Challenging the Autoregressive Paradigm

Jiaxin Wang

July 4, 2025

This presentation briefly introduces a new approach that challenges the **autoregressive modeling (ARM)** paradigm: **masked diffusion language models**, with a focus on **LLaDA**[2] — the first large-scale implementation. For more details, please refer to the **References** at the end.

Contents

Motivation

Generative Modeling

Language models fall entirely within the framework of generative modeling. Specifically, LLMs aim to capture the true but unknown language distribution $q(x)$ by optimizing a model distribution $p_\theta(x)$ through maximum likelihood estimation (MLE), or equivalently, by minimizing the KL divergence between the two distributions:

$$\max_\theta \mathbb{E}_{q(x)} \log p_\theta(x) \Leftrightarrow \min_\theta \mathbf{KL}(q(x) \| \| p_\theta(x))$$

This is the **generative modeling principle**.

Large Language Models

With the success of Transformers, LLMs have achieved remarkable performance across a wide range of tasks. As a result, expectations have shifted — from text generation tools to the emergence of *general-purpose agents*.

Autoregressive Modeling

The predominant approach to language modeling relies on **autoregressive modeling (ARM)** — commonly referred to as the next-token prediction paradigm — to define the model distribution:

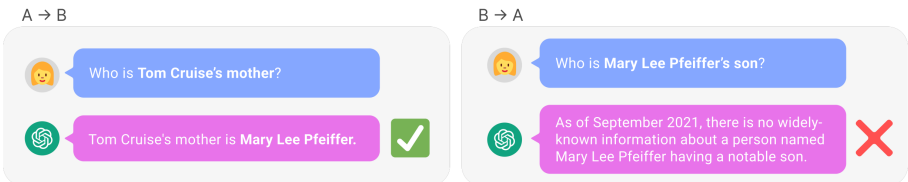
$$p_\theta(x) = p_\theta(x_1) \cdot \prod_{i=2}^{L} p_\theta(x_i \mid x_1, \ldots, x_{i-1})$$

Here, x is a token sequence of length L, and x_i is the i-th token. This formulation generates text in a strictly left-to-right fashion.

But despite its success, does ARM have to be the only choice?

The Reversal Curse[3]

If a model is trained on a sentence of the form "A is B", it will not automatically generalize to the reverse "B is A". This is the Reversal Curse.



This failure persists even with large-scale LLMs and extensive fine-tuning. ARMs lack explicit mechanisms for semantic symmetry or bidirectional reasoning.

Limitations of ARMs[1][2]

- **Sequential and Slow**
  AR models **generate text one token at a time**, always conditioned on previous tokens. This strict left-to-right order **prevents parallel generation**, making inference inherently slow and hard to scale efficiently.
- **Lack of Global Control**
  Because AR models **don't plan the full output in advance**, they struggle to enforce global structure — like fixed length, specific formats, or templates. Outputs are often free-form and difficult to constrain, which limits their usefulness in complex applications.
- **No Re-Attention or Adjustment**
  Causal attention means AR models process input once, left to right, with **no ability to look back or adjust focus**. They struggle with dynamic, task-aware attention, and often rely on costly chain-of-thought (CoT) reasoning to compensate.

The Time Has Come?[2]

The success of LLMs mainly comes from the combination of: **generative modeling principle, Transformer architecture, strong computing power, Large amounts of training data**.

ARM is helpful, but not essential. The success of diffusion transformers (DiT)[5] further supports this view.

Perhaps it's time to consider alternative approaches to language modeling.

ARM

Masked Diffusion Models

Diffusion Modeling

Denoising Diffusion Probabilistic Models (DDPM)[4] marked a turning point in **diffusion models** by demonstrating their power on high-quality image generation tasks.

It introduced a practical formulation that consists of two processes:

- **Forward Process $\Rightarrow$:** A Markov chain that gradually adds Gaussian noise to the input image over T steps, transforming it into pure noise.
- **Reverse Process $\Leftarrow$:** A learned denoising process that progressively removes noise to recover the data distribution, typically trained to predict the added noise at each step.

Why Diffusion Models?

Diffusion models are powerful generative models known for their stability, high sample quality, and controllability. They have achieved state-of-the-art results in **image**, **audio**, and **video** generation, and avoid the training instability of adversarial methods.
Diffusion Transformers (DiT)[5] further show that diffusion works well with the Transformer architecture, combining flexibility, scalability, and structured generation.
This naturally led to early efforts exploring diffusion for **text generation**.

Continuous Diffusion

Text is inherently **discrete**, making it incompatible with standard diffusion models that rely on adding Gaussian noise.

Early research explored several strategies to bridge this gap[6]:

- **Embedding-space noise:** Add Gaussian noise to continuous token embeddings, then denoise them back to text.
- **Latent diffusion:** Encode discrete text into a continuous latent space (e.g., via VAE), perform diffusion there, then decode.
- ...

