DREAMON：超越固定尺寸画布的代码填充扩散语言模型

Zirui Wu[1,4*], **Lin Zheng**[1*], **Zhihui Xie**[1], **Jiacheng Ye**[1], **Jiahui Gao**[1], **ShansanGong**[1], **冯砚松**[4], **Zhenguo Li**[3], **Wei Bi**[2], **周国瑞**[2], **Lingpeng Kong**[1†]
[1]香港大学 [2]快手科技 [3]华为诺亚方舟实验室 [4]北京大学
ziruiwu@pku.edu.cn, lzheng2@cs.hku.hk, lpk@cs.hku.hk

arXiv:2602.01326v1 [cs.CL] 1 Feb 2026

摘要

扩散语言模型（DLMs）为自回归模型提供了一种极具吸引力的替代方案，无需专门的提示设计即可实现灵活的任意顺序填充。然而，其实际效用受到一个关键限制的阻碍：生成时需要固定长度的掩码序列。当预定义的掩码大小与理想补全长度不匹配时，这一约束会严重降低代码填充性能。为解决这一问题，我们提出 DREAMON，一种支持动态、可变长度生成的新型扩散框架。DREAMON 为扩散过程引入两种长度控制状态，使模型能够仅依据自身预测自主扩展或收缩输出长度。我们将该机制集成到现有 DLMs 中，仅对训练目标做最小修改，且无需架构改动。基于 Dream-Coder-7B 和 DiffuCoder-7B 构建的 DREAMON，在 HumanEval-Infilling 和 SantaCoder-FIM 上的填充性能与最先进的自回归模型相当，并达到了使用真实长度所实现的预言机性能。我们的工作消除了 DLMs 实际部署中的一项根本障碍，显著提升了其在可变长度生成方面的灵活性与适用性。我们的代码发布于 https://github.com/DreamLM/DreamOn。

1 引言

近年来，自回归语言模型取得了显著进展（Comanici 等人，2025；OpenAI，2025；Guo 等人，2025；通义千问等人，2025）。它们将语言建模为以固定的从左到右方式依次生成文本。尽管这一范式仍占主导地位，如今它正逐渐由扩散语言模型（DLMs）补充（Hoogeboom 等人，2021；Austin 等人，2021；Zheng 等人，2023；Singh 等人，2023；Lou等人，2024；Sahoo等人，2024；Shi等人，2024；Nie 等人，2025；Ye 等人，2025；DeepMind，2025；Labs 等人，2025）。扩散语言模型已成为一种前景广阔的替代方案，并受到广泛关注。

扩散语言模型通过多步去噪过程运行，逐步优化掩码序列，从而支持灵活的任意顺序生成（Austin 等人，2021；Hoogeboom 等人，2021）。这一特性使其天然适用于代码填充任务——在给定前缀和后缀之间生成内容（Bavarian 等人，2022；Fried 等人，2023；Allal 等，2023）。相比之下，自回归模型必须采用繁琐的变通方法来完成填充，例如将目标跨度移至序列末尾（Fried 等人，2023；Guo 等人，2024a；Hui 等人，2024；Seed 等人，2025）。这类方法不仅会破坏自然的上下文结构，还要求在训练和推理过程中使用专门的提示。

尽管具有理论优势，扩散语言模型的实际应用却受到一个关键瓶颈的阻碍：对预先指定的固定长度掩码的依赖。当前的扩散语言模型（Ye 等人，2025年；Nie 等人，2025年；Xie 等，2025；Gong 等，2025b）要求输入序列和输出序列具有

*贡献相同。
†通讯作者。

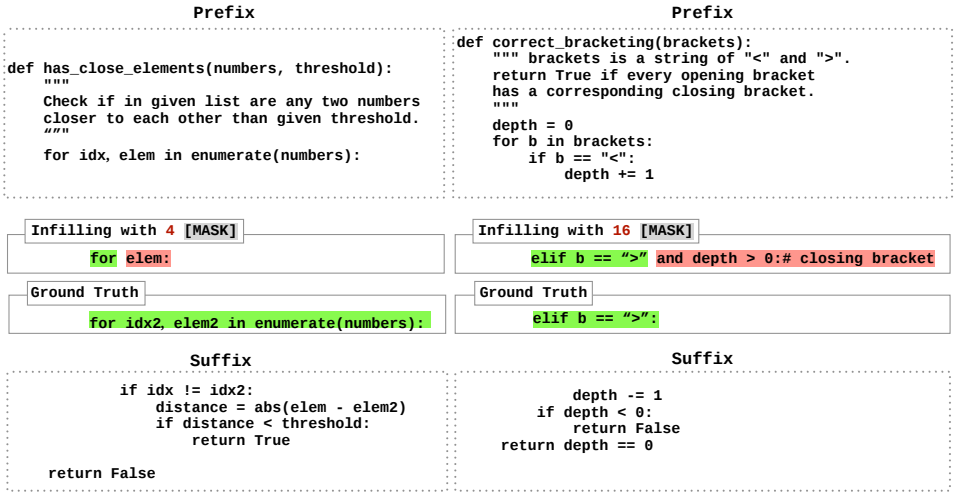


图1：DreamCoder-7B 因掩码输入与真实值之间的长度不匹配而在代码填充中失败的示例。错误代码和正确代码分别以红色和绿色标出。掩码标记过少时，扩散模型缺乏足够空间进行有意义的代码填充。掩码过多会导致不必要代码片段的过度生成（例如，深度 > 0 是不正确的）。

相同的长度，这使它们无法动态确定输出的长度。这一局限对代码填充尤其不利，因为不同示例的解决方案长度可能差异很大。如图 1 所示，当掩码长度与真实值不一致时，Dream-Coder-7B（Xie 等，2025）会生成不完整或过度生成的代码。更关键的是，我们观察到，当预定义的掩码长度与真实值长度不一致时，HumanEval-Infilling（Bavarian 等，2022）上的平均性能下降 38%（表 2），这凸显了当前扩散语言模型对该超参数的极度敏感性。

为解决这一瓶颈，我们提出 DREAMON，一种配备自适应长度调整的离散扩散语言建模框架（§3.1）。DREAMON 通过两个专用特殊标记 [扩展] 和 [删除], 引入动态长度自适应，无需架构修改。我们用辅助长度控制状态增强标准扩散训练过程，使 DREAMON 能够在与传统 DLM 目标最小偏离的情况下进行训练（§3.2）。在推理过程中，模型仅根据自身预测自适应地扩展和收缩掩码序列，无需外部引导（§3.3）。基于 Dream-Coder-7B（Xie 等，2025）和 DiffuCoder-7B（Gong 等，2025b），DREAMON 在 HumanEval-Infilling（Bavarian 等，2022）和 SantaCoder-FIM（Allal 等，2023）上取得了与最先进的自回归模型相当的填充性能（§4），并接近使用真实值长度所达到的预言机级性能（§5.1）。

• 我们通过引入 DREAMON 缓解了扩散语言模型（DLMs）的固定长度瓶颈，使模型能够在生成过程中动态扩展或收缩序列，而无需任何架构更改。

•我们的方法借助两种特殊状态 [扩展] 和[删除], 实现可变长度生成，并通过简单的增强技术支持长度自适应的可扩展端到端学习，且与标准扩散目标函数的偏差极小。

• 在多项填充基准测试中，DREAMON 相较扩散基线平均绝对性能提升了**26.4%**，与使用预言机长度所达到的性能相当，并使扩散模型接近或媲美最先进的自回归模型。

2 预备知识

令$\mathbf{x}_0 = [\mathbf{x}_{10}, \ldots, \mathbf{x}_0^N]$ 为从数据分布 $q(\mathbf{x})$ 中采样得到的离散词元序列 N。每个词元的取值来自大小为 $V + 1$ 的词汇表，该词汇表由 V 个常规符号和一个额外的吸收态 [掩码] 组成。我们将每个词元 $\mathbf{x}_{0,n}$ 以及吸收态

