ANY-ORDER ANY-SUBSET AUTOREGRESSIVE MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose Any-order Any-subset Autoregressive modeling (A3), a novel sequence generation framework that generalizes standard autoregressive (AR) factorization to support the prediction of arbitrary token groups in any order. A3 overcomes the limitations of conventional left-to-right decoding by enabling flexible groupwise generation while preserving probabilistic rigor and training stability. Our design combines a two-stream attention architecture with a progressive training strategy, allowing both efficient parallel decoding and robust modeling of diverse dependency structures. Empirical results demonstrate that A3 achieves a superior trade-off between generation speed, flexibility, and quality compared to state-of-the-art AR and diffusion-based methods. This work offers a unified approach for a flexible, efficient, and novel language modeling paradigm.

1 INTRODUCTION

Autoregressive (AR) modeling has been the dominant paradigm for text generation, underpinning the success of most large language models (Touvron et al., 2023; Bai et al., 2023a; Radford et al., 2018). In the AR framework, the joint probability of a token sequence is factorized in a fixed left-to-right order, generating one token at a time. While simple and effective, this formulation imposes several intrinsic limitations. For example, the strict left-to-right ordering prevents models from fully exploiting bidirectional context during generation. Also, the one-token-at-a-time decoding process creates a computational bottleneck that limits generation efficiency. These fundamental drawbacks constrain both modeling flexibility and decoding speed, especially in long-context and complex generation scenarios (Kuratov et al., 2024; Bai et al., 2023b).

Alternative approaches such as masked diffusion language models (Li et al., 2022; Gong et al., 2025; Gulrajani & Hashimoto, 2024) attempt to address these limitations by enabling parallel prediction of multiple tokens. By iteratively denoising partially masked sequences, diffusion models can achieve any-order generation and leverage bidirectional context. However, these methods introduce new challenges: they often require carefully tuned noise schedules and multi-step inference, which complicates training and slows decoding, and they typically yield less stable training dynamics and lower sample quality compared to AR approaches (Kim et al., 2025).

In this work, we propose **Any-order Any-subset Autoregressive modeling** (A3), a new framework that merges the strengths of AR modeling with the flexibility of parallel generation. A3 generalizes the standard AR factorization by partitioning the sequence into arbitrary groups of tokens and predicting them in any order. This groupwise factorization retains the probabilistic rigor of AR models while enabling flexible dependency structures, bidirectional conditioning, and efficient groupwise decoding.

To realize A3 in practice, we design a two-stream attention architecture that supports arbitrary group orderings, and develop a progressive training strategy that adapts pretrained AR models to any-order prediction. Our framework naturally supports diverse inference strategies, including groupwise AR sampling and dynamic resampling, offering a tunable trade-off between generation speed and quality.

Through comprehensive experiments on question answering (Joshi et al., 2017), commonsense reasoning (Zellers et al., 2019; Sakaguchi et al., 2020; Sap et al., 2019; Bisk et al., 2020), and story infilling tasks (Mostafazadeh et al., 2016), we show that A3 achieves strong performance across diverse benchmarks while enabling flexible and efficient generation. Notably, A3 outperforms state-of-the-art diffusion-based models despite using substantially less training data, and demonstrates

promising scaling behavior with model size. These results suggest that A3 provides a new direction for bridging the gap between AR and parallel generation paradigms.

Our contributions can be summarized as follows:

- **Conceptually**, we propose A3, a novel language modeling framework that leverages both insights from AR modeling and parallel generation. With groupwise token partition, A3 successfully enables flexible generation at *any subset* in *any order*.
- **Practically**, we implement A3 for both training and inference phases. Building on a novel attention architecture that supports arbitrary group orderings, A3 can be trained progressively to adapt pretrained AR models to any-order prediction during inference.
- **Empirically**, we evaluate A3 across diverse reasoning and QA benchmarks to demonstrate its strong performance while enabling flexible and efficient generation, showing its great potential serving as next-generation of language modeling paradigm.

2 Any-order Any-subset Autoregressive Modeling

2.1 Formulation

In this section, we will demonstrate the formulation of our proposed Any-order Any-subset Autoregressive modeling (A3). The dominant paradigm for text generation is autoregressive (AR) modeling, where the joint probability of a sequence $x_{1:N}$ with length N is factorized in a fixed left-to-right order:

$$P(x_{1:N}) = \prod_{t=1}^{N} P(x_t|x_{<t}). \tag{1}$$

This formulation is simple, effective, and fundamental to most large language models (Touvron et al., 2023; Bai et al., 2023a; Radford et al., 2018). However, it introduces two key limitations. First, the left-to-right constraint forces the model to generate tokens sequentially, preventing it from leveraging bidirectional context during both training and inference. Second, decoding proceeds one token at a time, creating a bottleneck of inference efficiency. Together, these limitations restrict the model's ability to fully exploit contextual information and achieve faster generation.

Alternative paradigms such as masked diffusion models (Li et al., 2022; Gong et al., 2025; Gulrajani & Hashimoto, 2024) attempt to overcome the speed bottleneck by generating multiple tokens in parallel. Concretely, the index set $1, 2, \ldots, N$ is partitioned into two disjoint subsets: a unmasked index group G_1 and its masked complement G_2. The model then predicts the masked tokens x_{G_2} conditioned on the visible tokens x_{G_1}:

$$P(x_{G_2}|x_{G_1}) = \prod_{t \in G_2} P(x_t|x_{G_1}). \tag{2}$$

During inference, the model starts from a fully masked sequence (i.e., $G_2 = \emptyset$) and iteratively denoises it by resampling subsets of positions, filling in multiple tokens at each step. This procedure alleviates the inference-speed limitation of AR models by enabling parallel token generation. However, masked diffusion models suffer from two key drawbacks. First, since only a subset of tokens is predicted in each training step, the learning signal is partial and less informative, often yielding lower sample quality compared to AR models. Second, the training dynamics are unstable: performance depends heavily on carefully tuned noise schedules, masking strategies, and iteration counts, which complicates optimization and make it difficult to achieve consistent performance (Kim et al., 2025).

We now seek a middle ground: a model that preserves the probabilistic rigor of AR modeling while enabling flexible prediction orders and parallelism in decoding. This motivates **Any-order Any-subset Autoregressive** modeling (A3), which extends the AR framework to group-level prediction. Instead of using a fixed left-to-right order with unit-sized steps, A3 factorizes the joint probability by partitioning the token sequence into groups:

$$P(x_{1:N}) = \prod_{k=1}^{K} P(x_{G_k}|x_{g_{<k}}). \tag{3}$$

where $G_1, G_2, \ldots, G_K$ is a partition of the tokens and each group G_k may contain one or more tokens. Crucially, the ordering of groups is arbitrary, which enables flexible dependency structures.

