COrAL: Order-Agnostic Language Modeling for Efficient Iterative Refinement

Anonymous authors
Paper under double-blind review

Abstract

Iterative refinement has emerged as an effective paradigm for enhancing the capabilities of large language models (LLMs) on complex tasks. However, existing approaches typically implement iterative refinement at the application or prompting level, relying on autoregressive (AR) modeling. The sequential token generation in AR models can lead to high inference latency. To overcome these challenges, we propose Context-Wise Order-Agnostic Language Modeling (COrAL), which incorporates iterative refinement directly into the LLM architecture while maintaining computational efficiency. Our approach models multiple token dependencies within manageable context windows, enabling the model to perform iterative refinement internally during the generation process. Leveraging the order-agnostic nature of COrAL, we introduce sliding blockwise order-agnostic decoding, which performs multi-token forward prediction and backward reconstruction within context windows. This allows the model to iteratively refine its outputs in parallel in the sliding block, effectively capturing diverse dependencies without the high inference cost of sequential generation. Empirical evaluations on reasoning tasks demonstrate that COrAL improves performance and inference speed, respectively, achieving absolute accuracy gains of 4.6% on GSM8K and 4.0% on LogiQA, along with inference speedups of up to $3.9\times$ over next-token baselines. Preliminary results on code generation indicate a drop in pass rates due to inconsistencies in order-agnostic outputs, highlighting the inherent quality–speed trade-off.

1 Introduction

Large Language Models (LLMs) have recently achieved remarkable success across a wide range of tasks (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2023; Dubey et al., 2024), such as mathematical problem-solving, logical reasoning, and programming (Yu et al., 2024; Pan et al., 2023; Schick et al., 2023; Rozière et al., 2023). Strategies that enable LLMs to learn from previous mistakes and iteratively refine their outputs have been particularly effective, achieving human-level performance and transforming both academic research and industrial applications (Pan et al., 2024; Ye et al., 2024; OpenAI, 2024). These iterative refinement approaches incorporate feedback—either external or internal—as supervision signals during training (Zelikman et al., 2022; Huang et al., 2023; Shinn et al., 2023; Lightman et al., 2024; Xie et al., 2024), or by developing prompting frameworks that guide the model toward improved generations through methods like guided search or self-refine (Yao et al., 2023; Xie et al., 2023; Madaan et al., 2023).

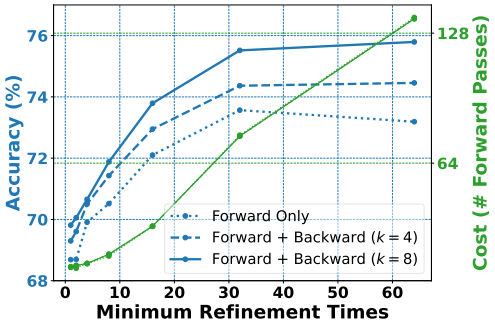


Figure 1: Scaling of performance and inference cost on GSM8K with increasing the minimum refinement times for each output position. k represents the backward context window size. We set the decoding block size as $b = 64$.

Despite their effectiveness, these approaches predominantly rely on autoregressive (AR) LLMs, which generate text by predicting the next token in a fixed left-to-right order using causally masked Transformers (Radford, 2018). This sequential generation process inherently limits the model's

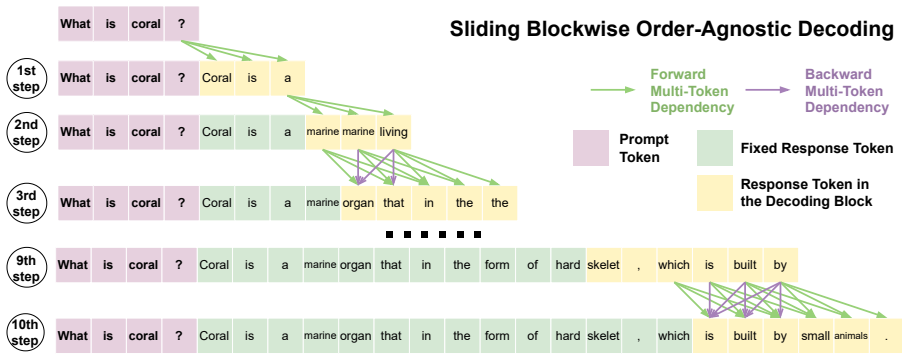


Figure 2: **Sliding Blockwise Order-Agnostic Decoding**. COrAL performs multi-token prediction and refinement in the sliding block with context window size $k = 3$ and block size $b = 6$.

ability to capture dependencies spanning beyond the immediate next token, especially those that require backward context (Hu et al., 2024). Moreover, the sequential nature of AR models leads to high inference latency, resulting in computational inefficiency for long sequences (Cai et al., 2024).

To address these limitations, researchers have explored order-agnostic architectures that enhance representation learning and accelerate inference. Previous studies mainly focus on two solutions: permutation-based AR and non-autoregressive (NAR) modeling, but each has its own strengths and limitations. For instance, permutation-based models propose diversity-enhanced pretraining objectives that predict multiple subsequent tokens in various orders to capture richer dependencies (Yang et al., 2019; Zhang et al., 2024b). Similarly, NAR models generate tokens in parallel, significantly reducing inference time (Gu et al., 2018). However, conventional NAR models often struggle with tasks involving variable-length generation and complex token dependencies, leading to degraded text quality. As a result, these models are typically task-specific and require additional mechanisms to ensure consistency (Gui et al., 2023; Shi et al., 2024). Inspired by the success of diffusion models in image generation (Austin et al., 2021), recent efforts have adapted denoising techniques to generative language modeling as an iterative extension of NAR models (Savinov et al., 2022; Gong et al., 2023). While these methods improve efficiency, they still lag behind AR models regarding generation quality and generalizability. Given the trade-offs among different models[1], a pivotal question arises:*Can we unify the strengths of denoising techniques with order-agnostic modeling to enhance the capabilities of AR-LLMs while mitigating their respective limitations*?

In this work, we propose **Context-Wise Order-Agnostic Language Modeling** (COrAL), which combines the advantages of AR and order-agnostic modeling. COrAL models token dependencies within manageable context windows, effectively balancing the capture of both local and long-range dependencies with computational efficiency. Through context-wise modeling, COrAL overcomes the limitations of fixed-order generation in AR models and the dependency modeling challenges in NAR models. Within each context window, COrAL models diverse dependencies in an order-agnostic manner, enhancing the model's ability to capture complex token relationships while maintaining computational efficiency. Leveraging COrAL, we introduce **Sliding Blockwise Order-Agnostic Decoding**, which performs forward multi-token prediction and backward reconstruction simultaneously. As shown in Figure 1, this strategy enables the model to perform iterative refinement internally to scale up inference performance. Additionally, to ensure that the model remains aware of target token positions without necessitating architectural changes, we apply a generalized Rotary Position Embedding (RoPE) (Su et al., 2024) to the last layer of the Transformer. This positional encoding technique preserves target-aware representations, which are essential for effective order-agnostic generation and iterative refinement.

We conduct extensive experiments on reasoning tasks, including arithmetic computation and logical reasoning, to evaluate the effectiveness and efficiency of COrAL. Our empirical results show that COrAL not only improves performance but also significantly accelerates inference. Specifically, COrAL achieves absolute accuracy gains of 4.6% on GSM8K and 4.0% on LogiQA, along with inference speedups of up to 3.9 times over next-token baselines. These findings demonstrate that COrAL effectively captures dependencies within context windows while maintaining computational

[1]We make conceptual comparison among different model architectures in Appendix A.

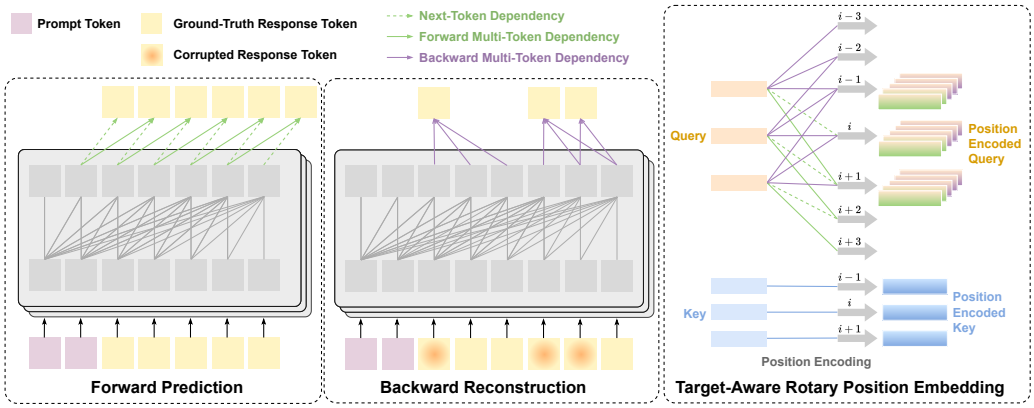


Figure 3: **Context-Wise Order-Agnostic Language Modeling**. We visualize the order-agnostic dependencies within a context window size $k = 2$. For target-aware position encoding, we show how COrAL obtains query representations for multiple positions within a context window size $k = 2$.

efficiency. However, preliminary experiments on code generation reveal a decrease in pass rates due to inconsistencies in order-agnostic outputs, highlighting the inherent quality–speed trade-offs. This suggests that further refinements are necessary for tasks that require strict syntactic coherence.

