Discrete Diffusion in Large Language and Multimodal Models: A Survey

Runpeng Yu*, Qi Li*, Xinchao Wang†
National Univerisity of Singapore
{r.yu, liqi}@u.nus.edu, xinchao@nus.edu.sg

***Abstract*—In this work, we provide a systematic survey of Discrete Diffusion Language Models (dLLMs) and Discrete Diffusion Multimodal Language Models (dMLLMs). Unlike autoregressive (AR) models, dLLMs and dMLLMs adopt a multi-token, parallel decoding paradigm using full attention and a denoising-based generation strategy. This paradigm naturally enables parallel generation, fine-grained output controllability, and dynamic, response-aware perception. These capabilities are previously difficult to achieve with AR models. Recently, a growing number of industrial-scale proprietary d(M)LLMs, as well as a large number of open-source academic d(M)LLMs, have demonstrated performance comparable to their autoregressive counterparts, while achieving up to *10*× acceleration in inference speed. These developments position discrete diffusion models as a promising alternative to intelligence based on the traditional autoregressive approach.**

The advancement of discrete diffusion LLMs and MLLMs has been largely driven by progress in two domains. The first is the development of autoregressive LLMs and MLLMs, which has accumulated vast amounts of data, benchmarks, and foundational infrastructure for training and inference. A significant portion of these resources has been or can be transferred to support the development of dLLMs and dMLLMs. The second contributing domain is the evolution of the mathematical models underlying discrete diffusion. Originally based on the continuous-space, the diffusion models for LLM and MLLM have transitioned to the currently prevalent discrete-space diffusion models built on absorbing states. The adaptation and simplification of the mathematical framework has played a crucial role in enabling the scaling and engineering optimization of dLLMs and dMLLMs. Together, these advancements have catalyzed a surge in dLLMs and dMLLMs research in early 2025.

In this work, we present a comprehensive overview of the research in the dLLM and dMLLM domains. We trace the historical development of dLLMs and dMLLMs, formalize the underlying mathematical frameworks, and categorize representative models. We further analyze key techniques for training and inference, and summarize emerging applications across language, vision-language, and biological domains. We conclude by discussing future directions for research and deployment.

Relative papers are collected in this repo.

***Index Terms*—Discrete Diffusion, Large Language Model, Multimodal Large Language Model, Diffusion Large Language Model, Diffusion Multimodal Large Language Model, Language Model, Unified Model**

I Introduction

In recent years, Large Language Models (LLMs) and Multimodal Large Language Models (MLLMs) have demonstrated remarkable advances, exhibiting capabilities that increasingly resemble, or even surpass, human-level performance in domains traditionally associated with intelligence. Modern LLMs and MLLMs achieve superior scores on standard benchmarks designed for general knowledge, comprehension, and reasoning, suggesting that these systems are no longer merely text completion engines but competent general-purpose agents.

To date, the predominant paradigm for both LLMs and MLLMs has been autoregressive (AR) [1, 2, 3, 4, 5]. Despite previous successes, AR models, which generate outputs sequentially in a left-to-right fashion, possess certain intrinsic limitations. First, their decoding strategy inherently restricts generation to be token-by-token, making parallelization during inference challenging and limiting efficiency improvements. Second, people do not merely envision LLMs and MLLMs as question-answering machines; rather, people aim for them to serve as the intelligent core of a system—akin to the human brain—with capabilities for tool use and task orchestration. However, in this context, unstructured natural language becomes highly inefficient for fine-grained control. AR models struggle to enforce precise structural constraints, such as specific lengths or formats, due to their lack of inherent mechanisms for such regulation. Finally, because of the nature of causal attention, AR models perceive visual inputs or textual prompts in a one-pass, static manner. This inhibits selective attention and makes task-aware, dynamic perception challenging without relying on costly iterative chain-of-thought (CoT) reasoning or multi-round processing integrated with external tools.

Discrete Diffusion Large Language Models (dLLMs) and Discrete Diffusion Multimodal Large Language Models (dMLLMs) [6, 7, 8, 9, 10, 11] have recently emerged as a promising direction. In contrast to AR generation, discrete diffusion models treat generation as an iterative denoising process over discrete token sequences. This paradigm eliminates the left-to-right constraint and enables generation that is parallel and structurally controllable with bidirectional attention mechanism.

- **Parallel Decoding:** Unlike AR models that decode one token at a time, discrete diffusion models generate multiple tokens simultaneously in each denoising step. This parallel decoding significantly accelerates inference speed.
- **Better Controllability:** Discrete diffusion treats generation as a denoising or infilling task rather than unbounded left-to-right generation. This allows for precise control over output properties such as response length, format, and even the reasoning structure—by conditioning the generation on predefined templates.
- **Dynamic Perception:** Enabled by bidirectional attention, discrete diffusion models continuously revise their per-

* Equal contribution, random order.
† Corresponding author.

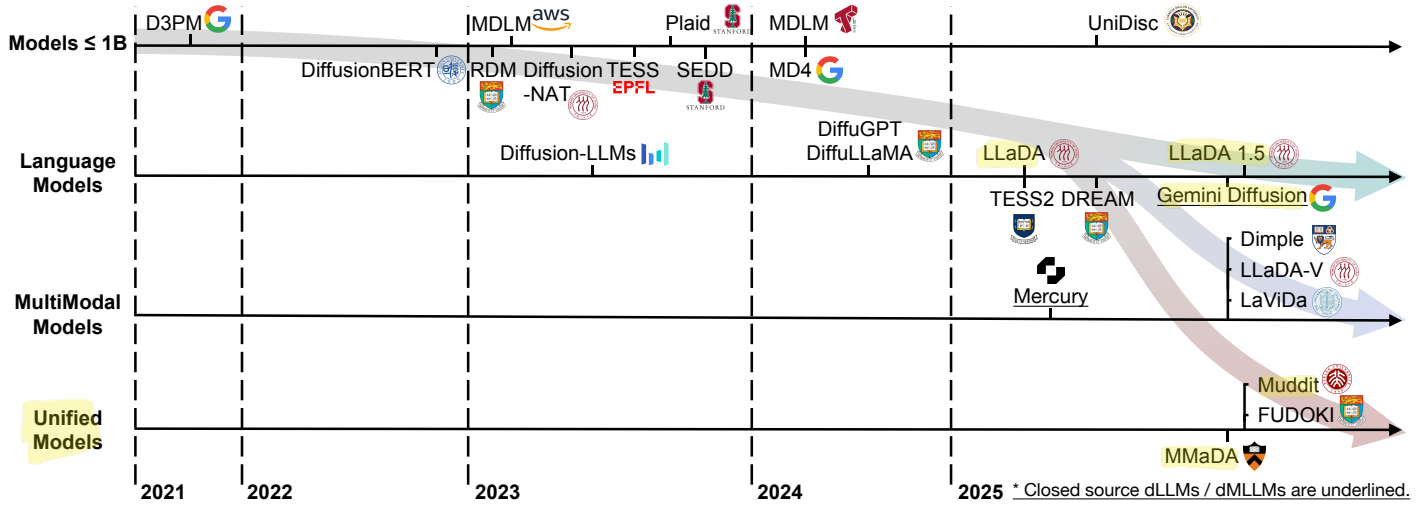


Fig. 1. A timeline of existing dLLMs and dMLLMs in recent years. The timeline is established mainly according to the release date (e.g., the submission date to arXiv) of the technical paper for a model. The affiliation marked in the figure is based on the first affiliation listed in each paper.

ception of visual and linguistic context throughout the generation process. This facilitates adaptive comprehension that evolves with the response, overcoming the static, one-pass limitations of AR models.

Originally proposed for continuous domains such as image generation [12], diffusion models have since been successfully extended to discrete spaces. Early efforts in this direction established the foundational mathematical formulations of discrete diffusion, introducing a token corruption schemes specifically designed for categorical data [13, 14]. These models demonstrated the feasibility of diffusion-based generation on various types of discrete data, including natural language [13, 15], images [13], and biological sequences such as DNA [16]. In this early stage, experiments were limited to models with around or fewer than 1 billion parameters. Through simplifications and reparameterizations [15, 16, 17], along with practical engineering efforts, the absorbing-state discrete diffusion formulation has gradually become the predominant mathematical framework adopted by open-source models. In this formulation, the inference process is an iterative masked token prediction, while the training process reduces to a masked language modeling task based on randomly sampled time indices and masked positions. The loss function can be written simply as a weighted cross-entropy loss. These simplifications significantly reduced the complexity and improved the stability in training and inference, paving the way for large-scale diffusion language models.

Recent advances have significantly improved the scalability and effectiveness of discrete diffusion models [18, 19]. A major breakthrough on the industrial front came with the presentation of discrete diffusion-based large language models by Inception Labs and Google, namely *Mercury* [20] and *Gemini Diffusion* [21]. These models reports comparable performance on code and mathematics benchmarks with their AR counterpart, while also achieving *10*× speedups in decoding, with about 1000 tokens per second.

In parallel, the research community has developed and open-sourced an increasing number of discrete diffusion-based language and multimodal models. Following a trajectory similar to that of autoregressive (AR) models, the development began with dLLM models trained on large-scale text corpora, such as *LLaDA* [6] and *Dream* [7]. Later, using the public available dLLMs as the backbones, dMLLMs, such as *Dimple* [8], *LaViDa* [10], and *LLaDA-V* [9], are developed through multimodal alignment, instruction tuning, preference learning, and then reasoning enhancement. Empirical results consistently show that these dLLMs and dMLLMs match or closely approach the performance of AR models trained at similar scale and data size. Crucially, with acceleration techniques, their parallel decoding ability enables significantly faster inference compared to traditional AR models.

Alongside the development of open-source dLLMs and dMLLMs, significant progress has been made in optimizing training techniques specific to this paradigm. Strategies such as model initialization [18, 8] and masking scheduling [22] have been systematically explored to accelerate convergence, improve training stability, and enhance the final generation quality.

With the availability of open-source models, academic research has also deepened into the inference mechanisms of discrete diffusion. For example, during inference, the fundamental question of discrete diffusion models lies in determining, at each generation step, which tokens should be unmasked (i.e., finalized) and which should be left masked to future iterations. To this end, various unmasking [7, 8] and remasking [23, 24] strategies have been proposed, aiming to balance generation quality, stability, and speed. Another example is that, although discrete diffusion models are inherently capable of parallel decoding, they suffer from increased per-step computational cost due to the use of full attention masks. As a result, without optimization, their actual inference speed cannot surpass that of AR models. A considerable amount of work has focused on adapting existing inference acceleration algorithms originally designed for AR models to dLLMs and dMLLMs, aiming to reduce extensive full attention computations and enhance inference efficiency [25, 26].

To provide a comprehensive framework for understanding discrete diffusion large language models (DLLMs) and discrete diffusion multimodal large language models (DMLLMs), this survey systematically explores recent advances in modeling,

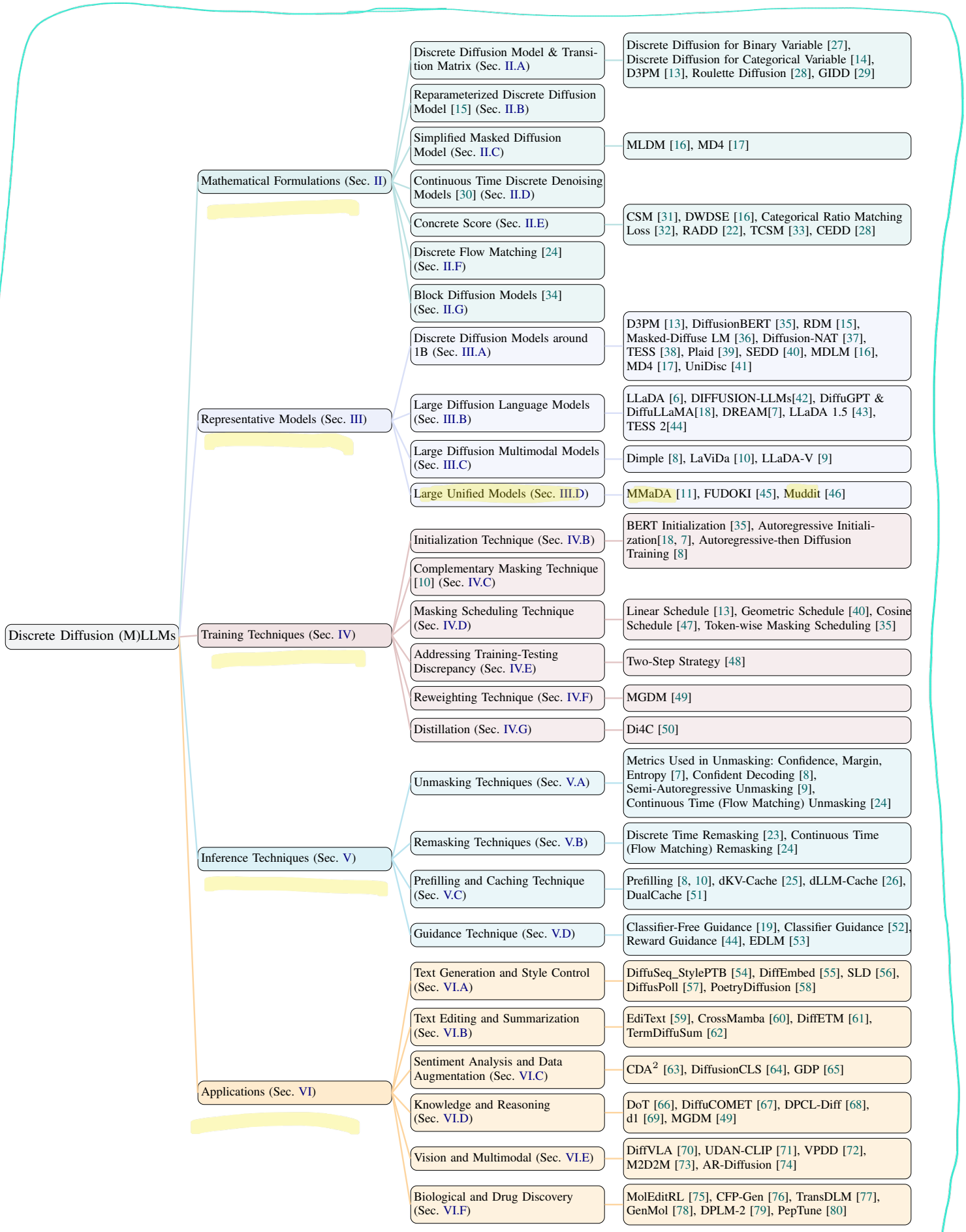


Fig. 2. Overview of our survey. We begin by introducing the mathematical foundations of discrete diffusion language models (Sec. II). Next, we present a high-level overview of representative base models (Sec. III), followed by discussions on training strategies (Sec. IV) and inference techniques (Sec. V). In addition, we also review a wide range of applications (Sec. VI) that adopt discrete diffusion language models as their core model.

training, generation and applications of discrete diffusion techniques in language, vision-language, and biology tasks.

In the rest of this paper, Sec. II presents the mathematical foundations of discrete diffusion models. The core of mathematical formulations lies in the definitions of the forward and reverse diffusion processes over discrete spaces. These are modeled via Markov chains that incrementally corrupt and denoise sequences. The framework encompasses both discrete-time and continuous-time formulations, supporting extensions such as flow matching and semi-autoregressive block diffusion model.Sec. III surveys representative discrete diffusion language models across varying scales. This includes early-stage models (e.g., D3PM [13], RDM [15], DiffusionBERT [35]), scaled dLLMs [6, 18, 7],dMLLMs [8, 10, 9], and unified models [11]. Sec. IV discusses the key training strategies used in dLLMs and dMLLMs, including initialization, masking strategies, and masking schedules. Sec. V lists various inference techniques used in dLLMs and dMLLMs, including unmasking, remasking, caching and guidance. Sec. VI reviews the broad range of applications powered by discrete diffusion models. Finally, Sec. VII summarize potential directions for future research. The organization of this survey is illustrated in Fig. 2.

II Mathematical Formulations

II.A Discrete Diffusion Model and Transition Matrix

Diffusion models with discrete state spaces were initially introduced in [27] for binary variables, and later extended to categorical variables in [14]. Based on the previous works, Discrete Denoising Diffusion Probabilistic Models (D3PMs) [13] provides a general and flexible framework.

Let $x_0 \sim q(x_0)$ denote the data distribution over sequences composed of K categorical values. The D3PM framework defines a forward Markov process $q(x_{1:T} \mid x_0)$ that gradually corrupts the data into noise, and a parameterized reverse process $p_\theta(x_{0:T})$ that learns to denoise:

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1}), \tag{1}$$

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t). \tag{2}$$

Each x_t is a discrete random variable and $q(x_t \mid x_{t-1})$ is defined via a time-dependent transition matrix Q_t, with categorical transition probabilities:

$$q(x_t \mid x_{t-1}) = \text{Cat}(x_t; p = x_{t-1} Q_t), \tag{3}$$

where x_{t-1} is a one-hot vector and $Q_t \in \mathbb{R}^{K \times K}$ is a row-stochastic matrix. The marginal distribution $q(x_t \mid x_0)$ and the posterior $q(x_{t-1} \mid x_t, x_0)$ are:

$$q(x_t \mid x_0) = \text{Cat}(x_t; p = x_0 Q_{1:t}), \tag{4}$$

$$Q_{1:t} = Q_1 Q_2 \dots Q_t, \tag{5}$$

$$q(x_{t-1} \mid x_t, x_0) = \text{Cat}\left(x_{t-1}; p = \frac{x_t Q_t^\top \circ x_0 Q_{1:t-1}}{x_0 Q_{1:t} x_t^\top}\right). \tag{6}$$

D3PM framework support various types of transition matrices Q_t, each inducing a different corruption behavior:

- **Uniform**: $Q_t = (1-\beta_t)I + \frac{\beta_t}{K}\mathbf{1}\mathbf{1}^\top$ yields a uniform stationary distribution. The uniform transition matrix looks like

$$Q_t^{\text{uniform}} = \begin{bmatrix} 1-\frac{K-1}{K}\beta_t & \frac{\beta_t}{K} & \cdots & \frac{\beta_t}{K} \\ \frac{\beta_t}{K} & 1-\frac{K-1}{K}\beta_t & \cdots & \frac{\beta_t}{K} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\beta_t}{K} & \frac{\beta_t}{K} & \cdots & 1-\frac{K-1}{K}\beta_t \end{bmatrix}. \tag{7}$$

- **Absorbing**: $Q_t = (1-\beta_t)I + \beta_t \mathbf{1} e_m^\top$, where e_m is a vector with a one on the absorbing state and zeros elsewhere. Tokens either remain unchanged or are mapped to a special [MASK] token with probability β_t. The absorbing transition matrix looks like

$$Q_t^{\text{absorb}} = \begin{bmatrix} 1-\beta_t & 0 & \cdots & 0 & \beta_t \\ 0 & 1-\beta_t & \cdots & 0 & \beta_t \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1-\beta_t & \beta_t \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}. \tag{8}$$

- **Hybrid**: Hybrid transition was initially discussed in [13], where multiple types of transition are combined to create more expressive diffusion processes. For example, the Routlette Diffusion in [28] and the Generalized Interpolating Discrete Diffusion (GIDD) [29] both study the linear combination of the absorbing transition and the uniform transition. A key motivation for such "Linear + Absorbing" transitions is to overcome a fundamental limitation of absorbing diffusion: once a token is denoised, it cannot be revised. Such "Linear + Absorbing" transitions address this by allowing previously denoised tokens to re-enter the diffusion process via the uniform component. This grants the model the ability to correct earlier generation errors during inference.
- **Discretized Gaussian**:

$$[Q_t]_{ij} = \begin{cases} \frac{\exp\left(-\frac{4|i-j|^2}{(K-1)^2\beta_t}\right)}{\sum_{n=-(K-1)}^{K-1}\exp\left(-\frac{4n^2}{(K-1)^2\beta_t}\right)} & \text{if } i \neq j \\ 1-\sum_{l\neq i}[Q_t]_{il} & \text{if } i = j \end{cases}$$

This matrix simulates Gaussian diffusion, suitable for ordinal data. Each state i is most likely to transition to nearby states j with probabilities resembling a Gaussian kernel. Closer states receive higher probabilities, while distant states receive lower ones. Diagonal values are chosen to ensure that each row sums to 1, yielding a uniform stationary distribution.
- **Band-diagonal**:

$$[Q_t]_{ij} = \begin{cases} \frac{1}{K\beta_t} & 0 < |i-j| \leq v \\ 0 & |i-j| > v \\ 1-\sum_{l\neq i}[Q_t]_{il} & i = j \end{cases}$$

Band-diagonal imposes local transitions only: state i can only transition to its v-nearest neighbors on the ordinal scale. It biases the forward process toward small, local perturbations.