However, these methods often suffer from[6]:

- **Semantic drift**: Noise moves representations off the data manifold.
- **Ambiguous decoding**: Mapping from continuous outputs back to discrete tokens is lossy and unstable.
- **Limited scalability**: Difficult to extend to high-quality, large-scale text generation.

Aside: Diffusion is Spectral Autoregression? (1/2)[7]

A recent perspective suggests that diffusion models can be interpreted as **autoregressive models in frequency space**.
In image generation, diffusion denoises from coarse to fine, mirroring autoregression—but over frequencies, not pixels. Natural images follow a **power-law frequency distribution**:

- Adding Gaussian noise acts as a low-pass filter, suppressing high-frequency components.
- Denoising then restores high-frequency details from low-frequency structure.

This explains its strong visual quality and perceptual alignment.

Aside: Diffusion is Spectral Autoregression? (2/2)[7]

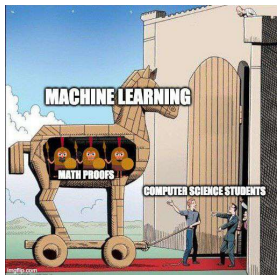
But the frequency-domain analogy does not generalize:

- **Audio waveforms**: Spectra do not follow a power law.
- **Language**: No meaningful frequency structure.

This may partly explain why diffusion models face challenges when applied directly to language modeling.

Discrete Diffusion

To address the challenges of applying continuous diffusion to text, attention has shifted toward an alternative: **discrete diffusion models**.
Unlike classical diffusion—which operates in continuous space via Gaussian noise—discrete diffusion *redefines* the process over categorical variables. It simulates forward noising and reverse denoising entirely in discrete space.
While this approach no longer fits the strict mathematical definition of diffusion, it aims to preserve key benefits.



Discrete Denoising Diffusion Probabilistic Models

D3PM[8] is the first major framework that formulates denoising diffusion directly over **discrete data** (e.g., text tokens).

The forward process in D3PM is a **Markov chain** over discrete token sequences. At each time step, the sequence

$$x_t = (x_{t,1}, x_{t,2}, \ldots, x_{t,L})$$

consists of L tokens, each chosen from a vocabulary of size K.

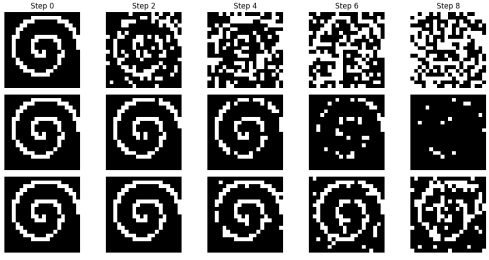
Each token is independently corrupted using a time-dependent **transition matrix** $Q_t = (Q_t[i,j]) \in \mathbb{R}^{K \times K}$: If token $x_{t-1,k} = i$, then $x_{t,k} = j$ with probability $Q_t[i,j]$. This defines a categorical corruption process over steps $t = 1$ to T. Importantly, transitions at different positions are *independent*.

The reverse model learns to denoise — that is, to predict x_{t-1} given the corrupted sequence x_t.

Designs of Q_t

D3PM supports various designs of the transition matrix Q_t, each modeling different corruption behaviors (more than the three listed below):

- **Uniform Corruption** Each token is randomly replaced by any token in the vocabulary with equal probability.
- **Absorbing Corruption** Tokens are either kept unchanged or replaced with a special `[MASK]` symbol. This mimics BERT-style masking and supports infilling-style tasks.
- **Gaussian-Like Corruption** Inspired by continuous Gaussian kernels.



From D3PM to SEDD

D3PM provides a principled formulation of discrete diffusion models, defining a corruption process over token sequences and training via the *evidence lower bound (ELBO)*.
While theoretically complete, this training objective requires computing expectations over the full diffusion trajectory, which quickly becomes *impractical at scale*.

To address this, SEDD[9] proposes a new loss, and in the absorbing corruption case — also known as the **Masked Diffusion Model** (MDM) — the formulation becomes especially simple and efficient.

SEDD: Score Entropy Discrete Diffusion models

Instead of optimizing the full ELBO, SEDD[9] introduces **Score Entropy Loss**, a new objective designed for discrete diffusion. It is inspired by score matching in continuous diffusion models.
In the absorbing case (MDM), this loss simplifies significantly: with structured transition matrices Q_{absorb}, the **Diffusion-Weighted Denoising Score Entropy (LDWDSE)** reduces to a form that is *functionally equivalent to* ***cross-entropy***, enabling the following training procedure:

- Sample a clean input x_0 from data
- Randomly choose a timestep $t \sim \mathcal{U}(0, 1)$
- Apply masking noise to obtain x_t
- Predict x_0 from x_t using cross-entropy loss

The model only needs to predict the original tokens from masked ones — making the training *efficient, scalable, and architecture-friendly*.

From SEDD to RADD

SEDD predicts the original input x_0 from a noisy version x_t, but the model **must be conditioned on the timestep** t. This requires explicit time encoding, which increases implementation complexity and makes the model *less compatible with standard Transformer architectures*.