2 Context-Wise Order-Agnostic Language Modeling

We present Context-Wise Order-Agnostic Language Modeling, a generalized AR framework that captures conditional textual distributions based on various orders in context windows. We first review order-agnostic autoregressive language modeling and non-autoregressive modeling with iterative refinement as background, before describing COrAL's objective and architecture.

2.1 Background

Given a prompt $\boldsymbol{x}$ and a target sequence of T tokens $\boldsymbol{y} = \{y_1, y_2, \cdots, y_T\}$, conventional AR models factorize the multivariate distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ into a product of univariate distributions using the probability chain rule $\log p(\boldsymbol{y} \mid \boldsymbol{x}) = \sum_{t=1}^{T} \log p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$, which requires T iterative sampling steps to generate the sequence. In contrast, order-agnostic AR modeling generalizes this by modeling multiple possible orderings $\sigma \in \mathcal{S}_T$ of the sequence:

$$\begin{aligned} \log p(\boldsymbol{y} \mid \boldsymbol{x}) &= \log \mathbb{E}_{\sigma \sim \mathcal{S}_T} \left[p(\boldsymbol{y} \mid \boldsymbol{x}, \sigma) \right] \\ &\geq \mathbb{E}_{\sigma \sim \mathcal{S}_T} \left[\log p(\boldsymbol{y} \mid \boldsymbol{x}, \sigma) \right] = \mathbb{E}_{\sigma \sim \mathcal{S}_T} \left[\sum_{t=1}^{T} \log p \left(y_{\sigma(t)} \mid \boldsymbol{y}_{\sigma(<t)}, \boldsymbol{x} \right) \right]. \end{aligned} \tag{1}$$

where $\mathcal{S}_T$ denotes the set of all possible permutations of the indices $\{1, 2, \cdots, T\}$. However, this permutation-based objective poses a significant optimization challenge and can lead to underfitting, as observed in prior works (Yang et al., 2019; Hoogeboom et al., 2022).

On the other hand, NAR modeling (Lee et al., 2018) breaks the sequential dependency to accelerate inference. This approach applies sequence-level denoising steps, enabling parallel reconstruction of multiple tokens with iterative refinement to enhance generation quality. To equip the model with denoising capabilities, it employs L intermediate latent variables $\{\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \cdots, \boldsymbol{y}^{(L)}\}$ and approximates their marginalization as follows:

$$\log p(\boldsymbol{y} \mid \boldsymbol{x}) \geq \sum_{t=1}^{T} \log p(y_t \mid \boldsymbol{y}^{(L)}, \boldsymbol{x}) + \sum_{l=1}^{L} \sum_{t=1}^{T} \log p(y_t^{(l)} \mid \boldsymbol{y}^{(l-1)}, \boldsymbol{x}) + \sum_{t=1}^{T} \log p(y_t^{(0)} \mid \boldsymbol{x}) \tag{2}$$

where the latent variables are constrained to match the type of the target output $\boldsymbol{y}$. While previous studies demonstrate the efficiency of NAR modeling in specific tasks such as machine translation (Gu et al., 2018; Ghazvininejad et al., 2019; Kasai et al., 2020), its potential in language modeling remains underexplored. Moreover, the use of corrupted data for denoising and the assumption of token-wise independence in each reconstruction step in NAR models can introduce instability, often resulting in reduced text quality compared to their AR counterparts (Savinov et al., 2022).

2.2 Objective: Context-Wise Order-Agnostic Autoregressive Modeling

To address the above limitations in AR language modeling, we propose Context-Wise Order-Agnostic Language Modeling (COrAL), unifying token-level dependency modeling and sequence-level denoising to advance the capabilities of current LLMs. Previous order-agnostic modeling works attempt to capture various factorization orders involving long dependencies that are difficult to model and fit. In contrast, COrAL learns the orderless relationships within predetermined context windows. Built on the AR foundation, our COrAL framework leverages the superior capability of sequential language modeling in LLMs.

COrAL tackles the problem of generative language modeling by combining forward multi-token prediction with backward denoising in a context-wise and order-agnostic framework. Denoting the context window size as k[2], we model the conditional probability distribution of each target token by considering an ensemble of dependencies over all possible positions in the context:

$$\log p_\theta(\boldsymbol{y} \mid \boldsymbol{x}) \geq \sum_{t=1}^{T} \mathbb{E}_{i\sim\mathcal{U}[t-k,t+k]} \mathbb{E}_{l\geq 0} \left[\log p_\theta(y_t \mid \boldsymbol{y}^{(l)}_{\leq i}, \boldsymbol{x}) \right] \tag{3}$$

where $\boldsymbol{y}^{(l)}$ represents an intermediate state of the target output sequence $\boldsymbol{y}$ during iterative refinement. The conventional AR modeling, in comparison, becomes a specific case where only the forward prediction with k=1, conditioned on previous tokens in the target sequence $\boldsymbol{y}$, is modeled.

Forward Prediction and Backward Reconstruction. As shown in Figure 3, we decompose the order-agnostic objective into **forward prediction** and **backward reconstruction**. In forward prediction, COrAL learns to predict multiple future tokens simultaneously given past tokens in the ground-truth sequence. For backward reconstruction, we randomly corrupt tokens in the input sequence to create the intermediate states $\boldsymbol{y}^{(l)}$ in Eq. 3. Similar to BERT (Devlin et al., 2019), we compute the loss only on the corrupted tokens. During training, we use the original data for prediction and the corrupted data for reconstruction. This decomposition disentangles the self-refinement capability from forward prediction, leveraging all data points to enhance sequence modeling.

Corruption Strategy. Our corruption and reconstruction process is a form of denoising autoencoding (Vincent et al., 2008) in language modeling. However, instead of representation learning, we aim to endow the model with the self-refinement capability to revise the generated content. Inspired by masked autoencoders (He et al., 2022), we divide the output sequence into non-overlapping patches and randomly sample a subset for corruption. Each patch is a fragment of text containing one or multiple consecutive tokens in the sequence. Specifically, we corrupt a patch by either (**i**) replacing it with a random patch sampled from the current sequence or (**ii**) repeating the first token to replace the other tokens in the patch. This design draws on insight from Ye et al. (2024) that model performance can be significantly improved by simply enhancing consistency across steps.

2.3 Architecture: Target-Aware Query Representation for Self-Attention

We build our framework by adapting the standard architecture of LLMs using decoder-only Transformers (Brown et al., 2020). Unlike prior NAR works employing encoder–decoder architectures (Lee et al., 2018; Kasai et al., 2020), the conventional AR foundation predicts the same distribution given the current context regardless of the target token position. While this demonstrates advanced capabilities of sequence modeling and generation, the typical parameterization of next-token distribution constrains its generalizability to the order-agnostic objective in Eq. 3.

Previous works on order-agnostic modeling have explored various ways to incorporate positional information, including scaling up the dimensionality of the final projection layer (Stern et al., 2018) and adding look-ahead tokens (Monea et al., 2023) or extra decoding heads (Cai et al., 2024; Gloeckle et al., 2024). Despite their promising performance, these methods introduce the overhead of additional self-attention network calls and new parameters for multi-position prediction. Instead, we propose a seamless adjustment without adding extra model parameters. Specifically, we apply a generalized Rotary Position Embedding (RoPE) (Su et al., 2024) at the final layer of the decoder-only Transformers to integrate target-aware information into the query representations.

[2]Without loss of generality, we can set different context window sizes for forward prediction and backward reconstruction in practice. Here, we present the objective with the same hyperparameter k to avoid clutter.

Target-Aware RoPE. RoPE encodes positional information into query and key representations, ensuring that their inner product inherently contains relative position information in self-attention: $f(\boldsymbol{q}_m, m)^\top f(\boldsymbol{k}_n, n) = g(\boldsymbol{q}_m, \boldsymbol{k}_n, m-n)$, where f is the positional encoding function applied to the query and key embeddings at m-th and n-th positions, respectively. Conventional RoPE integrates positional information of the current token to form the query representation. While this effectively enhances the position-aware representation of the input token in intermediate hidden states, it introduces inherent misalignment with the target token position when using the learned representation for output prediction. To avoid this problem, we propose Target-Aware RoPE (Figure 3), which modifies the positional encoding function at the final layer by considering the target token position in the query representation:

$$f(\boldsymbol{q}_m, \mu)^\top f(\boldsymbol{k}_n, n) = g(\boldsymbol{q}_m, \boldsymbol{k}_n, \mu - n), \quad \mu \in [m-k, m+k] \tag{4}$$

The rationale behind this modification is that the position encoding in RoPE can adapt the representation of the current token to be tailored for the target position. This simple yet effective adjustment endows the model with the target-aware capability, allowing it to predict tokens at various positions without the overhead of additional entire network calls.