- **Embedding-based**: Let A be an adjacent matrix build from based on the similarity between tokens in the embedding space, in [13] the Embedding-based transition matrix is defined as

$$Q_t = \exp(\alpha_t R) = \sum_{n=0}^{\infty} \frac{\alpha_t^n}{n!} R^n, \tag{9}$$

$$R_{ij} = \begin{cases} -\sum_{l \neq i} A_{il} & i = j \\ A_{ij} & i \neq j \end{cases}. \tag{10}$$

This transition matrix promotes a diffusion process where tokens are more likely to transition to others that are semantically or syntactically similar in the embedding space.

Following the x_0-parameterization, the model predicts $p_\theta(x_{t-1} \mid x_t)$ using:

$$p_\theta(x_{t-1} \mid x_t) \propto \sum_{\tilde{x}_0} q(x_{t-1}, x_t \mid \tilde{x}_0) \tilde{p}_\theta(\tilde{x}_0 \mid x_t), \tag{11}$$

where $\tilde{p}_\theta(\tilde{x}_0 \mid x_t)$ is a network predicting logits over x_0. This parameterization ensures the reverse distribution respects the sparsity pattern of Q_t.

The loss function combines the variational lower bound $\mathcal{L}_{\text{vb}}$ with an auxiliary cross-entropy denoising term $\mathcal{L}_{\text{ce}}$:

$$\mathcal{L}_\lambda = \mathcal{L}_{\text{vb}} + \lambda \mathcal{L}_{\text{ce}}, \tag{12}$$

$$\begin{aligned} \mathcal{L}_{\text{vb}} = \mathbb{E}_{q(x_0)} \Big[& \underbrace{\text{KL}(q(x_T \mid x_0) \,\|\, p(x_T))}_{\mathcal{L}_T} \\ & + \underbrace{\sum_{t=2}^{T} \mathbb{E}_{q(x_t \mid x_0)} \left[\text{KL}(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)) \right]}_{\mathcal{L}_{t-1}} \\ & - \underbrace{\mathbb{E}_{q(x_1 \mid x_0)} \left[\log p_\theta(x_0 \mid x_1) \right]}_{\mathcal{L}_0} \Big], \end{aligned} \tag{13}$$

$$\mathcal{L}_{\text{ce}} = \mathbb{E}_{q(x_0)} \mathbb{E}_{q(x_t \mid x_0)} [-\log \tilde{p}_\theta(x_0 \mid x_t)]. \tag{14}$$

In $\mathcal{L}_{\text{vb}}$,

- $\mathcal{L}_T$ is the KL divergence between the forward terminal distribution $q(x_T \mid x_0)$ and the prior $p(x_T)$,
- $\mathcal{L}_{t-1}$ is the KL divergence between the forward posterior $q(x_{t-1} \mid x_t, x_0)$ and the learned reverse model $p_\theta(x_{t-1} \mid x_t)$ at each intermediate step,
- $\mathcal{L}_0$ is the cross-entropy loss for reconstructing x_0 from x_1 using the reverse model.

Such decomposition enables the model to be trained efficiently by sampling time steps t uniformly and estimating each term using stochastic gradient descent. The forward posterior $q(x_{t-1} \mid x_t, x_0)$ has a closed-form expression under categorical diffusion, and the model is typically parameterized to predict $p_\theta(x_0 \mid x_t)$, from which $p_\theta(x_{t-1} \mid x_t)$ is derived analytically. The auxiliary $\mathcal{L}_{\text{ce}}$ is added to encourage accurate prediction of the original data x_0 from corrupted samples.

Besides a flexible representation of discrete diffusion, D3PM also unifies various paradigms such as BERT, autoregressive models, and masked language models within the discrete diffusion framework.

II.B Reparameterized Discrete Diffusion Model

Reparameterized Discrete Diffusion Models (RDMs) [15] reformulate the backward process of discrete diffusion in D3PM into a two-stage sampling procedure. The core idea is to introduce a latent routing variable v_t that determines decoding behavior for each token at every step.

Given a forward transition of the form:

$$q(x_t \mid x_{t-1}) = \beta_t x_{t-1} + (1 - \beta_t) q_{\text{noise}}, \tag{15}$$

$$q(x_t \mid x_0) = \alpha_t x_{t-1} + (1 - \alpha_t) q_{\text{noise}}, \tag{16}$$

where q_{noise} is the noise distribution, and $\alpha_t := \prod_{i=1}^{t} \beta_i$. The backward posterior $q(x_{t-1} \mid x_t, x_0)$ can be expressed as a mixture of two cases, depending on whether $x_t = x_0$:

$$q(x_{t-1} \mid x_t, x_0) = \begin{cases} \lambda_{t-1}^{(1)} x_t + (1 - \lambda_{t-1}^{(1)}) q_{\text{noise}}, & x_t = x_0, \\ \lambda_{t-1}^{(2)} x_0 + (1 - \lambda_{t-1}^{(2)}) q_{\text{noise}}(x_t), & x_t \neq x_0, \end{cases} \tag{17}$$

where $q_{\text{noise}}(x_t)$ is the interpolation between x_t and q_{noise}, and $\lambda_{t-1}^{(1)}$, $\lambda_{t-1}^{(2)}$ are scalar coefficients derived from β_t, α_t and $q_{\text{noise}}(x_t)$.

To reparameterize the backward transition, RDM introduces Bernoulli latent variables:

$$v_{t-1}^{(1)} \sim \text{Bernoulli}(\lambda_{t-1}^{(1)}), \quad u_t^{(1)} \sim q_{\text{noise}}, \tag{18}$$

$$v_{t-1}^{(2)} \sim \text{Bernoulli}(\lambda_{t-1}^{(2)}), \quad u_t^{(2)} \sim q_{\text{noise}}(x_t). \tag{19}$$

Let $b_t = \mathbf{1}[x_t = x_0]$, the reparameterized sample is computed as:

$$\begin{aligned} x_{t-1} = b_t & \left[v_{t-1}^{(1)} x_t + (1 - v_{t-1}^{(1)}) u_t^{(1)} \right] \\ & + (1 - b_t) \left[v_{t-1}^{(2)} x_0 + (1 - v_{t-1}^{(2)}) u_t^{(2)} \right]. \end{aligned} \tag{20}$$

This formulation allows the model to explicitly route tokens through different behaviors: either retaining the current token, resetting to noise, or denoising back to the ground truth.

Based on Eq. (20), RDMs define a joint generative model:

$$p_\theta(x_0, x_{1:T}, v_{1:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}, v_{t-1} \mid x_t), \tag{21}$$

and the evidence lower bound (ELBO) becomes:

$$\log p_\theta(x_0) \geq \mathcal{L}_1 - \underbrace{\sum_{t=2}^{T} \mathcal{L}_t}_{\mathcal{L}_T} + C, \tag{22}$$

where C is a constant.

For $t > 1$, the loss term decomposes as:

$$\begin{aligned} \mathcal{L}_t = \mathbb{E}_{x_{1:T}, v_{1:T} \mid x_0} \Big[& \text{KL}(q(v_{t-1}) \parallel p_\theta(v_{t-1} \mid x_t)) \\ & + \text{KL}(q(x_{t-1} \mid v_{t-1}, x_t, x_0) \parallel p_\theta(x_{t-1} \mid v_{t-1}, x_t)) \Big]. \end{aligned} \tag{23}$$

By aligning $p_\theta(v_{t-1} \mid x_t)$ with $q(v_{t-1})$ and using the x_0-parameterization, the loss can be simplified into

$$\mathcal{L}_t = \mathbb{E}_{x_0, x_{1:T}} \Big[-\lambda_{t-1}^{(2)} \sum_{n=1}^{N} (1 - b_{t,n}) x_{0,n} \log [f_\theta(x_t)]_n \Big], \tag{24}$$

where $\lambda_{t-1}^{(2)} = \frac{\alpha_{t-1} - \alpha_t}{1 - \alpha_t}$.

A central issue of RDM lies in its dependence on the ground truth x_0 to compute the backward transition probabilities Eq. (17). However, in the inference stage, x_0 is unknown, making it infeasible to directly evaluate the indicator function b_t required for determining the appropriate transition path. To overcome this limitation, the authors propose a recursive approximation for computing b_t by utilizing the Bernoulli routing variables v. Beginning with $b_T = 0$, which assumes the initial sequence is fully noisy, the clean token set is recursively updated via:

$$b_{t-1,n} = \left(b_{t,n} \wedge v^{(1)}_{t-1,n}\right) \vee v^{(2)}_{t-1,n}, \tag{25}$$

where $\wedge$ and $\vee$ denote logical conjunction and disjunction, respectively.

II.C Simplified Masked Diffusion Model

A widely adopted class of discrete diffusion models is based on an absorbing state and is often referred to as the *Masked Diffusion Model*. Both [17, 16] introduced simplifications to the diffusion process and the corresponding training objective for masked diffusion, yielding improved performance and computational efficiency.

In masked diffusion, the forward process progressively replaces input tokens with a special `[MASK]` token. Once a token is masked, it remains in that state throughout the remainder of the process, making the `[MASK]` token an absorbing state. At each time step t, the forward transition for a token x is defined as:

$$q(x_t \mid x_0) = \text{Cat}(x_t; \alpha_t x_0 + (1-\alpha_t)m), \tag{26}$$

where m is the one-hot vector corresponding to the `[MASK]` token, and $\alpha_t \in [0,1]$ is the monotonically decreasing schedule such that $\alpha_0 \approx 1$ and $\alpha_T = 0$.

The reverse process aims to denoise the masked sequence by substituting `[MASK]` tokens with predicted tokens. Importantly, unmasked tokens are carried out unchanged throughout the denoising process. The reverse posterior at a previous time s conditioned on x_t and x_0 is given by:

$$q(x_s \mid x_t, x_0) = \begin{cases} \text{Cat}(x_s; x_t), & \text{if } x_t \neq m, \\ \text{Cat}\left(x_s; \dfrac{(1-\alpha_s)m + (\alpha_s - \alpha_t)x_0}{1-\alpha_t}\right), & \text{if } x_t = m. \end{cases} \tag{27}$$

This formulation reflects two key properties of the masking process: (1) If the current token x_t is not masked, the posterior is deterministic: $x_s = x_t$. (2) If the token is masked, then the posterior is a linear interpolation between the mask vector m and the clean token x_0, scaled by the noise schedule parameters α_s and α_t.

Let $f_\theta(x_t)$ be the neural network output predicting the original token x_0 from the noisy input x_t. The above reverse transition is rewritten as:

$$p_\theta(x_s \mid x_t) = \begin{cases} \text{Cat}(x_s; x_t), & \text{if } x_t \neq m, \\ \text{Cat}\left(x_s; \frac{(1-\alpha_s)m + (\alpha_s - \alpha_t) f_\theta(x_t)}{1-\alpha_t}\right), & \text{if } x_t = m. \end{cases} \tag{28}$$

The variational lower bound for discrete diffusion can be simplified using the above formulation, leading to an final loss in the form of

$$\mathcal{L} = \sum_{t=2}^{T} \mathbb{E}_{x_0, x_{1:T}} \Big[-\frac{\alpha_{t-1} - \alpha_t}{1-\alpha_t} \sum_{n=1}^{N} \delta_m(x_{t,n}) x_{0,n} \log[f_\theta(x_t)]_n \Big], \tag{29}$$

where $\delta_m(x_{t,n})$ denotes the indicator function, $\delta_m(x_{t,n}) = 1$, if the n-th token of x_t is a masked token, otherwise, $\delta_m(x_{t,n}) = 0$. This loss is similar to Eq. (24) in the form, but the masked diffusion model eliminates the need to rely on ground truth for computing the indicator function; it can instead be determined based on the mask tokens.

By defining the discrete time series as $0, \frac{1}{T}, \ldots, 1 - \frac{1}{T}, 1$ and letting $T \to \infty$, both works [16, 17] extend the above loss Eq. (29) to the continuous-time setting:

$$\mathcal{L} = \int_0^1 \mathbb{E}_{x_{0:1}} \Big[\frac{\alpha_t'}{1-\alpha_t} \sum_{n=1}^{N} \delta_m(x_{t,n}) x_{0,n} \log[f_\theta(x_t)]_n \Big] dt. \tag{30}$$

This loss corresponds to a reweighted cross-entropy loss evaluated only over masked tokens. Such loss formulation is significantly simpler than the original variational bound and has become the standard training objective for subsequent large discrete diffusion models.

II.D Continuous Time Discrete Denoising Models

D3PM operates in discrete time, *i.e.*, with time steps $t = 0, 1, 2, \ldots, T$. [30] describes a continuous-time framework for discrete denoising models, formulated as a Continuous-Time Markov Chain (CTMC), where $t \in [0, T]$. This approach generalizes discrete diffusion models by allowing transitions at arbitrary time points. The infinitesimal transition probabilities are given by:

$$q_{t|t-\Delta t}(x' \mid x) = \delta_{x,x'} + R_t(x, x')\Delta t + o(\Delta t). \tag{31}$$

This process converges to a tractable reference distribution as $t \to T$. The time-reversed generative process is another CTMC with reverse rate matrix $\hat{R}_t$, expressed as:

$$\hat{R}_t(x, x') = R_t(x', x) \sum_{x_0} \frac{q_{t|0}(x' \mid x_0)}{q_{t|0}(x \mid x_0)} p_\theta(x_0 \mid x), \tag{32}$$

where $p_\theta(x_0 \mid x)$ is a learnable denoising model approximating $q_{0|t}(x_0 \mid x)$.

The training of $p_\theta(x_0 \mid x)$ is guided by a continuous-time version of the variational lower bound. Let $Z_t(x) = \sum_{x' \neq x} R_t(x, x')$ be the total outgoing rate and $r_t(x' \mid x) = R_t(x, x')/Z_t(x)$ the normalized jump probability. The continuous-time variational lower bound is:

$$\mathcal{L}_{\text{vb}}(\theta) = T\, \mathbb{E}_{t \sim \mathcal{U}(0,T)} \mathbb{E}_{x \sim q_t} \mathbb{E}_{x' \sim r_t(\cdot|x)} \Big[\sum_{x'' \neq x} \hat{R}^\theta_t(x, x'') - Z_t(x) \log \hat{R}^\theta_t(x', x) \Big] + C, \tag{33}$$

where C is constant with respect to θ. This objective can be efficiently optimized using stochastic gradient descent by sampling (x, x') pairs according to the forward process.

During inference, however, the exact simulation of the reverse CTMC can be computationally prohibitive. Instead, the tau-leaping algorithm [reference] approximates the reverse process by applying multiple transitions within a time interval τ simultaneously. For a current state x_t, the number of transitions to each x' during $[t-\tau, t]$ is modeled as:

$$P_{x'} \sim \text{Poisson}(\tau \cdot \hat{R}_t^\theta(x_t, x')). \tag{34}$$

The next state $x_{t-\tau}$ is obtained by aggregating the sampled transitions. This method supports parallel decoding by allowing simultaneous updates across multiple dimensions.

To further improve sample fidelity, predictor-corrector steps is used. After a tau-leaping step, corrector transitions with rate matrix $R_c = R_t + \hat{R}_t^\theta$ are applied to refine the sample distribution toward the target marginal $q_t(x)$. This approach is analogous to Langevin correctors in continuous diffusion models.

II.E Concrete Score

In the continuous-time discrete diffusion framework, as formulated in previous [30], the reverse process can also be analytically expressed in terms of the forward transition rate matrix and a function known as the concrete score [31]. This construction enables training via score matching, analogous to score-based models in continuous domains.

Let $R_t(x, x')$ be the forward transition rate matrix of a continuous-time Markov chain (CTMC) over a discrete state space $\mathcal{X}$. The reverse-time transition rate $\hat{R}_t(x, x')$ can be formulated as:

$$\hat{R}_t(x, x') = \begin{cases} \dfrac{p_t(x')}{p_t(x)} R_t(x', x), & x' \neq x, \\ -\displaystyle\sum_{k \neq x} \hat{R}_t(x, k), & x' = x. \end{cases} \tag{35}$$

Here, the scalar ratio $\frac{p_t(x')}{p_t(x)}$ is referred to as the concrete score. It quantifies the relative likelihood of two discrete states at time t and modulates the reverse transition rate accordingly.

Thus, instead of learning the full reverse transition distribution, one can train a model $s_\theta(x, t)$ to estimate the concrete score:

$$s_\theta(x, t) \approx \left[\frac{p_t(x')}{p_t(x)}\right]_{x' \in \mathcal{X}}. \tag{36}$$

II.E.1 Training Loss

Training is typically done by minimizing divergence-based losses that compare the predicted ratio to the true ratio.

II.E.1.(a) Concrete Score Matching (CSM) [31]

The most direct approach is Concrete Score Matching (CSM) [31], which uses a squared-error loss to match the predicted and true ratios:

$$\mathcal{L}_{\text{CSM}}(t) = \frac{1}{2}\mathbb{E}_{x \sim p_t}\left[\sum_{x' \neq x}\left(s_\theta(x, t)_{x'} - \frac{p_t(x')}{p_t(x)}\right)^2\right] \tag{37}$$

While theoretically sound, this ℓ_2 loss does not penalize invalid (e.g., negative or zero) predictions sufficiently, potentially leading to instability.

II.E.1.(b) Diffusion-Weighted Denoising Score Entropy [40]

Leveraging Bregman divergence, score entropy is formulated as another score matching loss. Score entropy is non-negative, symmetric, and convex. It is also an extension of the conventional cross-entropy to general positive-valued functions beyond the probability simplex. Score entropy enables the construction of an ELBO for discrete diffusion models, resulting in the Diffusion-Weighted Denoising Score Entropy (DWDSE) loss [40]

$$\begin{aligned}\mathcal{L}_{\text{DWDSE}}(x_0) = \int_0^T \mathbb{E}_{x_t \sim p_{t|0}(\cdot|x_0)} \sum_{x' \neq x_t} R_t(x_t, x') \Bigg(& \\ s_\theta(x_t, t)_{x'} - \frac{p_{t|0}(x'|x_0)}{p_{t|0}(x_t|x_0)} \log s_\theta(x_t, t)_{x'} & \\ + K\left(\frac{p_{t|0}(x'|x_0)}{p_{t|0}(x_t|x_0)}\right)\Bigg)\, dt, & \end{aligned} \tag{38}$$

where K is a normalizing constant function.

