Superposition Reasoning Model

Zheyang Xiong[w,m], **Shivam Garg**[m], **Vaishnavi Shrivastava**[m], **Haoyu Zhao**[p,m]
Anastasios Kyrillidis[r], **Dimitris Papailiopoulos**[w,m]
[w]University of Wisconsin-Madison, [m]Microsoft Research, [p]Princeton University, [r]Rice University

Abstract

While Large Language Models (LLMs) usually reason on the language space with discrete tokens, recent studies have found that LLMs can reason on more expressive spaces like continuous latent space. However, training LLMs on continuous latent space is challenging due to lack of sufficient training signals. In this work, we propose a way that teaches LLMs to reason on superpositions of discrete tokens. Our model takes in a superposition of token embeddings and outputs multiple tokens using a Multi-token Prediction (MTP) module. Our empirical results show that with superposition reasoning, the model use $\sim$30% fewer reasoning tokens on GSM8K compared to the baseline with no accuracy gap.

1 Introduction

Large Language Models (LLMs) typically reason on discrete tokens—often termed chain-of-thought (CoT) [10]. While such token-level reasoning can improve problem solving, long CoT reasoning can also be computationally expensive. A recent line of work explores reasoning in *continuous* latent space, where models use continuous tokens instead of discrete tokens [7, 3]. However, these approaches have been demonstrated primarily on relatively short reasoning chains and often hard to scale to longer CoT settings.

We propose *superposition reasoning*, a simple, scalable alternative that preserves the advantages of token-level supervision while also having the efficiency of operating in a richer space. Rather than completely discarding the discrete vocabulary, our model reads and writes on *superpositions of token embeddings*. Concretely, within each reasoning step the model (i) compresses a pair of CoT token embeddings into a single hidden representation via a learned $2H \to H$ projection and predicts the next CoT token, and (ii) uses a lightweight Multi-token Prediction (MTP) module to jointly predict an additional CoT token. We evaluate by post-training Qwen2.5-Math-1.5B-Instruct on a curated dataset. On GSM8K superposition reasoning reduces reasoning tokens by approximately 30% with no gap on accuracy.

2 Related Works

Latent Reasoning in LLMs. When prompted with a question, LLMs can generate intermediate reasoning via discrete tokens before answering the question, and such reasoning process is termed chain-of-thought (CoT) [10]. Recently, Several works focus on using CoT states beyond discrete tokens. Hao et al. [7] introduce a method that directly feeds back the last continuous hidden state as the input embedding for the next step. However, this method requires a complicated training curriculum and the length of CoT is short (fewer than 20 tokens). Cheng and Van Durme [3] compress CoT by training a compression module and a decoder module. Zhang et al. [11] and Zhuang et al. [12] introduce new training-free generation techniques that take in a weighted sum of token embeddings as input.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