3 Sliding Blockwise Order-Agnostic Decoding

Leveraging the order-agnostic capabilities of COrAL, we propose Sliding Blockwise Order-Agnostic Decoding, a parallel decoding strategy to enable efficient iterative refinement.

High inference latency significantly hinders the broader application of AR-LLMs. Recent studies have tackled this bottleneck from various angles to accelerate inference. For instance, speculative decoding employs a smaller, faster draft model to propose multiple continuations, which the larger target model then verifies and accepts (Leviathan et al., 2023; Miao et al., 2024). Blockwise parallel decoding directly leverages the large model to generate multiple tokens simutaneously (Stern et al., 2018; Cai et al., 2024). However, these studies increase memory consumption, which thus limits the scalability and impedes distributional deployment. Another promising line of work breaks the sequential dependency by adopting Jacobi decoding (Santilli et al., 2023; Fu et al., 2024) for iterative refinement without architectural add-ons. Kou et al. (2024) propose consistency LLMs to further improve the performance of Jacobi decoding inspired by consistency models (Song et al., 2023).

While these existing approaches improve inference efficiency, they rely on the conventional left-to-right AR foundation with monotonic dependencies. In this work, we leverage the order-agnostic nature of COrAL to perform backward sequence-level refinement and forward multi-token prediction simultaneously, significantly accelerating inference. At each step, we ensemble the output distributions based on multiple possible dependencies and construct a candidate set to fill a block of the output sequence. Furthermore, this process facilitates self-refinement by modifying previous generations at a higher-level horizon, enhancing output quality with advanced inference capabilities. Next, we detail the ensemble strategy in decoding for candidate construction and verification, corresponding to the "Collect" and "Verify and Slide" parts in Algorithm 1, respectively.

Prediction. Given a set of possible distributions $\{p_\theta(y_t \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x})\}_{i=t-k}^{t+k}$ for the t-th token in the output sequence, we obtain the ensemble distribution via model arithmetic (Dekoninck et al., 2024). Specifically, we apply different weights to the distributions to prioritize the more accurate dependencies, with distributions based on more qualified content generally leading to better generations:

$$\pi_\theta(y_t) = \mathrm{softmax}\left(\frac{1}{\sum_{i=t-k}^{t+k} \omega_{t-i}(\boldsymbol{y}_{\leq i}, \boldsymbol{x})} \sum_{i=t-k}^{t+k} \omega_{t-i}(\boldsymbol{y}_{\leq i}, \boldsymbol{x}) \log p_\theta(y_t \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x}) \right) \tag{5}$$

The weight function $\omega_{t-i}(\boldsymbol{y}_{\leq i}, \boldsymbol{x}) = \lambda_{t-i} \cdot c(\boldsymbol{y}_{\leq i} \mid \boldsymbol{x})$ is determined by the relative distance and direction of the dependency, as well as the confidence of the generated context $\boldsymbol{y}_{\leq i}$. Here, the factor $\lambda_{t-i} \in [0, 1]$ only depends on the relative position of the target token, decaying for longer dependencies. Using order-agnostic modeling, we calculate the confidence score c by gathering the predicted probabilities based on different dependencies, which we obtain in the verification stage. Generally, backward reconstruction and next-token prediction based on iteratively refined content will be associated with higher weights. See Section 4.3 for a detailed comparison among different dependencies. In practice, some of the distributions in Eq. 5 may not be available for all tokens at each step. We calculate the ensemble utilizing available dependencies within the context window.

Algorithm 1 Sliding Blockwise Order-Agnostic Decoding

Input: Order-agnostic generator π_θ and verifiers v_θ and v_θ^{CD} based on OA-LLM p_θ, prompt $\boldsymbol{x}$, decoding context window size k, decoding block size b, maximum output sequence length T.
Initialize $t \leftarrow 0$, $\boldsymbol{y} \leftarrow \emptyset$. ▷ Initialize the current length of the output sequence
Initialize $t_s \leftarrow 1$, $t_e \leftarrow \min(k, b)$. ▷ Initialize the start and end indices of the block to predict and refine
while $t_s < T$ **do**
  Construct $\mathcal{Y}_{t_s:t_e} \leftarrow \left\{\{\tilde{y}_i\}_{i=t_s}^{t_e}, \tilde{y}_i \sim \pi_\theta(y_i \mid \boldsymbol{y}, \boldsymbol{x})\right\}$. ▷ Collect candidates through tree construction
  Select $\boldsymbol{y}_{t_s:t_e} \leftarrow \arg\max_{\tilde{\boldsymbol{y}}_{t_s:t_e} \sim \mathcal{Y}_{t_s:t_e}} \frac{1}{t_e - t_s + 1} \sum_{i=t_s}^{t_e} \left(v_\theta(\tilde{y}_i \mid \boldsymbol{y}, \boldsymbol{x}) + v_\theta^{\text{CD}}(\tilde{y}_i \mid \boldsymbol{y}, \boldsymbol{x})\right)$. ▷ Verify
  Update $\boldsymbol{y} \leftarrow \text{concat}(\boldsymbol{y}_{<t_s}, \boldsymbol{y}_{t_s:t_e})$.
  Set $t \leftarrow t_e$.
  for $i = t_s$ **to** t_e **do**
    Sample $r \sim U[0, 1]$ from a uniform distribution
    if $r < c(y_i \mid \boldsymbol{y}, \boldsymbol{x})$ **then**
      Set $t_s \leftarrow t_s + 1$. ▷ Slide the decoding block based on rejection sampling
      if y_i == `[EOS]` **then**
        Exit while loop.
      end if
    else
      Exit for loop.
    end if
  end for
  Set $t_e \leftarrow \min(t_s + b - 1, t + k)$.
end while
Output: $\boldsymbol{y}$

Verification. Following Cai et al. (2024), we employ tree attention[3] to select from multiple candidates sampled from the ensemble distribution π_θ. Each candidate is a combination of tokens used to fill the sliding block. Unlike previous works that only adopt the original next-token probability for verification, we also incorporate the backward reconstruction probabilities to leverage the refinement ability of COrAL. The verification score can thereby be formulated as follows:

$$v_\theta(y_t) = \frac{1}{\sum_{i=t-1}^{t+k} \lambda_{t-i}} \sum_{i=t-1}^{t+k} \lambda_{t-i} \log p_\theta(y_t \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x}) \tag{6}$$

Here, we only consider the next-token and backward predictions for the verification score calculation. This scheme can be further enhanced by introducing a contrastive objective (Li et al., 2023) that penalizes the possible failure cases in forward multi-token prediction:

$$v_\theta^{\text{CD}}(y_t) = \max\left(0, \log p_\theta(y_t \mid \boldsymbol{y}_{\leq t-1}, \boldsymbol{x}) - \frac{1}{\sum_{i=t-k}^{t-2} \lambda'_{t-i}} \lambda'_{t-i} \log p_\theta(y_t \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x})\right) \tag{7}$$

where $\lambda'_{t-i} = 1/\lambda_{t-i}$ to apply a higher penalty to predictions based on longer dependencies. Combining v_θ with v_θ^{CD}, we keep the candidate of the highest average score. We allow several refinement iterations for each position within a sliding block to enhance the generation quality. Specifically, we propose an ensemble rejection sampling scheme to determine the sliding step size through majority voting across multiple dependencies, where we accept each token with the probability:

$$c(y_t \mid \boldsymbol{y}_{\leq t+k}, \boldsymbol{x}) = \frac{1}{k+2} \sum_{i=t-1}^{t+k} \mathbb{1}_{p_\theta(y_t \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x}) > \min\left(\epsilon, \sqrt{\epsilon} \exp\left(-H\left(p_\theta(\cdot \mid \boldsymbol{y}_{\leq i}, \boldsymbol{x})\right)\right)\right)} \tag{8}$$

where $H(\cdot)$ is the entropy and ϵ is a fixed threshold to reject low-probability predictions. This acceptance scheme is inspired by truncation sampling (Hewitt et al., 2022; Cai et al., 2024) to choose candidates that are more likely to be sampled from the reference distributions. The sliding step size for each step is set to the length of the longest accepted prefix of the current block. We detail the sliding decoding procedure in Algorithm 1.

[3] To balance exploitation and exploration in tree construction, we select nodes according to the estimated accuracy of each token. Detailed considerations of candidate selection can be found in Appendix C.