II.E.1.(c) Target Concrete Score Matching [33]

Target Concrete Score Matching (TCSM) introduces two score matching losses: the *score-based* and *distribution-based* losses. The score-based objective operates directly on the concrete score vectors:

$$\mathcal{L}_{\text{score}}(\theta) = \mathbb{E}_{\omega(t)p(x_t)h(x_1|x_t)} \sum_{i=1}^{L} \ell^i_{\text{score}}, \tag{39}$$

$$\ell^i_{\text{score}} = \mathcal{D}\left(\left[\frac{p_{1|t}([y^i, x_1^{\setminus i}]|x_t)}{p_{1|t}(x_1|x_t)}\right]_{y^i=1}^{V}, \left[\frac{p^\theta_{1|t}([y^i, x_1^{\setminus i}]|x_t)}{p_\theta^{1|t}(x_1|x_t)}\right]_{y^i=1}^{V}\right), \tag{40}$$

where V is the vocabulary size, $\mathcal{D}(\cdot, \cdot)$ denotes a divergence measure, $\omega(t)$ is the distribution used to sample time index t, and $h(x_1|x_t)$ is a proposal distribution such as the ground-truth conditional distribution $p_{1|t}(x_1|x_t)$. In the equation, $[y^i, x^{\setminus i}] := [x^1, \ldots, x^{i-1}, y^i, x^{i+1}, \ldots, x^L]$ is used to define a new sequence of the same length as x_1, where the token at position i is replaced by y^i and all other tokens remain identical to those in x_1. $x_1^{\setminus i}$ is used to indicate all tokens in x_1 except for the token at position i.

The distribution-based objective aligns the model and true conditional distributions:

$$\mathcal{L}_{\text{distrib}}(\theta) = \mathbb{E}_{\omega(t)p(x_t)} \sum_{i=1}^{L} \mathbb{E}_{h(x_1^{\setminus i}|x_t)} \ell^i_{\text{distrib}}, \tag{41}$$

$$\ell^i_{\text{distrib}} = \mathbb{D}\left(p_{1|t}(x_1^i|x_1^{\setminus i}, x_t) \,\|\, p^\theta_{1|t}(x_1^i|x_1^{\setminus i}, x_t)\right), \tag{42}$$

where $\mathbb{D}(\cdot, \cdot)$ is a statistical divergence measures the differences between probability distributions. Shown in Proposition 3 of [33], with $h(x_1|x_t) = p^{1|t}(x_1|x_t)$, the two objectives are equivalent:

$$\mathcal{L}_{\text{score}}(\theta; \mathcal{D} = \mathcal{D}_{\text{GKL}}) \equiv \mathcal{L}_{\text{distrib}}(\theta; \mathbb{D} = V\mathbb{D}_{\text{KL}} + \mathbb{D}_{\text{IS}}), \tag{43}$$

where $\mathcal{D}_{\text{GKL}}$, $\mathbb{D}_{\text{KL}}$ and $\mathbb{D}_{\text{IS}}$ refer to the generalized Kullback–Leibler divergence, the Kullback–Leibler divergence and Itakura–Saito divergence, respectively.

By selecting different discrepancy measures and proposal distributions, there are different instantiations of the TCSM loss.

For instance, choosing the generalized KL divergence as the discrepancy $\mathcal{D}$ and the true conditional distribution $p_{1|t}(x_1|x_t)$ as the proposal $h(x_1|x_t)$, the score-based TCSM becomes

$$\ell^i_{\text{score}} = \left(-\log p^\theta_{1|t}(x^i_1|x_t) + \frac{1}{V p^\theta_{1|t}(x^i_1|x_t)} \right) + \frac{1}{V}\sum_y \log p^\theta_{1|t}(y|x_t); \tag{44}$$

choosing the KL divergence as the discrepancy $\mathbb{D}$ and the true conditional distribution $p_{1|t}(x_1|x_t)$ as the proposal $h(x_1|x_t)$, the distribution-based TCSM becomes

$$\ell^i_{\text{distrib}} = -\mathbb{E}_{p_{1|t}} \log p^\theta_{1|t}(x^i_1|x_t) + C, \tag{45}$$

where C is a constant. The distribution-based objective reduces to a cross-entropy loss, maximizing the pseudo-likelihood of $p^\theta_{1|t}$ under the true denoising distribution.

II.E.2 Connection with CE Loss

[32] leverages the theorem stating that two probability distributions are identical if and only if their conditional distributions are equal for all values of the condition. Based on this, the original marginal probability distributions in the concrete score are transformed into conditional probability distributions. Since both the numerator and denominator of the concrete score share the same functional form, they can be represented by a single neural network. That is, the concrete score can be expressed as:

$$p_t(X^d \mid x^{\setminus d}; \theta) \approx q_t(X^d \mid x^{\setminus d}) \Rightarrow \frac{q_t(y^d, x^{\setminus d})}{q_t(x^d, x^{\setminus d})} = \frac{q_t(X^d_t = y^d \mid x^{\setminus d})}{q_t(X^d_t = x^d \mid x^{\setminus d})} \approx \frac{p_t(X^d_t = y^d \mid x^{\setminus d}; \theta)}{p_t(X^d_t = x^d \mid x^{\setminus d}; \theta)}. \tag{46}$$

From this perspective, [32] propose the categorical ratio matching loss, which is a cross-entropy loss to fit conditional probabilities. Thus, [32] shows that training a neural network with a cross-entropy loss is also able to fit the concrete score.

[28] also connects the concrete score matching with the CE loss. Consider two discrete sequences x and y that only differ at position i. By Bayesian Theorem, the probability ratio $\frac{p_t(y)}{p_t(x)}$ can be rewritten as

$$\frac{p_t(y)}{p_t(x)} = \sum_{h \in \mathcal{V}} \frac{p_{t|0}(y^i \mid h)}{p_{t|0}(x^i \mid h)} \cdot p^i_{0|t}(h \mid x_t), \tag{47}$$

where $p_{t|0}(\cdot \mid h)$ is the known conditional transition probability in the forward process, and $p^i_{0|t}(h \mid x_t)$ is the posterior of the clean token h at position i given the noised sequence x_t. Thus, the concrete score can be parameterized as

$$s^i_\theta(x_t, t) = \sum_{h \in \mathcal{V}} \frac{p_{t|0}(y^i \mid h)}{p_{t|0}(x^i \mid h)} \cdot f^i_\theta(x_t, t)[h], \tag{48}$$

where $f^i_\theta(x_t, t)[h]$ models the likelihood that token h was the original clean token at position i. This allows training to proceed via a simple cross-entropy loss over the posterior $p_{0|t}$, rather than requiring explicit score estimation:

$$\mathcal{L}_{\text{CEDD}} = \mathbb{E}_t \mathbb{E}_{x_0, x_t} \sum_{i=1}^{L} w(t) \cdot \log f^i_\theta(x_t, t)[x^i_0], \tag{49}$$

where $w(t)$ is a timestep-dependent weighting function.

II.E.3 Time Independency

An issue with the Concrete Score is its dependency on time, which prevents the use of caching techniques for inference and results in lower efficiency. [22] shows that the concrete score in absorbing discrete diffusion can be reparameterized as a product of a time-dependent scalar and a conditional distribution over clean data. Specifically, in practice, R_t can be parameterized as the multiplication between a scalar function and a constant rate, *i.e.*, $R_t = \sigma(t)R$. let $x_i = [\text{M}]$ denote a masked token at position i. Then, the concrete score for replacing x_i with a token $x'_i \neq [\text{M}]$ is defined as:

$$\frac{p_t(x')}{p_t(x)} = \frac{p_t(x_1, \ldots, x'_i, \ldots)}{p_t(x_1, \ldots, x_i, \ldots)} = \frac{e^{-\bar{\sigma}(t)}}{1 - e^{-\bar{\sigma}(t)}} \cdot p_0(x'_i \mid x^{\text{UM}}), \tag{50}$$

where:

- $\bar{\sigma}(t) = \int_0^t \sigma(s)ds$ is the cumulative noise schedule,
- x^{UM} consists of all unmasked tokens of x,
- $p_0(x'_i \mid x^{\text{UM}})$ is the conditional distribution of the clean data.

This reparameterization removes the dependence on t from the model output, enabling the training of a time-independent network c_θ:

$$c_\theta(x)[i, \hat{x}^{(i)}] \approx p_0(\hat{x}^{(i)} \mid x^{\text{UM}}). \tag{51}$$

Such a model, termed RADD (Reparameterized Absorbing Discrete Diffusion), permits caching of the network outputs across steps when tokens remain unchanged, reducing the number of function evaluations (NFEs) during inference.

II.F Discrete Flow Matching (DFM)

Build upon the continuous-time Markov chain (CTMC) framework in Continuous Time Discrete Denoising Models [30], Discrete Flow Matching (DFM) [24] extends the Flow Matching paradigm to categorical sequence data. The model defines a probability path p_t interpolating between a source distribution p (e.g., all-mask sequences) and a target distribution q (e.g., the data distribution), such that $p_0 = p$ and $p_1 = q$.

Given a coupling distribution $\pi(x_0, x_1)$ between source and target sequences, the marginal probability at time t is defined as:

$$p_t(x) = \sum_{x_0, x_1 \in D} p_t(x \mid x_0, x_1)\pi(x_0, x_1), \tag{52}$$

where the conditional path $p_t(x \mid x_0, x_1)$ factorizes over positions:

$$p_t(x \mid x_0, x_1) = \prod_{i=1}^{N} p_t(x^i \mid x_0, x_1), \tag{53}$$

with token-level conditional paths defined as a convex combination of basis distributions:

$$p_t(x^i \mid x_0, x_1) = \sum_{j=1}^{m} \kappa^i_{t,j} w_j(x^i \mid x_0, x_1), \tag{54}$$

where $\kappa^i_{t,j} \geq 0$, $\sum_j \kappa^i_{t,j} = 1$ form a scheduler controlling the path dynamics.

The generative process is defined via probability velocity fields $\{u_t^i\}_{i=1}^N$ guiding transitions between states. The update rule for sampling is:

$$x_{t+h}^i \sim \delta_{x_t^i}(\cdot) + h u_t^i(\cdot, x_t), \tag{55}$$

where u_t is said to generate p_t if the process satisfies:

$$p_{t+h}(x) = p_t(x) - h \operatorname{div}_x(p_t u_t) + o(h), \tag{56}$$

with the discrete divergence operator:

$$\operatorname{div}_x(v) = \sum_{z \in D} [v(z, x) - v(x, z)]. \tag{57}$$

For the convex interpolation path:

$$p_t(x^i \mid x_0, x_1) = (1 - \kappa_t)\delta_{x_0}(x^i) + \kappa_t \delta_{x_1}(x^i), \tag{58}$$

the corresponding generating velocity takes the closed form:

$$u_t^i(x^i, z) = \frac{\dot{\kappa}_t}{1 - \kappa_t} \left[p_{1|t}(x^i \mid z) - \delta_z(x^i) \right], \tag{59}$$

where $p_{1|t}(x^i \mid z)$ is the probability denoiser: the conditional probability of the target token x_1^i given the current state z.

To estimate the posteriors required in the generative process, the model minimizes the cross-entropy loss:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{N} \mathbb{E}_{t,(x_0,x_1),x_t} \left[\log p_{1|t}(x_1^i \mid x_t; \theta) \right], \tag{60}$$

where $x_t \sim p_t(\cdot \mid x_0, x_1)$.

II.G Block Diffusion Models

Block Diffusion models (BD3-LMs) [34] provide a hybrid framework that interpolates between autoregressive language models and fully parallel diffusion models. Instead of denoising all tokens simultaneously, BD3-LMs segment the sequence into blocks and perform discrete denoising diffusion within each block, while conditioning autoregressively on all preceding blocks.

Given a sequence $x = (x_1, \ldots, x_L)$, BD3-LMs partition it into B blocks of length L', denoted $x = (x^{(1)}, \ldots, x^{(B)})$. The joint likelihood under a Block Diffusion model is factorized autoregressively across blocks:

$$\log p_\theta(x) = \sum_{b=1}^{B} \log p_\theta(x^{(b)} \mid x^{(<b)}), \tag{61}$$

where each conditional distribution $p_\theta(x^{(b)} \mid x^{(<b)})$ is modeled by a discrete diffusion process applied within block b:

$$p_\theta(x_s^{(b)} \mid x_t^{(b)}, x^{(<b)}) = \sum_{x^{(b)}} q(x_s^{(b)} \mid x_t^{(b)}, x^{(b)})\, p_\theta(x^{(b)} \mid x_t^{(b)}, x^{(<b)}), \tag{62}$$

where $q(\cdot \mid \cdot)$ is the forward noise process, and $p_\theta(x^{(b)} \mid x_t^{(b)}, x^{(<b)})$ is the learned denoising model.

The model is parameterized by a transformer f_θ with a block-causal attention mask. For each block $x^{(b)}$, the model predicts:

$$f_\theta\left(x_t^{(b)}, x^{(<b)}\right) \to \hat{x}_0^{(b)}. \tag{63}$$

During inference, block sampling proceeds sequentially over blocks, but parallel sampling is used within blocks. Block-wise KV caching can be used to avoid recomputing transformer states for previously generated blocks.

The training loss for the full sequence is obtained by summing the variational lower bound over all blocks:

$$-\log p_\theta(x) \le \mathcal{L}_{\text{BD}}(x; \theta) := \sum_{b=1}^{B} \mathcal{L}(x^{(b)}, x^{(<b)}; \theta), \tag{64}$$

where each $\mathcal{L}(x^{(b)}, x^{(<b)}; \theta)$ follows the discrete diffusion loss, optionally adapted to continuous time or masking processes.

III Large Discrete Diffusion Language and Multimodal Models

In this section, we provide a high-level overview of the representative works. In the following sections, we give detailed discussions on the training paradigms and inference-time decoding strategies of the models scaled to sizes comparable to LLMs. An evolutionary diagram of representative dLLM and dMLLM models is shown in Fig. 1.

III.A Discrete Diffusion Models around 1B

Discrete diffusion has emerged as a compelling alternative to continuous diffusion for modeling both discrete data and continuous data. [27] first introduces a diffusion process over binary variables. This idea is generalized to categorical variables by [14], who demonstrates its effectiveness on image generation. Building on these foundations, D3PM [13] proposes a more flexible family of noising schedules that extends discrete diffusion to a broader class of discrete spaces (see Section II.A). DiffusionBERT [35] explores training BERT [81] to reverse a discrete diffusion process with an absorbing state, introducing a token-aware noise schedule for the forward pass and methods to embed time-step information into BERT.

[15] derives an alternative yet equivalent formulation of the sampling from discrete diffusion process (see Section II.B) and introduces a new model family named Reparameterized Discrete Diffusion Models (RDMs). It reformulate the backward process of discrete diffusion in D3PM [13] into a two-stage sampling procedure and yield a greatly simplified training objective and enable more flexible decoding algorithms for text generation. Masked-Diffuse LM (MDLM) [36] propose to leverage the inherit linguistic features of texts to encourage the model to recover the text following an easy-first-generation nature, and directly predict the discrete token with cross-entropy loss to stabilize the intermediate diffusion steps. Diffusion-NAT [37] uses the pretrained BART [82] as the language backbone and unifies the inference process of pretrained language models and the denoising process of discrete diffusion models in a non-autoregressive manner, thus the BART model plays the role of the parameterized denoiser in discrete diffusion models. Furthermore, TESS [38] leverages a new form of self-conditioning and applies diffusion on the logit simplex instead of the learned embedding space. Plaid [39] takes the first step towards closing the likelihood gap between autoregressive and diffusion-based language models through algorithmic improvements, scaling laws, and increased compute. SEDD [40] generalize the idea of score matching into the discrete spaces by proposing a novel loss named score entropy, which can be integrated seamlessly

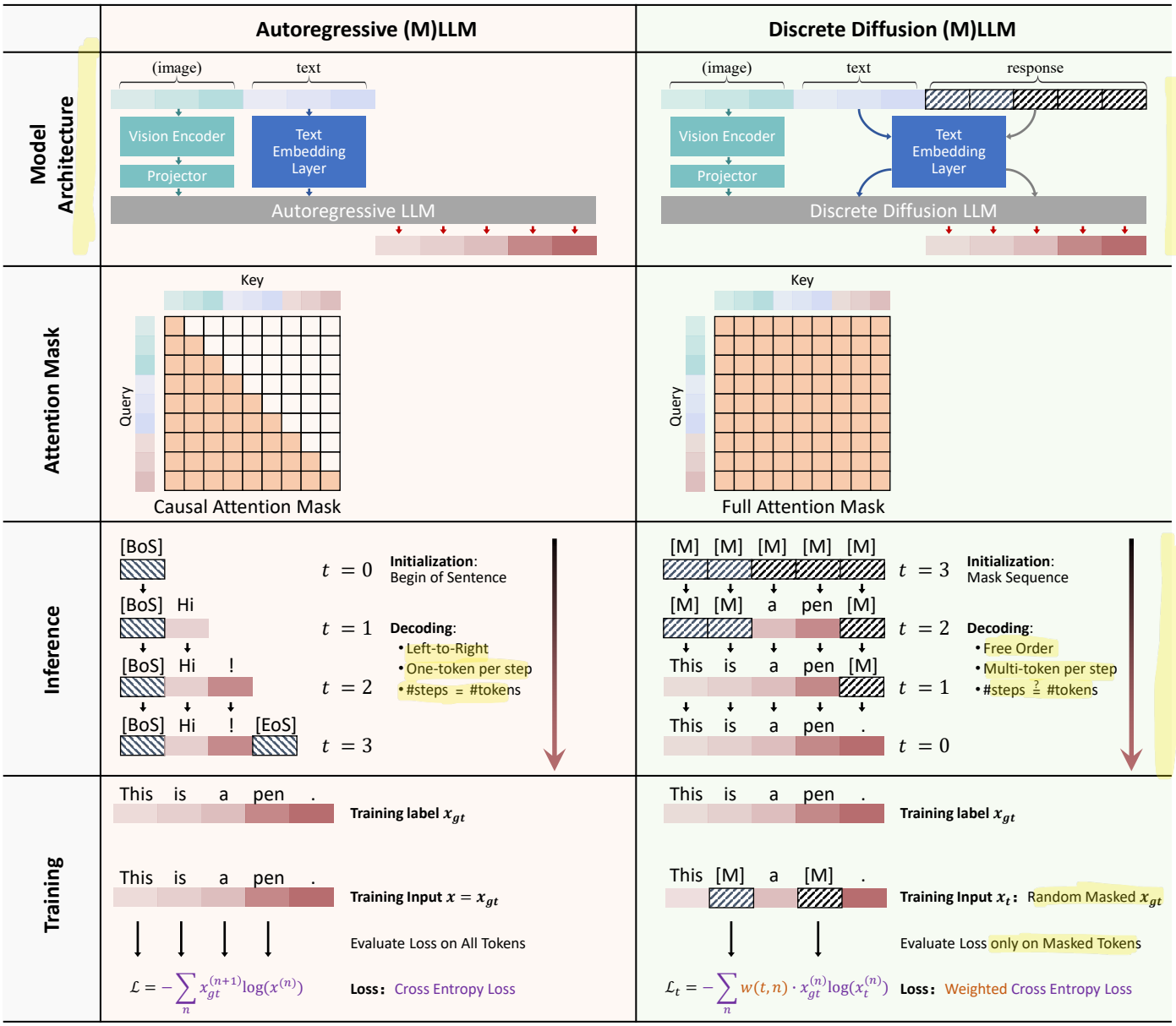


Fig. 3. This figure compares autoregressive models and discrete diffusion models from four perspectives. First, regarding model architecture, both models share the same network structure; the key difference lies in their generation mechanisms. In addition to the LLM, both MLLM and dMLLM require an additional vision encoder. In terms of the attention mask, the autoregressive model uses a causal attention mask, whereas the discrete diffusion model adopts a full (bidirectional) attention mask. During inference, the autoregressive model starts from a BoS token and generates tokens one by one from left to right. In contrast, the discrete diffusion model begins with a sequence of mask tokens and denoises all tokens in parallel. At each step, a subset of tokens is selected and replaced with non-mask tokens, continuing until no mask tokens remain. For training, the autoregressive model directly takes the input sequence and applies next-token prediction loss. The discrete diffusion model first randomly masks the input tokens and then computes a weighted cross-entropy loss over the masked positions.

to build discrete diffusion models and can significantly boost model performance.