RADD[10] removes this dependency by reparameterizing the reverse distribution:

$$p_t(x_0 \mid x_t) \propto p_0(x_0 \mid x_t) \cdot \frac{e^{-\bar{\sigma}(t)}}{1 - e^{-\bar{\sigma}(t)}}$$

Here, p_0 is a **shared, time-free model**, and $\bar{\sigma}(t) = \int_0^t \sigma(s)\, ds$ is the integrated noise schedule.

This allows us to use a *plain Transformer architecture*, shared across all corruption levels — without any time-aware components.

RADD: Reparameterized Absorbing Discrete Diffusion[10]

Training: We sample a clean sequence x_0, apply masking noise parameterized by a random timestep t to obtain x_t, and input x_t into a plain Transformer to predict x_0.

The model itself does not take t as input, but training involves sampling over different t values to cover various corruption levels. The loss is a **weighted cross-entropy** over masked positions:

$$\mathcal{L} = \mathbb{E}_{t,\, x_0,\, x_t} \left[-\frac{1}{\lambda(t)} \sum_{i \in M} \log p_0(x_{0,i} \mid x_t) \right]$$

where $\lambda(t) = 1 - e^{-\bar{\sigma}(t)}$ is the masking probability and M is the set of masked positions in x_t. This weighting comes from the analytic form of the reverse score under absorbing diffusion.

Inference: We apply the scalar to rescale logits and match the reparameterized reverse distribution.

RADD enables masked diffusion with a plain Transformer — fully reusable, time-free, and scalable.

Aside — Is MDM Still "Diffusion"?[11][6]

Removing the timestep input and using masking as corruption makes MDMs resemble **masked language models like BERT**. This has led some to question whether these models are still "diffusion", or simply MLMs in disguise.

But diffusion is not defined by Gaussian noise or time embeddings — it is defined by the generative reversal of a progressive corruption process. Even without explicit time inputs, MDMs like RADD are trained to learn the entire reverse trajectory — not just fill in a few masked tokens.

Whether we call it "diffusion" is beside the point. *What matters is that MDMs inherit many of the practical advantages of diffusion — and reframe them into a scalable, autoregressive-free generation paradigm.*

LLaDA

From Theory to Practice

D3PM[8] laid the foundation for discrete diffusion with a formal corruption-recovery process, but relied on ELBO-based objectives that were computationally intensive and difficult to scale.

SEDD[9] proposed a simplified training loss under absorbing corruption, enabling efficient and scalable optimization via a cross-entropy-style objective.

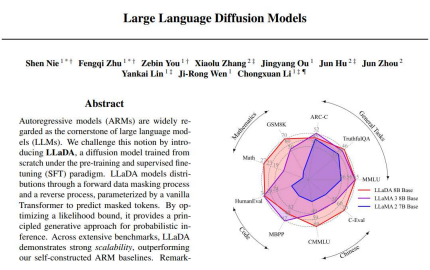
RADD[10] further removed the need for timestep inputs, enabling time-agnostic training with standard Transformer architectures.

Together, these developments paved the way for scalable diffusion-based language models — and laid the groundwork for LLaDA.

Then we will introduce the original paper [2] in this part.

Overview

LLaDA (Large Language Diffusion with mAsking)[2] is the first large-scale alternative to autoregressive LLMs based on MDM.
It uses a standard decoder-only Transformer **without causal masking**, trained to iteratively denoise masked token sequences.
A live demo is available at:
`https://huggingface.co/spaces/multimodalart/LLaDA`.



Large Language Diffusion Models

Shen Nie[1*†] Fengqi Zhu[1*†] Zebin You[1†] Xiaolu Zhang[2‡] Jingyang Ou[1] Jun Hu[2‡] Jun Zhou[2]
Yankai Lin[1‡] Ji-Rong Wen[1] Chongxuan Li[1‡¶]

Abstract

Autoregressive models (ARMs) are widely regarded as the cornerstone of large language models (LLMs). We challenge this notion by introducing **LLaDA**, a diffusion model trained from scratch under the pre-training and supervised fine-tuning (SFT) paradigm. LLaDA models distributions through a forward data masking process and a reverse process, parameterized by a vanilla Transformer to predict masked tokens. By optimizing a likelihood bound, it provides a principled generative approach for probabilistic inference. Across extensive benchmarks, LLaDA demonstrates strong *scalability*, outperforming our self-constructed ARM baselines. Remark-

Probabilistic Formulation

LLaDA adopts a model setup similar to RADD: given a clean sequence x_0, each token is independently masked with probability $t \sim \mathcal{U}(0, 1)$, producing the corrupted input x_t.
A **Transformer-based mask predictor** $p_\theta(\cdot \mid x_t)$ then predicts all masked tokens simultaneously — without timestep input.

