VaporTok: RL-Driven Adaptive Video Tokenizer with Prior & Task Awareness

Minghao Yang[1*] **Zechen Bai**[2*] **Jing Lin**[3] **Haoqian Wang**[1†] **Alex Jinpeng Wang**[4†]

[1]Tsinghua University [2]National University of Singapore
[3]Nanyang Technological University [4]Central South University

Abstract

Recent advances in visual tokenizers have demonstrated their effectiveness for multimodal large language models and autoregressive generative models. However, most existing visual tokenizers rely on a fixed downsampling rate at a given visual resolution, and consequently produce a constant number of visual tokens, ignoring the fact that visual information of varying complexity warrant different token budgets. Motivated by this observation, we propose an adaptive video tokenizer "VaporTok" with two core contributions: **Probabilistic Taildrop**: We introduce a novel taildrop mechanism that learns a truncation index sampling distribution conditioned on visual complexity of the video. During both training and inference, the decoder reconstructs videos at adaptive token lengths, allocating more tokens to complex videos and fewer to simpler ones. **Parallel Sample GRPO with Vapor Reward**: By leveraging the probability distribution produced by probabilistic taildrop, we reformulate the visual tokenization pipeline as a sequential decision process. To optimize this process, we propose a variant of GRPO and a composite reward encompassing token efficiency, reconstruction fidelity, and generative quality, thus enabling metrics-aware adaptive tokenization across diverse objectives. Extensive experiments on standard video generation benchmarks confirm our analysis, showing that our adaptive approach matches or outperforms fixed-rate baselines and naive taildrop while using fewer tokens.

1 Introduction

Visual generative models have undergone rapid advancements in recent years, progressing from VAEs[22] and GANs[17] to diffusion models[18]. More recently, autoregressive (AR) based approaches[7, 5, 42, 45] have emerged as a prominent direction, demonstrating competitiveness with diffusion models. The superior performance is largely due to the scalability and flexibility of the AR paradigm, as demonstrated by large language models (LLMs) [58, 46, 10]. Similar to LLMs, AR-based visual generative models necessitate a visual tokenizer, which is essential for converting image or video data into vector representations that the model can process. Consequently, research into visual tokenizers has become a central focus in visual generative models.

Despite their remarkable performance, AR models still face fundamental limitations inherent in the AR paradigm. Specifically, the computational complexity of processing token sequences grows quadratically with their length. In addition, the prediction errors can accumulate progressively as observed in numerous works[1, 8, 36]. Intuitively, a shorter, more compact visual token sequence can be a favorable option. Existing visual tokenizers[47, 60, 29, 62] generally output a predetermined number of latent tokens for subsequent generation tasks. While there are some attempts at adaptive

[*]Equal contribution.
[†]Corresponding authors.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

tokenization (e.g., [11, 57, 2, 31, 51]), most of them still depend on a manually specified range of token counts without an effective prior, limiting their capacity for truly adaptive tokenization.

Another representative limitation in AR-based visual generative models is that the training of tokenizers and AR models are usually divided into two separated stages, making the visual tokenizer sub-optimal and preventing it from generalizing well to downstream tasks. In fact, the above issues are intertwined. Determining visual representation and its adaptivity should not only satisfy the data prior but also align with downstream task characteristics. On the other hand, the supervision of visual generation can help optimize both AR generative model and, more importantly, its tokenizer, *if back-propagated properly*. This implies that an efficient, adaptive and downstream-aware visual tokenizer is the key to address the AR issues mentioned above.

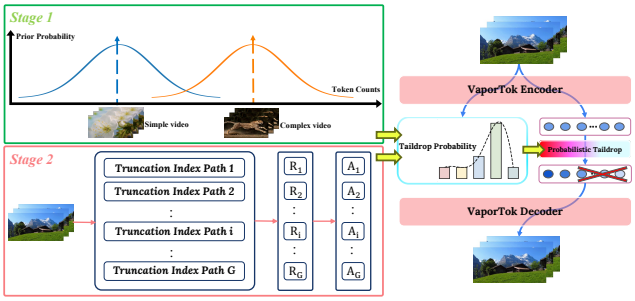


Figure 1: **VaporTok employs two-stage training:** the first stage uses visual complexity for supervision, the second stage uses GRPO with multiple task-aware rewards for supervision.

In this work, we introduce VaporTok, an adaptive video tokenizer that leverages both data prior and task-related signals to optimize its adaptivity. Specifically, we observe that **visual data inherently possesses varying degrees of complexity in terms of content across spatial and temporal dimensions.** Consequently, representing simple content with excessive tokens can lead to redundancy. Conversely, complex content may not be adequately captured if represented with too few tokens. Therefore, dynamically adjusting token number based on visual complexity would enable a more faithful alignment between visual information and its representation. Inspired by this, we propose "Probabilistic Taildrop", a method that leverages visual complexity to build a sampling distribution over token counts and then drops the excess tail tokens accordingly. The complexity informed taildrop not only helps mitigate quadratic computation by producing a compact token sequence, but also reduce error accumulation by condensing meaningful information at the head of the sequence.

When tackling the limitation of training disparity between tokenizer and AR model, the main technical challenge is the differentiability of the two models due to hard token indexing. Recently, reinforcement learning (RL), particularly GRPO[37], has shown considerable advantages in various domains[10, 13, 6, 59, 50, 21]. Notably, **the reward formulation in reinforcement learning does not necessitate differentiability with respect to the parameters of the policy model**. Therefore, we propose to leverage this characteristic to unlock task-aware adaptive tokenizer training. The core idea is that by optimizing the visual tokenizer using RL-based rewards, these non-differentiable supervisory signals could be effectively transmitted during its training. Although the idea seems intuitive, it is non-trivial to achieve. Our approach innovatively formulates visual tokenization as a sequential decision process, which is compatible with RL training framework. **To the best of our knowledge, this is the first work that employs RL framework to formulate and train a visual tokenizer for task-aware adaptive tokenization.** In addition, we design a novel "Vapor Reward" that accommodates multiple supervisory signals into the reward function, providing valuable insights for the community. Our contribution can be summarized as follows:

- **Probabilistic Taildrop**: An adaptive video tokenization technique adjusting token count based on visual complexity for more efficient sequences.

