Diffusion LLM-assisted Speech Recognition with LLaDA-style Masked Diffusion

Puneet Singh, Suraj Yadav
PhD24003 MT24098
IIIT-Delhi
puneetb@iiitd.ac.in suraj24098@iiitd.ac.in

1. Abstract

Automatic Speech Recognition (ASR) combines strong speech encoders with large language models, but standard autoregressive (AR) decoding is inherently sequential and can be latency-limited for long utterances. This report studies ASR decoding with a *masked diffusion language model* in the style of LLaDA (Large Language Diffusion with mAsking) [2], where generation proceeds by iterative denoising of masked tokens rather than left-to-right next-token prediction. Our pipeline (i) freezes a Whisper speech encoder [4], (ii) learns a lightweight projector to map speech representations into the diffusion decoder embedding space (following the frozen-encoder/frozen-LLM alignment paradigm [1]), and (iii) uses a frozen LLaDA-style mask predictor [2] to iteratively recover transcript tokens conditioned on speech and a transcription prompt. We present a probabilistic formulation of conditional masked diffusion for ASR, derive a projector-only training objective via a masked cross-entropy loss with LLaDA reweighting, and detail three inference configurations used in experiments: **(i)** single denoise, **(ii)** 8-step iterative low-confidence remasking, and **(iii)** 8-step + 16 blocks (semi-autoregressive blockwise diffusion for better controllability, inspired by blockwise/pipelined diffusion decoding ideas [5]). We outline an experimental protocol on LibriSpeech [3] 100h with Word Error Rate (WER) and real-time factor (RTF) style speed metrics.

2. Introduction

ASR maps speech signals to text while balancing accuracy, robustness, and latency. Modern encoder-decoder systems can achieve low WER, but deployment-oriented constraints remain challenging:

- **Latency/throughput:** AR decoders generate tokens sequentially; decoding time scales with transcript length.
- **Compute efficiency:** end-to-end training of large encoder+decoder stacks is expensive; adapting large decoders is slow.
- **Grounding:** strong text priors can hallucinate plausible words when acoustic evidence is weak.

Masked diffusion language models (MDMs) generate sequences by repeatedly predicting masked tokens [2]. Since each iteration can update many token positions in parallel, diffusion-style decoding replaces AR depth $\approx N$ with diffusion depth $\approx K$ (number of refinement steps), enabling a different latency/quality trade-off.

2.1. Problem Statement

Let $o_{1:T}$ be acoustic observations (features from waveform) and let $w_{1:N}$ be transcript tokens. ASR seeks:

$$\hat{w}_{1:N} = \arg\max_{w_{1:N}} P(w_{1:N} \mid o_{1:T}). \tag{1}$$

We study a compute-limited adaptation regime: freeze a strong speech encoder (Whisper) [4] and a strong diffusion LLM (LLaDA-style) [2], and train only a small projector so that the diffusion decoder can approximate $P(w \mid o)$ via conditional generation $P_\theta(w \mid o)$ (alignment-style training as in ASR [1]).

2.2. Contributions

- A **conditional masked diffusion** formulation for ASR with speech-prefix conditioning [2].
- A **projector-only** training objective compatible with a frozen Whisper encoder [4] and frozen LLaDA-style decoder [2].
- A detailed **inference design** for 1-step, 8-step, and 8-step+16-block decoding (blockwise refinement inspired by diffusion decoding acceleration ideas [5]).
- An experimental protocol (data, metrics, ablations) to map the WER–RTF trade-off on LibriSpeech [3].

3. Prior Work

3.1. Encoder + LLM ASR

A strong recent line of work shows that ASR can be achieved by connecting a frozen speech encoder to a frozen LLM using a small alignment module (projector) [1]. A particularly simple approach is to downsample long speech features and use a lightweight projector to match the LLM embedding dimension, then prompt the LLM to output the transcript. This establishes that most learning can be pushed into the alignment module while keeping large backbones frozen, which is attractive under limited compute.