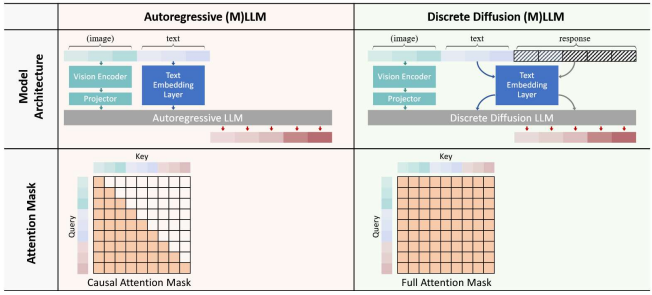
Training minimizes a cross-entropy loss over masked positions:

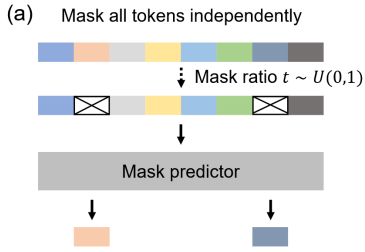
$$\mathcal{L}(\theta) = \mathbb{E}_{t,\, x_0,\, x_t} \left[-\frac{1}{t} \sum_{i \in M} \log p_\theta(x_{0,i} \mid x_t) \right]$$

Here, $\frac{1}{t}$ is a practical approximation of RADD's $\frac{1}{\lambda(t)} = \frac{1}{1 - e^{-\bar{\sigma}(t)}}$. Intuitively, earlier timesteps are more corrupted and harder to denoise.

This training objective upper-bounds the negative log-likelihood of $p_\theta(x_0)$, making LLaDA a **principled generative model** — unlike BERT-style masked predictors.

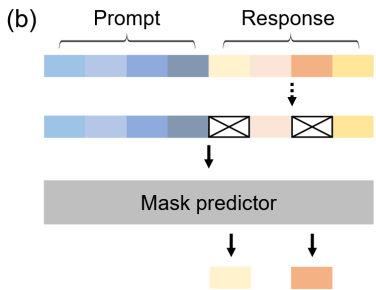
Pre-Training





Training uses 2.3T tokens, sequence length 4096, and standard optimization (AdamW, large-batch SGD, Warmup-Stable-Decay schedule).

Supervised Fine-Tuning (SFT)

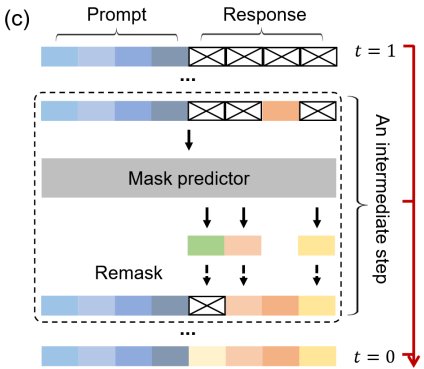


To enable instruction following, LLaDA is fine-tuned on 4.5M **prompt-response pairs** (p_0, r_0). The learning objective shifts to modeling the conditional distribution $p_\theta(r_0 \mid p_0)$.

During SFT, the prompt p_0 is **kept unmasked**, while tokens in the response r_0 are **independently masked** with a sampled ratio $t \sim \mathcal{U}(0, 1)$. The mask predictor then recovers the masked tokens conditioned on the full prompt.

This setup is fully **compatible with pre-training**: it reuses the same architecture and objective — only the masking pattern is constrained to the response.

Inference



LLaDA generates responses by **iteratively denoising** a fully masked sequence, conditioned on a given prompt. Starting from $t = 1$ where all response tokens are masked, the model uses its **mask predictor** to fill in masked tokens at each timestep. A subset of these predicted tokens is then **remasked**, gradually reducing uncertainty.
This process repeats until $t = 0$, yielding the final output.

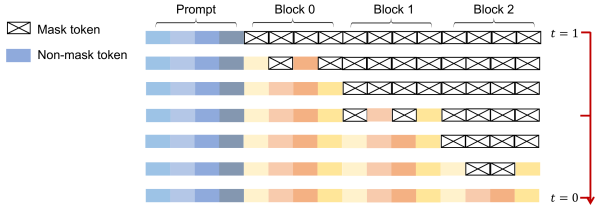
|EOS|

A key limitation of MDMs is the need to *fix sequence length before generation*. This makes it nontrivial to produce responses of varying length, as is common in instruction following.

To address this, LLaDA introduces an **end-of-sequence token** |EOS| during SFT:

- **Training:** |EOS| is appended to short responses in each mini-batch to align sequence lengths. These tokens are treated as part of the target: masked, predicted, and included in the loss.
  `What is the capital of France? Paris.`
  ⇒ `What is the capital of France? Paris. |EOS| |EOS| |EOS|`
- **Inference:** |EOS| is used as a stopping signal — generation halts when it is predicted, enabling the model to control output length dynamically.
  `The capital of France is Paris. |EOS| Thank you very much.`
  ⇒ `The capital of France is Paris.`

Semi-Autoregressive Generation: Block-wise Decoding



To improve generation quality, LLaDA adopts a **semi-autoregressive decoding strategy** at inference time.

