusion Model (ReMDM) Sampler allow for more flexible generation of already decoded tokens [16].

Define a discrete diffusion model with posterior constructed as:

$$q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1-\sigma_t)\mathbf{x} + \sigma_t \boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m}. \end{cases}$$

- Parameter σ_t controls the remasking flexibility; Increasing σ_t moves away from z_t, while $\sigma_t = 0$ recovers the MDLM posterior.
- **Confidence-based Schedule:** Define confidence score for unmasked tokens w.r.t. decoding probability at its last unmasking.
- Masked tokens are decoded using the approximate posterior from MDLM, and the unmasked tokens are remasked negatively proportional to their confidence.
- **Turning on/off ReMDM:** Use a Switch or a three-stage Loop from vanilla sampling, mixture, to pure posterior prediction.

ReMDM propose a training-free and alternative forward-backward processes that Facilitates diffusion guidance while enabling flexibility.

Soft-Masked Diffusion Language Models [arXiv 2025.10]

What's left, after unmasking and remasking strategies? Lets blend!

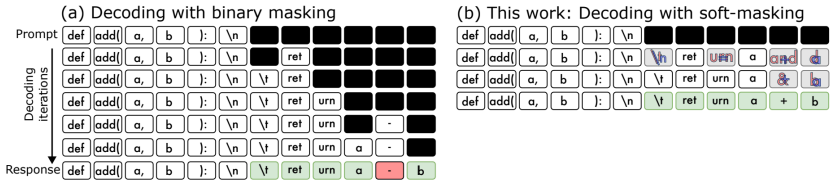


Figure: Soft-masking enriches the feedback for the next decoding step by superposing prior prediction information [17].

Soft-Masking (SM):

- Blend the mask token embedding with the top-k predicted token embeddings from the previous step (convex combination).
- Confidence-based token weighting by negative probability entropy and scaled sigmoid.
- Two-pass Training Scheme:
  1. Approximate the probability distribution of the previous denoising step by passing the corrupted data.
  2. Compute the soft-mask representation and passed through the backbone a second time to compute the loss.

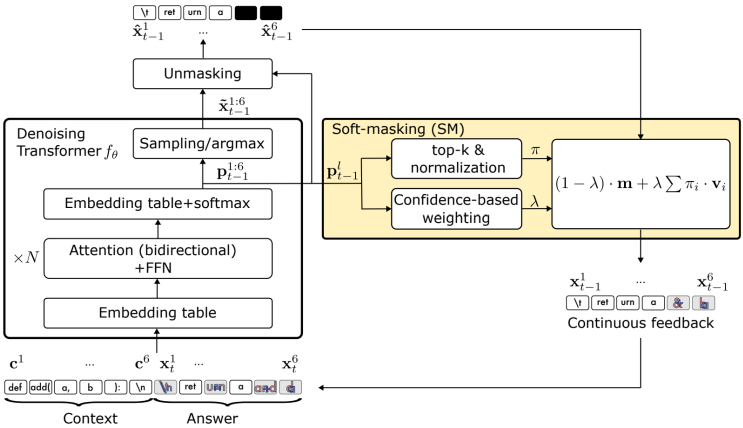


Figure: The soft-masking architecture predict via iterative denoising of an initially fully masked response [17]. Does it remind you latent reasoning?

Summary: Insights! (are they correct?)

1. Adaptive (valid) computing strategy can always improve efficiency and performance:
   - Model Level: Compute more on important layers.
   - Token Level: Compute less on confident predictions.
   - Data Level: Compute less on easy inputs.
2. Conditional Generation $\geq$ Uncon (in Reasoning)?
3. The Reasoning Scaling Law:
   - Knowledge by Model: Parameters $\Rightarrow$ Intelligence.
   - Depth by Prediction: Think-about-Think.
   - Width by Sampling: Let's ensemble! [18]

What scaling law is played by latent reasoning?

- Both vertical (iterative computation) and parallel scaling (more latent directions)!

Thank you for your attention!

References I

[1] Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

[2] Yuyang Wu, Yifei Wang, Ziyu Ye, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025.

[3] Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

[4] Siqi Fan, Xin Jiang, Xiang Li, Xuying Meng, Peng Han, Shuo Shang, Aixin Sun, Yequan Wang, and Zhongyuan Wang. Not all layers of llms are necessary during inference. *arXiv preprint arXiv:2403.02181*, 2024.

[5] David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

[6] Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron Courville, et al. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[7] Tianyu Fu, Yichen You, Zekai Chen, Guohao Dai, Huazhong Yang, and Yu Wang. Think-at-hard: Selective latent iterations to improve reasoning language models. *arXiv preprint arXiv:2511.08577*, 2025.

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

[9] Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations*, 2025.

References II

[10] Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[11] Zhenrui Yue, Bowen Jin, Huimin Zeng, Honglei Zhuang, Zhen Qin, Jinsung Yoon, Lanyu Shang, Jiawei Han, and Dong Wang. Hybrid latent reasoning via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[12] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[13] Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

[14] Zemin Huang, Zhiyang Chen, Zijun Wang, Tiancheng Li, and Guo-Jun Qi. Reinforcing the diffusion chain of lateral thought with diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[15] Metod Jazbec, Theo X. Olausson, Louis Béthune, Pierre Ablin, Michael Kirchhof, Joao Monterio, Victor Turrisi, Jason Ramapuram, and Marco Cuturi. Learning unmasking policies for diffusion language models, 2025.

[16] Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[17] Michael Hersche, Samuel Moor-Smith, Thomas Hofmann, and Abbas Rahimi. Soft-masked diffusion language models. *arXiv preprint arXiv:2510.17206*, 2025.

[18] Mouxiang Chen, Binyuan Hui, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Jianling Sun, Junyang Lin, and Zhongxin Liu. Parallel scaling law for language models. *arXiv preprint arXiv:2505.10475*, 2025.