- **RL for Task-Aware Tokenizer Training**: A novel framework that formulates video tokenization as a sequential decision process and uses RL to optimize the tokenizer, making its adaptivity aware of multiple metrics, including downstream AR generation performance.

- **Vapor Reward**: A new multi-signal reward function designed to effectively guide the RL-based tokenizer optimization.

2 Related work

Adaptive visual tokenizers. Most existing visual tokenizers [12, 29, 62, 27, 63, 48, 33, 9, 28, 49, 16, 68, 20, 55, 43, 67] are limited to representing visual features using a fixed number of tokens, ignoring the varying complexity of visual content. As a result, recent works have shifted toward adaptive tokenization schemes that dynamically adjust the token budgets based on visual complexity. ALIT [11] employs a recurrent encoding scheme to progressively assemble the token sequence of an image and the process can be halted. CAT[39] leverages a MLLM's complexity analysis of image captions to regress which compression rate to apply. FlexTok[2] and One-D-Piece[31] both employ nested dropout to encourage the model to prioritize core visual information in early tokens, yielding a coarse-to-fine representation without fixed-length constraints. ElasticTok[57] extends the taildrop technique to the video domain and adaptively selecting the token count at inference based on a reconstruction-quality threshold. ViLex[51] introduces a novel "visual language" that encodes image tokens after taildrop into the textual token space by self-supervised training on a frozen text-to-image diffusion model. These adaptive methods either pick from a fixed set of token counts or use random taildrop with thresholding, yet neither is truly adaptive: the first is limited to predefined choices, and the second ignores visual complexity during training. In addition, they focus only on reconstruction and neglect the impact of adaptive token selection on downstream performance.

GRPO in vision domains. Enhancing foundation models via reinforcement learning has become a major research focus. Motivated by the strong inference performance of DeepSeek R1[10], Group Relative Policy Optimization (GRPO) [37] has demonstrated clear advantages over PPO in both training efficiency and final model quality. In computer vision, core generation and understanding tasks, including Visual Question Answering[32, 59], Image Grounding[6], Video Question Answering[14, 38], and Visual Generation[50, 56, 21, 25], are actively investigating the integration of GRPO to boost existing methods, seeking to transfer the success of GRPO from large language models to vision.

3 Method

3.1 Probabilistic Taildrop

Conventional taildrop technique [23] simply samples truncation positions from *uniform distribution* without considering any prior information about visual complexity. This can lead to insufficient preservation of complex visual content when too few tokens are selected, and to unnecessary redundancy when too many tokens are retained for simpler visuals. In contrast, we propose probabilistic taildrop, which constructs a truncation-sampling distribution informed by the complexity of the visual input. During training, truncation index are drawn from this distribution to perform taildrop. This strategy **both preserves the fundamental principle of taildrop—compressing semantic information into earlier tokens while relegating detailed information to later tokens—and adaptively selects truncation points based on visual priors, thereby achieving efficient yet faithful visual encoding**.

To implement probabilistic taildrop, we introduce the Taildrop Probability Query Module in Section 3.1.1 to obtain the taildrop probabilities. To incorporate visual priors into the supervision of these probabilities, we construct a distribution from the visual information to regularize the predicted taildrop probabilities, as described in Section 3.1.2.

3.1.1 Taildrop Probability Query Module

Given a video $V \in \mathbb{R}^{T\times H\times W\times 3}$, VaporTok first patchify it into a sequence of video tokens P :

$$P = \text{Patchify}(V) \in \mathbb{R}^{(\frac{T}{f_T}\times\frac{H}{f_H}\times\frac{W}{f_W})\times D}, \tag{1}$$

where f_T, f_H, f_W are the temporal and spatial downsampling factors. And then P will be concatenated with K learnable query tokens $Q \in \mathbb{R}^{K\times D}$ and the combined sequence will be passed into the encoder :

$$Z_P \oplus Z_Q = \text{Enc}\big(P \oplus Q\big) \in \mathbb{R}^{(\frac{T}{f_T}\times\frac{H}{f_H}\times\frac{W}{f_W}+K)\times D}, \tag{2}$$

where $\oplus$ denotes concatenation and Z_P,Z_Q denotes the represatation of P, Q after encoder respectively. To enable VaporTok to learn a distribution for taildrop, we introduce Taildrop Probability

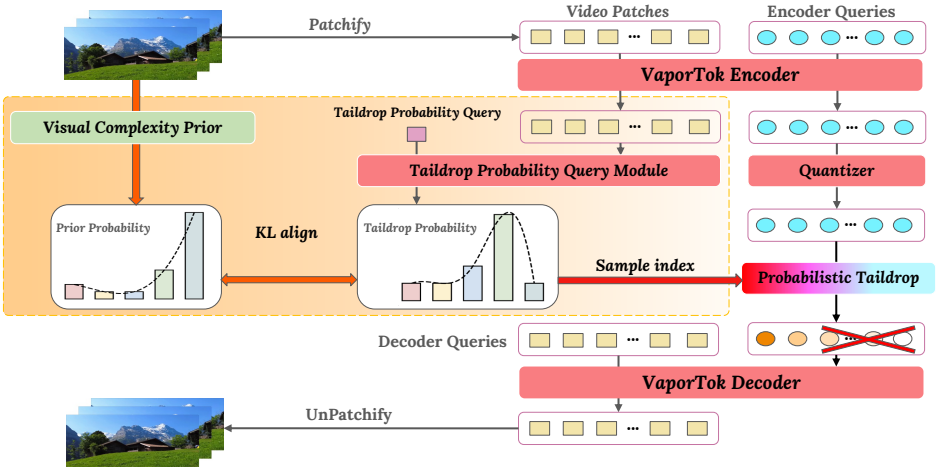


Figure 2: **VaporTok pipeline:** A taildrop probability query module constructs a taildrop probability supervised by a video-complexity prior. An index is then sampled according to this probability, and only the tokens preceding that index are retained for reconstruction training.