Table 1: Result comparison of performance (accuracy %), speed (accepted tokens per second), and cost (seconds per sample) on arithmetic reasoning tasks. We compare against the conventional autoregressive greedy decoding approach as our next-token prediction baseline (NT). "verifier" and "multi-forward" represent the verification stage and multiple forward token prediction in inference.

Approach	GSM8K				MATH			
	Accu.	Speed	Speedup	Cost	Accu.	Speed	Speedup	Cost
NT	74.1	39.7	1.0×	3.67	21.8	38.7	1.0×	5.41
SC@4	76.2	37.8	–	15.5	23.0	38.0	–	16.6
Ours	75.3$\uparrow_{1.2}$	43.4	1.1×	3.35	22.7$\uparrow_{0.9}$	44.4	1.1×	4.82
Ours $_{\text{w/o verifier}}$	72.4$\downarrow_{1.7}$	156.8	**3.9**×	0.96	20.0$\downarrow_{1.8}$	139.7	**3.6**×	1.47
Ours $_{\text{w/o multi-forward}}$	**78.7**$\uparrow_{4.6}$	14.9	–	9.81	**24.3**$\uparrow_{2.5}$	11.5	–	18.2

4 Experiments

In this section, we demonstrate the efficiency and breadth of COrAL regarding the quality–speed trade-offs across arithmetic, logical reasoning, and code generation.

Datasets. For arithmetic reasoning, we train COrAL on MetaMathQA (395K) (Yu et al., 2024) and evaluate it using GSM8K (Cobbe et al., 2021) on grade school math word problems and MATH (Hendrycks et al., 2021) of challenging competition mathematics problems. For logical reasoning, we filter LogiCoT (Liu et al., 2023b) with deduplication and reformulation, obtaining 313K training samples. We assess logical reasoning performance with multiple-choice reading comprehension tasks that test interpretation and decision-making skills: LogiQA (Liu et al., 2023a), based on the Chinese Civil Service Examination, and ReClor (Yu et al., 2020), sourced from Law School Admission Council exams. For code generation, we train on Magicoder-Eval-Instruct-110K (Wei et al., 2023) and evaluate using programming tasks from HumanEval (Chen et al., 2021).

Experimental Protocol. To address the discrepancy between the next-token-based pre-trained model and the target order-agnostic model, we adopt a two-stage training strategy (Kumar et al., 2022) to progressively enhance order-agnostic modeling. We begin with a domain-specific supervised fine-tuned (SFT) model. In the first stage, we perform order-agnostic training exclusively on the last target-aware layer, while freezing the other layers to preserve the output quality. In the second stage, we train the entire model by focusing on the previously frozen layers first and then unlocking the last layer to train together. We use Mistral-7B-v0.3 and DeepSeek-Coder-6.7B-base as the base models for reasoning and code generation tasks, respectively. During inference, we explore the effect of the verification stage and ablate the values of decoding context window size and block size. Given the order-agnostic training tax resulting from the discrepancy between pretraining and fine-tuning objectives, we use next-token prediction with the same model as the baseline to ensure a fair comparison. We detail our hyperparameter settings in Section 4.2 and Appendix D.

4.1 Main Results

We compare our order-agnostic decoding approach (Section 3) with its next-token counterparts across three tasks. We also show the quality–speed trade-offs in by ablating the decoding settings.

Arithmetic Reasoning. As shown in Table 1, COrAL enhances the effectiveness and efficiency through different mechanisms in order-agnostic generation. Using both verification and multiple forward token prediction in decoding, COrAL surpasses the corresponding next-token baseline with comparable inference-time cost. Furthermore, by trading inference speed with iterative generation and verification through backward refinement, we observe a substantial improvement in accuracy from 74.1% to 78.7 and 21.8% to 24.3%, on GSM8K and MATH, respectively. When skipping the verification stage for quality control, our approach significantly speeds up the decoding process up to 3.9×. This demonstrates the flexibility of COrAL in enhancing both the generation quality and inference speed in mathematical reasoning.

Logical Reasoning. Table 2 compares the performance and generation speed of model outputs under different decoding settings on logical reasoning tasks. Similarly, COrAL improves the reasoning

Table 2: Result comparison of performance and speed on logical reasoning tasks.

Approach	LogiQA			ReClor		
	Accu.	Speed	Speedup	Accu.	Speed	Speedup
NT	55.1	33.6	1.0×	63.2	33.2	1.0×
Ours	58.2$\uparrow_{3.1}$	62.1	1.8×	62.7$\downarrow_{0.5}$	38.2	1.2×
Ours $_{\text{w/o verifier}}$	55.7$\uparrow_{0.6}$	99.1	**2.9**×	61.6$\downarrow_{1.6}$	72.0	**2.2**×
Ours $_{\text{w/o multi-forward}}$	**59.1**$\uparrow_{4.0}$	8.9	–	**64.7**$\uparrow_{1.5}$	11.3	–

Table 3: Result comparison of pass rates and speed on code generation.

Approach	HumanEval		
	Pass@1	Speed	Speedup
NT	**64.6**	42.2	1.0×
Ours	13.0$\downarrow_{51.6}$	45.8	1.1×
Ours $_{\text{w/o verifier}}$	6.5$\downarrow_{58.1}$	119.0	**2.8**×
Ours $_{\text{w/o multi-forward}}$	61.6$\downarrow_{3.0}$	28.8	–

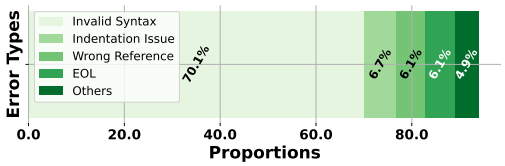


Figure 4: Meso-analysis of error cases in code generation (Ours $_{\text{w/o verifier}}$) on HumanEval. The primary failure cases come from syntax errors.

performance by augmenting next-token prediction exclusively with backward refinement. However, we observe a discrepancy in the performance improvements on LogiQA and ReClor with absolute increases of 4.0% and 1.5% in corresponding accuracies. We attribute this gap to the imbalanced proportions of the two tasks in our SFT data from LogiCoT (Liu et al., 2023b). This also implies the importance of high-quality data selection to boost the effect of order-agnostic training to model different dependencies related to the target tasks.

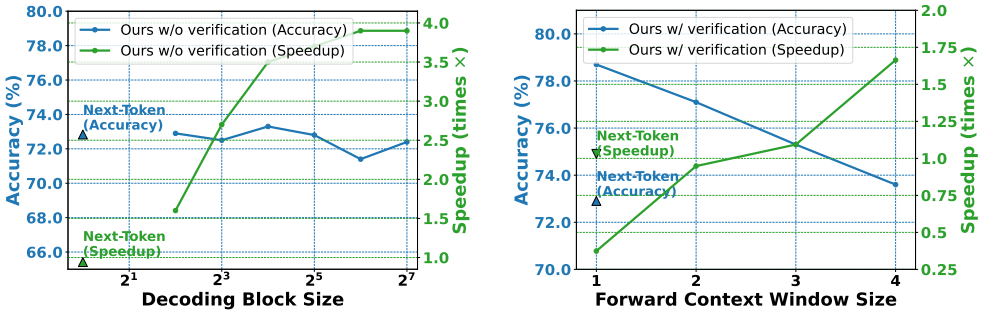
Code Generation. Results on code generation, however, show an opposite effect of order-agnostic modeling on performance. In Table 3, we observe substantial performance drops across different decoding settings using COrAL. For example, without verification, the pass rate on HumanEval decreases to 6.5% from 64.6% of next-token prediction. This gap remains to be large when applying verification for quality control. Error analysis in Figure 4 indicates that the major cause of this drop comes from the erroneous syntax, where the primary error type, *Invalid Syntax*, accounts for 70.1% of all samples. To mitigate this issue, we can turn off the mechanism of forward multi-token prediction and increase the threshold ϵ in Eq. 8 to reject tokens with low confidence scores. For example, with $\epsilon = 0.5$, COrAL achieves a comparable pass rate of 61.6% compared to 64.6% of the baseline. The absolute decrease of 3.0% indicates the deficiency of COrAL in producing incoherent content, showing the importance of specific designs for tasks requiring strict textual formats.

4.2 Ablation Studies

In this section, we analyze the core designs of COrAL to enable efficient iterative refinement. We probe the effect of different training and decoding hyperparameters.