By training the model on random groupings and permutations, we expose it to a wide variety of factorization orders, forcing it to learn robust conditional dependencies beyond simple left-to-right context. This resembles the permutation LM objective of XLNet (Yang et al., 2019) but at the group level, enabling richer structural modeling.

2.2 Discussion with Previous Paradigms

Comparison with Masked Diffusion Language Models. Masked diffusion language models (MDLMs) have recently emerged as an alternative to AR decoding, aiming to overcome the sequential bottleneck of one-token-at-a-time generation. By iteratively denoising partially masked sequences, MDLMs can update multiple tokens in parallel and exploit bidirectional context (Austin et al., 2021; Li et al., 2022; Gong et al., 2025; Nie et al., 2025). This flexibility enables controllable, any-order generation and supports tasks such as infilling and global rewriting. However, diffusion approaches face two major limitations. First, inference speed is constrained by the need for many iterative refinement steps, with generation efficiency depending critically on noise schedules and step counts (Gong et al., 2025). While large-scale efforts such as DiffuGPT and LLaDA demonstrate that diffusion LMs can match or surpass AR models in quality, they still require careful tuning and incur nontrivial decoding costs (Gong et al., 2025; Nie et al., 2025). Second, training stability is less favorable than AR: discrete diffusion objectives require complex forward–reverse processes and additional pretraining or adaptation, making optimization more resource-intensive and sensitive to hyperparameters (He et al., 2023).

In contrast, A3 preserves the probabilistic rigor and training simplicity of AR modeling while introducing flexible groupwise factorization. This allows parallel prediction of token subsets without relying on multi-step denoising schedules, thereby offering both efficiency and stability.

Comparison with AR Multi-Token Prediction. Another line of work seeks to improve AR efficiency by enabling the model to predict multiple future tokens per step (Gloeckle et al., 2024; Kou et al., 2024). Multi-token objectives and speculative decoding significantly reduce inference latency: for instance, predicting four tokens at once can yield up to threefold speedups while preserving or even improving generation quality, particularly in reasoning and code generation tasks (Gloeckle et al., 2024). These methods retain the training stability of standard AR, since the additional objectives can be implemented as auxiliary losses with negligible computational overhead. However, multi-token prediction remains bound to a fixed left-to-right ordering, limiting its modeling flexibility. The approach accelerates sequential decoding but does not enable infilling, bidirectional conditioning, or arbitrary ordering of token generation. Moreover, these methods depend on multiple linear heads, which limits maximum parallelism. Therefore, they cannot reach the same theoretical decoding parallelism as diffusion-style iterative refinement. A3, by contrast, can reuse diffusion-like scheduling using a single AR model, enabling it to achieve a higher theoretical upper bound on parallel decoding efficiency while remaining within an AR framework.

A3 generalizes beyond this paradigm by relaxing the strict AR factorization. Through arbitrary group partitions and orderings, A3 supports both sequential and parallel decoding strategies, combining the efficiency gains of multi-token prediction with greater structural flexibility. For more discussions with related work, refer to Appendix B.

3 Implementations of Training and Flexible Inference

In this section, we describe the implementation of A3. We begin with the architectural design of A3, where we use a two-stream attention mechanism to enable predictions in arbitrary orders. Next, we present our efficient continuous pretraining strategy, which progressively adapts the model from standard AR prediction to group-based prediction. Finally, we introduce the flexible inference strategy of A3, which leverages its general formulation to support diverse decoding modes.

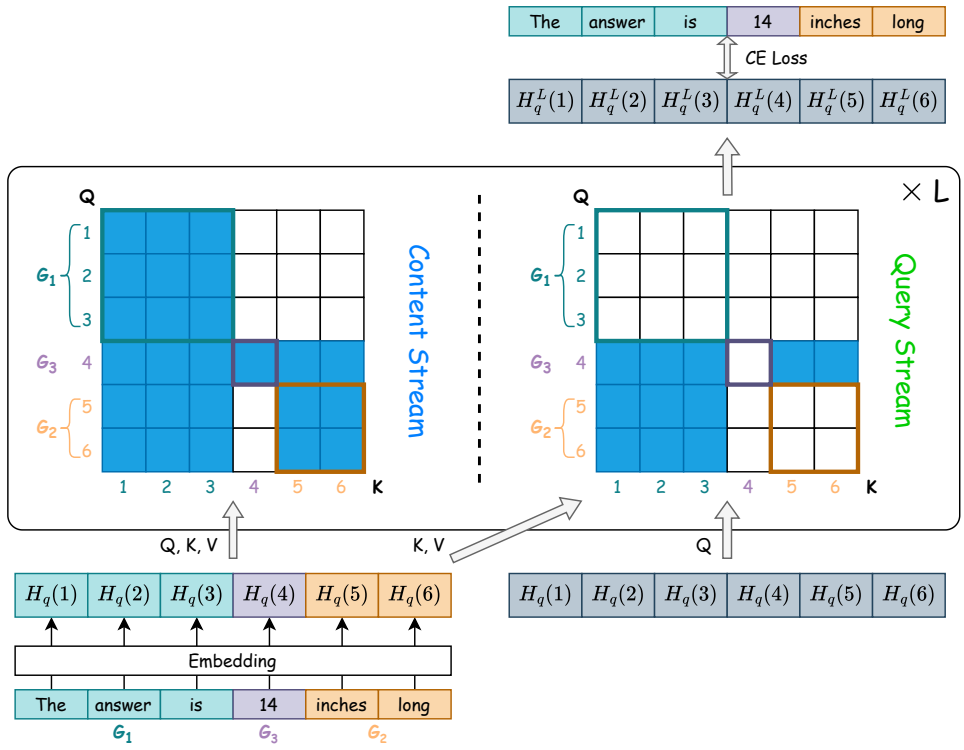

Figure 1: Architecture of the A3 model. Blue entries in the attention mask denote 0, and white entries denote $-\infty$. The model employs a two-stream attention module with distinct causal masks. The content stream encodes contextual information and attends to tokens within its own group as well as all preceding groups. The query stream encodes positional conditions and attends only to tokens in preceding groups. The final cross-entropy loss is computed between the input context and the query stream's output. For illustration, we provide an example grouping with $G_1 = \{1, 2, 3\}, G_2 = \{5, 6\}, G_3 = \{4\}$, showing how the forward process and causal masks are applied.