Query Module consisting of I successive transformer blocks and a softmax layer, as illustrated in Figure 2. A dedicated taildrop probability query $Q_{\text{tail}} \in \mathbb{R}^D$ is concatenated with Z_P and fed into transformer blocks of Taildrop Probability Query Module :

$$Q'_{\text{tail}} \oplus Z'_P = \text{TransformerBlocks}\big(Q_{\text{tail}} \oplus Z_P\big) \in \mathbb{R}^{(1+\frac{T}{f_T}\times\frac{H}{f_H}\times\frac{W}{f_W})\times D}, \tag{3}$$

Q'_{tail} is then passed through an MLP followed by a softmax layer to produce the taildrop probability distribution P :

$$P \;=\; \text{Softmax}\Big(\text{MLP}\big(Q'_{\text{tail}}\big)\Big) \;\in\; \mathbb{R}^K, \tag{4}$$

where K is the total token counts of latent query Z_Q. To enable adaptive token usage, a truncation index t is sampled according to the learned taildrop probability P :

$$t \sim \text{Categorical}(P) \tag{5}$$

All latent tokens of Z_Q whose index exceeds the sampled index t are discarded and only the first t tokens are concatenated with the decoder query M which are then passed to the decoder for reconstruction :

$$PTD\_Z_Q = \text{ProbabilisticTaildrop}(Z_Q) = Z_{Q,\,1:t} \in \mathbb{R}^{t\times D} \tag{6}$$

$$\hat{V} = \text{Dec}\left(M \oplus PTD\_Z_Q\right) \in \mathbb{R}^{T\times H\times W\times 3} \tag{7}$$

The first-stage training loss of VaporTok is composed of L1 reconstruction loss, LPIPS perceptual loss[66], GAN loss[17], quantizer loss[48] and prior loss[48]. The detail is provided in the Appendix.

3.1.2 Video Complexity Prior

Implicitly modeling video complexity is impractical. Therefore, we explicitly supervise the Taildrop Probability Query Module with a Gaussian distribution that contains visual prior information. First, the spatial and temporal complexities are computed separately and then these complexities are mapped to a corresponding token count via Equation 9. We then construct a Gaussian distribution to supervise the taildrop probabilities, with its mean set to the token count obtained from the preceding mapping and its variance determined by K, the total number of encoder query tokens. Specifically, let visual complexity of video i be :

$$c_i = \text{SC}_i \times \text{TC}_i, \tag{8}$$

where SC_i and TC_i are its spatial and temporal complexities whose details are provided in the Appendix. Then maintain an empirical CDF F over all observed complexities in training set and map each video complexity c_i to a token count k_i by :

$$k_i \;=\; \lfloor s + \Delta\, F(c_i) \rfloor\,, \quad F(c_i) \in [0,1]. \tag{9}$$

where s is a predefined minimum token count tolerable for reconstruction and Δ is the maximum increment. This yields a whole dataset wise mapping :

$$\{c_i\}_{i=1}^{|dataset|} \longmapsto \{k_i\}_{i=1}^{|dataset|}. \tag{10}$$

For each training sample i, we construct a 1D Gaussian distribution with mean k_i and variance $K \cdot \sigma_{\text{scale}}$ as the prior distribution :

$$GaussianPrior_i = \mathcal{N}\left(t;\, \mu = k_i,\, \sigma^2 = K \cdot \sigma_{\text{scale}}\right) \tag{11}$$

where k_i is the token count calculated by visual complexity prior for the i-th video, K is the total token counts of Z_Q, and σ_{scale} is a hyperparameter to control the trend of the prior distribution.

Then, the loss of sample video i to train the taildrop branch, specifically Taildrop Probability Query Module and VaporTok encoder, is calculated as the KL divergence between P_i and $GaussianPrior_i$:

$$Loss_i = \text{KL}(P_i \,||\, GaussianPrior_i) \tag{12}$$

3.2 Parallel Sample GRPO with Vapor Reward

Since the sampling operation employed in the VaporTok is non-differentiable, it is infeasible to propagate the reconstruction loss to the Probabilistic Taildrop branch through the latent space. Nevertheless, in reinforcement learning, the reward function can be treated as an arbitrary black box, whose information can be passed to the policy model, even though the reward is not differentiable with respect to the parameters of the policy model. In addition, except for the basic reconstruction feedback, **several helpful metrics can also be incorporated in reward definition to make the tokenizer more efficient and appropriate for downstream generation task.**

To this end, we cast the video tokenization as a sequential decision process in Section 3.2.1 and introduce Parallel Sampling GRPO in Section 3.2.2 to optimize this process while avoiding mode collapse. Furthermore, in Section 3.2.3, we introduce Vapor Reward, which enables GRPO to refine VaporTok's adaptivity with respect to both reconstruction and generation tasks. Finally, in Section 3.2.4, we define the optimization objective of Parallel Sample GRPO.

3.2.1 Definition of sequential decision process

Given an entire video $V_{entire} \in \mathbb{R}^{N_{\text{GRPO}} \times H \times W \times 3}$ of length N_{GRPO}, we first partition it into $L = \left\lceil \frac{N_{\text{GRPO}}}{N_{\text{VAE}}} \right\rceil$ video clips $(V_{clip}^1, V_{clip}^2, \ldots, V_{clip}^i, \ldots, V_{clip}^L)$, where N_{VAE} is the predefined frame count can be processed by VaporTok encoder at one time. For each video clip V_{clip}^i, VaporTok encoder computes its latent representation $Z_Q^{(i)}$ and Taildrop Probability Query Module computes its taildrop probability distribution P_i. We define the three key components of the sequential decision process in the context of token truncation problem as follows:

- **State** $\mathcal{S}$: the current input video clip $V_{clip}^i \in \mathbb{R}^{N_{\text{VAE}} \times H \times W \times 3}$;
- **Action** $\mathcal{A}$: the truncation index sampled from P_i to truncate the latent representation $Z_Q^{(i)}$ of current video clip V_{clip}^i;
- **Reward** $\mathcal{R}(s, a)$: we will introduce our proposed Vapor Reward in Section 3.2.3.

