Large Language Diffusion Models

LLaDA

NeurIPS 2025 Nie, Zhu, You, Zhang et al.
Renmin University of China & Ant Group

Presented by

Jingyi Feng & Tarun Mandapati

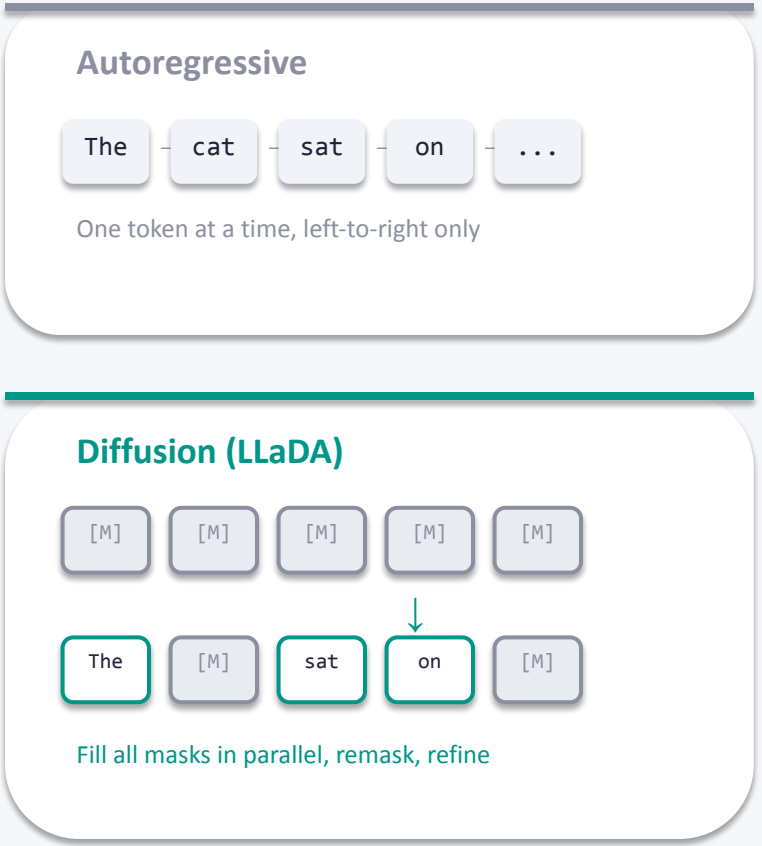
CS 8803-LLM

Do LLMs have to be "next-token" models?

Most LLMs: predict next token left-to-right

Some tasks need bidirectional context

Can we keep capabilities with a different generative principle?



Limitations of Autoregressive Models

Left-to-Right Only

I love [???]

Can never see future context
→ sequential bottleneck

Reversal Curse

A→B ✓ B→A
✗

"A is B" works but the
inverse "B is A" fails

Fixed Inference

1 token / step

Must decode token-by-token
No speed ↔ quality tradeoff

LLaDA vs Autoregressive Language Modeling

Generative modeling principles, not the AR formulation, underpin the capabilities of LLMs.

Autoregressive (AR)

$$p_\theta(x_0) = \prod_{i=1}^{L} p_\theta(x^i \mid x^1, \ldots, x^{i-1})$$

Chain-rule factorization

One way to decompose joint distribution

Causal attention ⟶ left-to-right only

VS

LLaDA (Masked Diffusion)

$$p_\theta(x_0) \text{ via } x_1 \longrightarrow x_{t-1} \longrightarrow \cdots \longrightarrow x_0$$

Diffusion-based factorization

Another valid way to model p(x)

Bidirectional attention ⟶ full context

From Discrete Diffusion to LLaDA

Evolution of diffusion language models

2021 — **D3PM [16]**

Austin et al.

Structured Denoising Diffusion Models in Discrete State-Spaces

First discrete diffusion for language modeling

2023–2024 — **MDLM [17] SEDD [18]**

Lou et al. / Shi et al.