MLDM [16] introduces substitution-based parameterization (SUBS) of the reverse unmasking diffusion process, allowing to derive a simple, continuous-time, Rao-Blackwellized objective that improves tightness and variance of the ELBO, further increasing performance. Besides the training design, the MLDM model is also equipped with fast samplers that support semi-autoregressive (SAR) generation. MD4 [17] begins by analyzing the properties of the forward process and its time reversal induced by the discrete diffusion model, then deriving a remarkably simple expression for the Evidence Lower Bound (ELBO), showing that it corresponds to a weighted time integral of cross-entropy losses, and can be rewritten in terms of signal-to-noise ratio, exhibiting invariance properties akin to those in continuous diffusion models. Building on this, a generalized masked diffusion model MD4 that allows state-dependent masking schedules is proposed, leading to improved predictive performance. For unified models, UniDisc [41] is a prior work on unified discrete diffusion model for text and image modalities. Conceptually, UniDisc treats an image and a caption as two token sequences (from discrete codebooks) and denoises them together. It uses a decoder-only Transformer with bidirectional attention, embedding both modalities via

appropriate positional encodings.

III.B Large Diffusion Language Models

III.B.1 LLaDA

[6] introduces LLaDA, the first discrete diffusion-based alternative to autoregressive LLMs. Instead of predicting one token at a time in a left-to-right manner, LLaDA generates text by gradually denoising masked text: a forward process randomly masks tokens in the input, and a reverse process uses a Transformer to predict the masked tokens. This Transformer mask predictor has an architecture similar to standard LLMs, but notably omits the causal masking, allowing bidirectional context since the model sees the entire sequence during prediction. LLaDA follows a principled generative modeling approach by optimizing a variational likelihood bound (ELBO) [83] rather than the exact log-likelihood. This design leverages the full context of text and the theoretical grounding of diffusion models to potentially achieve LLM capabilities beyond the autoregressive paradigm.

LLaDA demonstrates that dLLMs can attain competitive performance with same-size LLMs. Specifically, LLaDA 8B achieved comparable zero-/few-shot accuracy to LLaMA-3 8B [84] on a suite of 15 standard tasks [6], despite not using an autoregressive decoder. After instruction-tuning, LLaDA shows strong ability to follow complex instructions in dialogues, comparable to other fine-tuned LLMs. Notably, due to its bidirectional generation, LLaDA overcomes certain challenges that stump AR models: for example, it experimentally breaks the 'reversal curse' [6] and can correctly complete a reversed-order poem task, surpassing even GPT-4 in such evaluation [83]. These results establish that dLLMs are a viable and promising alternative to LLMs.

III.B.2 DIFFUSION-LLMs

[42] builds dLLMs in a multi-stage process. First, they leverage the intrinsic connection between masked language modeling and diffusion. A transformer is firstly pretrained on massive data using the masked LLM objective (similar to BERT [81]) to absorb world knowledge. This creates a strong base model with bidirectional representations. Then, they reprogram the pretrained masked LLM into a dLLM via a diffusive adaptation step. In practice, this means continuing to train the model to generate text by iterative denoising (rather than just single-step masking)—effectively converting the static masked LLM into a generative diffusion model. Finally, they perform supervised finetuning on specific tasks and crucially an instruction-tuning phase, where the model is trained on a broad set of natural language instructions and answers. This pipeline—scale up pretraining, adapt to diffusion generation, and then fine-tune on instructions—yields a dLLM that can follow human instructions to solve extensive tasks.

Results show that as the pretraining data and model size increased, the model's performance on downstream tasks rises steadily, mirroring classical scaling laws observed in AR LLMs [85, 86]. Moreover, after instruction-tuning, the dLLM begins to demonstrate zero-shot and few-shot learning abilities—it could tackle unseen tasks by following a natural language instruction, even with just a few in-context examples. This is a hallmark of modern LLMs like GPT-3 [87], and the fact that a dLLM can exhibit such in-context learning is a significant result. However, the study also notes that without the 'generative surgery' [42] of diffusive adaptation, simply scaling masked LLM pretraining alone can not confer strong generative abilities—underscoring that the adaptation step is essential. In summary, this work provides the first clear evidence that dLLMs can be general-purpose performers. It highlights the importance of scale and instruction-tuning, laying the groundwork for subsequent large diffusion models.

III.B.3 DiffuGPT & DiffuLLaMA

At the meantime, [18] proposes to convert a pretrained AR transformer (like GPT-2 [88] or LLaMA [89]) into a dLLM, thereby avoiding the need to train a large model purely from scratch. They demonstrate a simple adaptation process: continue training the AR model on a masking-based diffusion objective while reusing its learned weights. In practice, they transform AR-based language models ranging from 127M to 7B parameters into diffusion-based models dubbed 'DiffuGPT' (from GPT-2) and 'DiffuLLaMA' (from LLaMA). Crucially, [18] gives theoretical connections between AR model's next-token prediction and diffusion model's denoising objective, which allowed them to align the two paradigms during adaptation. By using the AR model's knowledge as an initialization, the diffusion model could be scaled up with far less data (under 200 billion tokens of additional training, vs. trillions for a scratch model). This work makes large dLLMs feasible with limited compute, addressing a key bottleneck in scaling.

The DiffuGPT & DiffuLLaMA model series shows comparable performance of AR LLMs. For example, DiffuGPT-355M not only exceeds previous dLLMs of similar size, but even outperforms the original GPT-2 on most evaluated tasks. The adapted DiffuLLaMA-7B achieves competitive results close to LLaMA-2-7B [89] (its AR counterpart), although it slightly trails the AR model on some benchmarks (likely due to having seen fewer total tokens). Notably, these dLLMs retain important abilities of the base LLMs: they produce fluent, coherent text and demonstrate in-context learning on par with AR models. Thanks to their bidirectional nature, they can also natively perform infilling without special prompt engineering or reordering, which AR models typically struggle with. In inference, DiffuGPT & DiffuLLaMA can trade generation speed for quality by adjusting the number of diffusion iterations, and the adapted models often require fewer refinement steps to reach good fluency compared to other dLLMs. Overall, scaling dLLMs via adaptation demonstrates a practical path to build high-performing dLLMs by capitalizing on the vast knowledge in pretrained AR LLMs. Besides, the released suite of adapted models (127M, 355M, 7B) provides evidence that dLLMs can be scaled up and offering a foundation for further research.

III.B.4 DREAM

DREAM 7B [7] is one of the most powerful open-source dLLMs to date. The name 'DREAM' underscores its focus on diffusion for reasoning and modeling—it is specifically designed to excel at complex reasoning tasks while remaining efficient and scalable. DREAM 7B achieves performance on par with, or exceeding, autoregressive models of similar size (e.g. it matches LLaMA3 8B [84] and Qwen 2.5-7B [90] on

many benchmarks). A key to DREAM's success is an optimized training recipe distilled from extensive experiments at smaller scales. [7] carefully explores the design choices on a 1B model and identify two especially valuable components: (1) AR weight initialization and (2) context adaptive noise scheduling.

For (1), DREAM's transformer is initialized with the pre-trained weights of a strong AR model (Qwen 2.5-7B and LLaMA3-8B). This is experimentally proved to be more effective than training from scratch, as it preserves useful left-to-right knowledge during early training. Setting a moderate learning rate is also crucial: too high would wash out the AR-inherited knowledge, while too low would hinder learning diffusion-specific patterns. By tuning this properly, DREAM leverages the existing AR knowledge to accelerate any-order (bidirectional) learning, reducing the data needed for training. For (2), DREAM introduce a novel context-adaptive, token-level noise rescheduling mechanism. In conventional discrete diffusion training, a single noise level is sampled for the whole sequence, meaning every token is masked with the same probability. This can be suboptimal: tokens with different context availability (e.g. a token whose neighbors are all masked vs. one whose neighbors are mostly visible) effectively experience different noise severity. Some tokens might be too difficult (no context to infer from) or too easy (fully predictable from context) at a given global noise level. To address this, DREAM measures each token's context information after masking and then dynamically reassigns an individual noise level to each token. In essence, the model reschedules the diffusion timestep per token: a token with little context left is treated as if at a higher effective noise (so the model knows it's very uncertain), whereas a token with rich context might be treated as lower noise. This token-level noise adaptation provides more precise guidance for learning each token, leading to better overall training efficiency and performance.

III.B.5 LLaDA 1.5

While LLaDA demonstrates strong performance after supervised finetuning, aligning a dLLM with human preferences (akin to RLHF [91] for AR models) remains challenging. LLaDA 1.5 [43] specifically addresses this by introducing Variance-Reduced Preference Optimization (VRPO) for dLLMs. The core difficulty is that reinforcement learning or direct preference optimization for a diffusion model requires estimating the model's log-likelihood for given outputs—but diffusion models cannot compute an exact log-probability and instead rely on ELBO approximations. These ELBO-based likelihood estimates are noisy (high-variance), which in turn makes preference-based gradient updates extremely unstable. LLaDA 1.5 presents a theoretical analysis of the variance and bias introduced by using ELBO estimates in preference optimization, and they derive bounds showing that reducing the variance of these estimates is key to improving alignment training. Building on this, VRPO provides unbiased variance reduction techniques to stabilize training. One technique is simply to increase the Monte Carlo sample size (using more random draws of diffusion time and mask patterns) when estimating the ELBO for a given candidate output. More cleverly, VRPO allocates the sampling budget in an optimal way: their analysis shows that it is best to sample many different diffusion timesteps but only one mask per timestep. This *optimal allocation* minimizes variance for a fixed total number of samples.

Another technique in VRPO is *antithetic sampling* for preference comparisons. In preference optimization (e.g. Direct Preference Optimization or DPO [92]), the model compares the log-likelihoods of a better output vs a worse output. VRPO uses the same random noise for both outputs when estimating their ELBO scores. By sharing diffusion timesteps and mask patterns between the winning and losing outputs, the random errors in their log-likelihood estimates tend to cancel out, greatly reducing variance in the difference. This is analogous to using the same random seed for paired evaluations to get a more reliable comparison. With these methods—increased samples, optimal allocation, and antithetic sampling—VRPO achieves a significant stability improvement in aligning the diffusion model. [43] applys VRPO to LLaDA and fine-tune it with a preference model (reward model) on reasoning, math, code, and some other general alignment tasks. The resulting LLaDA 1.5 shows a marked performance jump over the SFT-only LLaDA on all evaluated fronts. In summary, LLaDA 1.5 demonstrates that RLHF-style alignment is feasible for dLLMs and that addressing the ELBO variance issue is the key perspective.

III.B.6 TESS 2

TESS 2 [44] is a dLLM that is not only large-scale but also instruction-following and general-purpose. The training recipe for TESS 2 is a culmination of ideas from prior works: it starts by adapting a powerful AR base model via continued pretraining on the diffusion objective, and then applies thorough instruction-tuning to that adapted model. Tess 2 starts from using some of the strongest available AR LLMs as the starting point and performs adaptation similar to DiffuLLaMA [18]. By doing so, TESS 2 inherits a high-quality knowledge base and linguistic capability from the AR model, which is critical—the study also finds that both the adaptation procedure and the choice of base model are crucial for a good dLLM. After adapting the base model to a diffusion one, an instruction-tuning process is conducted. This step helps the model to follow a wide range of natural language instructions, similar to how ChatGPT or InstructGPT [93] are trained, thus making TESS 2 adept at answering user queries, explaining reasoning, etc.

III.C Large Diffusion Multimodal Models

III.C.1 Dimple

Dimple [8] is one of the first Discrete Diffusion Multimodal Large Language Models (dMLLMs). Its base architecture (vision encoder + transformer LLM) resembles existing vision-language models (e.g. Qwen-VL [94], LLaVA [95, 96, 97]). One of the Dimple's key innovations is its two-stage hybrid training. In Stage 1, with the weights of Dream 7B [7] as an initialization, it is fine-tuned autoregressively on vision-language instruction data (for alignment and instruction following). In Stage 2, it is then further fine-tuned with a discrete diffusion objective. This hybrid approach is devised because pure diffusion training is found to be unstable (leading to length bias and performance drop). By warming up with autoregressive training first, Dimple-7B achieves stable training and eventually surpasses the fully-autoregressive models.

During inference, Dimple introduces a *confident decoding* strategy for efficiency: the model dynamically chooses how many tokens to fill in at each step based on model confidence (see Sec. V.A). Empirically, this reduces the number of iterations to about $\frac{\text{response length}}{3}$. Dimple also re-implements an autoregressive *prefilling* trick: by filling in some tokens from the existing context, it speeds up inference by about 1.5×–7× with minimal impact on quality. Under the same training budget and dataset as LLaVA-NEXT [97], Dimple-7B achieves higher aggregate scores on multimodal benchmarks than LLaVA-NEXT-7B. This result shows that with a proper hybrid training recipe, a discrete dMLLM can match or exceed strong autoregressive baselines on vision-language tasks.

III.C.2 LaViDa

LaViDa [10] consists of a vision encoder (e.g. SigLIP-400M [98]) and a discrete diffusion Transformer. The input image is split into five views (four quadrants plus the full image); each view is encoded, average-pooled to reduce length, and concatenated into a single 1D context. An MLP connector projects these image embeddings into the language model's space. The language model is a standard discrete dLLM (such as LLaDA-8B or Dream-7B). In effect, the model treats image tokens and text tokens uniformly in a masked-denoising framework.

The Training of LaViDa leverages the masked diffusion objective over pairs (image, prompt, answer). A key innovation is complementary masking: for each training sample, two distinct mask patterns are created so that each token is masked in one of the two versions. This ensures that even short or rare answer tokens (e.g. object names in vision tasks) contribute to the loss and all tokens are learned efficiently, improving alignment between the visual encoder and the language model. During inference, LaViDa begins with a fully-masked output sequence and iteratively denoise these tokens. It employs a special Prefix-DLM [10] attention mask so that the encoded image and prompt tokens can be cached and reused. The model also uses a timestep-shifting schedule to improve sample quality. Together, these yield faster, more controllable decoding, and supports bidirectional infilling for format constraints.

III.C.3 LLaDA-V

LLaDA-V [9] is a purely dMLLM built on LLaDA [6]. It integrates a vision encoder and an MLP connector that projects visual features into the language embedding space, which helps the dLLM to process image inputs alongside text. LLaDA-V also uses masked diffusion rather than autoregressive decoding. The model undergoes a visual instruction-tuning process: it is fine-tuned on multimodal instruction-response data using the masked diffusion objective. Since the core model is diffusion-based, LLaDA-V naturally supports parallel decoding and controllable infilling. Even though the base language model of LLaDA-V is less performant on pure text tasks, experiments show it is highly competitive on vision-language benchmarks. When trained on the same multimodal data, LLaDA-V matches or approaches strong AR baselines (e.g. LLaMA3-V) and narrows the gap to the top models like Qwen2.5-VL. It achieves state-of-the-art results in multimodal understanding tasks, demonstrating that dMLLMs can be as effective as or better than hybrid or AR approaches in vision-language domains.

III.D Large Unified Model

III.D.1 MMaDA

MMaDA [11] employs a unified diffusion architecture with a shared probabilistic formulation across modalities. It uses a single diffusion-based transformer for all data types (text, images, etc.), rather than separate encoders for each modality. This modality-agnostic design simplifies the model by eliminating modality-specific components.

During training, MMaDA is fine-tuned with a *mixed long chain-of-thought* strategy. Reasoning steps from both text and vision tasks are converted into a unified CoT format so that the model learns aligned reasoning across modalities. For example, the rationale for answering a visual question is interleaved into the textual input. This CoT alignment provides a form of cold-start for the final reinforcement learning (RL) stage, allowing complex multi-step reasoning from the outset. Finally, MMaDA proposes a unified policy-gradient-based RL algorithm named *UniGRPO*. By incorporating diversified reward modeling, UniGRPO unifies the post-training process across both reasoning and generation tasks, leading to consistent performance improvements.

III.D.2 FUDOKI

FUDOKI [45] is a unified multimodal model built on discrete flow matching [99]. It uses a metric-induced probability path with kinetic-optimal velocities [99], which significantly improves over simple masking by enabling continuous self-correction during generation. For efficiency, FUDOKI is initialized from a pretrained AR-based multimodal LLM (Janus-1.5B [100]) and then transferred to the discrete flow matching framework. Architecturally, FUDOKI firstly replaces the standard causal mask with a full attention mask so that every token can attend to all others and thus improving global context modeling. Also, it shifts the output logits by one position (same as AR models) to retain the next-token prediction capability. In addition, unlike continuous diffusion models, FUDOKI does not use extra time-embedding layers; instead the model implicitly infers the corruption timestep from the input sequence. For input modalities, text is tokenized normally, while images are handled by separate pipelines: a semantic encoder (SigLIP [98]) extracts features for image understanding, and a pixel encoder/decoder [101] converts images to/from discrete image tokens for generation. At the output stage, FUDOKI has two output heads—one predicting text tokens and one predicting image tokens—and selects the appropriate head depending on the target modality during inference.

III.D.3 Muddit

Muddit [46] is a unified model that uses purely discrete diffusion to handle text and images under one framework. The architecture of Muddit comprises a single multimodal diffusion transformer (MM-DiT) [46], plus encoders/decoders for each modality. The MM-DiT follows a dual-/single-stream design (similar to FLUX [102]) and is initialized from the pretrained Meissonic [103]. Inputs are quantized into a shared token space: images are encoded by a pretrained VQ-VAE [104] into discrete codebook indices, and text is encoded by a CLIP text

encoder [105]. During training and inference, the MM-DiT predicts masked tokens in this joint space. A lightweight linear head maps the predicted tokens to actual text tokens for text output, while the VQ-VAE decoder reconstructs pixels from image tokens. Thus a single generator handles both text and image tokens.

IV Training Techniques

In this section, we summarize the techniques employed during the training of diffusion language models (dLLMs) and diffusion multimodal language models (dMLLMs).

IV.A Challenges

The challenges in training dLLMs stem from low corpus utilization, high variance due to stochastic masking, degraded generation quality, and length bias.

IV.A.1 Low Corpus Utilization

Unlike autoregressive training, where each token in the answer sequence contributes to the learning signal, discrete diffusion training applies supervision only to a randomly selected subset of tokens at each time step. Given an input sequence x of length $L = L_{\text{prompt}} + L_{\text{answer}}$, diffusion training samples a timestep $t \in [1, T]$ and computes loss only over the masked tokens at that timestep. This leads to sparse supervision across training samples, resulting in inefficient utilization of the available corpus.

IV.A.2 Random Sampling of Time Index

In diffusion training, the time index t is randomly sampled for each training instance. As a result, only a single generation step is supervised per sample, while the decoding process at inference time typically involves multiple time steps. This mismatch introduces a coverage gap between training and inference: although decoding requires iterative refinement over many steps, training provides gradient signals for only one of those steps per instance.