3.2. Masked diffusion models for text generation

Masked diffusion for text defines a forward process that independently masks tokens, and a reverse process that recovers tokens by predicting masked positions [2]. LLaDA [2] trains with a randomly sampled mask probability $t \sim \mathcal{U}(0, 1)$ and a masked-token cross-entropy computed only where tokens are masked. Importantly, LLaDA uses a reweighting factor $1/t$ in the loss, motivated by a generative modeling interpretation (likelihood-bound style) rather than a fixed-ratio masked LM objective.

3.3. Faster diffusion decoding and blockwise strategies

While diffusion updates many tokens in parallel, each diffusion step still requires a (non-causal) Transformer pass over the full sequence. Recent work explores architectural and training strategies for *faster-than-AR* inference in discrete diffusion, including block-wise decoding and pipelining [5]. Our blockwise diffusion configuration adopts the practical spirit of these ideas for stabilizing long outputs and enabling controllable termination.

3.4. Why diffusion is relevant for ASR

Diffusion-style decoding is interesting for ASR because:

- it provides **parallel token updates** (lower sequential depth than AR),
- it enables **iterative self-correction** (refinement over steps).

However, ASR additionally requires strong grounding to audio; hence the design focus in this project is a robust conditioning mechanism (speech prefix) plus conservative inference (low-confidence remasking, and a block strategy for long outputs).

4. Method

4.1. Notation

- $o_{1:T}$: acoustic observations/features for an utterance.
- $w_{1:N}$: transcript tokens; define $x_0 := w_{1:N}$ as the *clean* token sequence.
- $\mathcal{V}$: vocabulary set; $v \in \mathcal{V}$ is a candidate token.
- $M :=$ `[MASK]`: special mask token used in diffusion.
- $t \in [0, 1]$: *mask probability* (forward corruption level); larger t means more masking.
- K: number of diffusion refinement steps at inference.

During training, N is known from ground-truth transcripts. During inference, N can be set to a maximum decoding length and/or controlled via an `<eos>` token.

4.2. Bayesian view of ASR

Bayes rule decomposes ASR into acoustics and language:

$$w = \arg\max_w P(w \mid o) = \arg\max_w P(o \mid w)\, P(w). \tag{2}$$

In encoder-decoder ASR, these components are entangled. In our approach, the frozen speech encoder (Whisper) [4] provides acoustically grounded representations, while the diffusion decoder provides a strong language prior and a denoising mechanism [2], with the projector learning the interface (alignment regime as in [1]).

4.3. Model Architecture

Our system has three parts:

- **Frozen speech encoder** E_{wh} (Whisper encoder) [4].
- **Trainable projector** f_θ mapping speech states into text-embedding space.
- **Frozen diffusion decoder** (LLaDA-style) with parameters ϕ, acting as a *mask predictor* [2].

Only θ is trained.

4.4. Speech encoder and downsampling

Given acoustic features $o_{1:T}$, the encoder outputs frame-level states:

$$h_{1:T'} = E_{\text{wh}}(o_{1:T}), \qquad h_t \in \mathbb{R}^{d_s}. \tag{3}$$

Because T' may be large, we reduce sequence length by a factor k. We use pooling over windows (mean pooling shown):

$$\tilde{h}_i = \frac{1}{k} \sum_{t=(i-1)k+1}^{\min(ik,\, T')} h_t, \quad i = 1, \ldots, L, \;\; L = \left\lceil \frac{T'}{k} \right\rceil. \tag{4}$$

(The $\min(\cdot)$ makes the last, possibly shorter, window well-defined.)