Then we can get a taildrop probability sequence $(P_1, P_2, \ldots, P_i, \ldots, P_L)$ to sample truncation index for each latent representation $Z_Q^{(i)}$ respectively. Also, we define the policy in our pipeline as $\pi_\theta(P_i \mid V_{clip}^i)$, where θ is the parameter of Taildrop Probability Query Module proposed in Section 3.1.1. Notably, the key difference between Markov Decision Process (MDP) and our proposed sequential decision process is that the latter does not satisfy the Markov property, and its transition dynamics $\mathcal{P}(s' \mid s, a)$ are implicitly defined by the entire input video V_{entire}.

3.2.2 Parallel Sample GRPO

In our setting, the G trajectories within GRPO group share exactly the same state sequence $(V_{clip}^1, V_{clip}^2, \ldots, V_{clip}^i, \ldots, V_{clip}^L)$, and thus the same policy $\pi_\theta(P_i \mid V_{clip}^i)$ is applied. This leads to

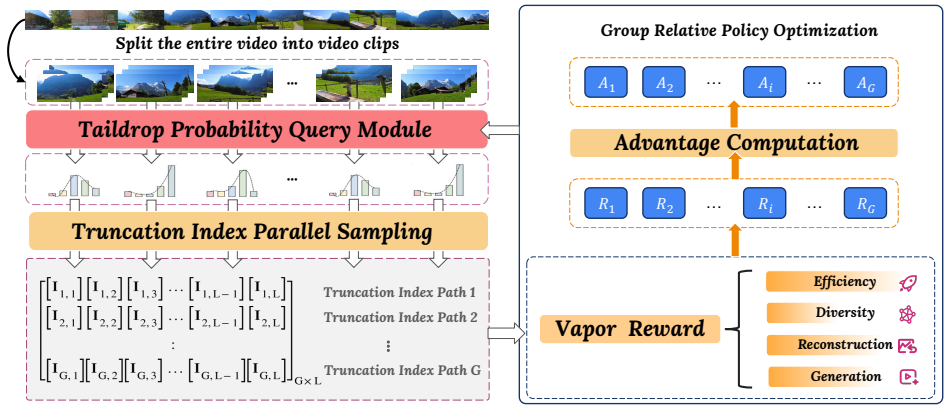


Figure 3: **Parallel Sample GRPO pipeline:** Split the video into a sequence of clips and apply the taildrop probability query module to generate corresponding taildrop probabilities. Then perform parallel sampling to derive a truncation matrix I, compute rewards using the Vapor Reward, and update the parameters of taildrop probability query module via GRPO.

highly similar or even identical sampled truncation index path across all G trajectories within a group, especially under greedy or top-k sampling strategies, which in turn causes severe mode collapse. To address this issue, we propose Parallel Sample GRPO, a variant of GRPO specifically designed for training GRPO in such non-Markovian scenario, which concludes two remedies:

Stochastic parallel sampling. Rather than restricting sampling to the top-k candidates, we draw samples from the whole categorical distribution defined by the taildrop probability distribution P_i. The sampling strategy increases the probability of selecting lower probability indexes and allows for greater diversity between trajectories within a group. Specifically, a truncation matrix $\mathbf{I} = [I_{i,j}] \in \mathbb{Z}^{G \times L}$ is obtained, where each row represents a truncation path that specifies how many tokens are used to represent each clip in the video and there are G such truncation paths in total. Besides, to further mitigate the mode collapse introduced by the non-Markovian nature of the defination, we augment the reward signal with an exploration bonus defined in Equation 14, which encourages diversity across trajectories and therefore promotes exploration.

3.2.3 Vapor Reward

We define the Vapor Reward to incorporate reconstruction, token-count, and downstream feedback into the policy model (Taildrop Probability Query Module) while avoiding collapsed sampling paths within a group. The Vapor Reward comprises of following four types of rewards:

Efficiency reward. To encourage the latent representation of entire video V_{entire} to be more efficient, we define the efficiency reward. Specifically, for the i-th path, the efficiency reward is defined as:

$$R^{(i)}_{\text{efficiency}} = N_{max} - N^{(i)}_{\text{curr}} \tag{13}$$

where N_{max} is the maximum permissible token count for the video, which can be calculated as $N_{max} = L \times K$, where K is the total token number of Z_Q. $N^{(i)}_{\text{curr}}$ is the actual number of tokens retained and can be calculated as $N^{(i)}_{\text{curr}} = \sum_{j=1}^{L} I_{i,j}$

Diversity reward. To encourage exploration and mitigate mode collapse, we define a diversity reward for each path based on how different its sampled index sequence is from others in the same group. Specifically, for the i-th path, the diversity reward is defined as:

$$R^{(i)}_{diversity} = \frac{1}{(G-1)L} \sum_{j=1}^{L} \sum_{\substack{k=1 \\ k \neq i}}^{G} \mathbf{1}\left[I_{i,j} \neq I_{k,j}\right] \tag{14}$$

where $\mathbf{I} = [I_{i,j}] \in \mathbb{Z}^{G \times L}$ is the sampled truncation matrix, G is the number of sampled paths, and L is the sequence length. The indicator function $\mathbf{1}[\cdot]$ is equal to 1 if its argument is true and 0 otherwise. A higher reward is assigned to a path if its sampled indices are more dissimilar from the others.

Reconstruction reward. To encourage the refined taildrop probability to preserve the reconstruction ability learned from former stage, we define reconstruction reward. Specifically, for the i-th path, the reconstruction reward is defined as:

$$R^{(i)}_{\text{reconstruction}} = -\sum_{j=1}^{L} \text{MSE}\big(\hat{V}^j_{clip},\, V^j_{clip}\big) \tag{15}$$

For j-th video clip, we use the truncation index path $I_{i,j}$ to truncate the latent representation and reconstruct the video clip via the VaporTok decoder. Then we compute the mean-squared error between reconstructed video clip $\hat{V}^j_{clip}$ and ground-truth video clip V^j_{clip}. The reconstruction reward for the i-th path is the negative of the sum of the reconstruction MSE of all video clips of current entire video.