IV.A.3 Length Bias

Diffusion models generate sequences of a pre-specified length, without a natural stopping mechanism like the `[EOS]` token in autoregressive models. Consequently, when trained purely with diffusion objectives, the model exhibits sensitivity to the target sequence length. Empirical results reveal significant performance fluctuation as the generation length varies, as shown in [8]. This phenomenon, referred to as *length bias*, indicates that diffusion models struggle to generalize across different output lengths under pure diffusion training.

IV.B Initialization Technique

To address the inefficiencies and instabilities in training dLLMs and dMLLMs, several works adopt advanced initialization strategies that convert the full diffusion training process into a fine-tuning task. This approach accelerates convergence and enhances final model performance.

IV.B.1 BERT Initialization

The diffusion generation process can be interpreted as a multi-step masked language modeling (MLM) procedure. [35] initializes diffusion models from pretrained BERT.

IV.B.2 Autoregressive Model Initialization

[18, 7] have explored direct adaptation from autoregressive language models by aligning the training objectives of the two paradigms. A key technique enabling this transition is the *shift operation*. In standard diffusion training, the model predicts the original token x_0 from its corrupted version x_t at each timestep. However, this formulation differs from AR training, where each hidden state h_i is trained to predict the next token x_{i+1} in a left-to-right fashion. To bridge this gap, [18, 7] propose shifting the output logits of the diffusion model by one position, such that the model's prediction at position i corresponds to token x_{i+1}. This allows the diffusion model to be initialized with pretrained autoregressive models.

IV.B.3 Autoregressive-then-Diffusion Training

Dimple [8] uses an *autoregressive-then-diffusion* training approach, demonstrating notable performance improvements for DMLLM. The Dimple training pipeline is divided into two distinct phases:

- **Phase I: Autoregressive Training.** In the first phase, Dimple is treated as an autoregressive model. It is trained using a causal attention mask and next-token prediction loss.
- **Phase II: Diffusion Fine-tuning.** After autoregressive training, the model transitions to a discrete diffusion training regime. Full attention masks and timestep-dependent masked language modeling losses are re-enabled.

This hybrid approach addresses several known limitations of pure diffusion training—such as low corpus utilization and length bias—by initializing the model with strong AR priors. With this hybrid training approach, Dimple achieves better alignment, improved instruction tuning, and enhanced robustness to response length variations.

IV.C Complementary Masking Technique

In [10], to ensure that all tokens participate in training, complementary masking constructs two complementary masked versions of each input sequence: X_t and X_t^C. These versions have non-overlapping masked spans. For example, consider the sentence:

``` ``The answer is dog.'' ```

One masked variant might be:

``` ``The [M] [M] dog.'' ```

and its complement:

``` ``[M] answer is [M].'' ```

This setup ensures that all tokens are eventually masked and optimized over the course of training.

IV.D Masking Scheduling Technique

Mask schedule governs the corruption process in the forward diffusion formulation. Specifically, the schedule defines the corruption level α_t at each timestep t, thereby determining the proportion of tokens masked during training. An effective schedule balances learning stability and generation quality by controlling the signal-to-noise ratio across timesteps.
```

IV.D.1 Uniform Masking Scheduling

Given a timestep $t \in [0, 1]$, the forward process defines the corruption as:

$$q(x_t \mid x_0) = \alpha_t x_0 + (1 - \alpha_t)m, \tag{65}$$

where m is the one-hot [MASK] token. The loss at each step is reweighted according to:

$$w_t = \frac{\alpha'_t}{1 - \alpha_t}, \tag{66}$$

where $\alpha'_t = \frac{d\alpha_t}{dt}$ is the derivative of α_t with respect to time. Several scheduling strategies have been proposed and empirically studied. The corresponding schedules are plotted in Fig. 4.

- **Linear Schedule [13]:**

$$\alpha_t = 1 - t, \tag{67}$$

$$w_t = -\frac{1}{t}. \tag{68}$$

- **Geometric Schedule [40, 17]:**

$$\alpha_t = \exp\left(-\bar{\beta}_{\min}^{1-t}\bar{\beta}_{\max}^{t}\right), \tag{69}$$

$$w_t = \left(\frac{\exp\left(-\bar{\beta}_{\min}^{1-t}\bar{\beta}_{\max}^{t}\right)}{1 - \exp\left(-\bar{\beta}_{\min}^{1-t}\bar{\beta}_{\max}^{t}\right)}\right)\bar{\beta}_{\min}^{1-t}\bar{\beta}_{\max}^{t}\log\left(\frac{\bar{\beta}_{\min}}{\bar{\beta}_{\max}}\right). \tag{70}$$

- **Cosine Schedule [47]:**

$$\alpha_t = 1 - \cos\left(\frac{\pi}{2}(1 - t)\right), \tag{71}$$

$$w_t = -\frac{\pi}{2}\tan\left(\frac{\pi}{2}(1 - t)\right). \tag{72}$$

- **Polynomial Schedule [17]:**

$$\alpha_t = 1 - t^r, \tag{73}$$

$$w_t = -\frac{r}{t}. \tag{74}$$

IV.D.2 Token-wise Masking Scheduling

Uniform masking scheduling uses the same scheduling for all tokens, which ignores the inherent variation in informativeness among different tokens. For example, different tokens carry different amounts of information, typically measured by entropy and dLLMs exhibit an *easy-first* decoding behavior—*i.e.*, common, low-entropy tokens are easier to predict earlier. [35] introduces a *token-wise masking schedule*.

The schedule defines a custom corruption probability α_t^i for each token position i at timestep t, determined by:

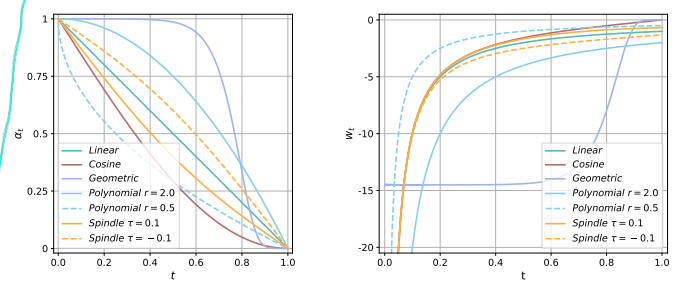
$$\alpha_t^i = 1 - \frac{t}{T} - S(t) \cdot \tilde{H}(x_0^i), \tag{75}$$

$$S(t) = \lambda \sin\left(\frac{t\pi}{T}\right), \tag{76}$$

$$\tilde{H}(x_0^i) = 1 - \frac{\sum_{j=1}^{n} H(x_0^j)}{n \cdot H(x_0^i)}, \tag{77}$$

where

- $H(x_0^i)$ denotes the entropy of the i-th token, measuring its information content.
- $S(t)$ is a sinusoidal scaling function that ensures zero informativeness contribution at $t = 0$ and $t = T$.



(a) α_t Schedules (b) w_t Schedules

Fig. 4. Different schedules for α_t and w_t. To unify notation, we also transformed the spindle schedule from the discrete-time format used in the original paper to a continuous-time format. The revised formulation is as follows: $\alpha_t = 1-t-\tau\sin(\pi t)$ and $w_t = -\frac{1+\tau\pi\cos(\pi t)}{t+\tau\sin(\pi t)}$, where τ corresponds to the original $\lambda\tilde{H}$.

- λ is a hyperparameter controlling the strength of entropy influence.

The above formulation defines a *Spindle-shaped Noise Schedule*, where the entropy-aware corruption pattern resembles a spindle curve: more informative tokens are masked earlier and less informative ones later. This design ensures that low-entropy, easier-to-predict tokens are decoded first, leading to more coherent and stable sequence generation.

IV.E Addressing Training-Testing Discrepancy

[48] mentioned a discrepancy between training and inference of dLLM. During training, the model receives ground-truth noisy tokens as input, while at inference time, the inputs of the model are the previously predicted tokens. To address this, [48] propose the two-step loss.

Let x_0 denote the ground truth sequence. During training, a time step t is randomly selected, and a noised version x_t is generated from x_0 via a diffusion process. The model then predicts the original sequence $\hat{x}_0 = f_\theta(x_t)$. Subsequently, a second input $\hat{x}_{t-1}$ is generated by applying the forward diffusion transition matrixto the predicted sequence $\hat{x}_0$ and the model again attempts to recover the ground truth $\hat{\hat{x}}_0 = f_\theta(\hat{x}_{t-1})$. The two-step loss is calculated between the ground truth x_0 and the twice-denoised output $\hat{\hat{x}}_0$.

To ease training in early stages, the model does not always use this two-step strategy. Instead, a mixed strategy is adopted. With probability $1 - p_k$, the two-step strategy is used and the loss is evaluated between $\hat{\hat{x}}_0$ and x_0. With probability p_k, the conventional one-step strategy is used and the loss is evaluated between $\hat{x}_0$ and x_0. p_k is set to be linearly increasing along the training step k.

IV.F Reweighting Technique

IV.F.1 Multi-Granularity Diffusion Modeling

Multi-Granularity Diffusion Modeling (MGDM) [49] introduces an additional token-level reweighting factor $v(x_{t,n})$, yielding a refined training loss:

$$\mathcal{L}_{\text{MGDM}} = \sum_{n=1}^{N}\sum_{t=1}^{T} w(t) \cdot v(x_{t,n}) \cdot \ell(x_0, x_t, n; \theta), \tag{78}$$

where $\ell(x_0, x_t, n; \theta)$ is the CE loss on the n-th token, and the adaptive token-level weight is defined as:

$$v(x_{t,n}) = \alpha(1 - \exp(-\ell(x_0, x_t, n; \theta)))^{\beta}, \tag{79}$$

with hyperparameters $\alpha > 0$, $\beta > 0$. This reweighting assigns larger weights to harder tokens (i.e., those with higher loss), thereby prioritizing difficult subgoals during training and accelerating convergence.

IV.G Distillation through Dimensional Correlations

To enable efficient few-step or even one-step generation while preserving performance, Di4C [50] explicitly models inter-dimensional correlations and introduces two principled loss functions: *Distillation Loss* and *Consistency Loss*. These losses are grounded in distributional matching and multi-path coherence.

IV.G.1 Distillation Loss

The distillation loss transfers the probabilistic knowledge from a teacher model, which performs multi-step denoising, to a student model that aims to generate outputs in fewer steps. This is achieved by aligning the student's predicted posterior with that of the teacher at an intermediate noise level δ. Formally, the loss is defined as:

$$\mathcal{L}_{\text{distil}}(\theta; \psi, r_\delta, \delta) = \mathbb{E}_{x_\delta \sim r_\delta}\Big[\mathrm{D}_{\mathrm{KL}}\big(p^{\psi}_{0|\delta}(\cdot \mid x_\delta) \,\|\, p^{\theta}_{0|\delta}(\cdot \mid x_\delta)\big)\Big] \tag{80}$$

where $p^{\psi}_{0|\delta}(\cdot|x_\delta)$ is the posterior distribution over clean data x_0 given intermediate noisy input x_δ under the teacher model; $p^{\theta}_{0|\delta}(\cdot|x_\delta)$ is the student model's predicted posterior; r_δ is a reference distribution over noisy states (typically chosen to match the forward diffusion at timestep δ). This loss encourages to transfer the full-step generative knowledge from teacher to student by matching posteriors.

IV.G.2 Consistency Loss

The consistency loss ensures that the student model behaves consistently when predicting from different intermediate noise levels. Specifically, if x_t is a noisy sample at step t, there should be agreement between: 1). First denoising $x_t \to x_u$ via the teacher, then $x_u \to x_s$ via the student; 2). Directly predicting $x_t \to x_s$ via the student. This intuition is captured by the following KL divergence:

$$\mathcal{L}_{\text{consis}}(\theta; \psi, r_t, s, u, t) = \mathbb{E}_{x_t \sim r_t}\Big[\mathrm{D}_{\mathrm{KL}}\big((p^{\theta}_{s|u} \circ p^{\psi}_{u|t})(\cdot \mid x_t) \,\|\, p^{\theta}_{s|t}(\cdot \mid x_t)\big)\Big] \tag{81}$$

where $p^{\psi}_{u|t}(x_u|x_t)$ is the teacher's distribution from timestep t to u; $p^{\theta}_{s|u}(x_s|x_u)$ is the student's distribution from u to s; $p^{\theta}_{s|t}(x_s|x_t)$ is the direct prediction by the student; $(p^{\theta}_{s|u} \circ p^{\psi}_{u|t})(\cdot|x_t)$ denotes the composite distribution over x_s via intermediate x_u. This loss enforces functional coherence across generation paths, capturing multi-dimensional correlations without assuming independence.

V Inference Techniques

V.A Unmasking Techniques

In dLLMs and dMLLMs, the model predicts all the response tokens at each step. However, only a subset of the masked tokens are selected to be unmasked at every iteration, while the remainder remain masked. This iterative unmasking process gradually reveals the complete output sequence. The core challenges in this procedure are: (1) determining which tokens to unmask at each iteration, and (2) deciding how many tokens should be unmasked at each iteration. Table I provides a summary of the unmasking strategies adopted by representative models. Figure 5 provides a detailed illustration of each type of unmasking strategy. In this section, we discuss each category in detail and describe the specific unmasking strategies proposed in each work.

V.A.1 Discrete-Time Unmasking

V.A.1.(a) Random Unmasking

The simplest strategy is to randomly select s_t masked tokens to unmask at step t. The value of s_t can be fixed across steps or controlled by a scheduling function as discussed in the training techniques, such as cosine scheduling [9]. Importantly, the inference-time masking schedule does not have to match that used during training. In fact, practically, it is often treated as a tunable hyperparameter that can vary across different tasks.

V.A.1.(b) Metric-Based Unmasking

Rather than relying on random selection, metric-based strategies assign a metric value to each token prediction and select tokens to be unmasked based on the metric value.

Let $p \in \mathbb{R}^K$ be the predicted probability distribution over the vocabulary for a given token, where K is the vocabulary size. The following metrics are commonly used [7]:

- **Maximum Probability (Confidence):**

$$c = \max(p), \tag{82}$$

  indicating the model's certainty about the most likely token. [51] provides a theoretical analysis of the equivalence between the parallel decoding using confidence and sequential decoding using confidence.
- **Margin:**

$$c = p_{\text{top1}} - p_{\text{top2}}, \tag{83}$$

  where p_{top1} and p_{top2} are the first and second highest probabilities, respectively. This measures how dominant the top prediction is.
- **Negative Entropy:**

$$c = -\sum_{i=1}^{K} p_i \log(p_i + \epsilon), \tag{84}$$

  with a small constant ϵ for numerical stability. This captures the peakedness of the distribution.

V.A.1.(c) Selection Policies

After obtaining the metric value for each token, the diffusion model performs selection based on different policies.

- **Top-s_t Strategy [13]:** Select the s_t tokens with the highest confidence scores for unmasking. The value of s_t follows the same scheduling principles as in random unmasking.

TABLE I
THE INFERENCE TECHNIQUES USED DURING THE INFERENCE PROCESS OF EACH MODEL.

Inference Technique		Language Model						Multimodal Model			Unified Model		
		LLaDA [6]	DIFFUSION-LLMs [42]	DiffuLLaMA [18]	DREAM [7]	LLaDA 1.5 [43]	TESS 2 [44]	Dimple [8]	LaViDa [10]	LLaDA-V [9]	MMaDA [11]	FUDOKI [45]	Muddit [46]
Unmasking	Top-s_t Strategy	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Confident Decoding							✓					
	Semi-Autoregressive Decoding	✓				✓							
Prefilling								✓	✓				
Reward Model Guidance							✓						

- **Confident Decoding:** As introduced in Dimple [8], this strategy dynamically selects the number of tokens to unmask based on a fixed confidence threshold $\gamma \in (0, 1)$. The motivation of this approach is that decoding should adapt to the semantic structure of the text: some steps may allow many tokens to be confidently predicted, while others may necessitate more caution. Thus, the number of decoded token should be adaptively adjusted at each step. At step t, the set of positions to decode is defined as:

$$\mathcal{I}_t = \{i \mid c_t^{(i)} \geq \gamma\}, \tag{85}$$

where $c_t^{(i)}$ is the confidence score at position i. If $\mathcal{I}_t$ is non-empty, all tokens in $\mathcal{I}_t$ are unmasked. Otherwise, only the token with the highest confidence is selected. This approach enables:
  - decoding multiple tokens in parallel when the model is confident, improving efficiency;
  - avoiding low-confidence predictions, preserving generation quality.
- **Local Unmasking:** The above strategies can generally be categorized as *global unmasking strategies*. When selection requires sorting, *global unmasking strategies* aggregate the metric values of all tokens and perform sorting or selection on this global set. However, the unmasking process does not necessarily have to be global; it can also be local. Specifically, all tokens can be divided into multiple subgroups, with sorting and selection performed independently within each subgroup, or different unmasking strategies applied across different subgroups. A typical example is the semi-autoregressive decoding strategy [6]. This approach divides the full response into multiple blocks, similar to block diffusion. During each forward pass, predictions are generated for all blocks simultaneously. However, the unmasking of tokens follows a left-to-right, block-by-block order. Tokens in the next block are only allowed to be unmasked once all tokens in the previous block have been unmasked.

Across these methods, a common pattern arises: (1) score every masked position by confidence; (2) select a subset according to a rule (threshold, top-k, decaying ratio, block order, etc.) and unmask them; (3) repeat until no masks remain. Such metric-based schedules prioritise 'easy' tokens first, reducing error propagation and enabling highly parallel generation.

V.A.2 Continuous-Time Unmasking (Flow Matching)

In continuous-time inference under the discrete flow matching framework (e.g., FUDOKI [45]), unmasking is modeled as a stochastic jump process along a probability path.

Let $\mathbf{x}_t$ denote the current sequence state at time $t \in [0, 1]$, and let $\mathbf{x}_1$ be the target sequence. For each token position i, the update from $\mathbf{x}_t$ to $\mathbf{x}_{t+h}$ is governed by:

1) Sample a predicted target $\hat{x}_1^{(i)} \sim p_\theta(x_1^{(i)} \mid \mathbf{x}_t)$;
2) Compute a total transition rate

$$\lambda^{(i)} = \sum_{x \neq x_t^{(i)}} u_t^{(i)}(x, x_t^{(i)} \mid \hat{x}_1^{(i)});$$

3) Draw a random variable $Z \sim \mathcal{U}(0, 1)$;
4) If $Z \leq 1 - e^{-h\lambda^{(i)}}$, update $x_t^{(i)}$ by sampling from:

$$x_{t+h}^{(i)} \sim \frac{u_t^{(i)}(x, x_t^{(i)} \mid \hat{x}_1^{(i)})}{\lambda^{(i)}}.$$

This process dynamically determines which tokens to update (i.e., unmask) based on local transition rates. The higher the rate $\lambda^{(i)}$, the more likely token i will jump to a new value, allowing the model to continuously refine its predictions in a semantically meaningful way.

V.B Remasking Techniques

For discrete diffusion models based on absorbing states, during inference, once a token is unmasked (i.e., decoded), it remains unchanged in subsequent steps. This static behavior limits the model's capacity to revise or refine earlier predictions. To address this, the remasking technique reintroduces masked tokens at previously unmasked positions, enabling iterative refinement of generated outputs. Remasking is able to extend the number of decoding steps beyond the response length, thereby allowing repeated updates to the response. This mechanism serves as a form of test-time scaling, improving output quality through progressive correction.

V.B.1 Remasking in General Masked Diffusion Models.