The response sequence is divided into fixed-length blocks (e.g., 32 tokens), which are generated **sequentially from left to right**, with the diffusion denoising process applied independently within each block.

This simple schedule aligns well with the hierarchical structure of instruction data, and enables more natural control over output length. Compared to fully autoregressive decoding, this strikes a *balance* between efficiency and coherence.

Remasking Strategies

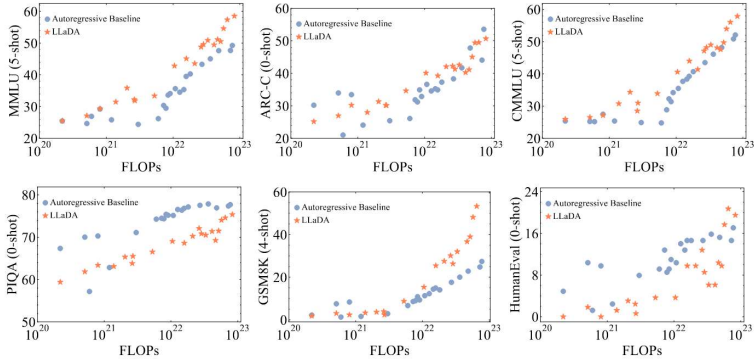
At each step in the reverse process, LLaDA predicts all masked tokens — but only keeps a subset, **remasking the rest** to continue denoising. This remasking ensures that the generation process mimics the forward corruption, maintaining training-inference consistency.

Two strategies are used to select which tokens to remask:

- **Random remasking:** Randomly remask a portion of the newly predicted tokens so that the overall masking ratio gradually decreases step by step — consistent with the original definition of diffusion.
- **Low-confidence remasking:** Choose tokens with the lowest predicted confidence (i.e., softmax probability of the selected token) to remask — improves stability and sample quality.

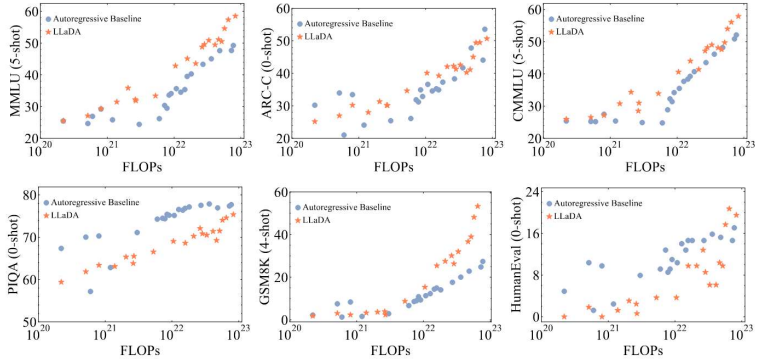
Best performance is achieved by combining both: Apply **low-confidence remasking within each block**, and **generate blocks semi-autoregressively** from left to right.

Experiments 1: Scalability (1/2)



To ensure fair comparison, the **ARM baselines** were constructed with matching configurations: at the 1B scale, LLaDA and ARM use the same architecture, data, and training setup; at larger scales, both were trained on the same dataset with similar model sizes and computational budgets.

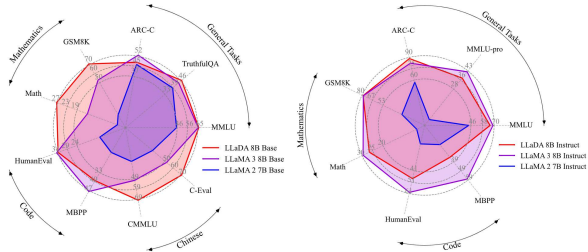
Experiments 1: Scalability (2/2)



Computational **FLOPs** are used as a unified metric to compare scalability. As shown, *LLaDA performs competitively across six diverse tasks*, surpassing ARMs on **MMLU** and **GSM8K**, and gradually closing the gap on others like **PIQA**.

Experiments 2: Benchmark (1/2)

LLaDA 8B Base is trained purely via masked diffusion pretraining on 2.3T tokens.
LLaDA 8B Instruct adds SFT on 4.5M instruction pairs — *without reinforcement learning or preference tuning*.
LLaDA Base outperforms LLaMA2 7B and rivals LLaMA3 8B on tasks like GSM8K, TruthfulQA, and CMMLU.
LLaDA Instruct matches LLaMA3 Instruct across most instruction-following benchmarks.



Experiments 2: Benchmark (2/2)

Compared with other open ARMs such as Qwen2.5, Gemma2, and DeepSeek, LLaDA remains highly competitive—even though it is **trained with fewer tokens** and **only SFT without RL alignment**.