Generation reward. To make the taildrop probability be aware of downstream generation performance, we define generation reward. The former work LARP[48] impose a lightweight AR prior model to encourage the latent space to be more suitable for downstream AR-based generation. Hence, the prior model in LARP[48] is reused in our VaporTok (the detail about lightweight ar prior used in VaporTok is provided in the Appendix), and its top-5 accuracy on latent-token predictions is employed as the AR generation reward, guiding the taildrop probability query module to improve efficiency without sacrificing downstream generation performance. Specifically, for the i-th path, the generation reward is defined as:

$$R^{(i)}_{\text{generation}} = \sum_{j=1}^{L} \text{Accuracy}_{\text{top5}}\big(PTD\_Z^j_Q\big) \tag{16}$$

3.2.4 Objective of Parallel Sample GRPO

For each entire input video V_{entire}, a batch of G candidate truncate index sequence $\{\mathbf{k}_i\}_{i=1}^{G}$,where $\mathbf{k}_i = (I_{i,1}, I_{i,2}, \ldots, I_{i,j}, \ldots, I_{i,L})$, is sampled from the old policy $\pi_{\theta_{\text{old}}}$ and scores by reward:

$$R_i \;=\; \sum_{m\in\mathcal{M}} \lambda_m\, R^{(i)}_m \tag{17}$$

where m is one element of $\mathcal{M} = \{\text{efficiency},\, \text{diversity},\, \text{reconstruction},\, \text{generation}\}$ and the non-negative weights $\{\lambda_{\text{m}}\}$ control the relative importance of each reward component. To obtain relative advantages, the rewards $\{R_i\}$ are normalized by their mean and standard deviation:

$$A_i \;=\; \frac{R_i \;-\; \text{mean}\{R_1,\, R_2,\, \ldots,\, R_G\}}{\text{std}\{R_1,\, R_2,\, \ldots,\, R_G\}}. \tag{18}$$

The parameter θ is then updated to maximize the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\mathbf{I}\sim\pi_{\theta_{\text{old}}}}\left[\frac{1}{G}\sum_{i=1}^{G} \min\Big(\rho_i\, A_i,\; \text{clip}(\rho_i,\, 1-\epsilon,\, 1+\epsilon)\, A_i\Big) \;-\; \beta\, D_{\text{KL}}\big(\pi_\theta \,\|\, \pi_{\text{ref}}\big)\right], \tag{19}$$

$$\rho_i = \frac{\pi_\theta(\mathbf{k}_i \mid V_{entire})}{\pi_{\theta_{\text{old}}}(\mathbf{k}_i \mid V_{entire})},$$

ϵ denotes the clipping threshold, β scales the KL-divergence penalty, $\mathbf{I} = [I_{i,j}] \in \mathbb{Z}^{G\times L}$ is the sampled truncation matrix, and A_i represents the advantage estimate for the ith sample.

4 Experiments

Dataset. We conduct video reconstruction and generation experiments using the Kinetics-600[4] and UCF-101[41] datasets. In the first stage, we use $N_{\text{VAE}} = 16$ frame video clips at a spatial resolution of 128 × 128 for VaporTok training and evaluation following [48]. In the second stage, Parallel Sample GRPO training operates on full $N_{\text{GRPO}} = 80$ frame videos and the sequence length optimized by GRPO is $L = \left\lceil \frac{N_{\text{GRPO}}}{N_{\text{VAE}}} \right\rceil = \lceil \frac{80}{16} \rceil = 5$.

Implementation details. VaporTok first patchifies the input video into a sequence of tokens. In all experiments, we set the patch sizes to $f_T = 4, f_H = 8, f_W = 8$, so that a $16 \times 128 \times 128$ video clip

Table 1: Comparison of generative video models. VaporTok refers to the evaluation results after Stage 1 training, while VaporTok-GRPO refers to the evaluation results after Stage 2 training. The reported token counts are the average number of tokens used per video.

Method	#Params		#Tokens	rFVD↓	gFVD↓	
	Tokenizer	Generator			K600	UCF
Diffusion-based generative models with continuous tokenizers						
VideoFusion [26]	—	2B	—	—	—	173
HPDM [40]	—	725M	—	—	—	66
MLM generative models with discrete tokenizers						
MAGVIT-MLM [61]	158M	306M	1024	25	9.9	76
MAGVIT-v2-MLM [62]	—	307M	1280	**8.6**	**4.3**	58
AR generative models with discrete tokenizers						
CogVideo [19]	—	9.4B	2065	—	109.2	626
TATS [15]	32M	321M	1024	162	—	332
MAGVIT-AR [61]	158M	306M	1024	25	—	265
MAGVIT-v2-AR [62]	—	840M	1280	**8.6**	—	109
OmniTokenizer [49]	82.2M	650M	1280	42	32.9	191
LARP-1024 [48]	173M	632M	1024	20	5.1	**57**
LARP-512 [48]	173M	632M	512	53.3	—	86
VaporTok (Ours)	195M	632M	**498**	53.9	8.3	80
VaporTok-GRPO (Ours)	195M	632M	**361**	66.6	10.4	98

is split into $4 \times 16 \times 16 = 1024$ patches. The number of encoder query tokens is set to $k = 1024$. The quantizer and prior model is set as same as [48], where the factorized codebook is employed of size 8192 with embedding dimension $d_{codebook} = 8$ and prior model is adapted from a small GPT-2 backbone[35]. For taildrop probability query module, we set the number of transformer blocks as $I = 2$, and the softmax temperature is set to 1.8. Due to the high computational cost of training, we trained for 30 epochs on the UCF101 and K600 datasets using the pretrained model provided by LARP[48], which required 90 hours on 8 A100 GPUs.