3.1 Architecture Design with Two-stream Attention

Limitations of Current Architecture. The decoder-only Transformer has become the backbone of modern large language models due to its simplicity and effectiveness in next-token prediction. It operates with a single stream of hidden states, where information flow is regulated by a causal attention mask. This mask ensures that the representation of the k-th token can only attend to the first k tokens, thereby enforcing the AR constraint required for language modeling. While well-suited for the standard left-to-right objective, this design assumes a fixed generation order: given the first k tokens, the model is trained to treat the $(k+1)$-th position as the unique next target. Such rigidity makes it incompatible with any-order prediction, where the next position to be generated need not follow the sequential index.

The encoder-only Transformer, widely used in masked language modeling, represents the opposite design. Rather than causal masking, it processes the full sequence bidirectionally, with missing information represented by mask tokens. Through position embeddings on these masks, the model identifies which locations are to be predicted, allowing arbitrary subsets of tokens to be reconstructed simultaneously. However, this formulation limits dependency modeling: masked positions are predicted in parallel and conditioned only on observed context in a single pass. Without recursive, multi-layered dependencies across tokens, the encoder-only approach struggles to match the generative fidelity of AR models.

Two-stream Attention Design for Any-order Prediction. To combine the flexibility of encoder-style masking with the dependency modeling strength of autoregression, A3 extend the two-stream attention mechanism proposed by XLNet (Yang et al., 2019). The model maintains two parallel representations for each position: a **content stream**, which encodes semantic and contextual in-

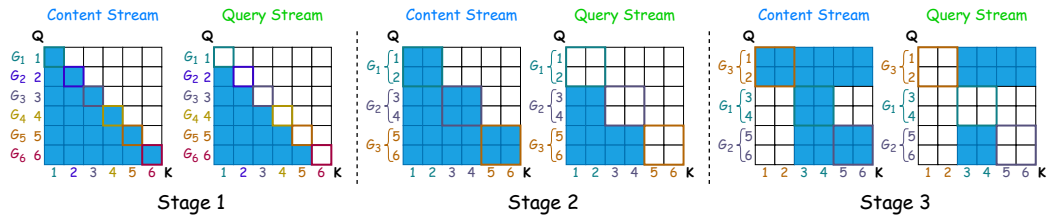

Figure 2: Causal masks for content stream and query stream in different stages. Blue for 0 and white for $-\infty$. Stage 1: **AR initialization** to reproduce AR factorization. Stage 2: **Group expansion** by allowing groups of size greater than one. Stage 3: **Order permutation** with introducing any-order prediction.

formation from the observed tokens, and a **query stream**, which provides position-aware signals to drive prediction of the next group. This separation allows A3 to retain the recursive structure of autoregression while relaxing the generation order constraint of decoder-only models. Figure 1 illustrates the pipeline of the forward process.

Formally, let $X = (x_1, \ldots, x_N)$ denote the sequence, partitioned into groups $\{G_1, \ldots, G_K\}$. In the **content stream**, the input consists of the observed tokens, embedded and passed through Transformer layers with a designed causal mask. This mask ensures that a token at group k can attend to all tokens in groups $\leq k$, i.e., both its own group and all groups before it. Thus, the content stream at group k aggregates all contextual evidence available up to that point. For group G_k, the hidden states in the content stream at layer l are computed as:

$$H_c^{(l)}(i) = \text{Attn}\Big(Q = H_c^{(l-1)}(i),\ K = H_c^{(l-1)}(\leq G_k),\ V = H_c^{(l-1)}(\leq G_k)\Big)\,. \tag{4}$$

In the **query stream**, the input is a shared learnable query vector injected at every position. The key and value matrices are tied to those of the content stream, while the queries are separate. With an appropriately designed causal mask, each query vector at group k can only attend to content tokens in groups $< k$, not including its own group. This forces the query representation to serve as the position-aware predictor for the tokens in group k, relying only on prior context rather than future information. Conceptually, the query stream specifies *where* to predict (positional conditioning), while the content stream provides *what* to predict (contextual grounding). The hidden states in the query stream at layer l for group G_k are:

$$H_q^{(l)}(i) = \text{Attn}\Big(Q = H_q^{(l-1)}(i),\ K = H_c^{(l-1)}(< G_k),\ V = H_c^{(l-1)}(< G_k)\Big)\,, \tag{5}$$

where the initialization $H_q^{(l-1)}(i) = w$ is a learnable query vector shared across positions, and the causal mask ensures the query stream at group k can only access content states from strictly earlier groups.

Finally, the predictive distribution for token $x_i \in G_k$ is parameterized by:

$$p(x_i \mid X_{<G_k}) = \text{Softmax}\Big(W \cdot H_q^{(L)}(i)\Big)\,, \tag{6}$$

where L is the final layer and W projects the query hidden state to the vocabulary.

3.2 Multi-Stage Training with Progressive Token Grouping

Building on the connection between standard AR and A3, we design a progressive adaptation strategy that smoothly transitions from left-to-right generation to fully flexible any-order prediction. To leverage the stability and strong initialization of existing AR models, we begin training A3 from a pretrained AR checkpoint and gradually relax its constraints through three stages:

- **Stage 1: AR Initialization.** We align A3 with conventional AR training by setting the two-stream causal masks to exactly reproduce left-to-right factorization (Figure 2 Stage 1). Formally, the sequence $x_{1:N}$ is partitioned into singleton groups:

$$G_1 = \{1\},\ G_2 = \{2\},\ \ldots,\ G_N = \{N\}. \tag{7}$$

This ensures that $P(x_{1:N}) = \prod_{t=1}^{N} P(x_t \mid x_{<t})$, identical to standard AR, providing a stable initialization.

- **Stage 2: Group Expansion.** We expand beyond token-level prediction by allowing groups of size greater than one (Figure 2 Stage 2). Concretely, the sequence is partitioned into contiguous segments of fixed size $s > 1$, e.g.,

$$G_1 = \{1, \ldots, s\}, \quad G_2 = \{s+1, \ldots, 2s\}, \quad \ldots \tag{8}$$

with s gradually increased from 1 to 4. This teaches the model to predict multiple tokens jointly within each group while still maintaining AR dependencies across groups.

- **Stage 3: Order Permutation.** We introduce any-order prediction within groups (Figure 2 Stage 3). The group structure $G_1, G_2, \ldots, G_K$ remains sequential, but the token indices assigned to each group are drawn from a random permutation of $\{1, \ldots, N\}$. For example, if π is a random permutation of indices, then

$$G_1 = \{\pi(1), \ldots, \pi(s)\}, \quad G_2 = \{\pi(s+1), \ldots, \pi(2s)\}, \quad \ldots \tag{9}$$

The model therefore learns to predict tokens in arbitrary subsets, while still preserving a group-to-group AR factorization:

$$P(x_{1:N}) = \prod_{k=1}^{K} P(x_{G_k} \mid x_{G_{<k}}). \tag{10}$$

This exposes the model to diverse intra-group orderings and enables it to generalize to arbitrary prediction targets at inference.