4.5. Projector and speech prefix conditioning

The projector maps $\tilde{h}_i \in \mathbb{R}^{d_s}$ to $z_i \in \mathbb{R}^{d_t}$ (decoder embedding dimension):

$$z_i = f_\theta(\tilde{h}_i), \qquad z_i \in \mathbb{R}^{d_t}. \tag{5}$$

We use an MLP projector:

$$f_\theta(x) = W_2\, \sigma(W_1 x + b_1) + b_2, \tag{6}$$

with σ as ReLU or GELU.

To avoid modifying the frozen diffusion decoder, we use **prefix conditioning** by concatenation (common in encoder–LLM alignment pipelines [1]): concatenate the speech prefix $z_{1:L}$ with a prompt $\pi_{1:M}$. Let $e(\cdot)$ be token embeddings:

$$u_{1:(L+M)} = [z_{1:L};\; e(\pi_{1:M})]. \tag{7}$$

The diffusion decoder then conditions on u when predicting transcript tokens. Concretely, the decoder input sequence is the concatenation of conditioning embeddings u and the noised transcript tokens x_t (embedded), with attention enabled from transcript positions to the prefix.

4.6. Conditional masked diffusion formulation (LLaDA-style)

Let $x_0 = w_{1:N}$ be the clean transcript tokens. LLaDA-style diffusion introduces a continuous corruption level $t \in [0, 1]$ controlling the masking probability [2]. Define x_t by independently masking each position:

$$q_{t|0}(x_t \mid x_0) = \prod_{i=1}^{N} q_{t|0}(x_t^i \mid x_0^i), \tag{8}$$

where

$$q_{t|0}(x_t^i \mid x_0^i) = \begin{cases} 1 - t, & x_t^i = x_0^i, \\ t, & x_t^i = M. \end{cases} \tag{9}$$

Let $\mathcal{S}_t = \{i : \; x_t^i = M\}$ be masked indices.

The frozen diffusion decoder provides a categorical distribution over vocabulary tokens at masked positions:

$$p_\phi(x_0^i \mid x_t, u), \qquad i \in \mathcal{S}_t, \;\; x_0^i \in \mathcal{V}. \tag{10}$$

Unlike AR decoding, the model can predict many masked positions in parallel.

We freeze ϕ and train only θ through gradients that flow from the decoder into the speech prefix $z = f_\theta(\tilde{h})$.

4.6.1. Projector-only conditional objective

A conditional LLaDA-style objective samples $t \sim \mathcal{U}(0, 1)$ and masks accordingly [2]. Because the factor $1/t$ is singular at $t = 0$, in practice we sample $t \sim \mathcal{U}(\varepsilon, 1)$ for a small $\varepsilon > 0$ (e.g., 10^{-3}):

$$\mathcal{L}_{\text{cond}}(\theta) = -\mathbb{E}_{t \sim \mathcal{U}(\varepsilon,1)} \mathbb{E}_{x_t \sim q_{t|0}(\cdot|x_0)} \left[\frac{1}{t} \sum_{i=1}^{N} \mathbf{1}[x_t^i = M] \;\log p_\phi(x_0^i \mid x_t, u) \right]. \tag{11}$$

The $\frac{1}{t}$ factor is characteristic of LLaDA-style training and is linked to a generative modeling interpretation [2]. In our ASR setup, the only learnable component affecting $p_\phi(\cdot \mid x_t, u)$ is the conditioning prefix u via $z = f_\theta(\cdot)$; i.e., the projector learns to encode audio evidence so that the frozen mask predictor can denoise correctly.

4.7. Inference

At inference we choose a target transcript length N (or decode up to a maximum length and rely on an end-of-sequence token in $\mathcal{V}$). Initialize $x^{(K)}$ as fully masked:

$$x^{(K)} = [M, M, \ldots, M], \qquad t^{(K)} = 1. \tag{12}$$

Use a decreasing schedule $1 = t^{(K)} > t^{(K-1)} > \cdots > t^{(0)} = 0$. For $s = K, K-1, \ldots, 1$ each step consists of:

1. **Predict:** fill currently masked positions using $p_\phi(\cdot \mid x^{(s)}, u)$ to obtain a fully-filled candidate $\hat{x}_0$.
2. **Remask:** re-mask the lowest-confidence positions so that the next iterate $x^{(s-1)}$ has approximately a fraction $t^{(s-1)}$ of tokens masked.