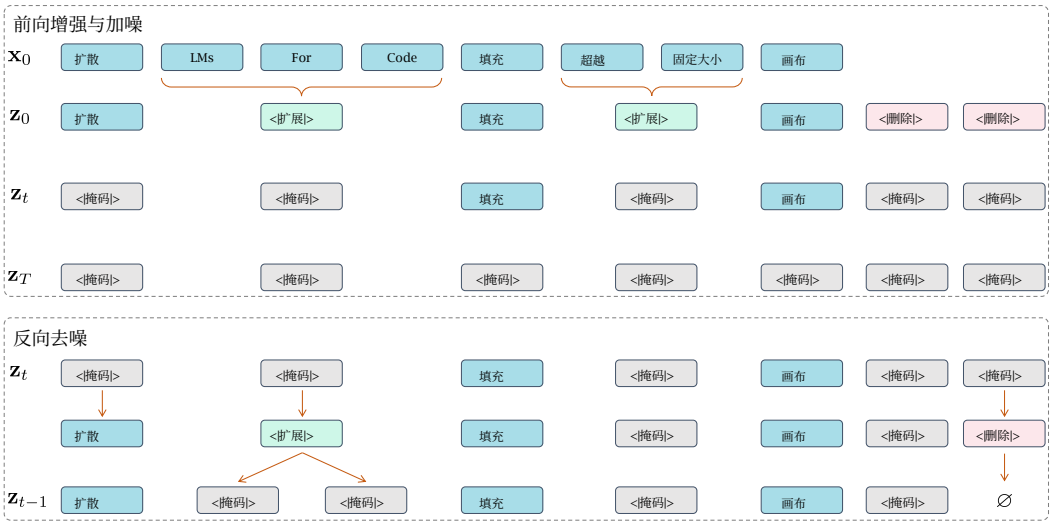


图 2：增强扩散过程概览。**上半部分**：前向增强与加噪过程将输入序列$\mathbf{x}_0$ 映射为增强潜变量$\mathbf{z}_0$ ，其中包含 [扩展]和 [删除] 状态；随后对$\mathbf{z}_0$ 应用标准掩码扩散过程，得到$\mathbf{z}_t$，最终得到$\mathbf{z}_T$。**下半部分**：单个去噪步骤，其中$\mathbf{z}_t$ 中的 [掩码] 位置可预测为常规词元或特殊状态；[扩展] 会确定性地扩展为两个 [掩码]词元，而 [删除] 则会移除相应位置，从而生成新的序列$\mathbf{z}_{t-1}$，其长度与$\mathbf{z}_t$不同。

状态 [掩码], 表示为 $\{0,1\}^{V+1}$ 中的独热向量。典型的离散时间掩码扩散模型被定义为此类序列上的一类潜变量模型，包含正向和反向转移过程。在正向过程中 q，每个词元以一定概率保持不变，否则被替换为 [掩码]，从而得到 $q(\mathbf{x}_t \mid \mathbf{x}_0) = \alpha_t \mathbf{x}_0 + (1-\alpha_t)$ [掩码]，其调度方案 α_t 预先定义。随着 t 增大，调度方案经过设计，使序列最终收敛为全掩码令牌。生成模型将反转这一过程，从 $\mathbf{x}_T$ 开始，并应用参数化转移 $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ 来近似真实后验分布 $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$。由此得到$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$。

这类生成模型可以通过考虑 $t \in [0,1]$推广到**连续时间** 参数化，从而避免预定义时间步离散化所引入的偏差。我们采用 Kingma 等人（2021）；Campbell 等人（2022）；Sahoo 等人（2024）；Shi 等人（2024）；Ou 等人（2025）的框架，并使用加权交叉熵损失目标函数训练 p_θ ，

$$\mathcal{L}(\theta) = -\mathbb{E}_{\substack{\mathbf{x}_0 \sim q(\mathbf{x}) \\ t \sim \mathcal{U}(0,1) \\ \mathbf{x}_t \sim q(\mathbf{x}_t \mid \mathbf{x}_0)}} \left[w(t) \sum_{n=1}^{N} \mathbf{1}_{[\mathbf{x}_t^n = \texttt{[mask]}]} \log p_\theta(\mathbf{x}_0^n \mid \mathbf{x}_t) \right], \tag{1}$$

其中，指示函数$\mathbf{1}_{[\mathbf{x}_t^n = \text{[掩码]}]}$ 表示损失仅在掩码位置上计算，而 $w(t) \in (0,1]$ 是根据噪声调度方案 α_t 推导出的时间相关加权项（Shi等人，2024年；Gong 等人，2025年a）。该目标函数为负对数似然提供了一个可处理的变分上界，并作为大规模扩散语言模型的有效训练目标。

3 方法

本节介绍我们将掩码扩散模型扩展到固定长度生成之外的方法。我们首先在 §3.1 概述框架，随后在第 3.2 和 3.3 节介绍训练和推理过程，并在 §3.4 说明实际实现细节。

算法 1 DREAMON 训练

要求： 模型参数 p_θ，合并率调度器 $\mathcal{S}$；
1: **重复**
2: 采样原始数据 $\mathbf{x}_0 \sim q(\mathbf{x}\ t \sim \mathcal{U}(0,1)$ 和一个时间步 $t \sim \mathcal{U}(0,1)$；
3: 使用 [扩展] 和 [删除] 以及 $\mathcal{S}$，从 $\mathbf{x}_0$ 构造增强序列$\mathbf{z}_0$；
4: 采样掩码序列$\mathbf{z}_t \sim q(\mathbf{z}_t|\mathbf{z}_0)$；
5: 通过式 (2) 计算加权损失 $\mathcal{L}(\theta)$；
6: 通过 $\nabla_{\boldsymbol{\theta}}\mathcal{L}$更新参数 $\boldsymbol{\theta}$；
7: **直到**收敛

3.1 使用增强状态的掩码扩散实现长度控制

DREAMON 的关键在于在扩散过程中引入两种新的特殊状态 [扩展] 和 [删除]。当某个词元转换到状态 [扩展], 时，我们会在序列的同一位置将其扩展为两个 [掩码] 词元；而当产生 [删除] 状态时，该词元会从序列中移除。通过对这些特殊状态进行恰当预测，模型便获得了原生的长度控制能力。

通过数据增强模拟特殊转移。 为训练模型预测[扩展] 和 [删除], ，我们引入一个由原始输入 $\mathbf{x}_0$ 构建的辅助增强序列 $\mathbf{z}_0$。该增强将随机词元片段合并为 [扩展], 并将[删除] 插入序列中。对于合并，我们首先采样一个时间步 $t \sim$ Uniform(0,1)，并计算一组掩码索引 $\mathcal{M}_t$，依据的是调度方案 α_t。我们并不对词元进行掩码，而是使用 $\mathcal{M}_t$ 来门控合并，使得仅 $\mathcal{M}_t$ 中连续掩码索引构成的片段会在由速率调度器（§3.4）控制的合并概率下被替换为 [扩展], 。这种伪掩码过程能够更精细地控制特殊词元与常规词元的比例，从而生成长度可变、且常规状态与特殊状态比例均衡的 $\mathbf{z}_0$。

增强 $\mathbf{z}_0$上的掩码扩散。 我们将掩码扩散过程 $p_{\boldsymbol{\theta}}$ 施加于$\mathbf{z}_0$。对于这些特殊状态 [扩展] 和 [删除], ，前向扩散过程始终会将其映射为 [掩码], ，确保所有此类词元都被掩码，并为学习信号作出贡献。按照构造，掩码扩散损失中的预测目标自然包含 [扩展]和[删除]。因此，模型现在经过训练，不仅能对常规词元进行去噪，也能对特殊哨兵词元进行去噪，从而学会长度控制行为，并且无需更改任何架构即可实现可变长度生成。

3.2 训练

与掩码状态 [掩码] 和任何常规词元类似，我们将 [扩展] 和 [删除]视为词汇表中的哨兵词元，并使用式（1）中的目标函数训练模型对其进行预测。然而，在训练过程中，我们观察到一种不平衡现象：许多 [掩码] 位置对应 [删除]目标，而对应 [扩展], 的要少得多，因为每个 [删除] 都会被转换为单个 [掩码], ，而多个 [掩码] 词元会合并为一个 [扩展]。因此， [删除] 词元对损失的贡献过大。为对此进行校准，我们引入一种损失加权方案，降低 [删除] 预测的贡献，使其总权重等同于单个 [掩码] 预测的权重。加权训练损失如下所示

$$\mathcal{L}(\theta) = -\mathbb{E}_{\substack{\mathbf{x}_0\sim q(\mathbf{x}) \\ t\sim\mathcal{U}(0,1) \\ \mathbf{z_0}\sim q(\mathbf{z_0}|\mathbf{x_0}) \\ \mathbf{z}_t\sim q(\mathbf{z}_t|\mathbf{z}_0)}} \left[w(t) \sum_{n=1}^{N} \mathbf{1}_{[\mathbf{z}_t^n = \texttt{[mask]}]} \cdot w_n \cdot \log p_\theta(\mathbf{z}_0^n \mid \mathbf{z}_t) \right], \tag{2}$$