By the end of this curriculum, the model is able to predict arbitrary subsets of tokens as coherent groups while preserving the recursive dependency structure of AR. Importantly, at every stage of training, each token in the sequence belongs to exactly one group, so all tokens are always predicted, maximizing the learning signal and computational efficiency.

3.3 Flexible Inference via Groupwise Decoding

Building on the A3 formulation, we propose flexible inference strategies that extend beyond conventional AR decoding. The first decoding method we introduce is **groupwise AR sampling**, which generalizes standard left-to-right generation by sampling groups of tokens sequentially rather than strictly one-by-one. Formally, let the token positions of a sequence be partitioned into K groups $\mathcal{G} = \{G_1, G_2, \ldots, G_K\}$, where $G_k \subseteq \{1, \ldots, n\}$ and $\bigcup_{k=1}^{K} G_k = \{1, \ldots, n\}$. Given a prompt covering groups $G_1, \ldots, G_{k_0}$, the model generates subsequent groups by conditioning on all preceding groups:

$$p_\theta(x_{G_{k_0+1}}, \ldots, x_{G_K} \mid x_{G_{\leq k_0}}) = \prod_{k=k_0+1}^{K} p_\theta(x_{G_k} \mid x_{G_{<k}}). \tag{11}$$

Here, x_{G_k} denotes the tokens within group G_k, and $x_{G_{<k}}$ the tokens of all earlier groups. This reduces to the classical AR factorization when $|G_k| = 1$ for all k, but naturally generalizes to larger groups. The procedure is summarized in Algorithm 1. Concretely, several grouping strategies can be applied:

1. **Token-wise grouping.** Each token is treated as its own group, i.e., $G_k = \{k\}$. The decoding reduces to the standard left-to-right AR generation:

$$p_\theta(x_1, \ldots, x_n) = \prod_{t=1}^{n} p_\theta(x_t \mid x_{<t}). \tag{12}$$

2. **Fixed-size grouping.** Tokens are partitioned into groups of size s, e.g., $G_k = \{(k-1)s + 1, \ldots, ks\}$ for $s \in \{2, 4\}$. In this case, the model predicts s tokens jointly per step and accelerates decoding.

3. **Task-specific grouping.** For infilling tasks we allow groups to be arbitrary index subsets and then assign group ids so that groups containing masked positions are decoded after groups used as context. Concretely, let the sequence be partitioned into left, middle and right index sets L, M, R (so $\{1, \ldots, n\} = L \cup M \cup R$). We choose an index k_0 such that every group G_k satisfying $G_k \cap M = \emptyset$ has $k \leq k_0$, while every group that contains any masked position satisfies $k > k_0$ (groups need not be contiguous and a single group may contain tokens from both L and R). Under this design, all context groups (those covering L and R) appear before the masked groups, and the model performs:

$$p_\theta(x_M \mid x_{G_{\leq k_0}}) = \prod_{k=k_0+1}^{K} p_\theta\big(x_{G_k} \mid x_{G_{<k}}\big), \tag{13}$$

which realizes AR dependencies inside each masked group while conditioning on both left and right contexts. This flexible assignment enables infilling where context groups are formed from arbitrary subsets of $L \cup R$, and masked spans are predicted group-by-group. This capability distinguishes A3 from conventional AR models, which cannot directly condition on future context during generation.

Dynamic Resampling Inference. Beyond fixed grouping, A3 also supports a more adaptive inference procedure inspired by iterative refinement (Li et al., 2022; Chen et al., 2024a). Here, the grouping $\mathcal{G}$ is not fixed. At each step, the model evaluates all unfinished positions simultaneously, conditioned on the completed tokens. Formally, suppose $U_t \subseteq \{1, \ldots, n\}$ is the set of unfinished (blank) positions at iteration t, and F_t is its complement of finished positions. The model computes predictive distributions

$$p_\theta(x_i \mid x_{F_t}), \quad \forall i \in U_t. \tag{14}$$

Based on these distributions, we then select a subset $S_t \subseteq U_t$ to be committed at this step, according to some criterion such as maximum confidence, lowest entropy (Kim et al., 2025), or simply random sampling. Once S_t is chosen, the tokens at S_t are sampled and added to the finished set:

$$F_{t+1} = F_t \cup S_t, \quad U_{t+1} = U_t \setminus S_t. \tag{15}$$

This process repeats until $U_T = \emptyset$, at which point the sequence is fully generated. The procedure is summarized in Algorithm 2. The advantage of this dynamic resampling strategy is twofold. First, it allows the model to adaptively choose the granularity of generation based on prediction confidence, committing to easy tokens early while deferring more uncertain positions until later. Second, unlike diffusion-style denoising which follows a pre-specified noise schedule, A3 inference directly uses the conditional distributions defined by the AR factorization, ensuring consistency between training and inference.

These inference strategies highlight a trade-off between efficiency and flexibility. Fixed-group sampling is fast but less adaptive, as performance depends on group alignment with text structure. Dynamic resampling is slower since all unfinished positions are reevaluated at each step, but it yields greater accuracy by adapting token commitment to model confidence. We will compare these strategies in the next section on real-world tasks.

4 Experiments

4.1 Setup

Training Setup. We initialize our models from the LLaMA series, including LLaMA-3.1-8B, LLaMA-3.2-3B, and LLaMA-3.2-1B (Dubey et al., 2024). For training data, we construct a mixture of the FineWeb dataset (Penedo et al., 2024) and the SlimPajama dataset (Soboleva et al., 2023), following prior work on DLMs and AR models. From this mixture, we sample 2B tokens and apply sequence packing with a maximum context length of 2048. All models are trained with full-parameter fine-tuning in `bf16`. In the progressive adaptation recipe, the first two training stages are trained for one epoch over 20% of the dataset, while the final stage is trained for one epoch over the full dataset. Additional training details are provided in Appendix A.