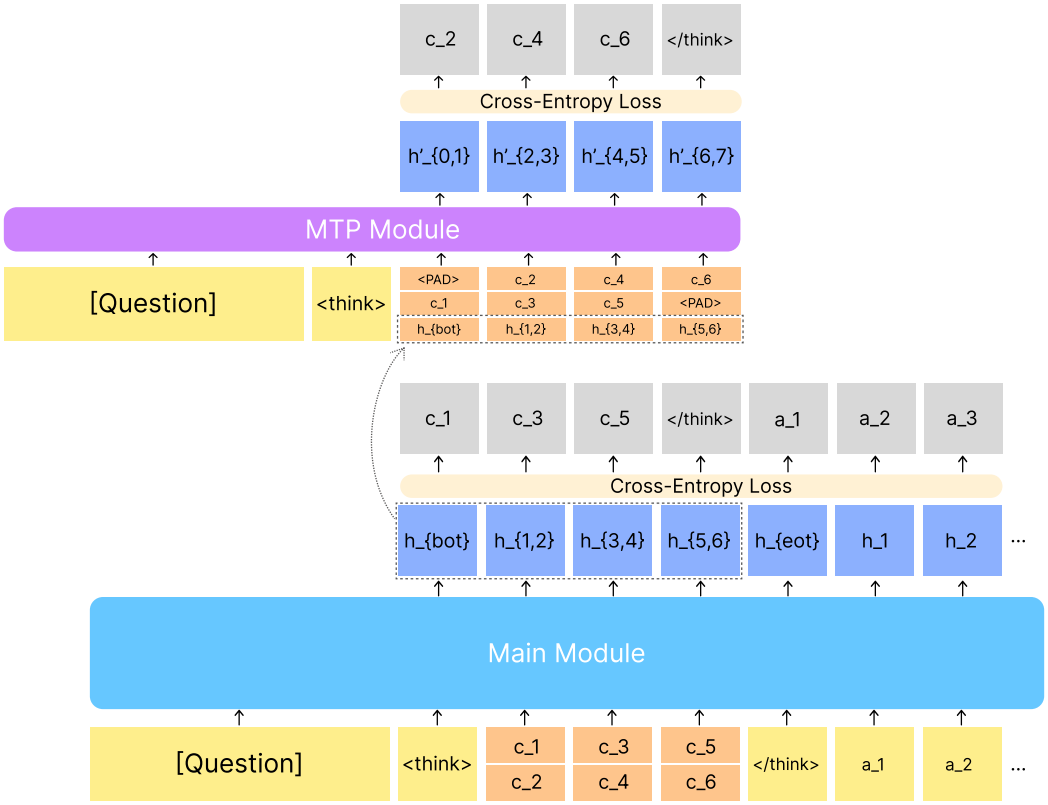

Figure 1: Illustration of our proposed model that reasons in superposition. At each reasoning step, the main module superposes CoT token embeddings using a learned $2H \to H$ projection and predicts a CoT token; the MTP module superposes the CoT token embeddings with the main module's hidden states using a learned $3H \to H$ projection and predicts an additional CoT token. The output language modeling heads are shared.

Multi-token Prediction. Traditionally, LLMs are trained with next-token prediction loss where the model is provided with a prefix and asked to predict the next token that follows the prefix [9]. Bachmann and Nagarajan [2] argue that teacher-forcing in next-token prediction results in inaccurate next-token predictor and proposes a solution that learns to predict multiple tokens. Gloeckle et al. [6] pre-train LLMs from scratch that predicts multiple future tokens at once using multiple output heads and show that multi-token prediction (MTP) is better than next-token prediction (NTP) on larger models. DeepSeek-V3 [8] also train the model with MTP objective but use a lightweight MTP module instead of an independent output head. Ahn et al. [1] propose joint multi-token prediction (JTP) by employing a representation bottleneck that encourages the model to encode richer information in the output hidden state.

3 Methods

3.1 Superposition Reasoning Model

Overview. We partition each sequence into question tokens $q_{1:L_q}$, CoT ("thinking") tokens $c_{1:L_c}$ bracketed by `<think>` and `</think>`, and answer tokens $a_{1:L_a}$. Question and answer segments use standard next-token prediction. Inside the CoT span, the model operates in *two-token superposition*: at each reasoning step, (i) the **Main** module compresses two CoT embeddings with a learned $2H \to H$ projection, runs a Transformer block stack, and predicts the next (*odd*-indexed) CoT token; then (ii) the **MTP** module consumes the Main hidden state together with the most recent two CoT tokens via a learned $3H \to H$ projection to predict the following *even*-indexed CoT token. Training uses teacher forcing and both modules share the output LM head.

Details. We introduce our model that reasons in superposition. Let $\texttt{Main}(\cdot)$ be the main module and $\texttt{MTP}(\cdot)$ be the MTP module that is similar to the MTP module in DeepSeek-V3 [8]. In this paper we assume to have only one MTP module that predicts an additional token.