其中逐词元权重 w_n 定义为

$$w_n = \frac{\mathcal{N}_{mask}}{\mathcal{N}_{mask} - \mathcal{N}_{delete} + 1} \times \begin{cases} 1, & \text{if } \mathbf{z}_0^n \neq \texttt{[delete]}, \\ \dfrac{1}{\mathcal{N}_{delete}}, & \text{if } \mathbf{z}_0^n = \texttt{[delete]}, \end{cases} \tag{3}$$

其中，$\mathcal{N}_{mask}$ 和 $\mathcal{N}_{delete}$ 分别表示序列中 [掩码] 和 [删除] 词元的数量。归一化因子可确保具有不同删除数量的序列，其期望损失幅度保持一致。

3.3 推理

我们的推理过程如算法 2 所示，它在标准的掩码扩散去噪框架基础上进行了关键改进，以支持可变长度生成。在每个扩散步骤中，我们同时预测所有掩码位置，然后遵循 Ye 等人（2025）的方法，根据预测熵选择性地重新掩码令牌。然而，先前的工作采用固定的掩码调度器来决定每步取消掩码的令牌数量，这并不适用于动态长度建模，因为它们假定输出序列长度预先指定。相反，我们通过指定 n（即每步去噪的掩码令牌数量）来直接控制去噪轨迹，从而实现自适应序列长度建模。在去噪过程中，预测为 [扩展] 的令牌会立即扩展为两个 [掩码]令牌，而生成的 [删除] 令牌则会从序列中移除。为确保稳定并防止无限增长，我们对最大输出序列长度 L_{max} 进行限制。当所有 [掩码] 位置都已处理完毕后，生成过程即告终止。

算法 2使用 DREAMON 的可变长度生成

要求：已训练的模型参数 $\boldsymbol{\theta}$、初始序列长度 L、最大长度 $L_{\max}$、解掩-
码预算 n 每次迭代，以及采样温度 τ；

对于 $l = 1, 2, \ldots, L$ **do**
  初始化$\mathbf{z}^l \leftarrow$ [掩码]；
结束循环
当 [掩码]位于$\mathbf{z}$ **中时执行**
  计算词元概率 $p \leftarrow p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{z})$；
  如果 $|\mathbf{z}| \geq L_{\max}$ **则**
    将 [扩展] 的概率设为 0 并重新归一化；
  结束条件判断
  选择置信度最高的至多 n 个掩码位置；
  对每个选定位置 i **执行**
    从分类分布(p^i/τ)中抽取 $\tilde{\mathbf{z}}^i \sim$ ；
    if $\tilde{\mathbf{z}}^i =$ [扩展]**然后**
      将 $\mathbf{z}[i]$ 替换为 [[掩码], [掩码]]；
    否则如果 $\tilde{\mathbf{z}}^i =$ [删除]**然后**
      从序列中移除 $\mathbf{z}[i]$；
    else
      设置$\mathbf{z}[i] \leftarrow \tilde{\mathbf{z}}^i$；
    结束 if
    如果长度发生变化，则更新位置索引；
  结束 for
结束 while
返回 $\mathbf{z}$。

3.4 实际实现

跨度合并调度器。 我们设计了两种经验性的掩码合并调度方案。（1）**静态调度方案**：以固定概率 p_{merge} 合并相邻的 [掩码] 词元；（2）**动态逆向调度器**：将合并概率设置为与序列中的 [掩码] 词元数量成反比。该调度器在 [掩码] 词元较多时减少合并，以避免合并过多可能影响基础模型原始性能的词元。我们发现，在训练期间混合使用两种调度方案能够取得最佳性能，详见 §5.2。

广播式删除作为长度预测器。 在实践中，我们观察到，当初始掩码跨度与真实目标长度相差较大时，性能会略有下降。这会导致推理效率低下，因为模型必须耗费大量前向传播

来通过增量扩展或收缩调整序列长度。为缓解这一问题，我们引入了一种免训练推理时自适应方法，以加速收敛。具体而言，每当模型预测出 [删除], ，如果其右侧的所有后续词元都是 [掩码] 词元，我们就将它们全部删除。该机制显著减少了不必要的计算，同时提升了推理效率，且不会牺牲生成质量。

4 实验

4.1 设置

我们在 OpenCoder 监督微调数据（Huang 等人，2024年）的教育指令子集上微调 Dream-7B（Ye 等人，2025年）、DiffuCoder-7B（Gong 等人，2025年b）和 DreamCoder-7B（Xie 等人，2025年）。该子集包含约 11 万组从高质量教育数据合成的 Python指令-解答对。我们的实验聚焦于代码填充，目标是根据周围的前缀和后缀上下文生成缺失片段。训练期间，我们随机将每个解答拆分为三个片段：前缀、中间片段和后缀。指令、前缀和后缀固定为上下文，而扩散仅应用于中间片段。

对于序列收缩，我们发现训练期间只需在中间片段末尾附加随机数量的 [删除] 个 token（0 到 64 个）即可。对于序列扩展，使用静态和动态逆调度器各占一半的混合方式，并将合并概率p_{merge} 设为 0.5，以构造 [扩展]token。模型训练 10 个 epoch，批量大小为 128，最大上下文长度为 1024，学习率为 1e-5 ，并采用余弦衰减调度方案和 10% 的预热步数。使用 8 个 H800 GPU 训练约需 5 小时。与预训练基础模型所需的计算量相比，DreamOn 的计算量仅为 0.15%（Ye 等人，2025年）。推理时，我们将温度设为 0.2，top-p 设为 0.9。为防止过度增长，我们将 DREAMON 中的掩码扩展限制在 $L_{max} = 128$。扩展 L_{max} 次后，我们还会在推理期间禁用掩码扩展。

4.2 评测

基线。 我们将其与采用填充目标进行预训练的最先进自回归 模型进行比较，具体包括 Deepseek-Coder-6.7B（Guo 等，2024b）、Qwen2.5-Coder-7B（Hui 等人，2024年）和 Seed-Coder-8B（Seed 等人，2025年）。对于规模相近的开源扩散语言模型基线，我们评测了 LLaDA-8B（Nie 等人，2025年）、Dream-7B（Ye 等人，2025年）、DiffuCoder-7B（Gong 等，2025b）和 DreamCoder-7B（Xie 等，2025）。

基准。 我们在 HumanEval-Infilling（Bavarian 等，2022年）基准上评测模型，包括单行和多行子集，以及 Santacoder-FIM（Allal 等，2023）的 Python 子集。我们使用官方评测脚本报告 HumanEval-Infilling 的 Pass@1 和 Santacoder-FIM 的精确匹配率。评测自回归语言模型时，我们使用其预训练期间采用的相应填充模板。对于所有扩散模型，掩码长度默认设为 64。DREAMON 的不同变体会动态调整该长度，详情见 §3.4。

4.3 结果

表1显示，基线扩散模型因其固定长度生成而难以进行代码填充，明显落后于自回归模型。DREAMON有效解决了这一局限，相较扩散基线平均绝对提升26.4%，并凸显了其作为一种模型无关增强方法的有效性。

值得注意的是，借助 DREAMON，DiffuCoder-7B 和 DreamCoder-7B 不仅达到了 Qwen2.5-Coder-7B 等领先自回归模型的性能，还在更具挑战性的多行代码填充基准测试中超越了它们。这表明，为扩散模型配备我们的长度自适应机制，能使其在代码填充任务中具有很强的竞争力。

表1：在 HumanEval-Infilling 上的 Pass@1 和在 SantaCoder-FIM 上的精确匹配率，对比开源自回归模型与扩散模型基线。扩散模型中的最佳结果以粗体显示，次佳结果以下划线标出。