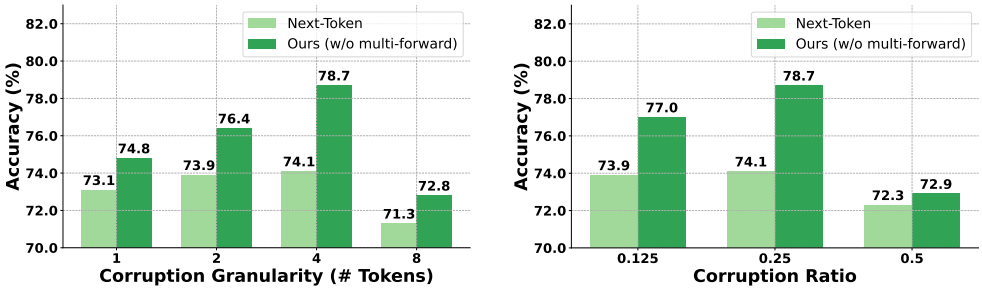
Backward Refinement Improves Generation Quality. Figure 1 shows how the performance and inference cost scale with iterative refinement. Note that even without backward dependencies in prediction, COrAL can still perform backward refinement using the next-token prediction. In this case, we examine the effect of backward dependencies with different context window sizes. Notably, the performance of iterative refinement scales faster than the inference cost as the iteration time increases. Furthermore, leveraging backward dependencies, COrAL reaches a higher plateau of performance compared to refining with forward dependencies only. However, the fast saturation of performance improvement with larger refinement times indicates a relatively low upper bound of the enhancement brought by backward reconstruction. We extensively discuss this problem attributed to the discrepancy between pre-training and fine-tuning objectives in Appendix E.

Quality–Speed Trade-off in Inference. In Section 4.1, we demonstrate the quality–speed trade-off by ablating the employment of verification and forward multi-token prediction. We now provide a detailed analysis of the decoding hyperparameters to show this trade-off. We consider the block size b and the forward context window size k, two variables closely related to the inference speed and quality. For block size, we probe its effect in reducing inference time in the verification-free case



(a) Block size for acceleration. (b) Quality–speed trade-off.

Figure 5: Quality–speed trade-offs on GSM8K. Generally, COrAL accelerates inference with larger decoding block size b and forward context window size k.



(a) Effect of corruption granularity. (b) Effect of corruption ratio.

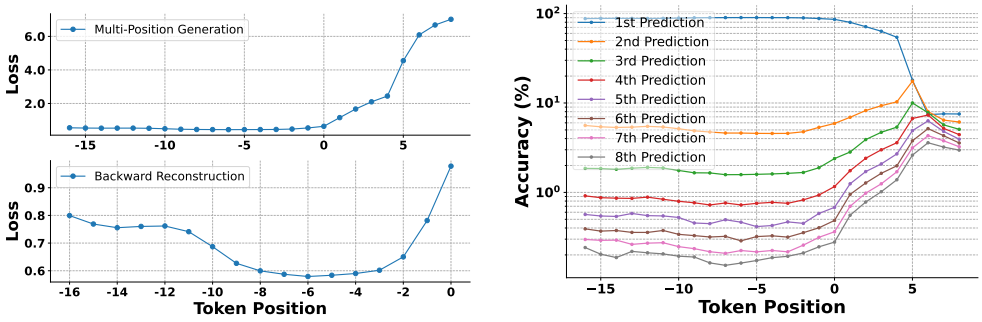
Figure 6: Ablation on the effects of corruption granularity and ratios in training for backward reconstruction. We probe the variation in model improvements from backward dependencies.

to maximize the speedup rate. Figure 5a shows that we can push the speedup boundary toward the corresponding upper bound of k with large b. For example, given $k=4$, we approach the maximum speedup rate $4\times$ with large block sizes such as $b=64$ and 128. Notably, leveraging the backward refinement capability of COrAL, this block size-driven acceleration process retains the generation quality at the same level, illustrating the balance of efficiency and effectiveness of our acceleration mechanism. For forward context window size, we adopt the two-stage prediction–verification setting to explore the quality improvement boundary of the iterative refinement mechanism. Figure 5b shows trends of performance drop and inference speedup when increasing k. We explain this trade-off as a reflection of the decreasing precision in predicting future tokens of longer dependencies.

Learning from Corruption Enhances Refinement Capability. One core design in COrAL is the denoising process to enable iterative refinement, where the corruption strategy is crucial for controlling data quality and model performance. In Figure 6, we analyze the variation in performance improvements from backward refinement (w/ multi-forward) when applying corruption with different granularity or ratios. Given a backward context window size $k = 8$, we observe a more significant improvement when applying corruption on longer pieces of text. For example, COrAL achieves an absolute increase of 4.6% with granularity 4, compared to 1.7% under token-level corruption. However, as the corrupted context gets longer, the model's capability to learn from mistakes may also degrade. One possible reason for this performance drop is the difficulty and inconsistency in simultaneous multi-token regeneration, as reconstructing more tokens brings higher uncertainty and noise. This indicates the importance of using a reasonable corruption granularity to obtain data of good quality and maintain training stability. Likewise, we see a similar trend when the corruption ratio varies. Specifically, a high corruption ratio such as 0.5 can damage the semantic meaning of the context, leading to a performance drop in both our and baseline approaches. Nevertheless, we can still benefit when increasing the corruption ratio within a reasonably lower range, such as 0.125 to 0.25, to enhance the reconstruction process.

4.3 Further Analysis

We analyze COrAL's capability to model different dependencies, and the potential computation overhead from order-agnostic modeling.



(a) Token-wise loss. (b) Token-wise accuracy of top-8 predictions.

Figure 7: Token-wise losses and accuracies corresponding to different dependencies.

How does COrAL model order-agnostic dependencies? We compare the model capabilities across different positions using token-wise losses and accuracies in Figure 7. Generally, COrAL performs better on backward reconstruction than forward prediction, as shown in the lower losses and higher accuracies on backward dependencies. Notably, we see better generalizability of backward reconstruction. For example, given the backward context window size $k = 8$ and forward context window size $k = 4$, we find that the loss and accuracy of backward reconstruction with dependencies longer than the training context window size, such as positions $|-9| > |-8|$, are also at the same level as other backward dependencies. Differently, we observe a dramatic increase in loss and a drop in accuracy from positions 4 to 5 on longer dependencies in forward prediction. This explains how backward refinement benefits from more information in sequence-level generation to improve performance. We observe decreased performance for forward prediction as the dependency gets longer, especially when it exceeds the forward context window size in training. However, we can mitigate this issue by aggregating multiple predictions for each position. As shown in Figure 7b, while forward positions with longer dependencies obtain lower accuracies on tghe first prediction, the accumulated accuracies of their non-first predictions are generally higher than those from other dependencies. This illustrates how COrAL can benefit from the tree construction and verification stage in decoding (Section 3) by considering multiple candidates for each position.

Computation Overhead. One concern regarding order-agnostic modeling is the potential computation overhead to accommodate more dependencies in the context windows. As target-aware RoPE is only applied on the last layer, this overhead scales relatively slower as we increase the number of positions to predict. For example, with forward and backward context window sizes each set as $k = 4$, each forward pass of COrAL costs 5.48 TFLOPS, compared with 2.81 TFLOPS of next-token prediction. In other words, COrAL predicts $8\times$ number of tokens with less than $2\times$ overhead in computational cost. This indicates the efficiency of COrAL in leveraging available computation resources to accelerate and enhance inference. Furthermore, we can adjust the forward and backward context window sizes to determine the number of tokens to predict in parallel, demonstrating the flexibility and generalizability of COrAL with target-aware RoPE.

5 Conclusion and Future Work

By unifying denoising with context-wise order-agnostic language modeling and introducing target-aware positional encoding, COrAL incorporates iterative refinement directly into the language generation process while keeping inference costs low. This approach offers a promising direction for developing more efficient and capable large language models by effectively capturing local dependencies within context windows and reducing inference latency.

The effectiveness and efficiency of COrAL underscores the promise of order-agnostic strategies as a generalized architecture to facilitate generative language modeling and text generation. Specifically, it suggests new opportunities to unify: (**i**) the sequence modeling and varying-length generation abilities of autoregressive modeling, (**ii**) the multi-dependency modeling and multi-token prediction mechanisms in order-agnostic modeling, and (**iii**) the efficient way of iterative refinement in denoising techniques. We hope our work will motivate future research to explore order-agnostic modeling and denoising in various tasks and other domains beyond sequence modeling.

REFERENCES

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *J. Mach. Learn. Res.*, 15(1):3563–3593, 2014. doi: 10.5555/2627435.2750359. URL https://dl.acm.org/doi/10.5555/2627435.2750359.

Shengnan An, Zexiong Ma, Siqi Cai, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Can llms learn from mistakes? an empirical study on reasoning tasks. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pp. 833–854. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.findings-emnlp.46.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 17981–17993, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 1171–1179, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/e995f98d56967d946471af29d7bf99f1-Abstract.html.

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, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=PEpbUobfJv.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *CoRR*, abs/2302.01318, 2023. doi: 10.48550/ARXIV.2302.01318. URL https://doi.org/10.48550/arXiv.2302.01318.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

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. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin T. Vechev. Controlled text generation via language model arithmetic. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=SLw9fp4yI6.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of LLM inference using lookahead decoding. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=eDjvSFOkXw.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pp. 6111–6120. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1633. URL https://doi.org/10.18653/v1/D19-1633.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=pEWAcejiU2.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=jQj-_rLVXsj.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1l8BtlCb.