For the main module, at the i-th superposition step (the i-th step after <think> as input), we first combine the embedding of the $(2i-1)$-th CoT token and $(2i)$-th CoT token with the linear projection:

$$\boldsymbol{x}_i = P[\texttt{Emb}(c_{2i-1}), \texttt{Emb}(c_{2i})],$$

where $P \in \mathbb{R}^{H \times 2H}$ is a learnable projection matrix and $[\cdot, \cdot]$ denotes concatenation. The main module outputs hidden state $\boldsymbol{h}_{2i-1,2i}$. We apply a head to this hidden state to get the next-token prediction (and similar for MTP).

For the MTP module, note that its <think> token occurs one position earlier than in the main module, and its first superposition step aligns with $\boldsymbol{h}_{\text{bot}}$. At the first superposition step, we combine the embedding of a <PAD> token, the embedding of c_1 and the hidden state $\boldsymbol{h}_{\text{bot}}$ with the linear projection:

$$\boldsymbol{x}'_1 = P'[\texttt{RMSNorm}(\texttt{Emb}(\texttt{<PAD>})), \texttt{RMSNorm}(\texttt{Emb}(c_1)), \texttt{RMSNorm}(\boldsymbol{h}_{\text{bot}})],$$

where $P' \in \mathbb{R}^{H \times 3H}$. At the i-th superposition step for $i > 1$, $\boldsymbol{x}'_i$ is defined as

$$\boldsymbol{x}'_i = P'[\texttt{RMSNorm}(\texttt{Emb}(c_{2i-2})), \texttt{RMSNorm}(\texttt{Emb}(c_{2i-1})), \texttt{RMSNorm}(\boldsymbol{h}_{2i-3,2i-2})].$$

The hidden state output from the MTP module at the corresponding position is $\boldsymbol{h}'_{2i-2,2i-1}$.

The NTP loss is defined by

$$\mathcal{L}_{\text{NTP}} = -\frac{1}{L_c/2 + L_a + 1}\Big(l(c_1|\boldsymbol{h}_{\text{bot}}) + \sum_{i=1}^{L_c/2-1} l(c_{2i+1}|\boldsymbol{h}_{2i-1,2i}) + l(\texttt{</think>}|\boldsymbol{h}_{L_c-1,L_c}) + l(a_1|\boldsymbol{h}_{\text{eot}}) + \sum_{i=2}^{L_a-1} l(a_i|\boldsymbol{h}_{i-1})\Big),$$

where $l(c|\boldsymbol{h}) := \text{CrossEntropy}(c, \text{head}(\boldsymbol{h}))$.

The MTP loss is defined by

$$\mathcal{L}_{\text{MTP}} = -\frac{1}{L_c/2 + 1}\Big(\sum_{i=1}^{L_c/2} l(c_{2i}|\boldsymbol{h}'_{2i-2,2i-1}) + l(\texttt{</think>}|\boldsymbol{h}_{2L_c,2L_c+1}\Big).$$

Note that the NTP loss targets both CoT and answer tokens while the MTP loss only targets the CoT tokens. The training loss is defined by

$$\mathcal{L}_{\text{Training}} = \mathcal{L}_{\text{NTP}} + \lambda \cdot \mathcal{L}_{\text{MTP}}.$$

3.2 Fast and Slow Thinking

Our superposition reasoning model introduced in Section 3.1 can think "fast" by taking two tokens as input and predicting two tokens within a single step. However, some tokens can be harder to predict and at some places the order of two adjacent tokens matter. We introduce two ways to "slow down" model's thinking.

3.2.1 Special Treatment on Number Tokens

One way is to consider all number tokens as special tokens and reason discretely on number tokens. In particular, in preparing the dataset, whenever we encounter two adjacent number tokens, we append a <PAD> token after each number token so that within a single step, the model will not take two number tokens as input and will not predict two number tokens.

Table 1: Accuracy and average correct CoT length on GSM8K test set with greedy decoding.