模型	HumanEval-Infilling（Pass@1）		SantaCoder（精确匹配）
	单行	多行	
开放权重自回归模型			
Deepseek-Coder-6.7B	73.0	45.7	76.3
Seed-Coder-8B	89.7	59.3	77.2
Qwen2.5-Coder-7B	92.6	58.7	79.8
开放权重扩散模型			
LLaDA-8B	48.3	21.1	35.1
Dream-7B	48.2	21.9	60.3
+ DREAMON	88.6 +40.4	53.3 +31.4	73.8 +13.5
DiffuCoder-7B	53.7	45.0	58.0
+ DREAMON	**92.2** +38.5	63.1 +18.1	77.4 +19.4
DreamCoder-7B	55.5	43.2	59.3
+ DREAMON	92.1 +36.6	**63.8** +20.6	**79.0** +19.7

表 2：扩散语言模型在不同设计下的填充性能。预言机：使用预言机目标长度时的性能，供参考。†：我们使用 AST 解析器计算精确匹配，以规范化模型输出与真实值之间巨大的语法差异。

模型	初始掩码长度					Avg.	预言机
	4	8	16	32	64		
单行（Pass@1）							
Dream-Coder-7B	24.9	61.2	72.6	62.4	55.5	55.3	93.3
+ DREAMON	**88.7**	**90.6**	**91.0**	**91.6**	**92.1**	**90.8**	91.6
不删除	87.8	77.9	71.2	62.3	37.8	67.4	93.3
不使用扩展	25.1	71.6	88.0	90.9	91.5	73.4	92.5
多行（Pass@1）							
Dream-Coder-7B	5.5	14.7	27.1	39.4	43.2	26.0	69.0
+ DREAMON	**50.2**	**53.8**	**56.9**	**60.9**	**63.8**	**57.1**	66.6
不使用删除	44.6	45.3	46.1	46.7	44.7	45.5	67.9
不使用扩展	5.5	16.5	30.7	48.2	61.3	32.4	63.2
SantaCoder-FIM（EM）							
Dream-Coder-7B	20.0	26.6	43.5	50.8	59.3	40.0	76.3†
+ DREAMON	**75.0**	**76.8**	**78.4**	**78.0**	**79.0**	**77.4**	82.0
不使用删除	74.2	44.3	40.2	50.0	56.2	53.0	84.2
不使用扩展	22.5	55.0	74.7	77.8	78.0	61.6	78.6†

5 分析

本节通过消融研究评估 DREAMON 的有效性。所有变体均在 DreamCoder-7B 的基础上进行微调，并使用从 4 到 64 的初始掩码长度进行评估。此外，我们还在预言机 设置下评估填充结果：此时初始掩码长度与真实解长度一致，从而为扩散语言模型的填充性能提供一个近似上限。

5.1 不同掩码长度下的性能

性能分析。 如表 2 所示，与预言机长度下的性能相比，未经微调的 DreamCoder-7B 在使用固定掩码长度时性能显著下降，凸显了填充质量对准确掩码长度的高度依赖。相比之下，DREAMON 在较宽范围的初始掩码长度下均能达到接近预言机级性能。重要的是，性能提升源于掩码扩展和收缩机制的结合使用。无论初始掩码长度如何，DREAMON 在单行填充和 SantaCoder-FIM 任务上都能保持稳定性能。我们在附录 D 中提供了两个去噪轨迹示例。

长度控制的消融实验。 为单独分析扩展和删除机制的作用，我们评估了 DREAMON 的以下消融变体：**(1) 不使用扩展**，禁用掩码扩展；以及 (2)**不使用删除**，禁用掩码删除。在较长的掩码长度下，移除删除机制会导致性能急剧下降，因为模型往往会过度生成，并填充给定的全部 [掩码] 标记。另一方面，移除扩展机制会严重损害短长度下的性能，因为模型无法动态延长掩码序列，以适应更复杂或更长的补全结果。我们还观察到，在不使用扩展机制时，长掩码下的性能会略有下降，这表明即使对于较长的掩码输入，扩展机制仍然有益，因为它能够进行细粒度的长度调整。

5.2 扩展机制设计

掩码合并会显著影响[扩展]标记的数量，以及初始序列长度与目标序列长度之间的差距。我们对此进行了深入研究，并评估了两种合并率调度器：一种是采用固定合并概率的静态调度器，另一种是合并概率与 [掩码] 标记数量成反比的动态逆向调度器。仅使用静态调度器即可实现有效扩展，在长度为 4 的掩码上达到 88.9% 的通过率。然而，其在更长掩码上的性能受到限制。

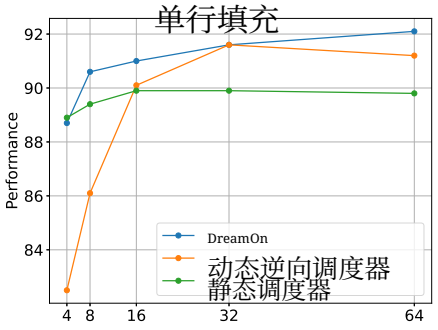


图 3：合并率调度方案设计选择的消融实验。

存在的掩码越多，动态逆向调度器的合并操作就越少。它在更长掩码上的性能更高，但在大规模扩展时表现欠佳，在长度为 4 的掩码上降至 82.5%。我们发现，1:1 的混合方案能够取得最佳的总体结果，在各种掩码长度之间实现良好平衡（附录 C）。

5.3 删除机制设计

我们通过以下实验消融分析处理 [删除] 标记的设计选择：**(1)不使用损失平衡**：训练模型时，不降低 [删除] 标记的损失权重，而是在损失计算中将其与其他标记同等对待；**(2) 不使用原位删除**：不删除 [删除]标记，而是将其保留在序列中，类似于在标准扩散语言模型中生成填充占位标记（Nie 等人，2025年；Ye 等人，2025年）。为实现这一点，我们在训练期间随机掩码或保留 [删除]；**(3)不使用删除广播**：禁用 §3.4 中所述的推理时机制。

如表 2 所示，移除损失平衡会导致性能大幅下降，平均 pass@1 率降至 84.6%，这证实了降低 [删除] 损失的权重对于防止模型过拟合删除信号至关重要。保留持续存在的 [删除] 标记同样表现不佳（平均 85.3%），这表明序列中的占位符式删除标记会破坏位置一致性并降低训练效果。禁用删除广播后，平均性能下降 0.6%，尤其是在给定掩码长度远超预期解长度时。删除广播机制还使生成速度提升了 2.1×。

表 3：掩码删除机制实现的消融研究。

模型	初始掩码长度					Avg.	预言机
	4	8	16	32	64		
HumanEval 单行填充（Pass@1）							
DREAMON	**88.7**	**90.6**	**91.0**	**91.6**	**92.1**	**90.8**	91.6
无损失平衡	75.8	82.5	87.0	87.2	90.4	84.6	88.6
无原位删除	85.9	85.7	88.5	84.8	78.0	84.6	93.1
无删除广播	**88.7**	90.5	90.0	90.2	91.4	90.2	91.6

5.4 删除广播的效率分析

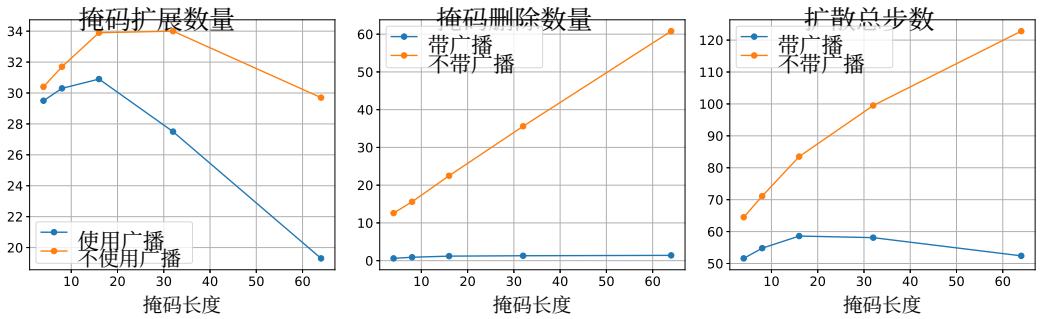


图 4：DreamCoder + DREAMON 在多行填充子集上的平均生成步数 et.