Let the model's confidence for position i be the probability of the chosen token:

$$c_i = \max_{v \in \mathcal{V}} p_\phi(v \mid x^{(s)}, u). \tag{13}$$

After producing a candidate $\hat{x}_0$, form $x^{(s-1)}$ by masking the $\lceil t^{(s-1)} N \rceil$ positions with the smallest c_i and

keeping the rest fixed. This yields a conservative refinement loop: high-confidence tokens persist, low-confidence tokens get multiple chances to be corrected.

4.8. Three inference configurations

4.8.1. Config A: 1-step (single denoise)

Set $K = 1$ and do one parallel prediction from fully masked:

$$\hat{x}_0^i = \arg\max_{v \in \mathcal{V}} p_\phi(v \mid x^{(1)}, u), \quad i = 1, \ldots, N. \tag{14}$$

This is fastest but can be brittle because no self-correction occurs.

4.8.2. Config B: 8-step iterative refinement

Set $K = 8$ with a uniform decreasing schedule

$$t^{(s)} = \frac{s}{8}, \quad s = 8, 7, \ldots, 0. \tag{15}$$

Repeat predict + low-confidence remask. This typically improves WER because early mistakes can be revised over multiple refinement opportunities [2].

4.8.3. Config C: 8-step + 16 blocks (semi-autoregressive block diffusion)

Long sequences are harder to control under fully parallel diffusion. We add left-to-right block commitment (motivated by blockwise diffusion decoding approaches [5]):

- Partition token positions into $B = 16$ contiguous blocks of size $m = \lceil N/16 \rceil$.
- Decode blocks left-to-right. When decoding block b, keep blocks $< b$ fixed, and run an 8-step diffusion process only on positions in block b (optionally with a small overlap window for boundary consistency).

Let $\mathcal{I}_b$ denote indices in block b. For block b, initialize those indices as masked, keep others fixed, and run the 8-step loop restricted to $\mathcal{I}_b$:

$$x_{t^{(8)}}^{(b)}[i] = \begin{cases} M, & i \in \mathcal{I}_b, \\ x_0^{(b-1)}[i], & i \notin \mathcal{I}_b. \end{cases} \tag{16}$$

This preserves diffusion parallelism *within blocks* while adding left-to-right structure for stability and improved termination control.

4.9. Complexity and speed intuition

AR decoding has sequential depth $\approx N$. Diffusion decoding has sequential depth $\approx K$ where $K \in \{1, 8\}$ here. Each diffusion step requires a forward pass over the (bi-directional) sequence, so gains depend on implementation and hardware. Blockwise variants can further trade parallelism for controllability and may enable practical acceleration strategies (e.g., pipelining, partial reuse) explored in diffusion decoding literature [5].

5. Experiment

We use LibriSpeech [3] under a compute-limited setup:

- **Training:** 100-hour subset.
- **Evaluation:** dev-clean, dev-other, test-clean, test-other.

We follow standard ASR preprocessing: resampling (if needed), feature extraction consistent with Whisper [4], normalization, and padding/truncation to batch sequences. Acoustic features are denoted $o_{1:T}$.

5.1. Model configuration

- **Speech encoder:** Whisper encoder frozen (E_{wh}) [4].
- **Projector:** 2-layer MLP (trainable), optional dropout.
- **Diffusion decoder:** frozen LLaDA-style mask predictor [2].
- **Conditioning:** speech prefix + prompt prefix (alignment setup as in [1]).