For parallel sample GRPO, we set the group size $G = 8$, the KL penalty weight $\beta = 0.1$, the number of inner iterations $\mu = 2$, and the clipping bounds to $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$ as in [64]. The default reward weights for efficiency, penalty, diversity, reconstruction, and generation are set to 1:1:1:1:1. The GRPO training process uses the UCF101 dataset for a single epoch, which takes 3 hours on a single A100 GPU.

For AR generative model, we adopt a LLaMA-style transformer [42]. In the class-conditional generation task on UCF-101 we prepend a `[cls]` token to represent the category, and a `[stop]` token to cease the generation process when encountering it. The generation task is trained on the UCF101 dataset for 3000 epochs, which takes 40 hours on 8 A100 GPUs.

Table 2: Comparison of different training techniques.

Base Model	#Tokens	Taildrop	Prob. Taildrop	Index	rFVD↓	gFVD↓	gFVD/ rFVD
LARP [48]	1024	✗	✗	—	20.00	57.00	2.85
VaporTok	1024	✓	✗	sample	49.45	62.34	1.26
LARP [48]	512	✗	✗	—	53.25	86.25	1.62
VaporTok	512	✓	✗	sample	81.94	93.34	1.14
VaporTok	509	✗	✓	argmax	59.49	90.65	1.52
VaporTok	498	✗	✓	sample	53.92	80.13	1.48
VaporTok	409	✗	✓	pre-sample	73.01	95.41	1.30

Table 3: Entropy of taildrop probabilities under different GRPO implementation on UCF101 validation set.

Model	Diversity Reward	Parallel Sampling	TopK Sampling	Entropy
VaporTok	—	—	—	5.11308
VaporTok-GRPO	✗	✗	✓	0.00642
VaporTok-GRPO	✗	✓	✗	0.01795
VaporTok-GRPO	✓	✓	✗	4.11323

4.1 Video reconstruction & generation comparison

On the UCF-101 class-conditional generation benchmark, we evaluate LARP against video generation approaches—spanning diffusion-based models, masked-language-modeling methods, and autoregressive methods as in [48]. As shown in Table 1, VaporTok achieves competitive performance with other video generators on the UCF-101 dataset even when using significantly fewer tokens. Notably, our VaporTok model shows a much smaller gap between gFVD and rFVD than other AR-based video generators. Besides, after GRPO training, we can further reduce the average latent token count from roughly 50% down to about 30% of the original while still preserving reconstruction and generation quality—thus achieving efficient adaptivity.

4.2 Comparison of training techniques

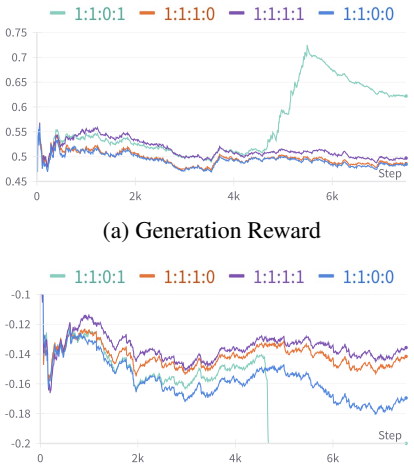
To demonstrate the effectiveness of proposed probabilistic taildrop technique, we make a comparison of different type of training strategies and show the evaluation result in Table 2. For *naive taildrop*, we conducted experiments to test whether taildrop can induce a progressively decreasing importance of tokens along the sequence. From first two rows in Table 2: taildrop causes a substantial performance drop in rFVD, from 20 to 49.45, yet gFVD almost fully recovers the gap introduced by reconstruction and achieve competitive gFVD 62.34 with 57. This shows that taildrop training can partially close the original gap between reconstruction and generation performance by enforcing a semantic-to-detail ordering in the latent space, which greatly reduces error accumulation during autoregressive inference. The same conclusion can also be drawn from the third and fourth rows in Table 2. For *probabilitstic taildrop*, we examine three strategies for obtaining the truncation index from the taildrop probability distribution: taking the argmax, directly sampling, and a pre-sampling variant that only samples from indices before the argmax. Among these strategies, directly sampling from the taildrop probability yields the best performance, achieve the best gFVD 80.13 with a similar token count to baseline methods. The argmax approach lacks variability across different lengths for the same video, thereby missing the core advantage of taildrop—structuring the latent space from semantic to detail. Pre-sampling best reflects this advantage, but it tends to reduce the average token count significantly, which slightly compromises reconstruction and generation quality.

4.3 Ablation studies

Impact of different sampling strategies. To verify that Parallel Sample GRPO alleviates mode collapse introduced by non-Markovian setting, we compute the average entropy of the taildrop probabilities on UCF101 validation set. The results appear in Table 3: The conclusion that probabilities adjusted by GRPO inevitably become more concentrated can be drawn from the lower entropy compared to that optimized by the prior probability of the first stage as shown at the first row in Table 3. This concentration is an unavoidable consequence of mode collapse introduced by our definition of sequential decision process. However, unlike the complete collapse observed with direct topk sampling, our parallel sampling combined with an exploration reward effectively mitigates the issue.

Impact of four rewards in Vapor Reward. During GRPO fine-tuning, we only adjust the taildrop probability without altering the latent-space distribution, our goal is to reduce token usage without degrading generation or reconstruction quality—that is, to achieve an efficiently task-aware adaptive tokenizer.

Firstly, we isolated the individual effects of the generation reward and reconstruction reward. We conducted three ablations: (a) dropping the generation reward, (b) dropping the reconstruction reward, and (c) dropping both. Figure 4 shows that relying solely on the generation reward yields a large boost in generation quality at the expense of reconstruction quality, whereas relying solely on the reconstruction reward greatly improves reconstruction with negligible impact on generation. If both rewards are removed, the model suffers its worst overall performance on both tasks. Notably, using both rewards simultaneously allows steady improvement in both without compromising either. These results confirm that jointly optimizing generation and reconstruction rewards outperforms using either one alone or none at all.