	LLaDA 8B*	LLaMA3 8B*	LLaMA2 7B*	Qwen2 7B†	Qwen2.5 7B†	Gemma2 9B†	Deepseek 7B¶
Model	Diffusion	AR	AR	AR	AR	AR	AR
Training tokens	2.3T	15T	2T	7T	18T	8T	2T
Post-training	SFT	SFT+RL	SFT+RL	SFT+RL	SFT+RL	SFT+RL	SFT+RL
Alignment pairs	4.5M	-	-	0.5M + -	1M + 0.15M	-	1.5M + -
General Tasks							
MMLU	65.5 (5)	**68.4** (5)	44.1 (5)	-	-	-	49.4 (0)
MMLU-pro	37.0 (0)	**41.9** (0)	4.6 (0)	44.1 (5)	56.3 (5)	52.1 (5)	-
Hellaswag	74.6 (0)	**75.5** (0)	51.5 (0)	-	-	-	68.5 (-)
ARC-C	**88.5** (0)	82.4 (0)	57.3 (0)	-	-	-	49.4 (-)
Mathematics & Science							
GSM8K	**78.6** (4)	78.3 (4)	29.0 (4)	85.7 (0)	91.6 (0)	76.7 (0)	63.0 (0)
Math	26.6 (0)	**29.6** (0)	3.8 (0)	52.9 (0)	75.5 (0)	44.3 (0)	15.8 (0)
GPQA	31.8 (5)	**31.9** (5)	28.4 (5)	34.3 (0)	36.4 (0)	32.8 (0)	-
Code							
HumanEval	47.6 (0)	**59.8** (0)	16.5 (0)	79.9 (0)	84.8 (0)	68.9 (0)	48.2 (-)
MBPP	34.2 (4)	**57.6** (4)	20.6 (4)	67.2 (0)	79.2 (0)	74.9 (0)	35.2 (-)

Experiments 3: Reversal Reasoning

User	'但闻人语响'的上一句是什么?
GPT-4o	"但闻人语响"的上一句是"无可奈何花落去",这两句出自唐代词人李清照的（武陵春，春晚: 无可奈何花落去，但闻人语响。这首词表达了作者在春天时节的孤独与感伤，借花落和人语声的描写，反映出词人内心的无奈与失落。
LLaDA	"但闻人语响"的上一句是"空山不见人"

	Forward	Reversal
GPT-4o (2024-08-06)	**82.7**	34.3
Qwen2.5 7B Instruct	75.9	38.0
LLaDA 8B Instruct	48.8	**42.4**

A **Chinese poem completion task** is used to evaluate reversal reasoning, where models generate either the **next** or **previous** line in a zero-shot setting.

LLaDA shows slightly lower accuracy on forward generation, as other models benefit from *larger datasets and greater computational resources*.

However, it significantly outperforms Qwen2.5 and GPT-4o on reversal generation, and achieves **balanced performance in both directions**. Intuitively, LLaDA treats tokens uniformly *without* inductive bias.

Conclusion

"What is now proved was once only imagined." — William Blake

LLaDA introduces a scalable and principled framework for language modeling based on **MDM**. It achieves strong performance across instruction following, in-context learning, and reasoning tasks—without autoregressive decoding.

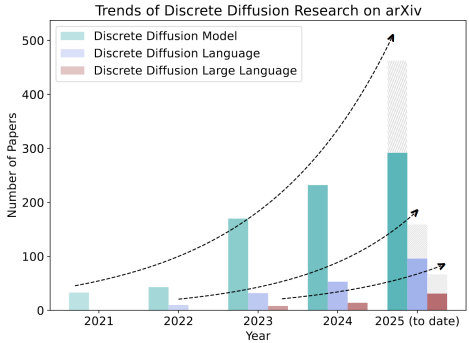
Key advantages include **bidirectional generation**, **robustness**, and **a simple, reusable architecture**. LLaDA demonstrates that diffusion can serve as a viable alternative to ARMs at scale.

Limitations remain in **alignment** and **inference robustness**. LLaDA currently exhibits sensitivity to inference hyperparameters such as sampling steps and block size, and has not undergone RL-based tuning or architectural optimization.

Future

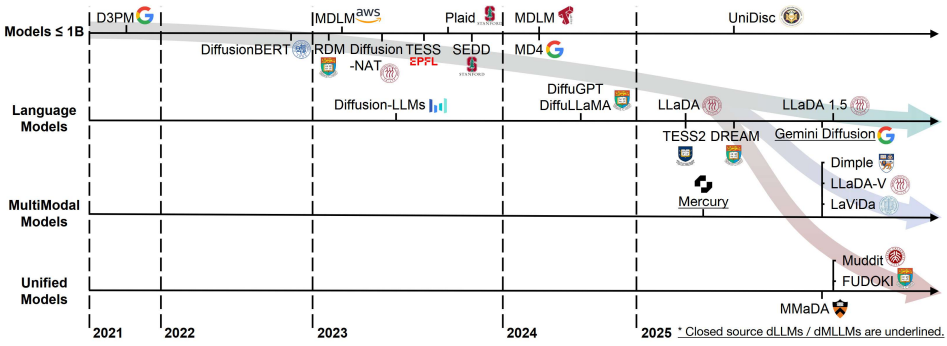
Trend[1]

As a promising alternative to autoregressive language models, **Discrete Diffusion Language Models (dLLMs)** have gained increasing attention, and the field is becoming more active.