Discrete Diffusion LM by Estimating the Ratios (MDLM) + Simplified and Generalized Masked Diffusion (SEDD)

Masked diffusion matches AR at GPT-2 scale + ELBO theory

2024 — **Scaling MDM [27]**

Nie et al.

Scaling Up Masked Diffusion Models on Text

Scaling laws for MDMs; GPT-2 scale QA tasks

2025 — **LLaDA**

Nie, Zhu, You et al.

Large Language Diffusion Models (this paper)

8B params from scratch; first competitive non-AR LLM

LLaDA = Masked Diffusion Language Model

1 Forward Process

Corrupt text by random masking

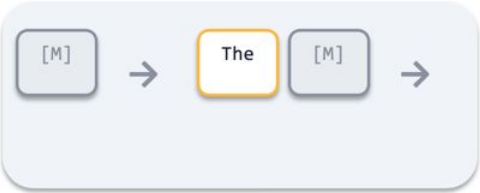
t = 0 → t = 1

2 Mask Predictor

Transformer predicts all [M] in parallel

Bidirectional

3 Reverse Process



Iterate: predict → remask → refine

t = 1 → t = 0

Same pipeline as LLMs: Pre-train → SFT → Sampling

LLaDA vs BERT: Why Masking Ratio Matters

BERT

Fixed 15% masking

Representation learning only

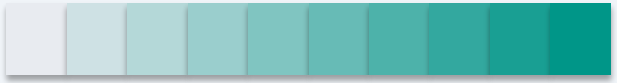
Cannot do iterative unmasking

Heuristic — no likelihood bound

VS

LLaDA

$t \sim \text{Uniform}(0, 1)$



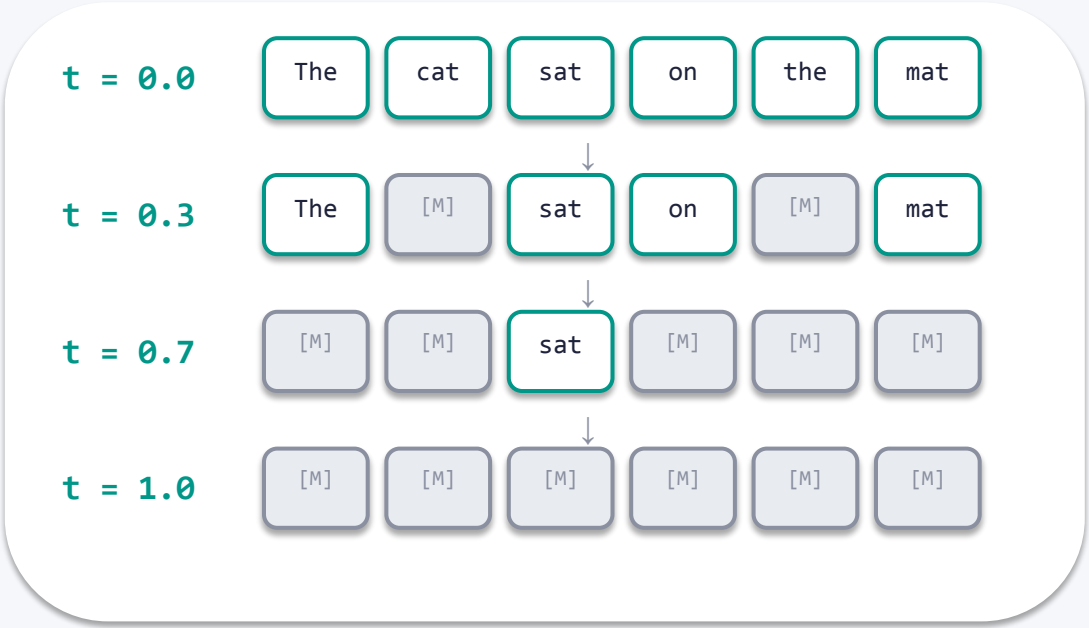
Designed for generation

Trains on all masking ratios

Principled likelihood lower bound

Forward Process: Random Masking

Each token independently masked with probability t



Transition Kernel

$$q(x_t^i \mid x_0^i) = \begin{cases} x_0^i & \text{w. p. } 1-t \\ \mathbf{M} & \text{w. p. } t \end{cases}$$