5.2. Training objective and optimization

We optimize Eq. (11). For a batch $\{(o^{(n)}, x_0^{(n)})\}_{n=1}^B$:

$$\theta \leftarrow \theta - \eta \nabla_\theta \left(\frac{1}{B} \sum_{n=1}^{B} \mathcal{L}_{\text{cond}}(\theta; o^{(n)}, x_0^{(n)}) \right). \tag{17}$$

Optimizer: AdamW. **Regularization:** weight decay; optional dropout. **Mask sampling:** $t \sim \mathcal{U}(\varepsilon, 1)$ with small ε for numerical stability.

5.3. Inference setups

We evaluate:

- **Config A:** 1-step.
- **Config B:** 8-step low-confidence remasking.
- **Config C:** 8-step + 16 blocks (blockwise diffusion).

We keep prompt format fixed and vary only decoding strategy for fair comparison.

5.4. Evaluation metrics

WER:

$$\mathrm{WER} = \frac{S + D + I}{R}$$

with substitutions S, deletions D, insertions I, and reference words R.

Speed: real-time factor style metrics under fixed hardware and batch size:

$$\mathrm{RTF} = \frac{\text{wall-clock decode time}}{\text{audio duration}}.$$

If reporting **RTFx**, define it explicitly as a normalized RTF (e.g., $\mathrm{RTFx} = \mathrm{RTF}/\mathrm{RTF}_{\text{baseline}}$) to avoid ambiguity. We also record insertion-vs-deletion bias to understand hallucination vs under-generation.

6. Result

We report WER and speed for each inference configuration. (If these are preliminary numbers, state the hardware and decoding hyperparameters used.)

Table 1: WER and inference speed comparison. **IFT** denotes instruction fine-tuning.

Model / Inference	Avg WER (%)	Avg RTFx
Whisper + MLP + AR-LLM (8B, IFT)	9.65	8.34
Whisper + MLP + LLaDA-8B (IFT, 1-step)	23.90	44.15
Whisper + MLP + LLaDA-8B (IFT, 8-step)	11.14	23.40
Whisper + MLP + LLaDA-8B (IFT, 8-step, 16 blocks)	10.15	1.74

6.1. Expected trends and diagnostic outcomes

Based on diffusion refinement:

- **1-step** is typically fastest but can show higher substitution and insertion errors (no opportunity for correction) [2].
- **8-step** typically reduces substitutions and improves rare-word recovery via multiple refinement chances, at higher compute cost [2].
- **8-step + 16 blocks** can improve stability on long utterances and reduce drift/hallucination by forcing left-to-right block commitment, often trading some parallelism for better controllability [5].

7. Conclusion

We presented a LLaDA-style masked diffusion framework for ASR [2] that freezes a Whisper speech encoder [4] and a diffusion LLM mask predictor, and trains only a lightweight projector to align speech representations with the decoder embedding space (alignment paradigm as in SLAM-ASR [1]). We provided a conditional masked diffusion formulation, a projector-only training objective with LLaDA reweighting, and an inference design based on low-confidence remasking. We defined and motivated three decoding configurations used in this project: 1-step, 8-step, and 8-step + 16 blocks (semi-autoregressive blockwise diffusion) to balance speed, stability, and accuracy, and described an evaluation protocol on LibriSpeech [3].

References

[1] Z. Ma et al. An embarrassingly simple approach for LLM with strong ASR capacity. arXiv preprint, 2024. SLAM-ASR alignment pipeline: frozen encoder + frozen LLM + trainable projector.

[2] S. Nie et al. Large language diffusion models. Conference / arXiv preprint, 2025. LLaDA (Diffusion LLM).

[3] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[4] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. arXiv preprint, 2023. Whisper.

[5] X. Wang, C. Xu, Y. Jin, J. Jin, H. Zhang, and Z. Deng. Diffusion LLMs can do faster-than-AR inference via discrete diffusion forcing. arXiv preprint, 2025. D2F: block-wise causal attention + asymmetric distillation + pipelined parallel decoding.