Timeline[1]

Since the introduction of **LLaDA**, the field of discrete diffusion language modeling has rapidly evolved, with a growing ecosystem of **LLaDA-based extensions**, **parallel approaches**, and **industry-scale implementations**.



Language Model Improvements

LLaDA 1.5[12]: Introduces *Variance-Reduced Preference Optimization* (VRPO), an alignment algorithm tailored for MDMs.

- Addresses the high variance of ELBO estimates in preference optimization (e.g., DPO)
- Allocates Monte Carlo budget optimally and balances timestep sampling
- Applies antithetic sampling for unbiased variance reduction
- Significantly improves alignment and math performance over SFT-only LLaDA

DREAM[13]: Optimizes dLLM training by leveraging autoregressive priors and adaptive supervision.

- Initializes from pretrained AR models to retain left-to-right inductive bias
- Uses token-wise noise scheduling based on context availability
- Trains faster and generalizes better, especially for long-form generation

These refinements enhance alignment and reasoning while preserving diffusion's core advantages.

Multimodal Extensions

LaViDa[14]: A vision-language model combining a SigLIP image encoder with a discrete diffusion language decoder.

- Splits input images into five views and projects them into token space
- Uses *complementary masking* and *prefix caching* for bidirectional generation
- Enables fast and controllable multimodal generation

LLaDA-V[15]: Extends LLaDA with joint image-text diffusion modeling.

- Denoises visual-text pairs with a unified masked diffusion objective
- Trained on instruction-following multimodal datasets
- Achieves competitive results without relying on AR decoding

These models show that LLaDA-style diffusion naturally extends to vision-language generation.

MMaDA: Multimodal Large Diffusion Language Models

MMaDA[16] is a large-scale unified diffusion model that supports modality-agnostic reasoning and generation. Key innovations include:
1) Unified Diffusion Backbone: A single Transformer handles all modalities (text, vision, etc.) under a shared probabilistic formulation, removing modality-specific components and improving scalability.
2) Cross-modal CoT Alignment: Unifies reasoning across text and vision into a single chain-of-thought (CoT) format, enabling joint training and multimodal inference from a shared input space.
3) Unified Reinforcement Learning (UniGRPO): Introduces a modality-agnostic policy gradient algorithm for diverse reward modeling, improving consistency and stability in post-training alignment.

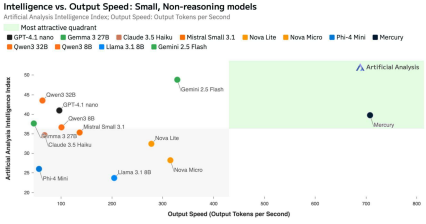
Live Demo: `https://huggingface.co/spaces/Gen-Verse/MMaDA`
Supports text generation, multimodal reasoning, and text-to-image generation.

Mercury

Mercury[17], developed by Inception Labs, is introduced as the fastest *commercial-grade* dLLM. Unlike theoretical prototypes, it is engineered for real-time use with a system-optimized architecture and decoding pipeline.

Despite its non-autoregressive nature, Mercury achieves decoding speeds over **1000 tokens/sec**, surpassing Gemini 2.5 Flash in speed while retaining strong task performance.

A public demo is available at: `https://chat.inceptionlabs.ai/`.



Gemini Diffusion

Gemini Diffusion[18] is an experimental language model developed by Google DeepMind. Unlike traditional autoregressive models that generate text one token at a time, it adopts a **diffusion-based approach**. This enables faster generation and supports iterative correction during decoding.

The model achieves a sampling speed of **1479 tokens/sec**, and its benchmark performance is comparable to much larger models. It shows strong results on text and code tasks, producing more coherent outputs and significantly faster responses.

https://deepmind.google/models/gemini-diffusion/

Aside: Why So Fast?

Mercury[17] (Inception Labs) and **Gemini Diffusion**[18] (Google DeepMind) both demonstrate impressive decoding speeds, outperforming strong autoregressive baselines by a wide margin.

What makes them fast? While full technical details remain undisclosed, likely contributing factors include (thanks to discussions in the official MMaDA WeChat group):

- *Step reduction:* fewer denoising steps via block-wise masking or early halting
- *Efficient attention:* sparse attention or lightweight masking strategies
- *System-level optimizations:* parallel decoding, KV cache reuse, and custom inference runtimes

These results show that diffusion-based LLMs can be both **fast** and **production-ready** — paving the way for real-time, autoregressive-free generation.

Challenges[1][6]

1. Length Bias
Diffusion models generate outputs of fixed length, which must be set before decoding starts. This makes variable-length generation difficult and often leads to truncated or padded outputs.