引入广播显著提升了推理效率，主要是将删除过程从逐个标记执行转变为长度预测操作。没有广播时，删除步数几乎随初始掩码长度线性增长，因为模型必须通过迭代逐个预测并移除多余的掩码标记。相比之下，采用删除广播机制的 DREAMON 将掩码删除数量维持在近乎恒定的水平，平均约为 1 步。这项优化消除了初始掩码跨度与真实目标长度之间差异过大所造成的计算瓶颈；当掩码长度为 64 时，推理步数从最高达 122.8 步（不使用广播）降至仅 52.4 步（使用广播）。因此，广播不仅大幅减少了不必要的前向传播，还稳定了整体推理成本，使生成过程更快，也更能适应初始化方差，同时不影响输出质量。

6 相关工作

使用自回归模型进行代码填充。 代码填充要求根据双向上下文生成缺失的代码片段，这项任务对标准的从左到右的自回归模型而言本身就具有挑战性。为此，一些方法对模型架构进行调整，以更好地捕捉双向依赖关系 Yang等人（2019年）；Stern等人（2019年）；Gu等人（2019年a）；Chan等人（2019年）；Welleck等人（2019年）；Shen等人（2020年）；Alon等人（2020年）；Nguyen等人（2023年）；Shen等人（2023年）。

一种广泛采用的替代方案保留了标准的从左到右自回归架构，将目标填充片段移至输入序列末尾，使模型能够以自回归方式生成缺失的代码（Raffel 等，2020；Tay 等，2023；2022；Anil 等，2023；Sun 等，2024）。这种方法兼容仅解码器架构（Bavarian 等，2022），并已成为现代代码语言模型的标准，包括 Codex（OpenAI 等，2022）、INCODER（Fried 等，2023）、SANTACODER（Allal 等，2023）、StarCoder（Li等，2023）和 StarCoder 2（Lozhkov 等，2024）、CODEGEN 2/2.5（Nijkamp 等，2023）、Code-Llama（Roziere 等，2023）、DeepSeek-Coder（Guo 等，2024a）、CodeGemma（CodeGemma 团队，2024）、Qwen-Coder（Bai 等，2023；Hui 等，2024）以及 Seed-Coder（Seed 等，2025）。

离散扩散语言模型。离散扩散模型近来已成为自回归模型一种颇具吸引力的替代方案。Austin 等人（2021）和 Hoogeboom 等人（2021）的奠基性工作为文本数据引入了离散扩散过程，使得能够通过迭代式双向去噪对词元序列进行概率建模。后续研究通过连续时间松弛（Campbell 等，2022）、改进训练目标（Zheng 等，2023；Lou等，2024）以及推广掩码扩散框架（Sahoo等，2024；Shi等，2024；Ou等，2025）进一步完善了这些方法。规模化工作催生了 Plaid（Gulrajani与Hashimoto，2023）和 LLaDA（Nie 等人，2025）等强大模型。利用预训练模型的适配技术，例如 DiffuLLaMA（Gong 等，2025a）和 Dream（Ye 等，2025），已缩小了与最先进自回归语言模型之间的性能差距。

具有长度控制的非自回归模型。生成变长序列仍是非自回归模型面临的一项重大挑战。以往研究探索了多种应对策略，包括学习独立的长度预测器（Gu等人，2018；Lee等人，2018；Ghazvininejad等人，2019年；Zheng等人，2023年）、通过对潜在对齐进行边缘化来缩短序列长度（Chan等人，2020年）、引入编辑操作（Gu等人，2019年a；b；Stern等人，2019年；Johnson等人，2021；Reid等人，2023年；Campbell等人，2023年；Patel等人，2025年；Havasi等人，2025年），以及在序列位置上执行扩散（Zhang等人，2025年；Kim等人，2025年）。近期并行开展的研究也在应对这一挑战。Edit flows（Havasi等人，2025年）提出了一种离散流匹配方法，在扩展空间中执行编辑操作，从而实现可处理且有效的训练；DDOT（Zhang等人，2025年）提出联合去噪词元状态与位置，以动态调整片段长度；FlexMDM（Kim等人，2025年）通过在词元状态和位置上采用联合插值框架，学习插入与去掩码速率，从而实现可变长度生成；DAEDAL（Li等人，2025年）则提供了一种无需训练的方法，利用推理时的预测置信度分数动态确定响应长度。

相比之下，我们的方法只需极少的额外训练且无需修改架构，即可在掩码扩散模型中实现原生长度控制，并直接适配预训练的扩散语言模型。这一设计保留了原始模型的简洁性，避免了复杂的多阶段推理流程，并显著提升了可变长度生成的灵活性和性能。

7 结论

在这项工作中，我们提出了 DREAMON——一种简单而有效的框架，无需架构更改，仅通过两个特殊标记（ [删除]和 [扩展]）即可实现动态长度控制。通过在扩散过程中加入辅助长度控制状态，DREAMON 学会仅根据模型置信度扩展或缩短序列。结果表明，DREAMON 的表现接近神谕长度性能，并取得了与最先进的自回归模型相当的结果。我们希望这项工作能为超越固定大小画布的、更实用灵活的扩散语言模型铺平道路。

局限性。 目前，我们将评估范围限定在需要强大可变长度生成能力的代码填充任务上。未来的工作将把研究范围扩展到更广泛的应用，以评估 DREAMON 的泛化能力。此外，DREAMON 的训练和推理过程依赖启发式方法，以简单而有效的方式实现可变长度生成；为掩码扩散模型中的灵活推理制定更具原则性的形式化方案，是未来研究的重要方向。最后，我们当前的设计采用单一扩展状态[扩展]，确定性地扩展为两个 [掩码] 令牌。这一选择使输出空间保持较小，并确保训练稳定，但当目标补全内容远长于初始掩码跨度时，需要进行多次扩展。未来一项很有前景的研究方向是引入包含多种扩展因子的更丰富词汇表，或将扩展与显式的长度预测头结合起来。这些方法可以减少去噪迭代次数，但也会扩大决策空间，并要求谨慎地重新平衡训练目标，以保持良好的长度调整动态。我们将这些更丰富的扩展方案留待未来探索。

致谢

我们感谢开源社区提供高质量的数据集和评估框架。本研究部分由国家自然科学基金委员会（NSFC）与研究资助局（RGC）的联合研究计划资助，资助编号为N HKU714/21。

伦理声明

我们的研究聚焦于开发一种能够进行可变长度文本生成的基于扩散的语言建模方法。我们未收集任何涉及人类受试者或私人信息的数据。本研究亦不包含任何人工评估。实验中使用的所有数据集均为公开可用的基准，且我们严格遵守其各自的使用许可。此外，我们的方法不存在任何可预见的滥用或社会危害风险。

可复现性声明

我们已采取审慎措施以确保本研究的可复现性。实验设置与超参数配置的详细说明见§4。

参考文献

Loubna Ben Allal、Raymond Li、Denis Kocetkov、Chenghao Mou、Christopher Akiki、Carlos Munoz Ferrandis、Niklas Muennighoff、Mayank Mishra、Alex Gu、Manan Dey 等。SantaCoder：不要好高骛远！*arXiv* 预印本 *arXiv:2301.03988*，2023年。

Uri Alon、Roy Sadaka、Omer Levy 和 Eran Yahav。代码结构语言模型。载于国际机器学习会议，第 245–256 页。PMLR，2020年。

Rohan Anil、Andrew M Dai、Orhan Firat、Melvin Johnson、Dmitry Lepikhin、Alexandre Passos、Siamak Shakeri、Emanuel Taropa、Paige Bailey、Zhifeng Chen 等。PaLM 2 技术报告。*arXiv* 预印本 *arXiv:2305.10403*，2023年。

Jacob Austin、Daniel D. Johnson、Jonathan Ho、Daniel Tarlow 和 Rianne van den Berg。离散状态空间中的结构化去噪扩散模型。载于《神经信息处理系统进展》，2021年。网址：https://openreview.net/forum?id=h7-XixPCAL。

Jinze Bai、Shuai Bai、Yunfei Chu、Zeyu Cui、Kai Dang、Xiaodong Deng、Yang Fan、Wenbin Ge、Yu Han、Fei Huang 等。通义千问技术报告。*arXiv* 预印本 *arXiv:2309.16609*，2023年。

Mohammad Bavarian、Heewoo Jun、Nikolas Tezak、John Schulman、Christine McLeavey、Jerry Tworek 和 Mark Chen。用于填充中间内容的高效语言模型训练。*arXiv* 预印本 *arXiv:2207.14255*，2022年。