$t \sim \text{Uniform}(0, 1)$

Model sees every masking ratio during training

Training: Learn to Fill Missing Tokens

1 Sample text

$x_0 =$ "The cat sat ..."

→

2 Mask it

$t \sim U(0,1),\ \text{mask} \to x_t$

→

3 Predict masked

`Transformer predicts only masked tokens`

Training Loss (Eq. 3)

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

Expectation over: $t \sim U(0,1)$, $x_0 \sim$ data, $x_t \sim$ forward process

- $\frac{1}{t}$ — Normalizes by expected number of masked tokens
- $1[x_t^i = \mathbf{M}]$ — Loss only computed on masked positions
- $p_\theta(x_0^i \mid x_t)$ — Transformer's prediction for each masked token

Theoretical Justification

Core Result (Eq. 4)

$$-\mathbb{E}_{p_{\text{data}}(x_0)}[\log p_\theta(x_0)] \leq \mathcal{L}(\theta)$$

Minimizing the training loss maximizes a variational lower bound on the log-likelihood

✓ Principled

Direct link to maximum likelihood estimation, not a heuristic like MaskGIT (misses 1/t)

≡ Proper Generative Model

Defines a real probability distribution — enables fair comparison with autoregressive LLMs

↗ Scalability

Fisher consistency + Transformer scaling = same ingredients that make AR models scale

Pre-training Setup

8B
Parameters

4096
Seq Length

2.3T
Tokens

Architecture

Standard Transformer, bidirectional attn
Vanilla MHA (no GQA, no KV cache)
Matches LLaMA3-8B scale

Data & Design Choices

Same data protocol as LLMs — no special tricks
Filtered online corpora + code, math, multilingual
Mixing guided by scaled-down ARMs
1% data truncated to [1, 4096] for variable length

Training Recipe

LR Schedule (WSD)

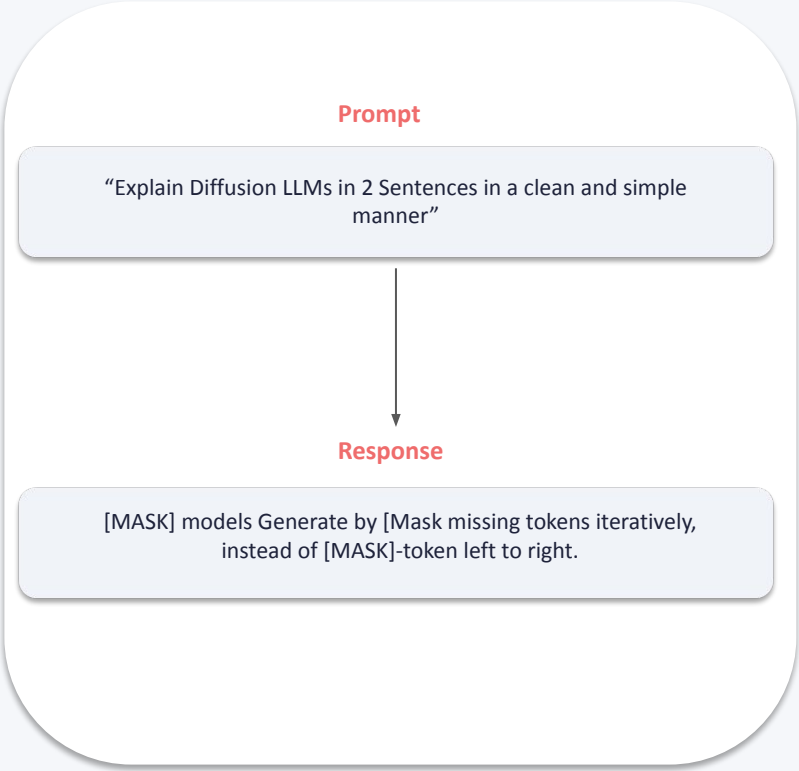
Warmup	`0` → `4e-4`	2K iters
Stable	`4e-4`	to 1.2T tok
Decay	`4e-4` → `1e-4`	0.8T tok
Final	`1e-4` → `1e-5`	0.3T tok