(a) Generation Reward

(b) Reconstruction Reward

Figure 4: The generation (top) and reconstruction (bottom) rewards under different reward weights where the weights order is efficiency, diversity, reconstruction, and generation.

To rigorously assess all four rewards, we carried out a full four-way ablation, removing each reward in turn. The results are summarized in Table 5. Without the efficiency reward, the performance of reconstruction and generation become stronger, but this leads to an increase in the average token cost, violating our efficiency goal. Leaving out the diversity reward produces comparable task performance but causes the taildrop probability to collapse to a few fixed indices, undermining true adaptivity. Omitting

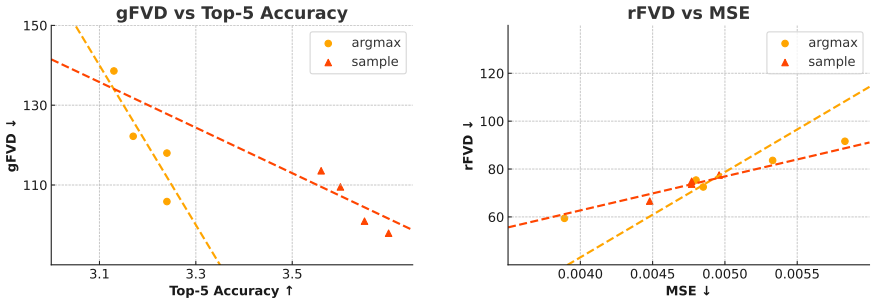
Table 4: Ablation of each rewards where each row "w/o R" indicates the model trained without reward R. The reported token counts are the average number of tokens used per video on the UCF101 validation set.

Missing Reward	#Tokens	rFVD↓	PSNR↑	gFVD↓	MSE↓	ACC↑
VaporTok-GRPO	361	66.6	24.49	97.92	4.48×10^{-3}	3.70%
w/o efficiency reward	874	42.2	27.08	60.17	2.52×10^{-3}	4.44%
w/o diversity reward	325	74.9	24.23	100.94	4.77×10^{-3}	3.65%
w/o reconstruction reward	318	73.7	24.19	109.52	4.77×10^{-3}	3.60%
w/o generation reward	297	77.5	24.04	113.56	4.96×10^{-3}	3.56%

either the reconstruction or generation reward leads to a performance drop in corresponding tasks. In conclusion, dropping any single reward prevents the model from maintaining reconstruction and generation quality in an efficiently adaptive manner.

4.4 The alignment between FVD and reward

To verify the alignment between the final FVD and the reward, the results of reconstruction and generation under both sampling and argmax settings are shown in the Figure 5. It can be observed that rFVD strongly correlates with the reconstruction reward, while gFVD shows a similarly strong correlation with the generation reward. This further supports the validity of our reward design: **the reconstruction MSE serves as a reliable proxy for reconstruction quality, and the top-5 accuracy of the prior model effectively reflects generation quality.**



(a) Alignment between gFVD and ACC (b) Alignment between rFVD and MSE

Figure 5: rFVD and MSE exhibit a strong positive correlation, while gFVD and ACC show a clear negative correlation, indicating the effectiveness and rationality of the proposed reward design.

5 Conclusion and future work

We introduce VaporTok, an efficient and adaptive video tokenizer with two key innovations. First, our probabilistic taildrop leverages visual complexity to dynamically determine truncation indexes, preserving semantic-to-detail token structure. Second, we introduce a parallel sample GRPO strategy guided by the Vapor Reward, a unified signal combining token count, reconstruction quality, and generation fidelity, to inject multiple task-related information to VaporTok. Our results show that adaptive tokenization can be effectively learned during training, and demonstrate the effectiveness of GRPO in optimizing tokenizers. However, the present work explores only how to employ GRPO to optimize the truncation probability rather than the entire VAE, and only the generation task is considered as downstream task. Moreover, although various methods have been employed to mitigate mode collapse, the truncation diversity of VaporTok after GRPO training remains notably lower than that after the prior training. Future work will investigate extending GRPO to entire VAE optimization, modeling video tokenization as a complete Markov decision process and applying the framework to understanding [34, 65, 44] and unified generation&understanding scenarios as in [28, 9, 33, 53]

References

[1] Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. *arXiv preprint arXiv:2403.06963*, 2024.

[2] Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. Flextok: Resampling images into 1d token sequences of flexible length, 2025.

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[4] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.

[5] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11315–11325, 2022.

[6] Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, and Vinci. R1-v: Reinforcing super generalization ability in vision-language models with less than $3. `https://github.com/Deep-Agent/R1-V`, 2025. Accessed: 2025-02-02.

[7] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

[8] Yinda Chen, Haoyuan Shi, Xiaoyu Liu, Te Shi, Ruobing Zhang, Dong Liu, Zhiwei Xiong, and Feng Wu. Tokenunify: Scalable autoregressive visual pre-training with mixture token prediction. *arXiv preprint arXiv:2405.16847*, 2024.

[9] Zisheng Chen, Chunwei Wang, Xiuwei Chen, Hang Xu, Jianhua Han, and Xiandan Liang. Semhitok: A unified image tokenizer via semantic-guided hierarchical codebook for multimodal understanding and generation. *arXiv preprint arXiv:2503.06764*, 2025.

[10] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[11] Shivam Duggal, Phillip Isola, Antonio Torralba, and William T. Freeman. Adaptive length image tokenization via recurrent allocation, 2024.

[12] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[13] Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025.

[14] Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms, 2025.

[15] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In *European Conference on Computer Vision*, pages 102–118. Springer, 2022.

[16] Yuying Ge, Yizhuo Li, Yixiao Ge, and Ying Shan. Divot: Diffusion powers video tokenizer for comprehension and generation, 2024.

[17] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[19] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.

[20] Huiwon Jang, Sihyun Yu, Jinwoo Shin, Pieter Abbeel, and Younggyo Seo. Efficient long video tokenization via coordinate-based patch reconstruction, 2025.

[21] Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot, 2025.

[22] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[23] Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.