Shangtong Gui, Chenze Shao, Zhengrui Ma, Xishan Zhang, Yunji Chen, and Yang Feng. Non-autoregressive machine translation with probabilistic context-free grammar. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/11c7f1dd168439884b6dfb43a7891432-Abstract-Conference.html.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 15979–15988. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01553. URL https://doi.org/10.1109/CVPR52688.2022.01553.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track*

on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

John Hewitt, Christopher D. Manning, and Percy Liang. Truncation sampling as language model desmoothing. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 3414–3427. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.FINDINGS-EMNLP.249. URL https://doi.org/10.18653/v1/2022.findings-emnlp.249.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=Lm8T39vLDTE.

Edward J. Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Ouj6p4ca60.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 1051–1068. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.67. URL https://doi.org/10.18653/v1/2023.emnlp-main.67.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. Non-autoregressive machine translation with disentangled context transformer. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5144–5155. PMLR, 2020. URL http://proceedings.mlr.press/v119/kasai20a.html.

Siqi Kou, Lanxiang Hu, Zhezhi He, Zhijie Deng, and Hao Zhang. Cllms: Consistency large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=8uzBOVmh8H.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=UYneFzXSJWh.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 1173–1182. Association for Computational Linguistics, 2018. doi: 10.18653/V1/D18-1149. URL https://doi.org/10.18653/v1/d18-1149.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 12286–12312. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.687. URL https://doi.org/10.18653/v1/2023.acl-long.687.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. Logiqa 2.0 - an improved dataset for logical reasoning in natural language understanding. *IEEE ACM Trans. Audio Speech Lang. Process.*, 31:2947–2962, 2023a. doi: 10.1109/TASLP.2023.3293046. URL https://doi.org/10.1109/TASLP.2023.3293046.

Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. Logicot: Logical chain-of-thought instruction tuning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 2908–2921. Association for Computational Linguistics, 2023b. doi: 10.18653/V1/2023.FINDINGS-EMNLP.191. URL https://doi.org/10.18653/v1/2023.findings-emnlp.191.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In Rajiv Gupta, Nael B. Abu-Ghazaleh, Madan Musuvathi, and Dan Tsafrir (eds.), *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3, ASPLOS 2024, La Jolla, CA, USA, 27 April 2024- 1 May 2024*, pp. 932–949. ACM, 2024. doi: 10.1145/3620666.3651335. URL https://doi.org/10.1145/3620666.3651335.

Giovanni Monea, Armand Joulin, and Edouard Grave. Pass: Parallel speculative sampling. *CoRR*, abs/2311.13581, 2023. doi: 10.48550/ARXIV.2311.13581. URL https://doi.org/10.48550/arXiv.2311.13581.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/. Accessed: 2024-09-12.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 3806–3824. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.248. URL https://doi.org/10.18653/v1/2023.findings-emnlp.248.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. Automatically correcting large language models: *Surveying the Landscape of Diverse Automated Correction Strategies*. *Trans. Assoc. Comput. Linguistics*, 12:484–506, 2024. doi: 10.1162/TACL_A_00660. URL https://doi.org/10.1162/tacl_a_00660.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In Dawn Song, Michael Carbin, and Tianqi Chen (eds.), *Proceedings of the Sixth Conference on Machine Learning and Systems, MLSys 2023, Miami, FL, USA, June 4-8, 2023*. mlsys.org, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html.

Alec Radford. Improving language understanding by generative pre-training. 2018.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950, 2023. doi: 10.48550/ARXIV.2308.12950. URL https://doi.org/10.48550/arXiv.2308.12950.

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodolà. Accelerating transformer inference for translation via parallel decoding. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 12336–12355. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.689. URL https://doi.org/10.18653/v1/2023.acl-long.689.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aäron van den Oord. Step-unrolled denoising autoencoders for text generation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=T0GpzBQ1Fg6.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Kunyu Shi, Qi Dong, Luis Goncalves, Zhuowen Tu, and Stefano Soatto. Non-autoregressive sequence-to-sequence vision-language models. *CoRR*, abs/2403.02249, 2024. doi: 10.48550/ARXIV.2403.02249. URL https://doi.org/10.48550/arXiv.2403.02249.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=WNzy9bRDvG.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 10107–10116, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/c4127b9194fe8562c64dc0f5bf2c93bc-Abstract.html.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/J.NEUCOM.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

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, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 467–475. JMLR.org, 2014. URL http://proceedings.mlr.press/v32/uria14.html.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *J. Mach. Learn. Res.*, 17:205:1–205:37, 2016. URL https://jmlr.org/papers/v17/16-272.html.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In William W. Cohen, Andrew McCallum, and Sam T. Roweis (eds.), *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pp. 1096–1103. ACM, 2008. doi: 10.1145/1390156.1390294. URL https://doi.org/10.1145/1390156.1390294.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Source code is all you need. *CoRR*, abs/2312.02120, 2023. doi: 10.48550/ARXIV.2312.02120. URL https://doi.org/10.48550/arXiv.2312.02120.

Sean Welleck, Kianté Brantley, Hal Daumé III, and Kyunghyun Cho. Non-monotonic sequential text generation. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6716–6726. PMLR, 2019. URL http://proceedings.mlr.press/v97/welleck19a.html.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Qizhe Xie. Self-evaluation guided beam search for reasoning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/81fde95c4dc79188a69ce5b24d63010b-Abstract-Conference.html.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P. Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *CoRR*, abs/2405.00451, 2024. doi: 10.48550/ARXIV.2405.00451. URL https://doi.org/10.48550/arXiv.2405.00451.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 5754–5764, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of language models: Part 2.2, how to learn from mistakes on grade-school math problems, 2024. URL https://arxiv.org/abs/2408.16293.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=N8N0hgNDRt.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. Reclor: A reading comprehension dataset requiring logical reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgJtT4tvB.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. Draft& verify: Lossless large language model acceleration via self-speculative decoding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 11263–11282. Association for Computational Linguistics, 2024a. doi: 10.18653/V1/2024.ACL-LONG.607. URL https://doi.org/10.18653/v1/2024.acl-long.607.

Qi Zhang, Tianqi Du, Haotian Huang, Yifei Wang, and Yisen Wang. Look ahead or look around? A theoretical comparison between autoregressive and masked pretraining. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024b. URL https://openreview.net/forum?id=2rPoTgEmjV.

Tianjun Zhang, Fangchen Liu, Justin Wong, Pieter Abbeel, and Joseph E. Gonzalez. The wisdom of hindsight makes language models better instruction followers. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 41414–41428. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhang23ab.html.

Xu Zhang, Xunjian Yin, and Xiaojun Wan. Contrasolver: Self-alignment of language models by resolving internal preference contradictions, 2024c. URL https://arxiv.org/abs/2406.08842.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. Take a step back: Evoking reasoning via abstraction in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=3bq3jsvcQ1.

Table 4: Conceptual comparison regarding desired features across different architectures.

Architectures	VL	BT	MV	MD	FS	EF	IT
Next-Token AR (Uria et al., 2016)	✓	✗	✗	✗	✓	✗	✗
Permutation-Based AR (Uria et al., 2014)	✗	✓	✓	✓	✗	✓	✗
NAR (Gu et al., 2018)	✗	✓	✓	✓	✓	✓	✓
Diffusion (Ho et al., 2020)	✗	✓	✓	✓	✓	✗	✓
Consistency Model (Song et al., 2023)	✗	✓	✓	✓	✓	✓	✓
COrAL (Ours)	✓	✓	✓	✓	✓	✓	✓

Limitations

This work proposes an approach to integrate iterative denoising with order-agnostic language modeling to enhance both the effectiveness and efficiency of LLM inference. While it offers a promising paradigm for mitigating issues related to monotonic dependencies and inference latency in conventional autoregressive models, several directions remain for further exploration, including designing corruption and decoding strategies to tailor the model to specific tasks, optimizing the training process to overcome the order-agnostic training tax, and probing the generalizability and scalability of COrAL across different context sizes, model scales, and tasks.

Specifically, order-agnostic language modeling can struggle with tasks that demand specific output formats or syntax due to inconsistencies in the multi-token predictions. This indicates the importance of a task-specific design of the acceptance scheme in order-agnostic decoding. For instance, the performance of the verification policy in Eq. 6 may vary by language and domain. Additionally, applying semantic-aware weights to different dependencies could further enhance task-specific features in the generated outputs. Future work can further explore the potential of incorporating different evaluation heuristics to guide the inference process.

Furthermore, incorporating corrupted data may introduce discrepancies between training- and inference-time objectives. For example, our experiments explore rule-based context-wise corruption strategies to create noisy data. Future work could focus on diversifying the types of corruption and scaling the difficulty level and proportion to better understand their impacts on model capabilities.