[23] formulates the reversal diffusion process with remasking as

$$q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1-\sigma_t)\mathbf{x} + \sigma_t \mathbf{m}), & \mathbf{z}_t \neq \mathbf{m}, \\ \mathrm{Cat}\left(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}\mathbf{m}\right), & \mathbf{z}_t = \mathbf{m}, \end{cases} \tag{86}$$

where σ_t is used to control the ratio of remasked tokens. When $\mathbf{z}_t \neq \mathbf{m}$, the token has already been decoded. The model samples the next token $\mathbf{z}_s$ from a distribution that mixes the input $\mathbf{x}$ and the mask token $\mathbf{m}$, controlled by σ_t. This enables remasking by reintroducing uncertainty into already decoded tokens. When $\mathbf{z}_t = \mathbf{m}$, the token is still masked. The sampling distribution is a weighted combination of $\mathbf{x}$ and $\mathbf{m}$, adjusted by both α_t and σ_t, allowing flexible control over how much information from the input or the mask dominates.

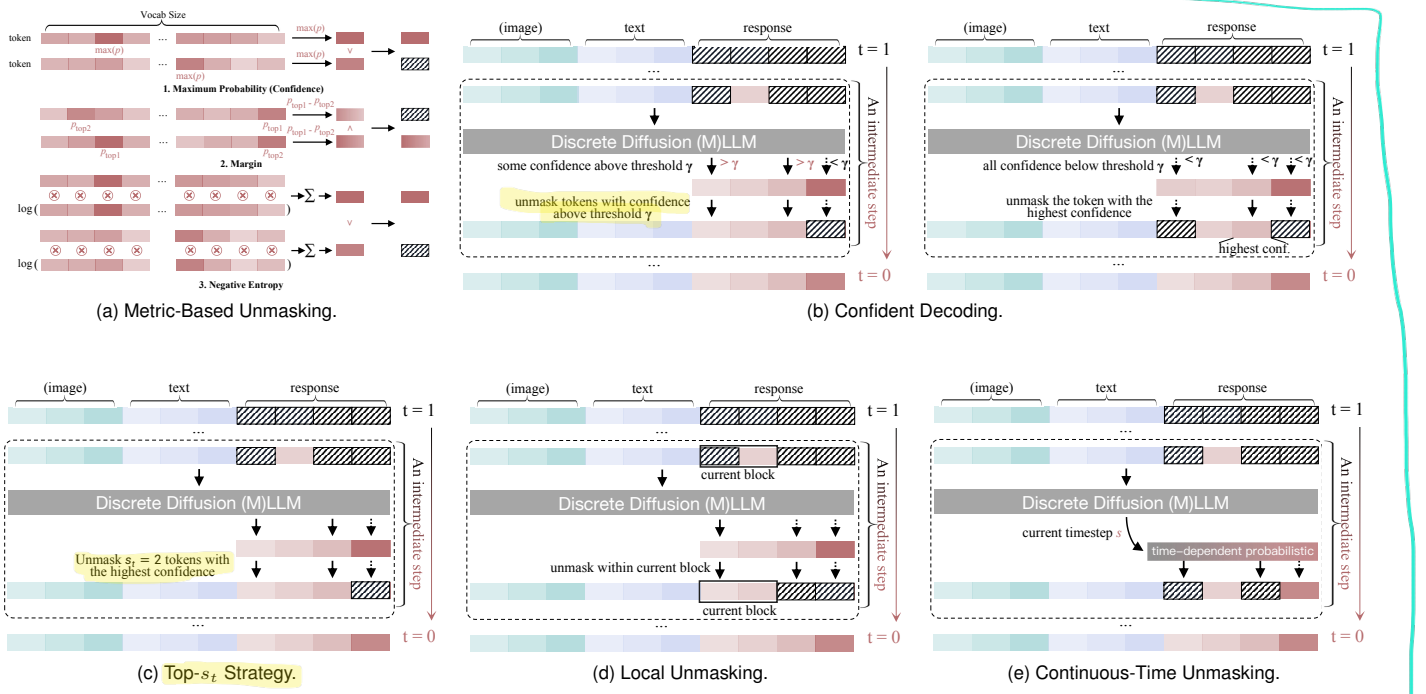


Fig. 5. **Unmasking strategies.** We divide the unmasking strategies used in dLLMs and dMLLMs into two categories: Discrete-Time Unmasking (a,b,c,d) and Continuous-Time Unmasking (e). In discrete-time unmasking, besides random unmasking, there are other two unmasking strategies: Metric-Based Unmasking (Maximum Probability (Confidence), Margin and Negative Entropy, see (a)) and Selection Policies (Top-s_t Strategy (see (c)), Confident Decoding (see (b)), and Local Unmasking (see (d))).

V.B.2 Remasking under Discrete Flow Matching.

In the discrete flow matching framework [24], remasking is incorporated via a velocity field that interpolates between forward and backward update directions:

$$\bar{u}_t^i(x^i, z) = \alpha_t \hat{u}_t^i(x^i, z) - \beta_t \check{u}_t^i(x^i, z), \tag{87}$$

where $\hat{u}_t^i$ and $\check{u}_t^i$ denote the forward-time and backward-time velocity fields, respectively. This combined velocity $\bar{u}_t^i$ is valid as long as $\alpha_t, \beta_t > 0$ and satisfies the probability flow condition for $t \in (0, 1)$. When $\alpha_t - \beta_t = 1$, each step progresses forward in time with remasking (corrector sampling), enabling iterative refinement. When $\alpha_t - \beta_t = 0$, the model operates in a stationary regime (corrector iteration), reintroducing noise and adjusting tokens within a fixed diffusion step.

V.C Prefilling and Caching Techniques

Prefilling and Key-Value Cache (KV-Cache) are standard inference acceleration techniques widely adopted in autoregressive language models. Intuitively, Prefilling and KV-Cache avoids redundant computation by storing the key and value representations from previous decoding steps, enabling the model to reuse them instead of recalculating at each new time step.

Formally, in a transformer decoder, the current hidden states $\mathbf{h}_i$ are projected into a query-key-value triplet $(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i)$ using learned projection matrices. For the first generation iteration $i = 1$, the hidden state $\mathbf{h}_0$ corresponding to the first predicted token and all prompt tokens is used to compute the new query-key-value triplet $(\mathbf{Q}_1, \mathbf{K}_1, \mathbf{V}_1)$. The calculated key tensor $\mathbf{K}_1$ and value tensor $\mathbf{V}_1$ are cached for reusing. At iteration $i > 1$, only the hidden state $\mathbf{h}_i$ corresponding to the i-th response token is used to compute the new query-key-value triplet. The new key $\mathbf{K}_i$ and new value $\mathbf{V}_i$ are concatenated with all key and value vectors from all previous steps to form the complete key tensor and the value tensor. These key tensor and value tensor are cached and will be reused in the subsequent steps.

$$\begin{aligned} \mathbf{K} &= \begin{cases} \mathbf{K}_i, & \text{if } i = 1, \\ \text{concat}\left(\mathbf{K}_{[1:i]}, \mathbf{K}_i\right), & \text{otherwise}, \end{cases} \\ \mathbf{V} &= \begin{cases} \mathbf{V}_i, & \text{if } i = 1, \\ \text{concat}\left(\mathbf{V}_{[1:i]}, \mathbf{V}_i\right), & \text{otherwise}, \end{cases} \end{aligned} \tag{88}$$

In this way, the attention computation at each step i leverages the cached key-value pairs from previous steps can be written as

$$\mathbf{z}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{89}$$

significantly reducing redundant operations and improving inference efficiency in autoregressive decoding.

In autoregressive models, the use of causal attention masks ensures that caching is theoretically lossless. This is attributed to the unidirectional nature of attention, where each token attends only to its preceding tokens. Consequently, previously computed key and value representations remain valid and unchanged throughout the generation process. Similarly, in semi-autoregressive generation paradigms such as block-wise decoding [], caching can be applied between different blocks without introducing approximation errors.

In contrast, dLLMs and dMLLMs employ full (bidirectional) attention mechanisms, wherein each token can attend to all other positions, regardless of their masking status. As a result, even tokens that have already been decoded and unmasked may have

their key and value vectors influenced by updates to other tokens during subsequent diffusion iterations.

Despite this theoretical limitation, empirical studies have consistently validated the effectiveness of incorporating prefilling and KV cache techniques in dLLMs and dLLMs. These methods yield substantial acceleration in inference with only minimal degradation in model performance.

V.C.1 Prefilling

In multimodal large models, the inclusion of visual inputs significantly increases the number of prompt tokens, often exceeding the length of the response even without inference. Prefilling the prompt greatly enhances inference efficiency. The papers Dimple [8] and LaViDa [10] were among the first to apply prefilling techniques to dMLLMs, demonstrating the feasibility of this approach in multimodal large models.

Dimple's experimental results demonstrate that the use of prefilling incurs negligible performance degradation on the majority of vision-language benchmarks. This observation implies that current multimodal models process and encode visual information in a largely static, one-pass manner, with limited dependence on the subsequent generation of textual responses. Furthermore, the adoption of prefilling yields substantial improvements in inference efficiency, achieving speedups ranging from 2× to 7×. Notably, the magnitude of acceleration increases with larger batch sizes, indicating a positive correlation between speedup and GPU utilization.

V.C.2 KV-Cache

Due to the use of bidirectional attention, the cached KV pairs, attention outputs, and other values in dLLMs and dMLLMs are not static. The associated caching algorithms typically consist of three components: caching, reuse, and update.

- **dKV-Cache** [25]. The core idea of dKV-Cache is to cache the key-value pairs when the tokens are unmasked and reuse the cached key-value pairs. Additionally, it introduces an interval hyperparameter to periodically update the cached key-value pairs.
- **dLLM-Cache** [26]. In addition to caching key-value pairs (KV), dLLM-Cache also stores attention outputs (AttnOut) and feed-forward network (FFNOut) outputs. During subsequent steps, it reuses the AttnOu and FFNOut rather than the raw KV pairs. The paper demonstrates that different interval hyperparameters can be set for the prompt and response segments, with the prompt cache requiring significantly less frequent updates than the response cache. For the response cache, besides periodical full updates, dLLM-Cache employs an Adaptive Partial Update strategy. In each iteration, a portion of the cache is selectively updated. Specifically, the cosine similarity between the current and cached value vectors is computed to identify tokens with significant value changes. After each inference forward pass, a subset of tokens is selected proportionally for KV cache updates.
- **DualCache** [51]. DualCache adopts a block-wise caching strategy, decoding text block by block. The block currently being decoded is not cached, while the surrounding blocks are cached.

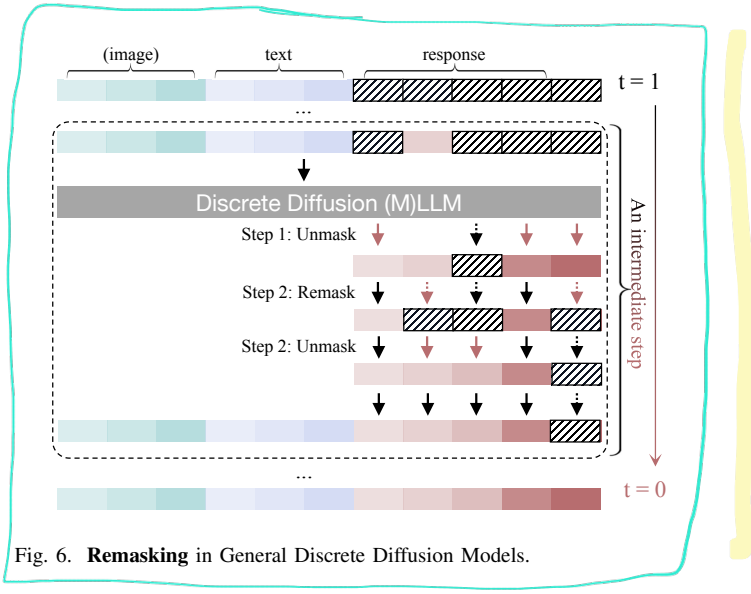


Fig. 6. **Remasking** in General Discrete Diffusion Models.

The general experimental observations are that caching in dLLMs leads to performance fluctuations, and in most cases, results in performance degradation. The severity of degeneration increases with larger update intervals. However, with smaller update intervals (*e.g.*, 2 to 8), performance degradation is minimal and often comparable to that of models without caching [25, 26].

V.D Guidance Techniques

In dLLMs and dMLLMs, post-processing on the predicted logits or sampling probabilities is commonly referred to as *guidance*, following terminology from image diffusion models. Guidance methods are used to influence the generation process toward desired characteristics, such as improved diversity or controllability.

V.D.1 Classifier-Free Guidance

[19] proposes an unsupervised *classifier-free guidance* strategy for discrete diffusion generation. The method performs two forward predictions at each diffusion timestep t:

- A **conditional prediction**, conditioned on both the prompt p_0 and the noisy response r_t;
- An **unconditional prediction**, conditioned on a sequence of mask token M and the same response r_t.

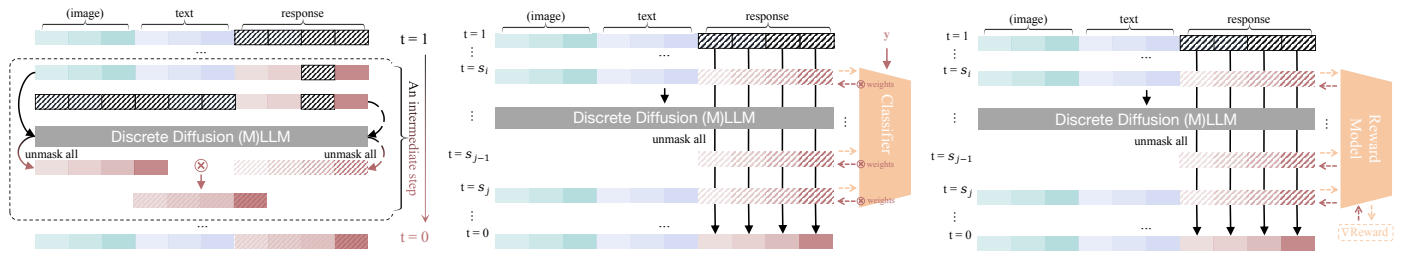
The unconditional prediction captures the model's inherent bias in the absence of instruction signals. The final guided prediction is adjusted as:

$$\tilde{p}_\theta(r_0 \mid p_0, r_t) \propto \frac{p_\theta(r_0 \mid p_0, r_t)^{1+w}}{p_\theta(r_0 \mid m, r_t)^w}, \tag{90}$$

where w is a tunable hyperparameter controlling the strength of the guidance. This guidance promotes *text diversity* by reducing the dominance of generic, encouraging prompt-independent responses.

V.D.2 Classifier Guidance

To improve controllability, for block diffusion model, [52] introduces a *classifier guidance* framework that integrates class-conditional preferences into the sampling process.



(a) Classifier-Free Guidance. (b) Classifier Guidance. (c) Reward Guidance.

Fig. 7. **Guidance Techniques.** We divide the guidance techniques (i.e., the post-processing on the predicted logits or sampling probabilities) into three categories: (a) Classifier-Free Guidance, (b) Classifier Guidance, (c) and Reward Guidance.

At each diffusion step t for block b, the guided reverse process modifies the original sampling distribution p_θ by incorporating the signal from a classifier p_ξ, yielding:

$$p_\gamma(x_b^s \mid x_b^t, x_{<b}, y) \propto p_\theta(x_b^s \mid x_b^t, x_{<b}) \cdot p_\xi(y \mid x_b^s, x_b^t, x_{<b})^\gamma, \tag{91}$$

where y is the desired class label and γ controls the influence of the classifier. To reduce computational complexity, the method assumes intra-block independence and approximates the classifier as:

$$p_\xi(y \mid x_b^s, x_b^t, x_{<b}) \approx \prod_{\ell=1}^{L} p_\xi(y \mid \hat{x}_{b,t|s}^\ell, x_{<b}), \tag{92}$$

where $\hat{x}_{b,t|s}^\ell$ denotes the sequence with the ℓ-th token in x_b^t replaced by the candidate token $x_{b,\ell}^s$. This allows the guided probability to be reformulated as:

$$p_\gamma(x_b^s \mid x_b^t, x_{<b}, y) = \prod_{\ell=1}^{L} \frac{p_\theta(x_{b,\ell}^s \mid x_b^t, x_{<b}) \cdot p_\xi(y \mid \hat{x}_{b,t|s}^\ell, x_{<b})^\gamma}{\sum_{x'} p_\theta(x' \mid x_b^t, x_{<b}) \cdot p_\xi(y \mid \hat{x}_{b,t|s}^{\prime\ell}, x_{<b})^\gamma}. \tag{93}$$

By integrating classifier predictions with the model's native probabilities, this approach enables fine-grained, attribute-conditioned generation across blocks while maintaining computational feasibility.

V.D.3 Reward Guidance

TESS 2 [44] represents a unique approach under the extra-model guidance category by leveraging an external reward model to guide token prediction. The main purpose of this method is to improve the quality of the generated response. Specifically, at each diffusion step, the model output $\hat{S}_\theta$ is first transformed into a token probability distribution:

$$\mathbf{p}_t = \text{softmax}(\hat{S}_\theta), \tag{94}$$

$$\mathbf{c}_w = \mathbf{E}\mathbf{p}_t, \tag{95}$$

where $\mathbf{E}$ is the embedding matrix. The resulting continuous representation $\mathbf{c}_w$ is fed into the reward model, which outputs a scalar reward $R \in \mathbb{R}$.

To maximize this reward, TESS 2 performs gradient ascent on the logits by computing $\nabla_\theta R$ and updates the model output:

$$\hat{S}_\theta := \hat{S}_\theta + \eta \cdot \nabla_\theta R, \tag{96}$$

where η is a tunable guidance coefficient. This process is performed during inference only, and no guidance is applied during training. By incorporating gradient signals from the reward model, TESS 2 can steer the generation towards more desirable outputs without modifying the base diffusion model.

V.D.4 Energy-Based Diffusion

[53] proposes the Energy-based Diffusion Language Model (EDLM), which augments a pretrained diffusion model $p_\theta(x_0 \mid x_t)$ with an unnormalized energy model $E_\phi(x_0, x_t, t)$, yielding a joint denoising distribution:

$$p_{\theta,\phi}(x_0 \mid x_t) = p_\theta(x_0 \mid x_t) \cdot \frac{\exp\left(-E_\phi(x_0, x_t, t)\right)}{Z_\phi(x_t)}, \tag{97}$$

where $Z_\phi(x_t)$ is the intractable partition function required for normalization. This residual formulation corrects the denoising distribution by reweighting samples from the diffusion model using the energy function. The energy function can be derived from either a pretrained autoregressive (AR) model or a finetuned diffusion model. To implement this energy function-based guidance, [53] adopts an importance sampling strategy. The decoding process at each time step can be summarized as follows.