Evaluation Setup. We adopt the evaluation protocol of Gong et al. (2025) to compare our models against both diffusion and AR baselines. For reading comprehension, we evaluate on TriviaQA

Algorithm 1 Groupwise AR Sampling

Require: Prompt tokens $x_{1:m}$, grouping strategy $\mathcal{G} = \{G_1, G_2, \ldots, G_K\}$, model f_θ
Ensure: Generated sequence $\hat{x}_{1:n}$
1: Initialize $\hat{x}_{1:m} \leftarrow x_{1:m}$
2: Find the last group index k_0 in the prompt
3: **for** $k = k_0 + 1$ to K **do**
4: Compute context representation $h \leftarrow f_\theta(\hat{x}_{G_{<k}})$
5: Sample tokens $\hat{x}_{G_k} \sim p_\theta(\cdot \mid h)$
6: **end for**
7: **return** Completed sequence $\hat{x}_{1:n}$

Algorithm 2 Dynamic Resampling

Require: : Prompt tokens, model f_θ, criterion
1: Initialize F_0 with prompt tokens, U_0 with blank positions
2: **while** $U_t \neq \emptyset$ **do**
3: **for** each $i \in U_t$ **do**
4: Compute $p_\theta(x_i \mid x_{F_t})$
5: **end for**
6: Select subset $S_t \subseteq U_t$ based on criterion
7: **for** each $i \in S_t$ **do**
8: Sample $\hat{x}_i \sim p_\theta(x_i \mid x_{F_t})$
9: **end for**
10: Update $F_{t+1} \leftarrow F_t \cup S_t$, $U_{t+1} \leftarrow U_t \setminus S_t$
11: **end while**
12: **return** Completed sequence $x_{1:n}$

(Joshi et al., 2017) using exact match accuracy. For commonsense reasoning, we consider HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2020), SIQA (Sap et al., 2019), and PIQA (Bisk et al., 2020), all assessed by multiple-choice accuracy. For story infilling, we use ROCStories (Mostafazadeh et al., 2016) and report ROUGE scores (Lin, 2004). We compare against two categories of baselines: (a) the base AR model LLaMA-3.1-8B, and (b) recent diffusion language models of varying sizes, including Plaid-1B (Gulrajani & Hashimoto, 2024), Dream-7B (Ye et al., 2025), and DiffuLlama-7B (Gong et al., 2025).

4.2 Main Results

The results in Table 1 show that A3 consistently outperforms diffusion-based models across QA, commonsense reasoning, and infilling tasks. For example, A3-8B achieves 19.4 accuracy on TriviaQA and 78.1 on PIQA, surpassing all the diffusion baselines, while also attaining competitive ROUGE scores for story infilling. In fact, A3 also achieves better latency number on the infilling task (0.15 s/sample for A3 v.s. 0.17 s/sample for DiffuLlama and 0.21 s/sample for Llama-3.1-8B). These gains are particularly noteworthy given that A3 is trained on only 2B tokens, whereas DiffuLlama is trained on 65B. Although A3 still underperforms the AR baseline, this gap is likely attributable to limited training data; with larger-scale pretraining, we expect A3 to close the difference further.

Importantly, A3 demonstrates clear scaling behavior: performance improves steadily from 1B to 3B to 8B parameters, indicating that the method benefits from larger models in the same way as conventional AR training. Overall, these results confirm that A3 offers a favorable balance between AR and diffusion paradigms, combining strong reasoning accuracy with flexible generation, and holds promise for further improvements under larger-scale training.

4.3 Ablation Study

Inference Strategies. To better understand the trade-offs between the two proposed inference strategies in Section 3.3, we conducted unconditional generation experiments under the A3 decoding framework. For groupwise AR sampling, we vary the group size from 1 to 4. For dynamic resampling, we vary the group size from 1 to 16 and implemented two selection criteria: (1) Confidence-based: selecting positions with highest maximum softmax probability. (2) Entropy-based: selecting positions with minimum output entropy. For each sequence, we sample with temperature of 1.5 and top-p of 0.95. Figure 3 reports the log of perplexity measured by Llama-3.1-8B and the average decoding time for one sequence.

We observe that dynamic resampling methods consistently achieve lower perplexity than groupwise AR sampling, indicating that they produce higher-quality generations. The confidence-based and entropy-based criteria yield very similar performance, with confidence being slightly better at smaller group sizes. However, all strategies show a trend of increasing perplexity as group size

Table 1: Comprehensive evaluation of different language models. There are 4 types of these models: AR for autoregressive, DD for discrete diffusion, CD for continuous diffusion and A3 for our proposed model. For the infilling task, we use ROUGE-1/2/L score; for other tasks, we use the accuracy (%) metric. * refers to the results reported in DiffuLlama (Gong et al., 2025).

Model	Size	Type	QA	CommonSense Reasoning				Infilling
			TriQA	HSwag	Wino.	SIQA	PIQA	ROCStories
Llama-3.1	8B	AR	52.1	76.0	63.9	46.7	80.3	11.7/2.3/10.5
Plaid*	1B	CD	1.2	39.3	51.3	32.3	54.5	12.1/1.1/11.2
Dream	7B	DD	18.3	26.9	51.8	36.6	55.8	11.7/2.3/10.5
DiffuLlama*	7B	DD	18.5	**58.7**	56.4	43.2	63.3	**23.3/5.5/21.2**
A3	1B	A3	10.2	40.2	52.8	35.1	64.7	11.8/1.7/11.1
	3B	A3	15.9	49.6	54.3	38.9	70.1	11.3/2.3/10.2
	8B	A3	**19.4**	58.4	**60.2**	**45.2**	**78.1**	19.2/4.6/18.6

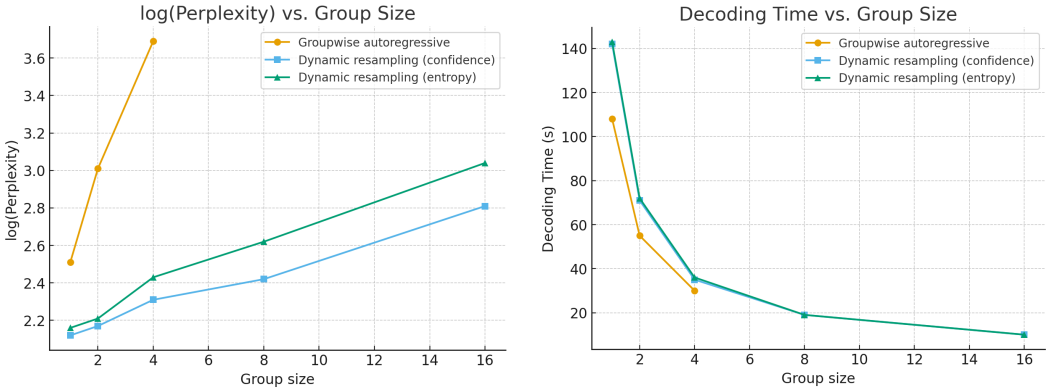

Figure 3: Unconditional generation log(perplexity) and speed using A3-8B. The perplexity is measured by Llama-3.1-8B and we compare several decoding strategies. Dynamic resampling will cost more time but have lower perplexity.

grows, reflecting the trade-off between decoding granularity and modeling accuracy. We can also see that decoding time decreases sharply with larger group sizes. Groupwise AR sampling is fastest at the same group size because it only generates the designated group per step, while dynamic resampling requires evaluating all unfinished tokens at each iteration, making it slower. However, as group size increases, dynamic resampling speeds up considerably, nearly matching the efficiency of groupwise sampling at large group sizes.