2. Training Inefficiency
Each training step only updates a subset of tokens via random masking, resulting in low data efficiency and slow convergence.

3. Slow Inference
Despite being parallelizable, diffusion decoding requires many steps with full attention, — making it slower than one-pass AR decoding — though recent models like Mercury have begun to overcome this limitation.

Directions[1]

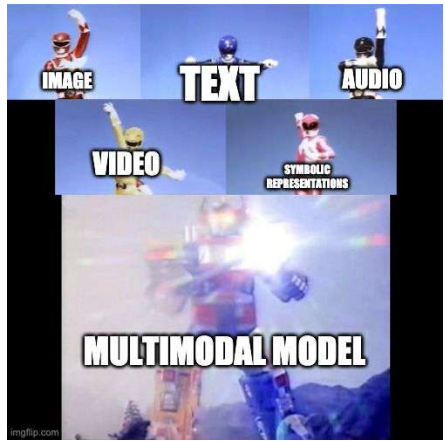
1. Bridging with AR Techniques
Adopting AR-derived techniques — such as instruction tuning, weight initialization, and RLHF — can improve training dynamics and downstream alignment for diffusion models.

2. Diffusion-native Optimization
To fully realize the potential of dLLMs, we need architectures tailored to the diffusion process — not simply borrowed from AR.

3. Multimodal and Task-specific Extensions
Diffusion's flexible corruption-reconstruction framework lends itself to complex input modalities (e.g., vision-language) and structured tasks such as step-by-step reasoning or code editing.

Some Thoughts

- Human language generation is not strictly autoregressive — our thoughts are formed iteratively, non-linearly, and often with partial information. Diffusion modeling may better reflect this process.
- Despite recent breakthroughs, the dLLM ecosystem is still in its early stages. Core components remain underdeveloped.
- With the rise of models like MMaDA, a unified, multimodal framework may become feasible in the near future.



Thank you!

References I

[1] Runpeng Yu *et al. Discrete Diffusion in Large Language and Multimodal Models: A Survey*. arXiv preprint arXiv:2506.13759, 2025.

[2] Shen Nie *et al. Large Language Diffusion Models*. arXiv preprint arXiv:2502.09992, 2025.

[3] Lukas Berglund *et al. The Reversal Curse: LLMs trained on "A is B" fail to learn "B is A"*. arXiv preprint arXiv:2309.12288, 2024.

[4] Jonathan Ho *et al. Denoising Diffusion Probabilistic Models*. arXiv preprint arXiv:2006.11239, 2020.

[5] William Peebles *et al. Scalable Diffusion Models with Transformers*. arXiv preprint arXiv:2212.09748, 2023.

References II

[6] Chongxuan Li *Answer to "How do you evaluate Google's newly released Gemini Diffusion?"*. Zhihu, https://www.zhihu.com/question/1908479621466396378/answer/1910672718174589774, edited on May 28, 2025, Beijing.

[7] Sander Dieleman *et al. Diffusion Is Spectral Autoregression*. Sander.ai blog, September 2, 2024.

[8] Jacob Austin *et al. Structured Denoising Diffusion Models in Discrete State-Spaces*. arXiv preprint arXiv:2107.03006, 2023.

[9] Aaron Lou *et al. Discrete Diffusion Modeling by Estimating the Ratios of the Data Distribution*. arXiv preprint arXiv:2310.16834, 2024.

[10] Jingyang Ou *et al. Your Absorbing Discrete Diffusion Secretly Models the Conditional Distributions of Clean Data*. arXiv preprint arXiv:2406.03736, 2025.

References III

[11] Kaiwen Zheng *et al. Masked Diffusion Models are Secretly Time-Agnostic Masked Models and Exploit Inaccurate Categorical Sampling*. arXiv preprint arXiv:2409.02908, 2025.

[12] Fengqi Zhu *et al. LLaDA 1.5: Variance-Reduced Preference Optimization for Large Language Diffusion Models*. arXiv preprint arXiv:2505.19223, 2025.

[13] Jiacheng Ye *et al. Dream 7B: The Most Powerful Open Diffusion Language Model to Date*. HKU NLP Group blog, April 2, 2025.

[14] Shufan Li *et al. LaViDa: A Large Diffusion Language Model for Multimodal Understanding*. arXiv preprint arXiv:2505.16839, 2025.

[15] Zebin You *et al. LLaDA-V: Large Language Diffusion Models with Visual Instruction Tuning*. arXiv preprint arXiv:2505.16933, 2025.

References IV

[16] Ling Yang *et al. MMaDA: Multimodal Large Diffusion Language Models*. arXiv preprint arXiv:2505.15809, 2025.

[17] Inception Labs *et al. Mercury: Ultra-Fast Language Models Based on Diffusion*. arXiv preprint arXiv:2506.17298, 2025.

[18] Google DeepMind *et al. Gemini Diffusion: A State-of-the-Art Text Diffusion Model*. Google DeepMind blog, May 20, 2025.