1) Generate k candidate samples $\{x_0^{(i)}\}_{i=1}^k \sim p_\theta(x_0 \mid x_t)$ using the diffusion model.
2) Compute the unnormalized energy scores $e^{(i)} = E_\phi(x_0^{(i)}, x_t, t)$ for all samples in $\{x_0^{(i)}\}_{i=1}^k$.
3) Sample one x_0 from the candidate pool according to importance weights:

$$w^{(i)} = \frac{\exp(-e^{(i)})}{\sum_{j=1}^k \exp(-e^{(j)})}. \tag{98}$$

4) Use the sampled x_0 to perform one denoising step via the backward posterior:

$$x_{t-1} \sim q(x_{t-1} \mid x_t, x_0). \tag{99}$$

VI Applications

VI.A Text Generation and Style Control

[54] uses a diffusion-based language model for fine-grained text style transfer, trained on the StylePTB benchmark and achieving incredible results even with limited data and without external knowledge. [55] poses adopting diffusion language models for text embeddings, motivated by their inherent bidirectional architecture and recent success in matching or surpassing LLMs especially on reasoning tasks. Building on the idea of leveraging diffusion for controllable generation, [56] propose the Segment-Level Diffusion (SLD) framework, which

segments long outputs and decodes them sequentially with an autoregressive decoder; as a result, diffusion predictions are simplified while coherence and scalability improve, and experiments confirm SLD's fluency and contextual superiority over both diffusion and autoregressive baselines. For the social media applications, [57] introduces DiffusPoll, a model that generates diversified, high-quality poll options from user comments by combining a task-specific masking strategy with attribute-tag guidance; consequently DiffusPoll not only maintains quality and diversity but also better captures minority viewpoints. In the creative-writing domain, [58] present PoetryDiffusion, which jointly enforces semantic meaning and metrical form. By generating entire lines in a single pass and integrating a novel metrical-control module, PoetryDiffusion flexibly adjusts rhyme and structure, and experiments show it produces poems that meet both semantic and metrical requirements with high quality.

VI.B Text Editing and Summarization

EdiText [59] is a controllable text-editing framework that operates at both coarse and fine levels to achieve target attributes; specifically, it fuses an SDEdit-style module for large-scale edits with a novel self-conditioning mechanism for subtle refinements, thereby enabling substantial stylistic changes while preserving meaning in tasks such as toxicity and sentiment control. Moving from editing to long-form generation, [60] designs a discrete diffusion model for efficient abstractive summarization, where a semantic-aware noising schedule, together with an adapted CrossMamba backbone, allows transformers to process long sequences under diffusion and yields strong results on Gigaword and CNN/DailyMail. Complementing summarization with unsupervised exploration of document themes, DiffETM [61] injects diffusion into the Embedded Topic Model so that the document–topic distribution is sampled via a more realistic stochastic process—consequently, the model better fits true topic structures while keeping optimization tractable, as confirmed on two standard corpora. Furthermore, focusing on domain-specific summarization, TermDiffuSum [62] leverages term-aware attention during diffusion and a re-ranking loss to highlight legally salient sentences, thereby surpassing previous diffusion-based extractive methods on legal-document benchmarks.

VI.C Sentiment Analysis and Data Augmentation

The CDA^2 framework [63] employs counterfactual diffusion augmentation to improve cross-domain adaptation in low-resource sentiment analysis. It generates counterfactual target samples by substituting domain-relevant words in source data and guiding the diffusion model to produce high-quality target domain examples. Experiments show that CDA^2 generates higher-quality target samples and achieves state-of-the-art performance in cross-domain sentiment tasks. DiffusionCLS [64] examines the use of a diffusion language model for data augmentation in low-resource sentiment classification. It generates pseudo-samples by reconstructing strongly sentiment-related tokens, thereby increasing data diversity while preserving label-consistent information. This approach balances consistency and diversity without introducing excessive noise. Pushing diffusion further into structured sentiment tasks, the Grid Noise Diffusion Pinpoint Network (GDP) [65] introduces a novel diffusion-based and template-guided approach for Aspect Sentiment Quad Prediction. It includes three modules: Diffusion Vague Learning to enhance learning under uncertainty, Consistency Likelihood Learning to capture commonalities of sentiment elements and mitigate noise, and a GDP-FOR generation template for more natural outputs. Extensive experiments demonstrate the remarkable effectiveness of GDP on ASQP tasks. [106] addresses the layout sticking issue in discrete diffusion layout generation, where poorly placed elements become hard to fix. It proposes Layout-Corrector, a simple module that scores elements' layout quality, resets poorly positioned tokens, and guides refined placement conditioned on high-quality elements. Combined with discrete diffusion models, Layout-Corrector consistently enhances layout quality, flexibility, and sampling efficiency.

VI.D Knowledge and Reasoning

Diffusion-of-Thought (DoT) [66] firstly integrates chain-of-thought reasoning into dLLMs to enhance reasoning capabilities. Unlike traditional left-to-right autoregressive reasoning, DoT allows reasoning steps to diffuse over multiple steps, offering a flexible trade-off between compute and reasoning performance. In addition, DiffuCOMET [67] develops a series of models that leverage diffusion to infer contextually relevant commonsense knowledge from narratives. The method progressively refines a commonsense fact representation anchored to the input context through multiple diffusion steps, producing inferences that are both contextually appropriate and diverse. On benchmarks like ComFact and WebNLG+, DiffuCOMET achieves a better trade-off between commonsense diversity, contextual relevance, and alignment to known facts. DPCL-Diff [68] combines graph node diffusion with dual-domain periodic contrastive learning for temporal knowledge graph reasoning. Its Graph Node Diffusion (GNDiff) model introduces noise into sparsely related historical events to generate new event data that better matches the true distribution, enhancing the ability to reason about future events; the dual-domain learning maps periodic event entities to a Poincaré space and non-periodic entities to a Euclidean space to improve representation power. Most recently, the d1 framework [69] adapts pretrained dLLMs into reasoning models via a combination of supervised fine-tuning and reinforcement learning. It introduces a novel critic-free policy-gradient algorithm called diffu-GRPO and employs masked SFT to distill reasoning knowledge directly from existing datasets, thereby enhancing the model's reasoning abilities. [49] provides an insight into DLLM reasoning: the difficulty of decoding individual tokens in a response varies, and it is not necessarily the case that tokens on the left are easier to decode than those on the right. Therefore, autoregressive (AR) reasoning may not be the optimal approach. Diffusion-based LLMs demonstrate clear advantages over AR models in tasks involving planning, reverse reasoning, and global context understanding. [49] presents examples such as Sudoku and the 24 Game to illustrate these points. NeSyDM [107] introduces a discrete diffusion process in the symbolic space to overcome the limitations of traditional neurosymbolic systems, which often rely on the assumption of conditional independence between symbols. The framework first extracts

symbolic representations from perceptual inputs using a neural network, and then performs a multi-step denoising diffusion process over these symbols. This allows the model to explicitly capture inter-symbol dependencies through the structure of the diffusion trajectory.

VI.E Vision and Multimodal

DiffVLA [70] introduces a vision-language-guided diffusion policy for autonomous-driving planning. Specifically, it employs a hybrid sparse–dense diffusion strategy, guided by a vision-language model, thereby improving both efficiency and action diversity in multi-modal driving scenarios. In a image-enhancement context, UDAN-CLIP [71] proposes an image-to-image diffusion framework for underwater enhancement that jointly integrates a vision-language model and a spatial-attention module. During pre-training, the model leverages synthetic underwater data and a CLIP-guided loss to retain in-air natural priors while correcting local degradations such as haze and low contrast. Extending diffusion policies to robotics, [72] present a framework VPDD that first pre-trains on large-scale actionless human videos and then transfers the learned discrete diffusion policy to robot-control tasks. Here, both human and robot videos are encoded into unified video tokens; a discrete diffusion model is trained generatively on the human corpus before being fine-tuned on a small, action-labeled robot set. Turning to motion synthesis, M2D2M [73] employs a discrete diffusion model to generate continuous human-motion sequences from textual descriptions of multiple actions. By adapting dynamic transition probabilities to token-level similarities and adopting a two-phase sampling strategy—independent followed by joint denoising—it produces long-term, smooth, and contextually coherent motions, ultimately outperforming state-of-the-art multi-motion baselines. Finally, AR-Diffusion [74] introduces a novel architecture that blends autoregressive and diffusion techniques for flexible, asynchronous video generation. It gradually corrupts video frames during both training and inference to reduce phase discrepancies; meanwhile, a non-decreasing length-control mechanism—borrowed from autoregressive generation—enables consistent handling of variable-length sequences, thereby enhancing temporal coherence. Besides, the understanding tasks, discrete diffusion is also largely used in the vision generation tasks [47, 103].

VI.F Biological and Drug Discovery

Several recent studies leverage dLLMs and dMLLMs to advance molecular editing, protein engineering, and drug discovery. MolEditRL [75] is a molecular-editing framework that combines a discrete graph-diffusion model with reinforcement learning to optimize molecular properties while preserving structural similarity. It operates in two stages: first, a conditional diffusion model reconstructs a target molecule from the source structure and textual instructions; second, reinforcement learning fine-tunes the editing actions, thereby further improving property alignment and structural conservation. Building on the idea of diffusion for biomacromolecules, CFP-Gen [76] adopts a diffusion language model for combinatorial functional protein generation, thus enabling de novo design under simultaneous functional, sequence, and structural constraints. It introduces two key modules—Annotation-Guided Feature Modulation (AGFM) and Residue-Controlled Functional Encoding (RCFE)—that dynamically adjust protein features according to functional annotations and precisely control residue-level interactions. As a result, CFP-Gen can create novel proteins whose functionality rivals that of natural counterparts, and it attains a high success rate in designing multifunctional proteins. In a related vein, TransDLM [77] proposes a text-guided, multi-property molecular-optimization method that leverages a diffusion language model. By encoding molecules with standardized chemical nomenclature and embedding property requirements directly into textual descriptions, TransDLM implicitly enforces multiple objectives and thereby reduces error propagation during optimization. Further generalizing diffusion to drug discovery, GenMol [78] presents a single, discrete diffusion model that serves as a versatile generator across diverse pharmaceutical tasks. It produces Sequential Attachment-based Fragment Embedding (SAFE) sequences via non-autoregressive bidirectional decoding, thus avoiding token-order constraints and boosting sampling efficiency. GenMol also treats molecular fragments as basic building blocks and introduces a fragment-remasking strategy to refine candidate structures. Addressing sequence–structure co-design, DPLM-2 [79] is a multimodal protein language model capable of understanding and generating both protein sequences and their three-dimensional structures. It converts 3-D coordinates into discrete tokens through quantization, then jointly trains on sequence and structure data, thereby capturing complex sequence-structure relationships and improving tasks that demand structure-conditioned sequence generation. PepTune [80] targets therapeutic-peptide design with a multi-objective, discrete diffusion framework built on a masked language-model backbone. It introduces bond-dependent masking and a Monte-Carlo Tree Guidance algorithm that balances exploration and exploitation during inference, iteratively refining peptides toward multiple goals. Consequently, PepTune generates chemically diverse peptides that are simultaneously optimized for properties such as binding affinity, membrane permeability, solubility, and hemolysis.

VII Future Directions

In this survey, we have reviewed the recent progress of dLLMs and dMLLMs, and summerized the key mathematics, base models, and training/inference techniques for understanding and utilizing this type of model. We mainly focus on the large-sized models while shortly introducing the contents of early small-sized models (e.g., D3PM, Plaid, SEDD, etc.) that have been well explored and covered in the existing literature. Next, we introduce the challenges and future directions for dLLMs and dMLLMs in the following aspects.

VII.A Training and Infrastructure

Currently, dMLLMs predominantly adopt architectures borrowed from their autoregressive counterparts. These models typically employ an autoregressive LLM [108, 90, 109] (Transformer) as the text encoder, alongside a separate vision encoder [95, 94, 97, 96, 97] to extract image embeddings. A lightweight projector or connector module—often a simple multilayer perceptron (MLP)—is then used to align vision tokens with textual

representations. While convenient and compatible with existing pretrained components, this architecture transfer is primarily driven by engineering convenience rather than by the modeling needs inherent to diffusion. However, diffusion models differ fundamentally from autoregressive models: they model the joint data distribution via iterative denoising steps, rather than sequentially modeling conditional probabilities. This difference becomes more pronounced at scale. Also, the infrastructure for dLLMs remains relatively underdeveloped compared to their autoregressive counterparts. In the autoregressive paradigm, the community has benefited from mature open-source models, standardized training frameworks, and reproducible pipelines that facilitate rapid iteration and deployment at scale. Therefore, establishing standardized modular and scalable training frameworks, and open-sourced pretrained models will be critical directions for the community. Building a robust infrastructure will not only promote fair comparisons and accelerate innovation, but also enable practical deployment across a wide range of real-world applications.

VII.B Inference Efficiency

Despite their recent successes, dLLMs still face substantial limitations in inference efficiency and system scalability [6, 10, 25, 26]. Future work can explore several key directions to improve their deployability and performance. At the architectural level, incorporating efficient attention mechanisms (e.g., FlashAttention [110] or block-wise attention) and multi-scale token representations may help reduce the compute burden during inference. In terms of the denoising process itself, advancing fast sampling techniques—such as progressive distillation [111] and adaptive timestep scheduling [112, 47]—could accelerate generation without compromising quality. Moreover, moving the diffusion process into a continuous latent space, as seen in latent-space diffusion models, presents a promising approach to balancing modeling power with inference efficiency. On the system side, integration with quantized inference (e.g., INT8 or INT4) [113] may yield high-throughput, low-latency generation pipelines. Especially in multimodal scenarios, exploring deeper vision-language coupling—such as cross-modal interaction modules embedded within the diffusion process or modality-aware denoising networks—may enhance the model's ability to reason across modalities. In summary, a holistic fusion of architectural refinement, sampling acceleration, representation compression, and deployment-level optimization constitutes a promising roadmap for advancing dLLMs toward practical, efficient real-world use.

VII.C Security and Privacy

The security and privacy implications of dLLMs are an emerging concern as these models become more widely used. On the privacy front, diffusion models share similar risks with other large generative models [114, 115, 116, 117, 118, 119, 120, 121]: they can inadvertently memorize and regurgitate sensitive training data, raising the possibility of privacy breaches or copyright violations. Recent studies have demonstrated that diffusion models trained on vast internet data can reproduce portions of their training examples, much like LLMs. For instance, [122] managed to extract specific images from an image diffusion model's training set, suggesting that memorization does occur and could pose legal or privacy issues. In the language domain, dLLMs could also analogously spit out memorized phrases, quotations, or personal information from its corpus. Mitigating this will require techniques such as differential privacy training [123], regularization to discourage overfitting [124, 125], or filtering of the training data to remove uniquely identifying content [126, 127]. In addition, security in terms of model misuse and alignment is another crucial aspect. Like any powerful language model, dLLMs could be misused to generate harmful, false, or biased content [128, 129]. One challenge is that controlling a diffusion model's output may require new methods: unlike AR models that can be guided token-by-token or halted upon generating disallowed tokens [130], diffusion models generate content in a more holistic way. This makes real-time content moderation non-trivial—dLLMs might only reveal problematic content once the final denoised text is produced. These areas remain critical future directions to address before dLLMs can be responsibly deployed at scale.

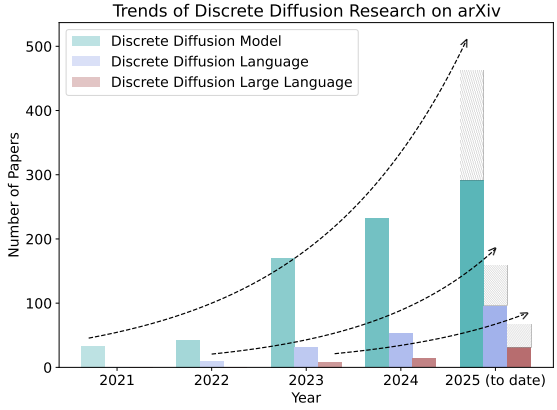


Fig. 8. Number of arXiv publications retrieved via keyword-based search (*Discrete Diffusion Model*, *Discrete Diffusion Language*, and *Discrete Diffusion Large Language*) under the *Computer Science (cs)* category using the *All fields* search option, which scans across all metadata including titles, abstracts, and author information. The results show a consistent year-over-year increase, reflecting the growing research interest in this area.

VIII Conclusion

In summary, this survey provides a comprehensive overview of Discrete Diffusion Large Language Models (dLLMs) and Discrete Diffusion Large Multimodal Models (dMLLMs). We present a detailed exposition of their mathematical foundations and landmark developments. We further detail the training and inference strategies behind them, and summarize the current application domains and potential future directions of them. As a promising alternative to autoregressive LLMs, dLLMs have attracted growing attention (see Figure 8) and show great potential in a variety of real-world scenarios. We hope this survey will serve as a valuable foundation for future research and development in this fast-evolving and important field.

References

[1] OpenAI , "Gpt-4o system card," 2024. [Online]. Available: https://arxiv.org/abs/2410.21276
[2] OpenAI, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[3] DeepSeek-AI, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2501.12948

[4] Gemini Team , "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," 2024. [Online]. Available: https://arxiv.org/abs/2403.05530

[5] Gemini Team, "Gemini: A family of highly capable multimodal models," 2025. [Online]. Available: https://arxiv.org/abs/2312.11805

[6] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li, "Large language diffusion models," *arXiv preprint arXiv:2502.09992*, 2025.

[7] J. Ye, Z. Xie, L. Zheng, J. Gao, Z. Wu, X. Jiang, Z. Li, and L. Kong, "Dream 7b," 2025. [Online]. Available: https://hkunlp.github.io/blog/2025/dream

[8] R. Yu, X. Ma, and X. Wang, "Dimple: Discrete diffusion multimodal large language model with parallel decoding," *arXiv preprint arXiv:2505.16990*, 2025.

[9] Z. You, S. Nie, X. Zhang, J. Hu, J. Zhou, Z. Lu, J.-R. Wen, and C. Li, "Llada-v: Large language diffusion models with visual instruction tuning," *arXiv preprint arXiv:2505.16933*, 2025.

[10] S. Li, K. Kallidromitis, H. Bansal, A. Gokul, Y. Kato, K. Kozuka, J. Kuen, Z. Lin, K.-W. Chang, and A. Grover, "Lavida: A large diffusion language model for multimodal understanding," *arXiv preprint arXiv:2505.16839*, 2025.

[11] L. Yang, Y. Tian, B. Li, X. Zhang, K. Shen, Y. Tong, and M. Wang, "Mmada: Multimodal large diffusion language models," *arXiv preprint arXiv:2505.15809*, 2025.

[12] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf

[13] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. Van Den Berg, "Structured denoising diffusion models in discrete state-spaces," *Advances in neural information processing systems*, vol. 34, pp. 17 981–17 993, 2021.

[14] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling, "Argmax flows and multinomial diffusion: Learning categorical distributions," *Advances in neural information processing systems*, vol. 34, pp. 12 454–12 465, 2021.

[15] L. Zheng, J. Yuan, L. Yu, and L. Kong, "A reparameterized discrete diffusion model for text generation," *arXiv preprint arXiv:2302.05737*, 2023.

[16] S. Sahoo, M. Arriola, Y. Schiff, A. Gokaslan, E. Marroquin, J. Chiu, A. Rush, and V. Kuleshov, "Simple and effective masked diffusion language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 130 136–130 184, 2024.

[17] J. Shi, K. Han, Z. Wang, A. Doucet, and M. Titsias, "Simplified and generalized masked diffusion for discrete data," *Advances in neural information processing systems*, vol. 37, pp. 103 131–103 167, 2024.

[18] S. Gong, S. Agarwal, Y. Zhang, J. Ye, L. Zheng, M. Li, C. An, P. Zhao, W. Bi, J. Han *et al.*, "Scaling diffusion language models via adaptation from autoregressive models," *arXiv preprint arXiv:2410.17891*, 2024.

[19] S. Nie, F. Zhu, C. Du, T. Pang, Q. Liu, G. Zeng, M. Lin, and C. Li, "Scaling up masked diffusion models on text," 2025.

[20] Inception Labs, "Mercury," https://www.inceptionlabs.ai/introducing-mercury, 2025, accessed: 2025-06-16.

[21] DeepMind, "Gemini diffusion," https://deepmind.google/models/gemini-diffusion/, 2025, accessed: 2025-06-16.