Overall, these results demonstrate a speed–accuracy trade-off. Groupwise AR sampling is faster but less accurate, while dynamic resampling achieves better perplexity at the cost of slower decoding. Importantly, A3 provides the flexibility to choose between these strategies depending on the requirements of the application, making it more flexible than conventional AR or diffusion-based methods.

Curriculum schedule. A3 introduces a different causal mask and attention flow from a standard AR transformer, and the model must progressively adapt from strict left-to-right prediction to multi-token and eventually arbitrary-order factorization. To assess the sensitivity of the schedule, we train two variants on 0.5B tokens: 1. original curriculum, and 2. skipping stage 1 and 2 (directly training on stage 3: order permutations). Results are shown in Table 2. Skipping the early stages consistently hurts performance by 4–6 points on several benchmarks, which proves the importance of such adaptation stage. An adaptive schedule, e.g., based on training loss, may further improve robustness. We plan to investigate this direction in the future work.

Performance with more data. Since the training budget for A3 is much less than the baseline (2B for A3, 60B for DiffuLlama and 15T for Llama-3.1-8B), in order to isolate the architecture effect on the worse performance than the AR baseline, we track how A3 improves under increasing post-

Table 2: Performance with different training curriculum schedule. We evaulate two variants trained on 0.5B tokens: 1. original curriculum and, 2. skipping stage 1 and 2 (directly training on stage 3: order permutations).

Schedule	QA	CommonSense Reasoning				Infilling
	TriQA	HSwag	Wino.	SIQA	PIQA	ROCStories
Original	**15.6**	**49.3**	**56.7**	**39.6**	**69.4**	**13.2/2.3/12.6**
Skipping Stage 1 & 2	11.3	44.2	54.1	37.3	64.2	13.1/2.2/12.4

Table 3: Performance of A3 with different training data on TriviaQA and perplexity measured by Llama-3.1-8B.

Model	TriviaQA	log(Perplexity)
A3 (1.5B tokens)	16.2	2.9
A3 (2B tokens)	19.4	2.5
A3 (2.5B tokens)	22.5	2.3
AR (15T)	52.1	0.8

Table 4: Model loss of A3 across context lengths, which is stably small.

Length	Model loss
256	3.54
512	3.51
1024	3.34
2048	3.23

training data. We use 1.5B, 2B (default) and 2.5B tokens to train A3. The results are shown in Table 3. Performance increases steadily with more data. This confirms that A3 benefits strongly from data scale and that the gap to fully-trained AR models is due to training budget, not a limitation of the A3 architecture.

Robustness on context length. In order to investigate whether A3 is robust across different context lengths, we input contexts with length of 512, 1024 and 2048 to A3 and calculate the loss. The results are shown in Table 4. The model loss keeps stable within the training length, indicating the robustness of A3 across different context lengths.

5 Conclusion

We have presented Any-order Any-subset Autoregressive modeling (A3), a novel framework that generalizes traditional autoregressive factorization to enable flexible, groupwise generation of tokens in arbitrary orders. By combining a two-stream attention architecture with a progressive training strategy, A3 achieves the dual goals of generation flexibility and modeling stability. Our approach supports a wide range of decoding strategies, including groupwise autoregressive sampling and dynamic resampling, offering a tunable trade-off between speed and accuracy. Through comprehensive experiments, we demonstrate that A3 outperforms diffusion-based models in reasoning, question answering, and infilling tasks. These results highlight A3's ability to balance efficiency, flexibility, and quality, making it a promising direction for future sequence modeling. In the future, we plan to explore scaling A3 to larger models and datasets, as well as applying it to more challenging tasks such as long-context reasoning.

Ethics Statement

This work complies with the ICLR Code of Ethics. While our methods are general, they may be applied in contexts with societal implications, including risks related to bias, fairness, and privacy. We encourage responsible use and declare no conflicts of interest.

Reproducibility Statement

We provide detailed descriptions of our methodology, datasets, model configurations, and evaluation metrics in both the main text and the Appendix. Codes will be released upon acceptance.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021. 3, 15

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023a. 1, 2

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023b. 1, 17, 18

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *AAAI*, 2020. 1, 8

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. *arXiv preprint arXiv:2401.10774*, 2024. 16

Pinzhen Chen, Zhicheng Guo, Barry Haddow, and Kenneth Heafield. Iterative translation refinement with large language models. In *EAMT*, 2024a. 7

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. LongloRA: Efficient fine-tuning of long-context large language models. In *ICLR*, 2024b. 15

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022. 15

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024. 7

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024. 15

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. In *ICML*, 2024. 3

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022. 15, 16

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, et al. Scaling diffusion language models via adaptation from autoregressive models. In *ICLR*, 2025. 1, 2, 3, 7, 8, 9, 15

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*, 2017. 15

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. *Advances in neural information processing systems*, 32, 2019. 16

Ishaan Gulrajani and Tatsunori B Hashimoto. Likelihood-based diffusion language models. In *NeurIPS*, 2024. 1, 2, 8, 15

Gabe Guo and Stefano Ermon. Reviving any-subset autoregressive models with principled parallel sampling and speculative decoding. *arXiv preprint arXiv:2504.20456*, 2025. 17

Junliang Guo, Xu Tan, Linli Xu, Tao Qin, Enhong Chen, and Tie-Yan Liu. Fine-tuning by curriculum learning for non-autoregressive neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7839–7846, 2020. 16

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020. 15

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. *arXiv preprint arXiv:2210.17432*, 2022. 15, 16

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuan-Jing Huang, and Xipeng Qiu. Diffusionbert: Improving generative masked language models with diffusion models. In *ACL*, 2023. 3

Yosuke Higuchi, Shinji Watanabe, Nanxin Chen, Tetsuji Ogawa, and Tetsunori Kobayashi. Mask ctc: Non-autoregressive end-to-end asr with ctc and mask predict. *arXiv preprint arXiv:2005.08700*, 2020. 16

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 15

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *NeurIPS*, 2021. 15

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017. 1, 8

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. Continual pre-training of language models. In *ICLR*, 2023. 15

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *ICML*, 2025. 1, 2, 7

Siqi Kou, Lanxiang Hu, Zhezhi He, Zhijie Deng, and Hao Zhang. Cllms: consistency large language models. In *ICML*, 2024. 3