Finally, due to the computation constraint, we explore the model capabilities in order-agnostic modeling with fixed context window sizes during the SFT stage only. Future work may investigate the effect of scaling context window sizes in both forward and backward directions. Moreover, increasing the context window sizes may exacerbate the discrepancy between autoregressive pre-training and order-agnostic fine-tuning. We thus anticipate future work to extend COrAL to the pre-training stage to further enhance model capabilities.

Potential Broader Impact

Compared to conventional autoregressive modeling, COrAL leverages multi-token prediction and reconstruction to backtrack and iteratively refine past generations. This strategy mirrors the human decision-making process in real-world task completion. We anticipate COrAL to inspire the community to design more efficient and effective frameworks to enhance interpretability and alignment with the reasoning and planning process of humans.

A Conceptual Comparison among Model Architectures

We consider the properties an ideal architecture should have as follows:

- **VL**: varying-length generation
- **BT**: backtrack / look-ahead
- **MV**: multi-variable generation
- **MD**: multi-dependency (inter-sample connection) modeling
- **FS**: fitting feasibility
- **EF**: inference efficiency
- **IT**: mechanism of iterative refinement

B Further Related Work

Order-Agnostic Language Modeling. Order-agnostic architectures have been explored to overcome the limitations of sequential generation in autoregressive models. Uria et al. (2014) propose permutation-based autoregressive models to learn different data orderings for density estimation. In language modeling, Yang et al. (2019) further explore the idea of order-agnostic autoregressive modeling as a generalized pretraining method. Welleck et al. (2019) explore the possibility of non-monotonic text generation in a tree-structure manner and achieve competitive performance with the conventional left-to-right sequential generation. To avoid the high latency in autoregressive decoding, Gu et al. (2018) introduce non-autoregressive machine translation by breaking the sequentially causal dependency across time into conditionally independent per-step distributions with latent variables as intermediate steps. Lee et al. (2018) adopt iterative refinement to interpret the latent variable model, inspired by the design of denoising autoencoders (Alain & Bengio, 2014). Follow-up works on non-autoregressive machine translation show promising performance of the iterative refinement process of mask-predict (Ghazvininejad et al., 2019; Kasai et al., 2020). Our work explores the potential of unifying the strengths of order-agnostic modeling and denoising to advance sequential modeling in LLMs, demonstrating an efficient way to conduct iterative refinement internally.

LLM Self-Refinement. Self-refinement in LLMs focuses on various feedback mechanisms to improve the model performance dynamically. Existing works utilize the feedback mainly in two directions. The first one relates to prompting-based frameworks such as instance-level refinement (Madaan et al., 2023), step-level guided search (Yao et al., 2023; Xie et al., 2023), and principle-driven reasoning (Zheng et al., 2024). Another line of work adapts the feedback as training signals to further enhance the performance of LLMs, including rationale-augmented refinement (Zelikman et al., 2022), hindsight-driven alignment (Zhang et al., 2023; 2024c), and search-enhanced preference learning (Xie et al., 2024). Unlike existing works relying on the AR foundation in conventional LLMs, we leverage the order-agnostic modeling ability of COrAL to conduct the iterative refinement internally while foregoing the computation overhead in AR-LLMs to maintain efficiency.

Parallel Decoding. Parallel decoding methods aim to accelerate LLM inference by generating multiple tokens simultaneously rather than sequentially. Non-autoregressive models (Gu et al., 2018) and blockwise decoding approaches (Stern et al., 2018; Monea et al., 2023; Cai et al., 2024) have enabled faster generation but often struggle with output inconsistencies. Speculative decoding techniques (Leviathan et al., 2023; Chen et al., 2023; Miao et al., 2024) adopts a faster draft model to speedup inference while struggling with the deficiency in scalability. Self-speculative decoding (Zhang et al., 2024a) uses the same model for drafting by selectively skipping certain intermediate layers. Look-ahead (Santilli et al., 2023) and Jacobi (Fu et al., 2024) decoding, on the other hand, directly utilize the AR LLMs to enhance performance iteratively. Consistency LLMs (Kou et al., 2024) further reduces this iteration time drawing inspiration from consistency models (Song et al., 2023; Song & Dhariwal, 2024). In this work, we realize parallel decoding leveraging the multi-token generation ability of COrAL. Instead of decoding toward the forward direction only, we support backward refinement simultaneously to enhance the generation quality further.

Iterative Refinement. Prompt engineering approaches (Madaan et al., 2023; Shinn et al., 2023) exploit incorrect attempts in historical data to improve the performance of a frozen LLM. In contrast, our method enables the model to directly correct generated mistakes via backward refinement. Verifier-based methods (Cobbe et al., 2021; Lightman et al., 2024) train separate models to re-rank outputs. These strategies are orthogonal to our method, which could further enhance COrAL by providing stronger verification mechanisms. An et al. (2024) demonstrate that the mistake reasoning data can be directly utilized through a standard fine-tuning approach. However, this approach relies on AR-LLMs and sequential prediction, whereas COrAL introduces a fundamentally new paradigm by enabling mistake correction through backward dependencies.

Scheduled Sampling. Scheduled sampling (Bengio et al., 2015) aims to mitigate the discrepancy between training and inference, it gradually transitions from teacher-forcing to self-generated inputs using curriculum learning. In contrast, COrAL decomposes the order-agnostic training into two separate objectives: forward prediction with ground-truth input and backward reconstruction with corrupted input. Inspired by scheduled sampling, future iterations of COrAL could explore cur-

riculum strategies to gradually increase corrupted input ratios, enhancing robustness and stability. Furthermore, scheduled sampling is designed for sequential decoding at inference, while COrAL employs blockwise order-agnostic decoding, enabling multi-token forward prediction for speedup and backward refinement for quality improvement.

C Candidate Tree Construction in Order-Agnostic Decoding

Our specific design of tree construction aims to explore promising combinations of multi-position predictions with a fixed budget for the number of total nodes in the tree. Unlike selecting promising nodes based on the accuracies of the top predictions of different heads in Cai et al. (2024), we forego the need of a validation set for accuracy calculation by leveraging the model confidence of each prediction with a dedicated scaling factor. Let $p_t^{(i)}$ denote the model-predicted probability of the i-th top candidate for the t-th token. For a candidate sequence composed by the top $[i_{t_s}, i_{t_s+1}, \cdots, i_{t_e}]$ predictions of tokens at different positions, we estimate its accuracy as:

$$\prod_{j=t_s}^{t_e} \left(p_j^{(i_j)} / \gamma_j \right) \tag{9}$$

where γ_i is a scaling factor to up weight the predictions based on nonconsecutive forward dependencies. As shown in Figure 7, this process benefits from the fact that COrAL obtains higher accuracies on non-first predictions on such dependencies. Empirically, we set these factors to be 1.1, 1.2, 1.3 for the second, the third, and the fourth tokens to predict, respectively.

Following Eq. 9, we construct the tree in a greedy manner, adding the node with the highest confidence to the tree one by one. This process considers the token-wise confidence as the expected contribution of each prediction to the tree. We repeat the node-adding process until the total number of nodes reaches the desired number to accommodate the maximum sequence length the model can deal with.

D Hyperparameter Setting

Training. For order-agnostic training, we train for 3 epochs at each stage with a batch size of 128 on all tasks. We fix the context window size in training as 4 and 8 for forward and backward dependencies. At different training stages, we recommend employing different learning rates. We set the learning rate as $5\text{e}{-6}$ and $1\text{e}{-4}$ in reasoning and code generation, respectively, for the last-layer tuning stage. We increase the learning rates at the second stage to be $1\text{e}{-6}$ and $2\text{e}{-5}$ for corresponding tasks following the general SFT settings.

We corrupt the training data with granularity 4 and ratio 0.25 across all tasks for backward reconstruction. As discussed in Section 4.2, we ablate the granularity and ratios on mathematical reasoning data to study their respective effects on enhancing model's refinement abilities. Note that as the training context window sizes are fixed as 4 and 8 for forward and backward dependencies, the optimal corruption hyperparameters may vary as we scale the context window sizes. Due to the computation constraint, we leave it to future work to explore the combinations of different granularities and corruption strategies.

Decoding. For order-agnostic decoding, we suggest adopting different context window sizes and block sizes to balance the quality and inference speed in different tasks. We report the experiment results (Section 4.1) under the same context window and block sizes across the three tasks, where the forward and backward context window and block sizes are 4, 8, and 64, respectively. For verification, we set $\epsilon = 0.2$ and 0.5 for reasoning and code generation. We implemented our order-agnostic decoding and corresponding next-token baseline without KV-Cache (Pope et al., 2023). During decoding, we set the batch size 1 and conduct inference on a single GPU.

Computation. For reasoning tasks with maximum sequence length 512, all training experiments were done on single-node eight 40GB A100s. For code generation task with maximum sequence length 2048, we conduct training and inference on single-node four 80GB H100s.