[22] J. Ou, S. Nie, K. Xue, F. Zhu, J. Sun, Z. Li, and C. Li, "Your absorbing discrete diffusion secretly models the conditional distributions of clean data," 2025.

[23] G. Wang, Y. Schiff, S. Sahoo, and V. Kuleshov, "Remasking discrete diffusion models with inference-time scaling," *arXiv preprint arXiv:2503.00307*, 2025.

[24] I. Gat, T. Remez, N. Shaul, F. Kreuk, R. T. Q. Chen, G. Synnaeve, Y. Adi, and Y. Lipman, "Discrete flow matching," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37. Curran Associates, Inc., 2024, pp. 133 345–133 385.

[25] X. Ma, R. Yu, G. Fang, and X. Wang, "dkv-cache: The cache for diffusion language models," *arXiv preprint arXiv:2505.15781*, 2025.

[26] Z. Liu, Y. Yang, Y. Zhang, J. Chen, C. Zou, Q. Wei, S. Wang, and L. Zhang, "dllm-cache: Accelerating diffusion large language models with adaptive caching," *arXiv preprint arXiv:2506.06295*, 2025.

[27] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International conference on machine learning*. pmlr, 2015, pp. 2256–2265.

[28] E. Haxholli, Y. Z. Gurbuz, O. Can, and E. Waxman, "Efficient perplexity bound and ratio matching in discrete diffusion language models," in *The Thirteenth International Conference on Learning Representations*, 2025.

[29] D. von Rütte, J. Fluri, Y. Ding, A. Orvieto, B. Schölkopf, and T. Hofmann, "Generalized interpolating discrete diffusion," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=rvZv7sDPV9

[30] A. Campbell, J. Benton, V. De Bortoli, T. Rainforth, G. Deligiannidis, and A. Doucet, "A continuous time framework for discrete denoising models," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 28 266–28 279.

[31] C. Meng, K. Choi, J. Song, and S. Ermon, "Concrete score matching: Generalized score matching for discrete data," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 34 532–34 545.

[32] H. Sun, L. Yu, B. Dai, D. Schuurmans, and H. Dai, "Score-based continuous-time discrete diffusion models," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[33] R. ZHANG, S. Zhai, Y. Zhang, J. Thornton, Z. Ou, J. M. Susskind, and N. Jaitly, "Target concrete score matching: A holistic framework for discrete diffusion," in *Forty-second International Conference on Machine Learning*, 2025.

[34] M. Arriola, A. Gokaslan, J. T. Chiu, Z. Yang, Z. Qi, J. Han, S. S. Sahoo, and V. Kuleshov, "Block diffusion: Interpolating between autoregressive and diffusion language models," in *The Thirteenth International Conference on Learning Representations*, 2025.

[35] Z. He, T. Sun, K. Wang, X. Huang, and X. Qiu, "Diffusionbert: Improving generative masked language models with diffusion models," *arXiv preprint arXiv:2211.15029*, 2022.

[36] J. Chen, A. Zhang, M. Li, A. Smola, and D. Yang, "A cheaper and better diffusion language model with soft-masked noise," *arXiv preprint arXiv:2304.04746*, 2023.

[37] K. Zhou, Y. Li, W. X. Zhao, and J.-R. Wen, "Diffusion-nat: Self-prompting discrete diffusion for non-autoregressive text generation," *arXiv preprint arXiv:2305.04044*, 2023.

[38] R. K. Mahabadi, H. Ivison, J. Tae, J. Henderson, I. Beltagy, M. E. Peters, and A. Cohan, "Tess: Text-to-text self-conditioned simplex diffusion," *arXiv preprint arXiv:2305.08379*, 2023.

[39] I. Gulrajani and T. B. Hashimoto, "Likelihood-based diffusion language models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 16 693–16 715, 2023.

[40] A. Lou, C. Meng, and S. Ermon, "Discrete diffusion modeling by estimating the ratios of the data distribution," *arXiv preprint arXiv:2310.16834*, 2023.

[41] A. Swerdlow, M. Prabhudesai, S. Gandhi, D. Pathak, and K. Fragkiadaki, "Unified multimodal discrete diffusion," *arXiv preprint arXiv:2503.20853*, 2025.

[42] J. Ye, Z. Zheng, Y. Bao, L. Qian, and Q. Gu, "Diffusion language models can perform many tasks with scaling and instruction-finetuning," *arXiv preprint arXiv:2308.12219*, 2023.

[43] F. Zhu, R. Wang, S. Nie, X. Zhang, C. Wu, J. Hu, J. Zhou, J. Chen, Y. Lin, J.-R. Wen *et al.*, "Llada 1.5: Variance-reduced preference optimization for large language diffusion models," *arXiv preprint arXiv:2505.19223*, 2025.

[44] J. Tae, H. Ivison, S. Kumar, and A. Cohan, "Tess 2: A large-scale generalist diffusion language model," *arXiv preprint arXiv:2502.13917*, 2025.

[45] J. Wang, Y. Lai, A. Li, S. Zhang, J. Sun, N. Kang, C. Wu, Z. Li, and P. Luo, "Fudoki: Discrete flow-based unified understanding and generation via kinetic-optimal velocities," *arXiv preprint arXiv:2505.20147*, 2025.

[46] Q. Shi, J. Bai, Z. Zhao, W. Chai, K. Yu, J. Wu, S. Song, Y. Tong, X. Li, X. Li *et al.*, "Muddit: Liberating generation beyond text-to-image with a unified discrete diffusion model," *arXiv preprint arXiv:2505.23606*, 2025.

[47] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, "Maskgit: Masked generative image transformer," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 315–11 325.

[48] M. Asada and M. Miwa, "Addressing the training-inference discrepancy in discrete diffusion for text generation," in *Proceedings of the 31st International Conference on Computational Linguistics*. Association for Computational Linguistics, Jan. 2025, pp. 7156–7164.

[49] J. Ye, J. Gao, S. Gong, L. Zheng, X. Jiang, Z. Li, and L. Kong, "Beyond autoregression: Discrete diffusion for complex reasoning and planning," in *The Thirteenth International Conference on Learning Representations*, 2025.

[50] S. Hayakawa, Y. Takida, M. Imaizumi, H. Wakaki, and Y. Mitsufuji, "Distillation of discrete diffusion through dimensional correlations," *arXiv preprint arXiv:2410.08709*, 2024.

[51] C. Wu, H. Zhang, S. Xue, Z. Liu, S. Diao, L. Zhu, P. Luo, S. Han, and E. Xie, "Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding," 2025.

[52] C. Huang and H. Tang, "Ctrldiff: Boosting large diffusion language models with dynamic block prediction and controllable generation," 2025. [Online]. Available: https://arxiv.org/abs/2505.14455

[53] M. Xu, T. Geffner, K. Kreis, W. Nie, Y. Xu, J. Leskovec, S. Ermon, and A. Vahdat, "Energy-based diffusion language models for text generation," in *The Thirteenth International Conference on Learning Representations*, 2025.

[54] Y. Lyu, T. Luo, J. Shi, T. C. Hollon, and H. Lee, "Fine-grained text style transfer with diffusion-based language models," *arXiv preprint arXiv:2305.19512*, 2023.

[55] S. Zhang, Y. Zhao, L. Geng, A. Cohan, A. T. Luu, and C. Zhao, "Diffusion vs. autoregressive language models: A text embedding perspective," *arXiv preprint arXiv:2505.15045*, 2025.

[56] X. Zhu, G. Karadzhov, C. Whitehouse, and A. Vlachos, "Segment-level diffusion: A framework for controllable long-form generation with diffusion language models," *arXiv preprint arXiv:2412.11333*, 2024.

[57] L. Cheng and S. Li, "Diffuspoll: Conditional text diffusion model for poll generation," in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 925–935.

[58] Z. Hu, C. Liu, Y. Feng, A. T. Luu, and B. Hooi, "Poetrydiffusion: Towards joint semantic and metrical manipulation in poetry generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, 2024, pp. 18 279–18 288.

[59] C. H. Lee, H. Kim, J. Yeom, and S. Yoon, "Editext: Controllable coarse-to-fine text editing with diffusion language models," *arXiv preprint arXiv:2502.19765*, 2025.

[60] D. A. Do, L. A. Tuan, W. Buntine *et al.*, "Discrete diffusion language model for efficient text summarization," in *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025, pp. 6278–6290.

[61] W. Shao, M. Liu, and L. Song, "Diffetm: Diffusion process enhanced embedded topic model," *arXiv preprint arXiv:2501.00862*, 2025.

[62] X. Dong, W. Li, Y. Le, Z. Jiang, J. Zhong, and Z. Wang, "Termdiffusum: A term-guided diffusion model for extractive summarization of legal documents," in *Proceedings of the 31st International Conference on Computa-*

tional Linguistics, 2025, pp. 3222–3235.

[63] D. Xin, K. Zhao, J. Sun, and Y. Li, "Cdaˆ2: Counterfactual diffusion augmentation for cross-domain adaptation in low-resource sentiment analysis," in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 61–72.

[64] Z. Chen, L. Wang, Y. Wu, X. Liao, Y. Tian, and J. Zhong, "An effective deployment of diffusion lm for data augmentation in low-resource sentiment classification," *arXiv preprint arXiv:2409.03203*, 2024.

[65] L. Zhu, X. Chen, X. Guo, C. Zhang, Z. Zhu, Z. Zhou, and X. Kong, "Pinpointing diffusion grid noise to enhance aspect sentiment quad prediction," in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 3717–3726.

[66] J. Ye, S. Gong, L. Chen, L. Zheng, J. Gao, H. Shi, C. Wu, X. Jiang, Z. Li, W. Bi *et al.*, "Diffusion of thoughts: Chain-of-thought reasoning in diffusion language models," *arXiv preprint arXiv:2402.07754*, 2024.

[67] S. Gao, M. Ismayilzada, M. Zhao, H. Wakaki, Y. Mitsufuji, and A. Bosselut, "Diffucomet: Contextual commonsense knowledge diffusion," *arXiv preprint arXiv:2402.17011*, 2024.

[68] Y. Cao, L. Wang, and L. Huang, "Dpcl-diff: Temporal knowledge graph reasoning based on graph node diffusion model with dual-domain periodic contrastive learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 14, 2025, pp. 14 806–14 814.

[69] S. Zhao, D. Gupta, Q. Zheng, and A. Grover, "d1: Scaling reasoning in diffusion large language models via reinforcement learning," *arXiv preprint arXiv:2504.12216*, 2025.

[70] A. Jiang, Y. Gao, Z. Sun, Y. Wang, J. Wang, J. Chai, Q. Cao, Y. Heng, H. Jiang, Z. Zhang *et al.*, "Diffvla: Vision-language guided diffusion planning for autonomous driving," *arXiv preprint arXiv:2505.19381*, 2025.

[71] A. Shaahid and M. Behzad, "Underwater diffusion attention network with contrastive language-image joint learning for underwater image enhancement," *arXiv preprint arXiv:2505.19895*, 2025.

[72] H. He, C. Bai, L. Pan, W. Zhang, B. Zhao, and X. Li, "Learning an actionable discrete diffusion policy via large-scale actionless video pre-training," *arXiv preprint arXiv:2402.14407*, 2024.

[73] S. Chi, H.-g. Chi, H. Ma, N. Agarwal, F. Siddiqui, K. Ramani, and K. Lee, "M2d2m: Multi-motion generation from text with discrete diffusion models," in *European Conference on Computer Vision*. Springer, 2024, pp. 18–36.

[74] M. Sun, W. Wang, G. Li, J. Liu, J. Sun, W. Feng, S. Lao, S. Zhou, Q. He, and J. Liu, "Ar-diffusion: Asynchronous video generation with auto-regressive diffusion," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 7364–7373.

[75] Y. Zhuang, D. Shen, and Y. Sun, "Moleditrl: Structure-preserving molecular editing via discrete diffusion and reinforcement learning," *arXiv preprint arXiv:2505.20131*, 2025.

[76] J. Yin, C. Zha, W. He, C. Xu, and X. Gao, "Cfp-gen: Combinatorial functional protein generation via diffusion language models," *arXiv preprint arXiv:2505.22869*, 2025.

[77] Y. Xiong, K. Li, W. Liu, J. Wu, B. Du, S. Pan, and W. Hu, "Text-guided multi-property molecular optimization with a diffusion language model," *arXiv preprint arXiv:2410.13597*, 2024.

[78] S. Lee, K. Kreis, S. P. Veccham, M. Liu, D. Reidenbach, Y. Peng, S. Paliwal, W. Nie, and A. Vahdat, "Genmol: A drug discovery generalist with discrete diffusion," *arXiv preprint arXiv:2501.06158*, 2025.

[79] X. Wang, Z. Zheng, F. Ye, D. Xue, S. Huang, and Q. Gu, "Dplm-2: A multimodal diffusion protein language model," *arXiv preprint arXiv:2410.13782*, 2024.

[80] S. Tang, Y. Zhang, and P. Chatterjee, "Peptune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion," *ArXiv*, pp. arXiv–2412, 2025.

[81] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[82] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[83] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[84] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[85] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[86] Y. Bahri, E. Dyer, J. Kaplan, J. Lee, and U. Sharma, "Explaining neural scaling laws," *Proceedings of the National Academy of Sciences*, vol. 121, no. 27, p. e2311878121, 2024.

[87] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, pp. 681–694, 2020.

[88] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[89] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[90] Qwen, :, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng,

B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu, "Qwen2.5 technical report," 2025.

[91] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier, "A survey of reinforcement learning from human feedback," *arXiv preprint arXiv:2312.14925*, vol. 10, 2023.

[92] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 728–53 741, 2023.

[93] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[94] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou, "Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond," *arXiv preprint arXiv:2308.12966*, 2023.

[95] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," 2023.

[96] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," 2023.

[97] H. Liu, C. Li, Y. Li, B. Li, Y. Zhang, S. Shen, and Y. J. Lee, "Llava-next: Improved reasoning, ocr, and world knowledge," January 2024. [Online]. Available: https://llava-vl.github.io/blog/2024-01-30-llava-next/

[98] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 11 975–11 986.

[99] N. Shaul, I. Gat, M. Havasi, D. Severo, A. Sriram, P. Holderrieth, B. Karrer, Y. Lipman, and R. T. Chen, "Flow matching with general discrete paths: A kinetic-optimal perspective," *arXiv preprint arXiv:2412.03487*, 2024.

[100] C. Wu, X. Chen, Z. Wu, Y. Ma, X. Liu, Z. Pan, W. Liu, Z. Xie, X. Yu, C. Ruan *et al.*, "Janus: Decoupling visual encoding for unified multimodal understanding and generation," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 12 966–12 977.

[101] P. Sun, Y. Jiang, S. Chen, S. Zhang, B. Peng, P. Luo, and Z. Yuan, "Autoregressive model beats diffusion: Llama for scalable image generation," *arXiv preprint arXiv:2406.06525*, 2024.

[102] B. F. Labs, "Flux," https://github.com/black-forest-labs/flux, 2024.

[103] J. Bai, T. Ye, W. Chow, E. Song, Q.-G. Chen, X. Li, Z. Dong, L. Zhu, and S. Yan, "Meissonic: Revitalizing masked generative transformers for efficient high-resolution text-to-image synthesis," in *The Thirteenth International Conference on Learning Representations*, 2024.

[104] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[105] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PmLR, 2021, pp. 8748–8763.

[106] S. Iwai, A. Osanai, S. Kitada, and S. Omachi, "Layout-corrector: Alleviating layout sticking phenomenon in discrete diffusion model," in *European Conference on Computer Vision*. Springer, 2024, pp. 92–110.

[107] E. van Krieken, P. Minervini, E. Ponti, and A. Vergari, "Neurosymbolic diffusion models," *arXiv preprint arXiv:2505.13138*, 2025.

[108] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.

[109] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[110] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *Advances in neural information processing systems*, vol. 35, pp. 16 344–16 359, 2022.

[111] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," *arXiv preprint arXiv:2202.00512*, 2022.

[112] D. Watson, J. Ho, M. Norouzi, and W. Chan, "Learning to efficiently sample from diffusion probabilistic models," *arXiv preprint arXiv:2106.03802*, 2021.

[113] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale," *Advances in neural information processing systems*, vol. 35, pp. 30 318–30 332, 2022.

[114] C. Zhang, J. X. Morris, and V. Shmatikov, "Extracting prompts by inverting llm outputs," *arXiv preprint arXiv:2405.15012*, 2024.

[115] Y. Chen, H. Lent, and J. Bjerva, "Text embedding inversion security for multilingual language models," *arXiv preprint arXiv:2401.12192*, 2024.

[116] J. X. Morris, W. Zhao, J. T. Chiu, V. Shmatikov, and A. M. Rush, "Language model inversion," *arXiv preprint arXiv:2311.13647*, 2023.

[117] Q. Li, R. Yu, and X. Wang, "Vid-sme: Membership inference attacks against large video understanding models," *arXiv preprint arXiv:2506.03179*, 2025.

[118] Z. Li, Y. Wu, Y. Chen, F. Tonin, E. Abad Rocamora, and V. Cevher, "Membership inference attacks against large vision-language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 98 645–98 674, 2024.

[119] Y. He, B. Li, L. Liu, Z. Ba, W. Dong, Y. Li, Z. Qin, K. Ren, and C. Chen, "Towards label-only membership inference attack against pre-trained large language models," in *USENIX Security*, 2025.

[120] J. Zhang, J. Sun, E. Yeats, Y. Ouyang, M. Kuo, J. Zhang, H. F. Yang, and H. Li, "Min-k%++: Improved baseline for detecting pre-training data from large language models," *arXiv preprint arXiv:2404.02936*, 2024.

[121] C.-L. Wang, Q. Li, Z. Xiang, Y. Cao, and D. Wang, "Towards lifecycle unlearning commitment management: Measuring sample-level unlearning completeness," *arXiv preprint arXiv:2506.06112*, 2025.

[122] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, B. Balle, D. Ippolito, and E. Wallace, "Extracting training data from diffusion models," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 5253–5270.

[123] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[124] C. F. G. D. Santos and J. P. Papa, "Avoiding overfitting: A survey on regularization methods for convolutional neural networks," *ACM Computing Surveys (Csur)*, vol. 54, no. 10s, pp. 1–25, 2022.

[125] X. Ying, "An overview of overfitting and its solutions," in *Journal of physics: Conference series*, vol. 1168. IOP Publishing, 2019, p. 022022.

[126] D. Chen, Y. Huang, Z. Ma, H. Chen, X. Pan, C. Ge, D. Gao, Y. Xie, Z. Liu, J. Gao *et al.*, "Data-juicer: A one-stop data processing system for large language models," in *Companion of the 2024 International Conference on Management of Data*, 2024, pp. 120–134.

[127] M. Li, Y. Zhang, S. He, Z. Li, H. Zhao, J. Wang, N. Cheng, and T. Zhou, "Superfiltering: Weak-to-strong data filtering for fast instruction-tuning," *arXiv preprint arXiv:2402.00530*, 2024.

[128] Y. Qu, X. Shen, X. He, M. Backes, S. Zannettou, and Y. Zhang, "Unsafe diffusion: On the generation of unsafe images and hateful memes from text-to-image models," in *Proceedings of the 2023 ACM SIGSAC conference on computer and communications security*, 2023, pp. 3403–3417.

[129] Y. Zhang, J. Jia, X. Chen, A. Chen, Y. Zhang, J. Liu, K. Ding, and S. Liu, "To generate or not? safety-driven unlearned diffusion models are still easy to generate unsafe images... for now," in *European Conference on Computer Vision*. Springer, 2024, pp. 385–403.

[130] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, ""do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 1671–1685.