Yury Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *NeurIPS*, 2024. 1

Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *arXiv preprint arXiv:1802.06901*, 2018. 16

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *ICML*, 2023. 16

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. In *NeurIPS*, 2022. 1, 2, 3, 7, 15

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024. 16

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004. 8

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023. 15, 16

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *NAACL*, 2016. 1, 8

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, JUN ZHOU, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025. 3

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024. 15

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. In *NeurIPS*, 2024. 7

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. 1, 2

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019. 17

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and jiuVolodymyr Kuleshov. Simple and effective masked diffusion language models. In *NeurIPS*, 2024. 15

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *AAAI*, 2020. 1, 8

Mohammad Samragh, Iman Mirzadeh, Keivan Alizadeh Vahid, Fartash Faghri, Minsik Cho, Moin Nabi, Devang Naik, and Mehrdad Farajtabar. Scaling smart: Accelerating large language model pre-training with small model initialization. *arXiv preprint arXiv:2409.12903*, 2024. 15

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. In *EMNLP*, 2019. 1, 8

Tianxiao Shen, Hao Peng, Ruoqi Shen, Yao Fu, Zaid Harchaoui, and Yejin Choi. Film: Fill-in language models for any-order generation. *arXiv preprint arXiv:2310.09930*, 2023. 16

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama, 2023. URL https://huggingface.co/datasets/cerebras/SlimPajama-627B. 7

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 15

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 15

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31, 2018. 16

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pp. 5976–5985. PMLR, 2019. 16

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 1, 2

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023. 15

Yiheng Xu, Hongjin SU, Chen Xing, Boyu Mi, Qian Liu, Weijia Shi, Binyuan Hui, Fan Zhou, Yitao Liu, Tianbao Xie, Zhoujun Cheng, Siheng Zhao, Lingpeng Kong, Bailin Wang, Caiming Xiong, and Tao Yu. Lemur: Harmonizing natural language and code for language agents. In *ICLR*, 2024. 15

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019. 3, 4, 16

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025. 8

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL*, 2019. 1, 8

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. In *COLM*, 2024. 15

A Training Hyperparameters

We list the hyperparameters in the training stage in Table 5

Table 5: The hyperparameter list

Hyperparameter	Value
Training	
Batch Size	64
Epoch	[0.2, 0.2, 1]
Optimizer	AdamW
LR	2e-5
Betas	(0.9, 0.999)
Weight Decay	0.01
LR Schedule	WarmupLR
Warmup Iters	[50, 50, 50]
Max Sequence Length	2048
Sampling (Section 4.3)	
Top-p	0.95
Temperature	1.5

B Additional Related Work

Continue Pre-training. Pretraining large language models has been proven to be complex and computationally expensive (Samragh et al., 2024). Consequently, continued pre-training has been proposed as an effective method to adapt existing large language models to specific domains (Ke et al., 2023; Gururangan et al., 2020) or to endow them with new capabilities, such as handling longer contexts (Chen et al., 2024b; Fu et al., 2024; Xiong et al., 2023) or code generation (Xu et al., 2024). Notably, certain continued pre-training efforts, such as those in scaling diffusion language models (Gong et al., 2025), have transcended autoregressive (AR) language modeling by converting large language models into diffusion-based architectures, thereby enabling parallel token generation. In contrast, our work retains autoregressive language modeling but innovatively incorporates a two-stream architecture and semi-autoregressive decoding to similarly support parallel prediction of multiple tokens, achieving significant reductions in inference latency compared to both autoregressive and diffusion-based baselines.

Diffusion Language Model. Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) have demonstrated remarkable capabilities in image generation. Consequently, a series of works have sought to extend diffusion models to text generation, which can be roughly divided into the continuous diffusion model and the discrete diffusion model. One straightforward approach involves embedding text data into a continuous space and directly applying diffusion models (Li et al., 2022; Gong et al., 2022; Han et al., 2022; Dieleman et al., 2022). However, the scalability of continuous diffusion methods remains a challenge, as they require substantially greater computational cost compared to AR models to achieve equivalent performance(Gulrajani & Hashimoto, 2024). To better accommodate the discrete nature of text, an alternative paradigm replaces continuous diffusion with a discrete process, introducing an absorbing [MASK] state as noise (Austin et al., 2021; Hoogeboom et al., 2021; Zheng et al., 2024; Sahoo et al., 2024). Lou et al. (2023) demonstrated that masked diffusion models (MDMs) achieve perplexity comparable to or even surpassing that of AR models at the GPT-2 scale. Ou et al. (2024) established foundational theoretical results, affirming the feasibility of MDMs. In comparison to MDMs, our method similarly enables parallel prediction of groups of tokens and leverages bidirectional context. However, by retaining autoregressive modeling, our approach utilizes every token during training, thereby facilitating faster convergence relative to MDMs.

Non-autoregressive Generation. Non-autoregressive (NAR) generation (Gu et al., 2017) accelerates inference by producing target tokens in parallel, eliminating the dependency on previously generated tokens inherent in traditional AR models. This approach substantially improves genera-

tion speed, but potentially at the cost of some accuracy. Conventional NAR models have focused mainly on machine translation and automatic speech recognition, emphasizing the trade-off between speed and quality (Higuchi et al., 2020; Lee et al., 2018). For instance, Guo et al. (2020) employed sequence-to-sequence labels in translation tasks, training NAR models via a curriculum learning strategy on attention masks. However, early NAR methods in text generation remain constrained by issues of global coherence and diversity. In recent years, insertion-based models(Stern et al., 2019; Gu et al., 2019) and diffusion language models (Gong et al., 2022; Lou et al., 2023; Shen et al., 2023) have progressively addressed these limitations. For example, SSD-LM (Han et al., 2022) leverages a diffusion-based language model to generate text blocks in a semi-autoregressive manner, enabling local bidirectional context updates. Compared to diffusion language models, our method similarly achieves parallel multi-token prediction and bidirectional context modeling, while benefiting from faster training convergence due to the absence of masked tokens during training.

Multi-token Prediction. Multi-Token Prediction (MTP) seeks to move beyond strictly stepwise next-token prediction by enabling models to forecast multiple future tokens per autoregressive step, improving inference efficiency with minimal quality degradation. Some methods achieve MTP by modifying the Transformer architecture. For instance, Stern et al. (2018) demonstrated through blockwise decoding that multi-token acceptance is feasible when supported by verification mechanisms. Medusa (Cai et al., 2024) add parallel heads to predict multi-step continuations with lightweight acceptance rules. Other approaches realize MTP via multi-model collaboration. Speculative decoding (Leviathan et al., 2023) uses a draft model to propose multi-token candidates that the main model verifies, while EAGLE-like validators (Li et al., 2024) further strengthen multi-token verification. Our method similarly modifies the Transformer structure, but innovatively leverages a two-stream architecture and semi-autoregressive decoding to achieve MTP, significantly reducing generation latency while attaining higher accuracy than baselines.