Hyperparameters

Optimizer	AdamW, wd = 0.1
Batch	1280 global
Compute	0.13M H800 hrs
Runs	Single, no HP tuning

SFT: Teach it to Follow Instructions

- Same idea as pre-training, but now we have (prompt → response) pairs
- The model knows a lot about language, but it doesn't know how to follow instructions yet. It can't answer questions properly.
- Keep the prompt visible; randomly mask ONLY the response tokens
- Model learns to complete an answer conditioned on the prompt
- The exact same masking trick from pre-training, just applied only to the answer side.



SFT: Implementation Details

Dataset	4.5 million (prompt, response) pairs
Domains	Code, math, instruction-following
Epochs	3 epochs on SFT data
|EOS| Handling	Append |EOS| for equal-length batches; remove during sampling
LR Schedule	Warmup ⟶ constant ⟶ linear decay (last 10%: reduce to 2.5×10^{-6})
Batch Size	256 global (2 per GPU)
Optimizer	AdamW, weight decay 0.1
RL Alignment	Not applied (left for future work)

Key point: SFT uses the same protocol as existing LLMs. There are no “special” techniques needed for diffusion.

Sampling: Parallel Fill → Remask → Refine

- Start from an all-[MASK] response (length chosen by user)
- Each step predicts ALL masked tokens in parallel
- Optionally "remask" low-confidence tokens and try again
- Number of steps is a speed means this can aact as a quality knob (typically 16–256 steps)

t = 1	[MASK]	[MASK]	[MASK]	[MASK]	[MASK]
step 1	This	[MASK]	generates	text	.
step 2	This	model	generates	text	.
step 3	This	model	generates	text	well.

Remask "uncertain" tokens ⟶ fixes errors over steps

Flexible Sampling Modes

Diffusion	Default — best quality	**Autoregressive**	Left-to-right compatible	**Block Diffusion**	Hybrid approach

Sampling Strategies Ablation

Table 7: **Ablation on Sampling Strategies for LLaDA 8B Base.** L' is the block length. Pure diffusion sampling achieves the best overall performance.

		BBH	GSM8K	Math	HumanEval	MBPP
Autoregressive		38.1	63.1	23.6	18.3	33.4
Block Difusion	$L' = 2$	37.3	62.6	25.2	14.6	33.6
	$L' = 4$	40.0	65.7	26.6	15.9	36.0
	$L' = 8$	42.0	68.2	27.7	19.5	39.2
	$L' = 32$	45.7	68.6	29.7	29.9	37.4
Block Diffusion LLaDA	$L' = 2$	48.0	70.0	30.8	26.2	**40.0**
	$L' = 4$	48.5	**70.3**	31.3	27.4	38.8
	$L' = 8$	48.6	70.2	30.9	31.1	39.0
	$L' = 32$	48.3	**70.3**	31.2	32.3	**40.0**
Pure Diffusion		**49.7**	**70.3**	**31.4**	**35.4**	**40.0**

Table 8: **Ablation on Sampling Strategies for LLaDA 8B Instruct.** The block length is set to 32 for efficiency. Pure diffusion sampling achieves the best overall performance.