Andrew Campbell、Joe Benton、Valentin De Bortoli、Thomas Rainforth、George Deligiannidis 和 Arnaud Doucet。离散去噪模型的连续时间框架。神经信息处理系统进展，35：28266–28279，2022 年。

AndrewCampbell、WilliamHarvey、ChristianDietrich Weilbach、ValentinDeBortoli、TomRain- forth 和 Arnaud Doucet。通过跳跃扩散模型进行跨维度生成建模。发表于第三十七届神经信息处理系统会议，2023年。URL https://openreview.net/forum?id=t6nA7x3GAC。

William Chan、Nikita Kitaev、Kelvin Guu、Mitchell Stern 和 Jakob Uszkoreit。Kermit：面向序列的基于生成式插入的建模。*arXiv* 预印本 *arXiv:1906.01604*，2019年。

William Chan、Chitwan Saharia、Geoffrey Hinton、Mohammad Norouzi 和 Navdeep Jaitly。《Im- puter：通过插补和动态规划进行序列建模》。载于 国际机器学习大会，2020年。

谷歌有限责任公司 CodeGemma 团队。CodeGemma：基于 Gemma 的开放代码模型，2024年。网址：https://storage.googleapis.com/deepmind-media/gemma/ codegemma_report.pdf。

Gheorghe Comanici、Eric Bieber、Mike Schaekermann、Ice Pasupat、Noveen Sachdeva、Inderjit Dhillon、Marcel Blistein、Ori Ram、Dan Zhang、Evan Rosen 等。Gemini 2.5：以先进推理、多模态、长上下文和新一代智能体能力拓展前沿。*arXiv* 预印本 *arXiv:2507.06261*，2025年。

DeepMind。Geminidiffusion。2025年。网址：https://deepmind.google/models/gemini-diffusion/。

Daniel Fried、Armen Aghajanyan、Jessy Lin、Sida Wang、Eric Wallace、Freda Shi、Ruiqi Zhong、Scott Yih、Luke Zettlemoyer 和 Mike Lewis。InCoder：用于代码填充和合成的生成模型。载于第十一届国际学习表征会议，2023年。网址：https://openreview.net/forum?id=hQwb-lbM6EL。

Marjan Ghazvininejad、Omer Levy、Yinhan Liu 和 Luke Zettlemoyer。Mask-predict：条件掩码语言模型的并行解码。载于 *2019*年自然语言处理实证方法会议和第九届自然语言处理国际联合会议（*EMNLP-IJCNLP*）论文集，2019年。网址：https://aclanthology.org/D19-1633/。

Shansan Gong、Shivam Agarwal、Yizhe Zhang、Jiacheng Ye、Lin Zheng、Mukai Li、Chenxin An、Peilin Zhao、Wei Bi、Jiawei Han、Hao Peng 和 Lingpeng Kong。通过从自回归模型迁移来扩展扩散语言模型。国际学习表征会议，2025年a。

Shansan Gong、Ruixiang Zhang、Huangjie Zheng、Jiatao Gu、Navdeep Jaitly、Lingpeng Kong 和 Yizhe Zhang。Diffucoder：理解并改进用于代码生成的掩码扩散模型。*arXiv* 预印本 *arXiv:2506.20639*，2025年b。

Jiatao Gu、JamesBradbury、Caiming Xiong、VictorO.K. Li 和 RichardSocher。非自回归神经机器翻译。发表于 国际学习表征会议，2018。网址：https://openreview.net/forum?id=B1l8BtlCb。

Jiatao Gu、Qi Liu 和 Kyunghyun Cho。基于自动推断生成顺序的插入式解码。计算语言学协会会刊，2019年a。网址：https://aclanthology.org/Q19-1042/。

Jiatao Gu、Changhan Wang 和 Junbo Zhao。Levenshtein 变换器。神经信息处理系统进展，2019b。

Ishaan Gulrajani 和 Tatsunori B Hashimoto。基于似然的扩散语言模型。神经信息处理系统进展，36：16693–16715，2023年。

Daya Guo、Qihao Zhu、Dejian Yang、Zhenda Xie、Kai Dong、Wentao Zhang、Guanting Chen、Xiao Bi、Y Wu、YKLi 等。Deepseek-coder：当大语言模型遇上编程——代码智能的崛起。*arXiv* 预印本 *arXiv:2401.14196*，2024a。

Daya Guo、Qihao Zhu、Dejian Yang、Zhenda Xie、Kai Dong、Wentao Zhang、Guanting Chen、Xiao Bi、Yu Wu、YK Li等。Deepseek-coder：当大语言模型遇见编程——代码智能的兴起。*arXiv*预印本 *arXiv:2401.14196*，2024b。

Daya Guo、Dejian Yang、Haowei Zhang、Junxiao Song、Ruoyu Zhang、Runxin Xu、Qihao Zhu、Shirong Ma、Peiyi Wang、Xiao Bi等。Deepseek-r1：通过强化学习激励大语言模型的推理能力。*arXiv*预印本 *arXiv:2501.12948*，2025年。

Marton Havasi、Brian Karrer、Itai Gat 和 Ricky TQ Chen。《Edit flows：基于编辑操作的流匹配》。*arXiv* 预印本 *arXiv:2506.09018*，2025年。

Emiel Hoogeboom、Didrik Nielsen、Priyank Jaini、Patrick Forr´e 和 Max Welling。《Argmax 流与多项式扩散：学习类别分布》。载于神经信息处理系统进展，2021年。网址：https://openreview.net/forum?id=6nbpPqUCIi7。

Siming Huang、Tianhao Cheng、Jason Klein Liu、Jiaran Hao、Liuyihan Song、Yang Xu、J Yang、Jiaheng Liu、Chenchen Zhang、Linzheng Chai 等。Opencoder：顶级代码大语言模型的开放食谱。*arXiv* 预印本 *arXiv:2411.04905*，2024年。

Binyuan Hui、Jian Yang、Zeyu Cui、Jiaxi Yang、Dayiheng Liu、Lei Zhang、Tianyu Liu、Jiajun Zhang、Bowen Yu、Keming Lu 等。Qwen2.5-Coder 技术报告。*arXiv* 预印本 *arXiv:2409.12186*，2024年。

Daniel D. Johnson、Jacob Austin、Rianne van den Berg 和 Daniel Tarlow。《超越原位损坏：去噪概率模型中的插入与删除》。载于*ICML* 可逆神经网络、归一化流与显式似然模型研讨会，2021年。网址：https://openreview.net/forum?id=cAsVBUe1Rnj。

Jaeyeon Kim、Lee Cheuk-Kit、Carles Domingo-Enrich、Yilun Du、Sham Kakade、Timothy Ngotiaoco、Sitan Chen 和 Michael Albergo。任意顺序的灵活长度掩码扩散。*arXiv* 预印本 *arXiv:2509.01025*，2025年。

Diederik Kingma、Tim Salimans、Ben Poole 和 Jonathan Ho。变分扩散模型。神经信息处理系统进展，34:21696–21707，2021。

Inception Labs、Samar Khanna、Siddhant Kharbanda、Shufan Li、Harshit Varma、Eric Wang、Sawyer Birnbaum、Ziyang Luo、Yanis Miraoui、Akash Palrecha 等。Mercury：基于扩散的超快速语言模型。*arXiv* 预印本 *arXiv:2506.17298*，2025年。

Jason Lee、Elman Mansimov 和 Kyunghyun Cho。通过迭代细化进行确定性非自回归神经序列建模。载于*2018*年自然语言处理实证方法会议论文集，2018年。网址：https://aclanthology.org/ D18-1149/。

Jinsong Li、Xiaoyi Dong、Yuhang Zang、Yuhang Cao、Jiaqi Wang 和 Dahua Lin。《超越固定长度：扩散大语言模型的免训练变长去噪》。*arXiv* 预印本 *arXiv:2508.00819*，2025年。

Raymond Li、Loubna Ben Allal、Yangtian Zi、Niklas Muennighoff、Denis Kocetkov、Chenghao Mou、Marc Marone、Christopher Akiki、Jia Li、Jenny Chim 等。《Starcoder：愿源代码与你同在！》*arXiv* 预印本 *arXiv:2305.06161*，2023年。