Permutation Network. Previous permutation-style networks like XLNet (Yang et al., 2019) models the text sequence in a bidirectional AR style, achieving better performance on language understanding tasks. However, they were all about boosting contextual understanding like hink QA and text classification. A3 repurposes these tools to realize the ability that traditional AR models never have: generate text in any order, or even just specific subsets of it. And it does this while hanging onto AR's biggest strengths–solid probabilistic rigor and stable training, which diffusion models have to sacrifice. Besides, A3 is the first to take permutation-style AR and scale it up to 7B+ parameter models for generative tasks. The earlier ideas behind permutation LMs (like the core concept in XLNet) never got past single-token prediction. It's A3's group-based curriculum and decoding strategies that make large-scale, parallel, and flexible generation actually work.

C The Use of Large Language Models (LLMs)

In this work, LLMs are primarily employed for polishing the language of the manuscript to ensure grammatical correctness and coherence. Importantly, all conceptual development, experimental design, and result interpretation are conducted independently by the authors. The use of LLMs is strictly limited to auxiliary tasks, ensuring that the scientific contributions of this paper remain entirely unaffected by such tools.

D More analysis experiments

Comparison with speculative decoding. We additionally conduct generation experiments as in Figure 3. We use speculative decoding (Leviathan et al., 2023) with Llama-3.2-1B as the draft model and Llama-3.1-8B as the target model, matching A3's group size (max 4). Perplexity is measured by Llama-3.1-70B. Results are shown in Table 6. Speculative decoding achieves lower perplexity but at higher wall-clock cost, while A3 provides competitive quality with faster decoding in this setting.

Importantly, semi-AR methods such as speculative decoding and multi-token prediction are decoding-level accelerators: they maintain the same left-to-right AR factorization and improve efficiency through draft-model proposals or multi-step predictions. In contrast, A3 changes the model factorization itself by enabling groupwise and permutation-based prediction. This makes A3 **or-**

Table 6: Comparison between speculative decoding and A3 on generating perplexity and time.

	log(perplexity)	Time
Speculative decoding	**1.9**	1.2×
A3	2.1	**1**×

thogonal to speculative/MTP: these accelerations can also be applied on top of A3's factorization in principle.

Comparison with recent any-subset AR. A recent any-subset AR work ASSD (Guo & Ermon, 2025) starts to provide provable joint-distribution correct parallel sampling by speculative decoding, which is not used in our A3 decoding process. The motivation for ASSD using correct-by-construction decoding is that, they model the sequence one token by one token using absorning state DTMC and assume

$$\sum_{i\in[m,N)} \log p(x_{\sigma(i)}|\mathbf{x}_{\sigma(<m)}) \neq \log p(\mathbf{x}_{\sigma(\geq m)}|\mathbf{x}_{\sigma(<m)}). \tag{16}$$

Therefore, when they sample a new group $\sigma(m),\ldots,\sigma(N-1)$, they need to use rejection sampling to get the right distribution for the new group. However, A3 directly models the sequence group by group. Therefore, the sampling results from $P(x_{G_t} \mid x_{\cup_{j<t}G_j})$ in each step faithfully represent the true distribution.

We compare A3's dynamical resampling with confidence and ASSD sampling method in unconditional generation as the same setting in Figure 3 using a group size of 4. We show the results in Table 7. With comparative results on generation quality, ASSD costs 2.4× time due to additional computation for resampling. This proves the high quality and high efficiency of A3's dynamic resampling method.

Practical speed–quality trade-off comparison. We now explicitly measure decoding time for Llama-3.1-8B (AR baseline) and DiffuLlama (Diffusion baseline) under the same setting in Figure 3. We evaluate all models' log-perplexity with Llama-3.1-70B. The results are shown in Table 8. Compared with the AR baseline, with small groups (size 1 & 2), A3 achieves better performance at the trade of longer time due to more complex architecture. With moderate groups (size 4), A3 achieves faster decoding than the AR baseline (67s → 37s) at a small quality tradeoff. Comapred with diffusion baseline, A3 consistently performs better with the same group size or with the same time (e.g. A3 2.1 37s v.s. DiffuLlama 2.2 51s). These results prove A3's practical decoding efficiency.

Results on longer contexts. To evaluate whether A3 remains stable under significantly longer contexts than 2k tokens, we finetuned both Llama-3.1-8B and A3-8B on 8k-length sequences from PG19 (Rae et al., 2019) using the same training budget (100 steps, batch size 64). We then evaluated them on the single-document QA task from LongBench v1 (Bai et al., 2023b), which requires reasoning over long passages. We used dynamic sampling for A3 with group size 1 and group size 2. The results are shown in Table 9.

A3 (group size 1) improves over the AR baseline with 1.7%, suggesting that the A3 factorization does not degrade long-context modeling and may offer small gains without parallel decoding. A3 (group size 2) achieves 30% faster decoding, demonstrating that A3's groupwise inference can provide real latency benefits in longer contexts. This result shows that larger groups introduce more parallelism but can slightly reduce accuracy, which is consistent with our analyses in shorter contexts.

Our 8k experiments indicate that A3 can scale to significantly longer sequences without degradation and provides decoding-time advantages via groupwise generation. These results support the potential of A3 for future long-context extensions and we will explore longer context in future works.

Table 7: Comparison between A3's dynamical resampling and ASSD sampling method.

	log(perplexity)	Time
ASSD	2.3	2.4×
A3	**2.2**	**1**×

Table 8: Comparison with Llama-3.1-8B and DiffuLlama on speed-quality tradeoff.

	log(perplexity)	Time
Llama-3.1-8B (baseline)	1.9	67s
DiffuLlama (group size = 1)	1.9	102s
DiffuLlama (group size = 2)	2.2	51s
DiffuLlama (group size = 4)	2.3	**25s**
A3 (group size = 1)	**1.7**	142s
A3 (group size = 2)	1.8	71s
A3 (group size = 4)	2.1	37s

Table 9: Accuracy and time comparison on Single Document QA task of LongBench v1 (Bai et al., 2023b).

	QA task acc (%)	Average Time
Llama-3.1-8B (baseline)	25.4	1.0×
A3 (group size = 1)	**27.1**	1.3×
A3 (group size = 2)	22.5	**0.7**×