	GSM8K	Math	HumanEval	MBPP	GPQA	MMLU-Pro	ARC-C
Autoregressive	0	9.5	0	0	0	0	84.4
Block Diffusion	24.6	23.5	17.1	21.2	29.3	32.5	88.1
Block Difusion LLaDA	**77.5**	**42.2**	46.3	34.2	31.3	34.8	85.4
Pure Diffusion	69.4	31.9	**49.4**	**41.0**	**33.3**	**37.0**	**88.5**

Interface / Sampling

The Reverse Process

- Generation starts from a fully masked sequence of length L with every token [M] at t=1
- At each step the mask predictor fills all masked positions simultaneously, then remaskes a fraction of them to obtain the next noisier state
- This continues from t=1 down to t=0, at which point all tokens are unmasked and the response is complete
- The number of diffusion steps is a free hyperparameter more steps give better quality, fewer steps give faster generation, with no retraining required

Two Remasking Strategies

Random Remasking

A random s/t fraction of predicted tokens are sent back to [M] at each step — theoretically principled but high variance

Low-confidence remasking

The s/t tokens the model was least confident about are remasked — the model keeps its best predictions and revisits uncertain ones

Scalability: LLaDA Scales Like ARMs

- Evaluated on 6 standard benchmarks (MMLU, ARC-C, GSM8K, PIQA, CMMLU, HumanEval)
- LLaDA and ARM baselines trained on the same data with increasing compute (FLOPs)
- LLaDA matches the overall scaling trend of ARMs across all tasks
- On MMLU and GSM8K, LLaDA shows even stronger scalability than ARMs

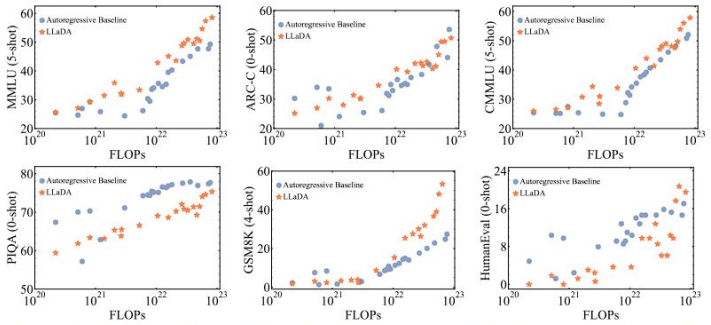


Figure 3: **Scalability of LLaDA.** We evaluate the performance of LLaDA and our ARM baselines trained on the same data across increasing pre-training computational FLOPs. LLaDA exhibits strong scalability, matching the overall performance of ARMs on six tasks.

Benchmark Results: Pre-trained Models (Base)

- LLaDA 8B Base surpasses LLaMA2 7B Base on nearly all 15 standard tasks
- Overall competitive with LLaMA3 8B Base which is remarkable for a diffusion model
- Particularly strong on math (+45% over LLaMA3) and Chinese tasks (+38%)
- Trained on only 2.3T tokens (vs 15T for LLaMA3) — 6.5× less data

Table 1: **Benchmark Results of Pre-trained LLMs.** * indicates that models are evaluated under the same protocol, detailed in Appendix B.6. Results indicated by † and ¶ are sourced from Yang et al. [25, 26] and Bi et al. [32] respectively. The numbers in parentheses represent the number of shots used for in-context learning. "-" indicates unknown data.