Aaron Lou、Chenlin Meng 和 Stefano Ermon。《通过估计数据分布的比率进行离散扩散建模》。载于第四十一届国际机器学习大会，2024年。网址：https://openreview.net/forum?id=CNicRIVIPA。

Anton Lozhkov、Raymond Li、Loubna Ben Allal、Federico Cassano、Joel Lamy-Poirier、Nouamane Tazi、AoTang、Dmytro Pykhtar、JiaweiLiu、YuxiangWei 等。《Starcoder2 与 The Stack v2：下一代》。*arXiv* 预印本 *arXiv:2402.19173*，2024年。

Nasrin Mostafazadeh、Nathanael Chambers、Xiaodong He、Devi Parikh、Dhruv Batra、Lucy Vander- wende、Pushmeet Kohli 和 James Allen。用于深入理解常识故事的语料库与完形填空评测。载于《*2016*年北美计算语言学协会分会会议论文集：人类语言技术》，第839–849页，2016年。

Anh Nguyen、Nikos Karampatziakis 和 Weizhu Chen。中间相遇：一种新的预训练范式。*arXiv* 预印本 *arXiv:2303.07295*，2023年。

Shen Nie、Fengqi Zhu、Zebin You、Xiaolu Zhang、Jingyang Ou、Jun Hu、Jun Zhou、Yankai Lin、Ji-Rong Wen 和 Chongxuan Li。《大型语言扩散模型》。*arXiv* 预印本 *arXiv:2502.09992*，2025年。

Erik Nijkamp、Hiroaki Hayashi、Caiming Xiong、Silvio Savarese 和 Yingbo Zhou。Code-Gen2：关于在编程语言和自然语言上训练大语言模型的经验。*arXiv* 预印本 *arXiv:2305.02309*，2023年。OpenAI。《OpenAI o3 和 o4-mini 系统卡》，2025年。网址：
h
t
t
p
s://
c
dn.openai.co
m/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf。系统卡。OpenAI、Mohammad Bavarian、Angela Jiang、Heewoo Jun 和 Henrique Pond´e。《GPT-3 的新能力：编辑与插入》。*OpenAI* 博客，2022年。网址：https://openai.com/blog/gpt-3-edit-insert/。Jingyang Ou、Shen Nie、Kaiwen Xue、Fengqi Zhu、Jiacheng Sun、Zhenguo Li 和 Chongxuan Li。你的吸收离散扩散模型实际上在建模干净数据的条件分布。国际学习表征会议，2025年。Dhruvesh Patel、Aishwarya Sahoo、Avinash Amballa、Tahira Naseem、Tim GJ Rudner 和 Andrew McCallum。插入语言模型：通过任意位置插入实现序列生成。*arXiv* 预印本 *arXiv:2505.05755*，2025年。通义千问，：An Yang、Baosong Yang、Beichen Zhang、Binyuan Hui、Bo Zheng、Bowen Yu、Chengyuan Li、Dayiheng Liu、Fei Huang、Haoran Wei、Huan Lin、Jian Yang、Jianhong Tu、Jianwei Zhang、Jianxin Yang、Jiaxi Yang、Jingren Zhou、Junyang Lin、Kai Dang、Keming Lu、Keqin Bao、Kexin Yang、Le Yu、Mei Li、Mingfeng Xue、Pei Zhang、Qin Zhu、Rui Men、Runji Lin、Tianhao Li、Tianyi Tang、Tingyu Xia、Xingzhang Ren、Xuancheng Ren、Yang Fan、Yang Su、Yichang Zhang、Yu Wan、Yuqiong Liu、Zeyu Cui、Zhenru Zhang 和 Zihan Qiu。《通义千问2.5 技术报告》，2025年。网址：https://arxiv.org/abs/2412.15115。Colin Raffel、Noam Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter J. Liu。探索统一文本到文本转换器迁移学习的极限。《机器学习研究杂志》，21：140:1–140:67，2020年。Machel Reid、Vincent Josua Hellendoorn 和 Graham Neubig。DiffusER：通过基于编辑的重构实现扩散。载于第十一届国际学习表征会议，2023年。网址：https://openreview.net/forum?id=nG9RF9z1yy3。Baptiste Roziere、Jonas Gehring、Fabian Gloeckle、Sten Sootla、Itai Gat、Xiaoqing Ellen Tan、Yossi Adi、Jingyu Liu、Tal Remez、J´er´emy Rapin 等。Code Llama：面向代码的开放基础模型。*arXiv* 预印本 *arXiv:2308.12950*，2023年。Subham Sekhar Sahoo、Marianne Arriola、Aaron Gokaslan、Edgar Mariano Marroquin、Alexan- der M Rush、Yair Schiff、Justin T Chiu 和 Volodymyr Kuleshov。简单而有效的掩码扩散语言模型。载于第三十八届神经信息处理系统年会，2024年。网址：https://openreview.net/forum?id=L4uaAR4ArM。字节跳动 Seed、Yuyu Zhang、Jing Su、Yifan Sun、Chenguang Xi、Xia Xiao、Shen Zheng、Anxiang Zhang、Kaibo Liu、Daoguang Zan 等。Seed-Coder：让代码模型为自身筛选数据。*arXiv* 预印本 *arXiv:2506.03524*，2025年。Tianxiao Shen、Victor Quach、Regina Barzilay 和 Tommi Jaakkola。空白语言模型。*arXiv* 预印本 *arXiv:2002.03079*，2020年。Tianxiao Shen、Hao Peng、Ruoqi Shen、Yao Fu、Zaid Harchaoui 和 Yejin Choi。填充式语言模型：用于任意顺序生成。*arXiv* 预印本 *arXiv:2310.09930*，2023年。Jiaxin Shi、Kehang Han、Zhe Wang、Arnaud Doucet 和 Michalis Titsias。针对离散数据的简化且泛化的掩码扩散。载于第三十八届神经信息处理系统年会，2024年。网址：https://openreview.net/forum?id=xcqSOfHt4g。Mukul Singh、Jos´e Cambronero、Sumit Gulwani、Vu Le、Carina Negreanu 和 Gust Verbruggen。CodeFusion：用于代码生成的预训练扩散模型。载于*2023*年自然语言处理实证方法会议论文集，2023年。网址：https://aclanthology.org/2023.emnlp-main.716/。

Mitchell Stern、William Chan、Jamie Kiros 和 Jakob Uszkoreit。插入变换器：通过插入操作实现灵活的序列生成。载于国际机器学习会议，第 5976–5985 页。PMLR，2019年。

Qiushi Sun、Zhirui Chen、Fangzhi Xu、Kanzhi Cheng、Chang Ma、Zhangyue Yin、Jianing Wang、Chengcheng Han、Renyu Zhu、Shuai Yuan 等。神经代码智能综述：范式、进展及未来。*arXiv* 预印本 *arXiv:2403.14734*，2024年。

Yi Tay、Jason Wei、Hyung Won Chung、Vinh Q Tran、David R So、Siamak Shakeri、Xavier Garcia、Huaixiu Steven Zheng、Jinfeng Rao、Aakanksha Chowdhery 等。以 0.1% 的额外计算量超越缩放定律。*arXiv* 预印本 *arXiv:2210.11399*，2022年。

Yi Tay、Mostafa Dehghani、Vinh Q. Tran、Xavier Garcia、Jason Wei、Xuezhi Wang、Hyung Won Chung、Dara Bahri、Tal Schuster、Steven Zheng、Denny Zhou、Neil Houlsby 和 Donald Met- zler。《UL2：统一语言学习范式》。载于第十一届国际学习表征会议，2023年。网址：https://openreview.net/forum?id=6ruVLB727MC。

Sean Welleck、Kiant´e Brantley、Hal Daum´e Iii 和 Kyunghyun Cho。非单调序列文本生成。载于国际机器学习会议，第 6716–6726 页。PMLR，2019年。

Zhihui Xie、Jiacheng Ye、Lin Zheng、Jiahui Gao、Jingwei Dong、Zirui Wu、Xueliang Zhao、Shansan Gong、Xin Jiang、Zhenguo Li 等。《Dream-coder 7b：一种面向代码的开放式扩散语言模型》。*arXiv* 预印本 *arXiv:2509.01142*，2025年。

Zhilin Yang、Zihang Dai、Yiming Yang、Jaime Carbonell、Russ R Salakhutdinov 和 Quoc V Le。XLNet：面向语言理解的广义自回归预训练。载于神经信息处理系统进展，第 32 卷，2019年。