Method	Acc (%)	Avg. correct CoT length
Qwen2.5-finetuned (baseline)	86.43	292.91
Superposed Reasoning ($\lambda = 1.0$)	78.32	142.97
w/ num. special treatment ($\lambda = 1.0$)	82.57	169.92
w/ adaptive superposition ($\lambda = 0.1, \tau = 0.95$)	83.04	162.55
w/ adaptive superposition ($\lambda = 0.02, \tau = 0.99$)	85.14	182.45
w/ adaptive superposition ($\lambda = 0.02, \tau = 0.999$)	86.43	204.91

3.2.2 Adaptive Superposition

Another way to decide when to reason discretely is to inspect the model's confidence in each prediction by the MTP module. For every token proposed by the MTP module at step i, compute the softmax over logits and take the maximum probability $p_{\max}$. If $p_{\max} < \tau$ (a chosen threshold), set the MTP output at step i to `<PAD>`; the main module will then re-predict that token at the next step. Otherwise, accept the MTP token.

4 Experiments

4.1 Experimental Setup

In this section, we introduce our experiment to train a model that reasons in superposition. We use Qwen2.5-Math-1.5B-Instruct as the model we start from and post-train it to a superposition reasoning model. The main module is initialized the same as Qwen2.5 and the MTP module is initialized using the weights of the last layer of Qwen2.5. The projection matrices P and P' are initialized as a map that averages the input embeddings.

To train the model, we curate a synthetic reasoning dataset. We collect questions from a synthetic dataset by Deng et al. [5] that contains 400K math questions that have similar style as GSM8K [4]. For each question, we let Qwen2.5 to generate 3 responses with CoT. To verify the response, we look at the last sentence of the response (determined by `\n`) and verify if the text within the last "`\boxed{}`" matches exactly with the solution. If a response is verified, we consider the whole response as the CoT part and the last sentence as the answer part and select it as a part of our training dataset. The curated training dataset consists of 1M sequences of data.

We finetune Qwen2.5 with batch size 512 and learning rate 1×10^{-4} for 2 epochs. For model trained with adaptive superposition, we use $\lambda = 0.1$ so that we adaptively decode during inference time, the Main module should focus more on the first decoded token since we can reject the second token and use the Main module to decode that token again on the next step. We also finetune Qwen2.5 on the same dataset as a baseline.

4.2 Results

The results for Qwen2.5 are shown in Table 1. For Qwen2.5, while there is an 8.11% gap on accuracy between fully superposed reasoning model and the baseline, having special treatment on number tokens or adaptive decoding further improves the accuracy. Additionally, for adaptive decoding, with higher threshold $\tau = 0.999$, we can close the gap between the baseline while saving 30% of the tokens. For smaller threshold, we can gain more CoT saving while having a slightly larger gap.

References

[1] K. Ahn, A. Lamb, and J. Langford. Efficient joint prediction of multiple future tokens. *arXiv preprint arXiv:2503.21801*, 2025.

[2] G. Bachmann and V. Nagarajan. The pitfalls of next-token prediction. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine*

Learning Research, pages 2296–2318. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/bachmann24a.html.

[3] J. Cheng and B. Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

[4] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[5] Y. Deng, K. Prasad, R. Fernandez, P. Smolensky, V. Chaudhary, and S. Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023.

[6] F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, and G. Synnaeve. Better & faster large language models via multi-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[7] S. Hao, S. Sukhbaatar, D. Su, X. Li, Z. Hu, J. Weston, and Y. Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[8] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[10] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[11] Z. Zhang, X. He, W. Yan, A. Shen, C. Zhao, S. Wang, Y. Shen, and X. E. Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *arXiv preprint arXiv:2505.15778*, 2025.

[12] Y. Zhuang, L. Liu, C. Singh, J. Shang, and J. Gao. Text generation beyond discrete token sampling. *arXiv preprint arXiv:2505.14827*, 2025.