	LLaDA 8B*	LLaMA3 8B*	LLaMA2 7B*	Qwen2 7B†	Qwen2.5 7B†	Mistral 7B†	Deepseek 7B¶
Model	Diffusion	AR	AR	AR	AR	AR	AR
Training tokens	2.3T	15T	2T	7T	18T	-	2T
General Tasks							
MMLU	**65.9** (5)	65.4 (5)	45.9 (5)	70.3 (5)	74.2 (5)	64.2 (5)	48.2 (5)
BBH	49.7 (3)	**62.1** (3)	39.4 (3)	62.3 (3)	70.4 (3)	56.1 (3)	39.5 (3)
ARC-C	45.9 (0)	**53.1** (0)	46.3 (0)	60.6 (25)	63.7 (25)	60.0 (25)	48.1 (0)
Hellaswag	70.5 (0)	**79.1** (0)	76.0 (0)	80.7 (10)	80.2 (10)	83.3 (10)	75.4 (0)
TruthfulQA	**46.1** (0)	44.0 (0)	39.0 (0)	54.2 (0)	56.4 (0)	42.2 (0)	-
WinoGrande	74.8 (5)	**77.3** (5)	72.5 (5)	77.0 (5)	75.9 (5)	78.4 (5)	70.5 (0)
PIQA	73.6 (0)	**80.6** (0)	79.1 (0)	-	-	-	79.2 (0)
Mathematics & Science							
GSM8K	**70.3** (4)	48.7 (4)	13.1 (4)	80.2 (4)	85.4 (4)	36.2 (4)	17.4 (8)
Math	**31.4** (4)	16.0 (4)	4.3 (4)	43.5 (4)	49.8 (4)	10.2 (4)	6.0 (4)
GPQA	25.2 (5)	**25.9** (5)	25.7 (5)	30.8 (5)	36.4 (5)	24.7 (5)	-
Code							
HumanEval	**35.4** (0)	34.8 (0)	12.8 (0)	51.2 (0)	57.9 (0)	29.3 (0)	26.2 (0)
HumanEval-FIM	**73.8** (2)	73.3 (2)	26.9 (2)	-	-	-	-
MBPP	40.0 (4)	**48.8** (4)	23.2 (4)	64.2 (0)	74.9 (0)	51.1 (0)	39.0 (3)
Chinese							
CMMLU	**69.9** (5)	50.7 (5)	32.5 (5)	83.9 (5)	-	-	47.2 (5)
C-Eval	**70.5** (5)	51.7 (5)	34.0 (5)	83.2 (5)	-	-	45.0 (5)

Results: Post-trained (SFT) Models

- SFT boosts LLaDA on most downstream tasks, particularly on instruction-following, code generation, and math reasoning.
- Case studies show multi-turn dialogue, multilingual responses, and constrained generation showcasing chatting capabilities from a non-autoregressive model.

Table 12: **Comparison on iGSM Dataset.**

	4 steps	5 steps	6 steps
LLaMA3 8B Base	38.0	35.0	34.0
LLaDA 8B Base	**64.0**	**41.0**	**44.0**

Table 2: **Benchmark Results of Post-trained LLMs.** LLaDA only employs an SFT procedure, while other models have extra reinforcement learning (RL) alignment. * indicates models are evaluated under the same protocol, detailed in Appendix B.6. Results indicated by † and ¶ are sourced from Yang et al. [26] and Bi et al. [32] respectively. The numbers in parentheses represent the number of shots used for in-context learning. "-" indicates unknown data.

	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	69.4 (4)	**78.3** (4)	29.0 (4)	85.7 (0)	91.6 (0)	76.7 (0)	63.0 (0)
Math	**31.9** (0)	29.6 (0)	3.8 (0)	52.9 (0)	75.5 (0)	44.3 (0)	15.8 (0)
GPQA	**33.3** (5)	31.9 (5)	28.4 (5)	34.3 (0)	36.4 (0)	32.8 (0)	-
Code							
HumanEval	49.4 (0)	**59.8** (0)	16.5 (0)	79.9 (0)	84.8 (0)	68.9 (0)	48.2 (-)
MBPP	41.0 (4)	**57.6** (4)	20.6 (4)	67.2 (0)	79.2 (0)	74.9 (0)	35.2 (-)

Breaking the Reversal Curse

Task: Given a famous Chinese poem line, generate the next (forward) or previous (reversal) line.