Jiacheng Ye、Zhihui Xie、Lin Zheng、Jiahui Gao、Zirui Wu、Xin Jiang、Zhenguo Li 和 Lingpeng Kong。Dream 7B：扩散大语言模型。*arXiv* 预印本 *arXiv:2508.15487*，2025年。

Andrew Zhang、Anushka Sivakumar、Chiawei Tang 和 Chris Thomas。灵活长度的文本填充 gf或离散扩散模型。*arXiv* 预印本 *arXiv:2506.13579*，2025年。

Lin Zheng、Jianbo Yuan、Lei Yu 和 Lingpeng Kong。一种用于文本生成的重参数化离散扩散模型。*arXiv* 预印本 *arXiv:2302.05737*，2023年。

A 大语言模型的使用

我们主要使用大语言模型来润色书面文本，例如纠正语法、提升表达清晰度。然而，大语言模型并未在核心研究活动中发挥重要作用，包括构思、实验设计或论文的实质性写作。

B 代码填充之外的泛化能力

为证明 DREAMON 并不局限于代码填充，我们进一步在 ROCStories语料库 Mostafazadeh 等人（2016）上评估模型的常识叙事理解能力。我们在 ROCStories 的训练集上对 Dream-7B（Ye 等人（2025））进行微调，采用两种方案：使用 DREAMON，以及按照 Ye 等人（2025）的方案训练的监督微调基线。我们考虑两种设置：（1）**叙事填充**，模型需要在故事中间生成缺失的句子，因此必须具备双向上下文理解能力；（2）**前缀引导生成**，向模型提供前面的故事内容，并由模型生成最后的总结句，作为一般补全任务的替代评估方式。

表 4 展示了不同初始掩码长度下的 Rouge-L 分数。在填充和前缀引导设置中，当初始化掩码长度与缺失内容的自然长度不匹配时（例如长度为 4 和 32 时），基线性能都会显著下降。相比之下，DREAMON 利用其长度自适应逻辑——通过 [扩展] 和 [删除]实现——取得了更优的性能。

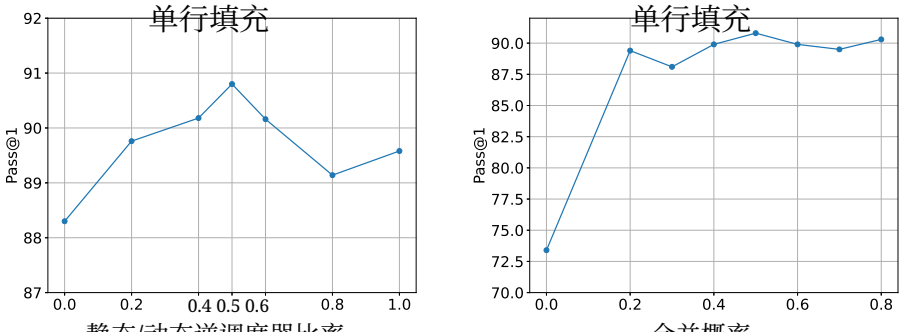
至关重要的是，我们的方法表现出很高的稳定性：无论初始掩码长度如何，生成质量都始终如一。这证实，所提出的长度适应机制成功地将生成质量与初始掩码长度解耦，展现出对富有创造性、长度可变的自然语言任务的稳健泛化能力。

表 4：ROCStories语料库在不同初始掩码长度下的 Rouge-L 得分。

方法	初始掩码长度			
	4	8	16	32
叙事填充				
Dream + 监督微调	19.2	29.8	26.5	18.9
DREAMON	**31.6**	**31.4**	**31.3**	**30.6**
故事结尾生成				
Dream + 监督微调	16.3	**25.1**	22.4	16.7
DREAMON	**24.5**	24.6	**24.4**	**24.1**

C 超参数消融实验

本节展示了 DreamCoder-7B 在不同训练超参数下的结果。如图5a所示，当采用静态与动态逆向调度器以 1:1 的比例混合时，单行填充的 Pass@1 分数达到峰值，约为 90.9%。这一结果凸显了这种协同组合带来的显著性能提升。同样，右侧面板显示，当合并概率为 0.5 时，模型的 Pass@1 分数最高，约为 90.5%。根据这些发现，我们在 DREAMON配置中采用静态与动态调度方案 1:1 的混合比例，并将合并概率设为 0.5，以最大限度地提升性能。



(a) 不同调度方案合并比例下的结果。(b) 不同合并概率下的结果 p_{merge} y

图5：训练期间采用不同超参数时，HumanEvalInfilling-FIM 单行子集上的性能。性能以掩码长度为 4、8、16、32 和 64 时 Pass@1 的平均值计算。

D 填充示例

DREAMON 的一项关键优势在于推理过程中能够自适应地处理序列长度变化。这是通过两种互补状态实现的：［扩展］和［删除］。首先，如图 6 所示，DREAMON 能够扩展掩码序列。这种动态扩展使模型能够生成长度超过初始输入掩码的输出，从而有效避免截断，并生成完整的序列。其次，图 7 展示了删除广播机制的有效性。该机制发挥着至关重要的作用



图 6：DREAMON 会按需添加掩码令牌。

通过有选择地移除冗余掩码令牌，促进序列长度快速收敛至最优预测序列长度，从而简化生成过程并提高效率。

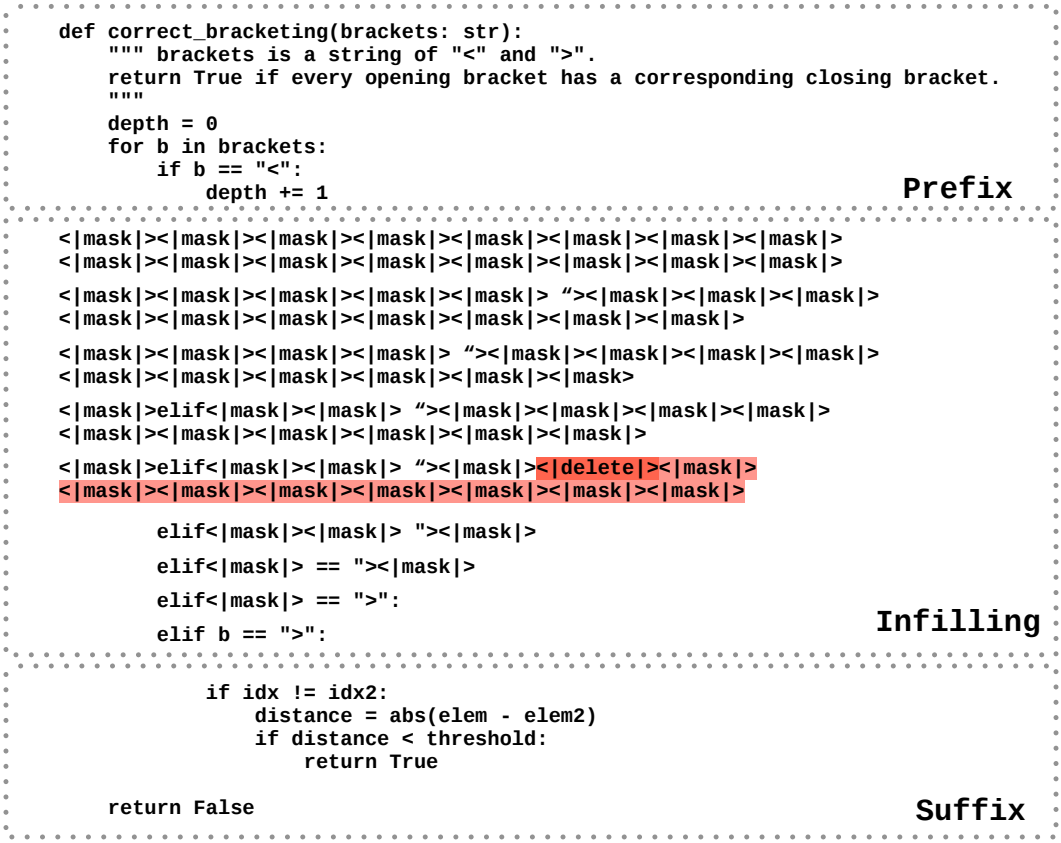


图 7：DREAMON 使用删除广播机制删除多余的掩码令牌 .