Prompt

The Kennel house keeps 3 German Shepherds and 2 Bulldogs.
If a German Shepherd consumes 5 kilograms of dog food and a bulldog consumes 3 kilograms of dog food per day.
How many kilograms of dog food will they need in a week?

Responses

Sliding Blockwise Order-Agnostic Decoding

1 The bakeraked

2 The bakeraked 200

3 The bakeraked 200 loaves of bread

.....

6 The bakeraked 200 loaves of bread.
They sold 93 loaves the in3

7 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning morning and

8 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in

.....

38 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 1226 loaves

39 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves. left...

40 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves left.
Therefore, they266

.....

47 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves left.
Therefore, they had 200 - 126 =

48 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves left.
Therefore, they had 200 - 126 = 72

49 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves left.
Therefore, they had 200 - 126 = 74 loaves of bread

.....

59 The bakeraked 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They returned 6 loaves, so they had 132 - 6 = 126 loaves left.
Therefore, they had 200 - 126 = 74 loaves of bread left.
74
The answer is: 74 ✓

Next-Token Based Greedy Decoding

1 The

2 The b

3 The bery

.....

6 The bery b had 2

7 The bery b had 20

8 The bery b had 200

.....

38 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold

39 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a

40 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total

.....

47 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 3

48 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39

49 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 =

.....

59 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They also

.....

106 The bery b had 200 loaves of bread.
They sold 93 loaves in the morning and 39 loaves in the afternoon,
so they sold a total of 93 + 39 = 132 loaves.
They also returned 6 unsold loaves,
so they had 200 - 132 + 6 = 72 loaves of bread left.
72
The answer is: 72 ✗

Figure 8: Qualitative result comparison on GSM8K.

E Extended Discussion

In this section, we extensively discuss the training protocol we design to endow AR-LLMs with order-agnostic ability without pretraining. Lastly, we illustrate how COrAL efficiently corrects mistakes in previous generations in qualitative analysis.

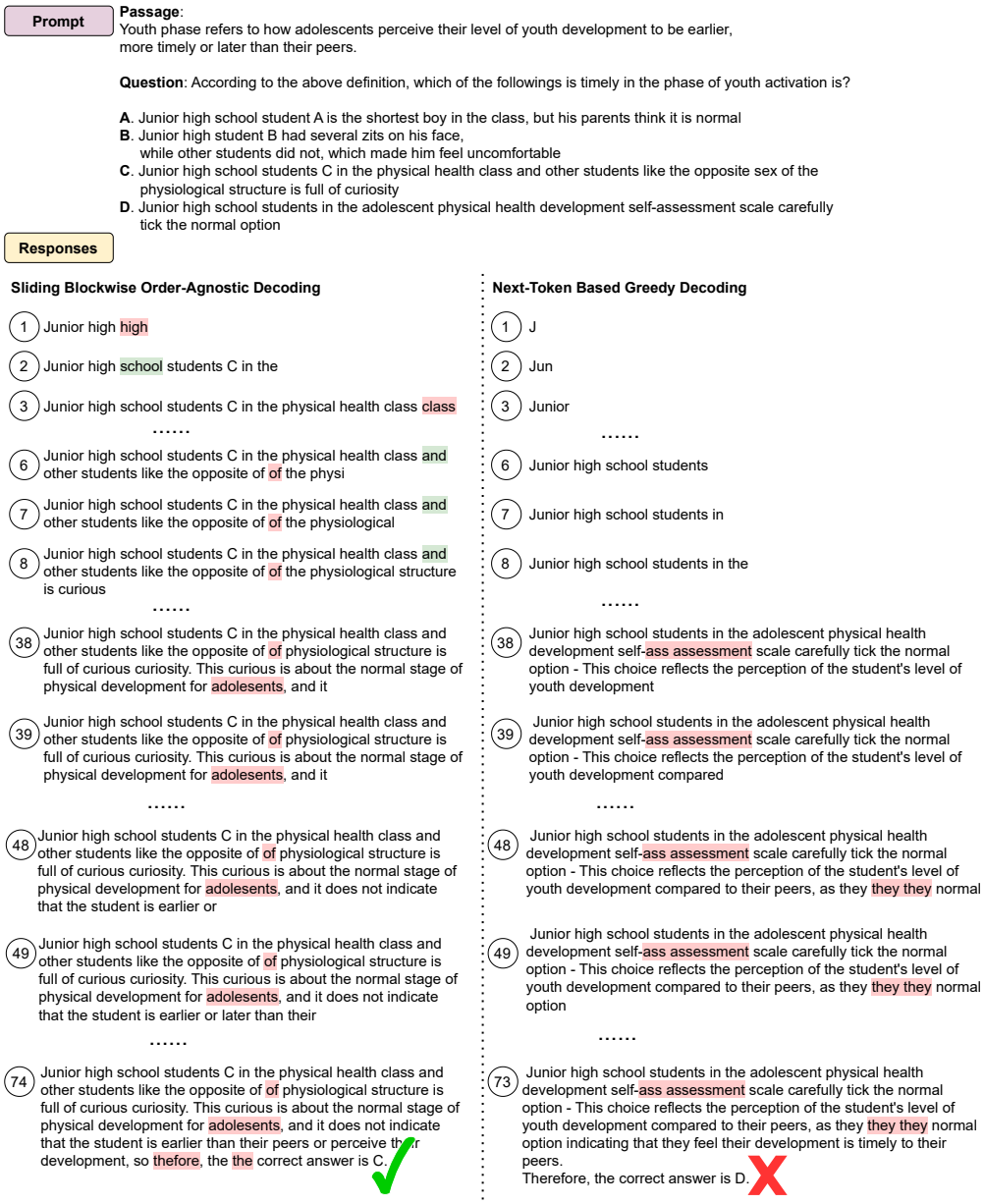


Figure 9: Qualitative result comparison on LogiQA.

Effect of Two-Stage Training. As discussed in Section 4.2, a high corruption ratio can cause a collapse in model performance as the noisy data contains corrupted information in a format that the model has not seen in pretraining. Furthermore, we are also faced with the order-agnostic training tax to endow an AR-based LLM with denoising and multi-token prediction abilities. In this section, we elaborate on the two-stage training we designed to mitigate this issue. Following Cai et al. (2024), we first tune the last layer where we apply target-aware RoPE. This adapts the previous parameterization on next-token prediction to target-aware multi-position prediction. Due to the discrepancy of training objectives in pretraining and fine-tuning, full fine-tuning is still essential to ensure better performance on multi-token prediction. To stabilize the training process, we then freeze the last layer and gradually unlock it through the second training stage of full fine-tuning.

Empirically, we find this strategy effective for stabilizing the autoregressive loss changes in forward prediction. However, we observe an order-agnostic training tax where the next-token prediction performance drops from 77.0% to 76.5% and then 74.1% after the first and second stages, respectively. This performance degradation possibly comes from two aspects: the difference in training objectives and the incorporation of corrupted data in fine-tuning. We leave it to future work to further explore the effect of applying our order-agnostic framework to the pretraining stage.

Qualitative Analysis. Our qualitative analysis on GSM8K and LogiQA showcases how COrAL corrects previously generated mistakes through the iterative internal process. In Figure 8, COrAL obtained a wrong calculated result 72 at the 48-th step. However, the backward refinement mechanism enables it to backtrack and refine the result to the correct number, 74, as shown at the 49-th step. In contrast, the next-token baseline cannot correct the erroneous 72, leading to the wrong final result. On the other hand, we observe the incoherence in COrAL's generation where COrAL can fail in correcting the mistakes when it happens to skip some positions during generation. For example, at the 1-st step, COrAL outputs "bakeraked" instead of "baker baked". This error incurs a chain reaction where the subsequent outputs all omit the correct token " b" right after "baker", indicating the need for further enhancement on the generation fluency of order-agnostic methods.

On LogiQA, interestingly, we observe a higher frequency of the inconsistencies in COrAL's generation. As discussed in Section 4.1, we attribute this scenario to the relatively low proportion of LogiQA-related training data in LogiCoT, where there are only 5K samples out of the 313K data points. As shown in Figure 9, while the COrAL produces several grammatical errors in a generation, it still achieves the correct result. This indicates the advanced ability of COrAL to sematically escape from paths that may lead to dead ends through iterative refinement.

Further Analysis on Computation Overhead. As discussed in Section 4.3, the computation overhead in COrAL scales efficiently relative to the number of predicted positions, with target-aware RoPE applied only to the last layer. We now provide a detailed computation comparison between COrAL and the baseline approaches.

Table 5: Computation comparison across different decoding approaches on GSM8K.

Approach	TFLOPS (per forward pass)	Accuracy (%)	Speed (tokens per second)	Speedup
Next-Token (NT)	2.81	74.1	39.7	1.0×
Ours	13.6	75.3	43.4	1.1×
Ours w/o verifier	5.48	72.4	156.8	3.9×
Ours w/o multi-forward	17.9	78.7	14.9	–