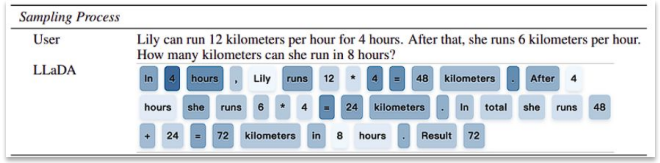
Model	Forward ↓	Reversal ↓
GPT-4o	82.7	34.3
Qwen2.5-7B	75.9	38.0
LLaDA-8B Instruct	**51.8**	**45.6**

Why?

- ARMs only see left-to-right so they inherent directional bias
- LLaDA uses bidirectional attention which is balanced forward & reversal performance
- The gap (51.8 vs 45.6) is much smaller than GPT-4o's (82.7 vs 34.3)

Case Studies: Sampling & Multi-Turn Dialogue

Sampling Visualization



Sampling Process	
User	Lily can run 12 kilometers per hour for 4 hours. After that, she runs 6 kilometers per hour. How many kilometers can she run in 8 hours?
LLaDA	In 4 hours , Lily runs 12 * 4 = 48 kilometers . After 4 hours she runs 6 * 4 = 24 kilometers . In total she runs 48 + 24 = 72 kilometers in 8 hours . Result 72

Math problem solving:
"How many km can she run in 8 hours?"
LLaDA solves step-by-step, filling in
all tokens in parallel at each step.

Multi-Turn Dialogue

Table 3: **Visualization of the Sampling Process and a Generated Multi-round Dialogue.** In the response of LLaDA, darker colors indicate tokens predicted in the later stages of sampling, while lighter colors correspond to earlier predictions.

Sampling Process	
User	Lily can run 12 kilometers per hour for 4 hours. After that, she runs 6 kilometers per hour. How many kilometers can she run in 8 hours?
LLaDA	In 4 hours , Lily runs 12 * 4 = 48 kilometers . After 4 hours she runs 6 * 4 = 24 kilometers . In total she runs 48 + 24 = 72 kilometers in 8 hours . Result 72
Multi-round Dialogue Case	
User	Could you please share with me what the initial two lines of the renowned poem 'The Road Not Taken'?
LLaDA	Certainly! The initial two lines of the renowned poem "The Road Not Taken" by Robert Frost are: "Two roads diverged in a yellow wood, And sorry I could not travel both." These lines set the stage for the poem, introducing the idea of a speaker facing a choice between two paths.
User	Please help me translate into Chinese.
LLaDA	Sure! The first two lines of "The Road Not Taken" by Robert Frost can be translated into Chinese as: "两条路分岔在黄色的树林中，遗憾我不能同时走"
User	Please translate into German.
LLaDA	Sure! The first two lines of "The Road Not Taken" by Robert Frost can be translated into German as: "Zwei Wege trennten sich im gelben Wald, und leider konnte ich nicht beide tun."
User	Please also write a poem about life choices. I require 5 sentences, and the first word of each sentence starts with C as the first letter.
LLaDA	Certainly! Here's a poem about life choices, with each sentence starting with the letter "C": Choices cascade through life's journey, Creating connections with each decision, Careful consideration leads to better paths, Courageous choices shape our destiny, Choosing wisely guides us through the years.

Key Takeaways

1

First competitive diffusion LLM at 8B scale

Challenges the assumption that autoregression is necessary for LLM capabilities.

2

Matches LLaMA3 with 6.5× less training data

65.9 vs 65.4 MMLU; strong math and multilingual performance.

3

Breaks the reversal curse

Bidirectional attention enables balanced forward and reversal reasoning.

4

Opens a new paradigm for language modeling

Diffusion LMs naturally support bidirectional context ⟶ better for infilling, editing, and beyond.

Thank You!

Questions & Discussion

Jingyi Feng & Tarun Mandapati

CS 8803-LLM • Spring 2026

Paper: arxiv.org/abs/2502.09992