[24] Dongyang Liu, Shitian Zhao, Le Zhuo, Weifeng Lin, Yu Qiao, Hongsheng Li, and Peng Gao. Lumina-mgpt: Illuminate flexible photorealistic text-to-image generation with multimodal generative pretraining. *arXiv preprint arXiv:2408.02657*, 2024.

[25] Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470*, 2025.

[26] Zhengxiong Luo, Dayou Chen, Yingya Zhang, Yan Huang, Liang Wang, Yujun Shen, Deli Zhao, Jingren Zhou, and Tieniu Tan. Videofusion: Decomposed diffusion models for high-quality video generation. *arXiv preprint arXiv:2303.08320*, 2023.

[27] Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-magvit2: An open-source project toward democratizing auto-regressive visual generation, 2024.

[28] Chuofan Ma, Yi Jiang, Junfeng Wu, Jihan Yang, Xin Yu, Zehuan Yuan, Bingyue Peng, and Xiaojuan Qi. Unitok: A unified tokenizer for visual generation and understanding. *arXiv preprint arXiv:2502.20321*, 2025.

[29] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

[30] Tsvetomila Mihaylova and André FT Martins. Scheduled sampling for transformers. *arXiv preprint arXiv:1906.07651*, 2019.

[31] Keita Miwa, Kento Sasaki, Hidehisa Arai, Tsubasa Takahashi, and Yu Yamaguchi. One-d-piece: Image tokenizer meets quality-controllable compression, 2025.

[32] Yingzhe Peng, Gongrui Zhang, Miaosen Zhang, Zhiyuan You, Jie Liu, Qipeng Zhu, Kai Yang, Xingzhong Xu, Xin Geng, and Xu Yang. Lmm-r1: Empowering 3b lmms with strong reasoning abilities through two-stage rule-based rl. *arXiv preprint arXiv:2503.07536*, 2025.

[33] Liao Qu, Huichao Zhang, Yiheng Liu, Xu Wang, Yi Jiang, Yiming Gao, Hu Ye, Daniel K Du, Zehuan Yuan, and Xinglong Wu. Tokenflow: Unified image tokenizer for multimodal understanding and generation. *arXiv preprint arXiv:2412.03069*, 2024.

[34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[35] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[36] Dhruv Rohatgi, Adam Block, Audrey Huang, Akshay Krishnamurthy, and Dylan J Foster. Computational-statistical tradeoffs at the next-token prediction barrier: Autoregressive and imitation learning under misspecification. *arXiv preprint arXiv:2502.12465*, 2025.

[37] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[38] Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. Vlm-r1: A stable and generalizable r1-style large vision-language model, 2025.

[39] Junhong Shen, Kushal Tirumala, Michihiro Yasunaga, Ishan Misra, Luke Zettlemoyer, Lili Yu, and Chunting Zhou. Cat: Content-adaptive image tokenization, 2025.

[40] Ivan Skorokhodov, Willi Menapace, Aliaksandr Siarohin, and Sergey Tulyakov. Hierarchical patch diffusion models for high-resolution video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7569–7579, 2024.

[41] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[42] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

[43] Anni Tang, Tianyu He, Junliang Guo, Xinle Cheng, Li Song, and Jiang Bian. Vidtok: A versatile and open-source video tokenizer. *arXiv preprint arXiv:2412.13061*, 2024.

[44] Zineng Tang, Long Lian, Seun Eisape, XuDong Wang, Roei Herzig, Adam Yala, Alane Suhr, Trevor Darrell, and David M. Chan. Tulip: Towards unified language-image pretraining, 2025.

[45] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.

[46] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[47] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[48] Hanyu Wang, Saksham Suri, Yixuan Ren, Hao Chen, and Abhinav Shrivastava. Larp: Tokenizing videos with a learned autoregressive generative prior. *arXiv preprint arXiv:2410.21264*, 2024.

[49] Junke Wang, Yi Jiang, Zehuan Yuan, Bingyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. *Advances in Neural Information Processing Systems*, 37:28281–28295, 2024.

[50] Junke Wang, Zhi Tian, Xun Wang, Xinyu Zhang, Weilin Huang, Zuxuan Wu, and Yu-Gang Jiang. Simplear: Pushing the frontier of autoregressive visual generation through pretraining, sft, and rl, 2025.

[51] XuDong Wang, Xingyi Zhou, Alireza Fathi, Trevor Darrell, and Cordelia Schmid. Visual lexicon: Rich image features in language space, 2024.

[52] Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. *arXiv preprint arXiv:2410.13848*, 2024.

[53] Yecheng Wu, Zhuoyang Zhang, Junyu Chen, Haotian Tang, Dacheng Li, Yunhao Fang, Ligeng Zhu, Enze Xie, Hongxu Yin, Li Yi, Song Han, and Yao Lu. Vila-u: a unified foundation model integrating visual understanding and generation, 2025.

[54] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[55] Yazhou Xing, Yang Fei, Yingqing He, Jingye Chen, Jiaxin Xie, Xiaowei Chi, and Qifeng Chen. Large motion video autoencoding with cross-modal video vae. *arXiv preprint arXiv:2412.17805*, 2024.

[56] Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. Dancegrpo: Unleashing grpo on visual generation, 2025.

[57] Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. Elastictok: Adaptive tokenization for image and video, 2025.

[58] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[59] Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, Bo Zhang, and Wei Chen. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization, 2025.

[60] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

[61] Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, et al. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10459–10469, 2023.

[62] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.

[63] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation, 2024.

[64] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025.

[65] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[66] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[67] Sijie Zhao, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Muyao Niu, Xiaoyu Li, Wenbo Hu, and Ying Shan. Cv-vae: A compatible video vae for latent generative video models. *arXiv preprint arXiv:2405.20279*, 2024.

[68] Ziqin Zhou, Yifan Yang, Yuqing Yang, Tianyu He, Houwen Peng, Kai Qiu, Qi Dai, Lili Qiu, Chong Luo, and Lingqiao Liu. Hitvideo: Hierarchical tokenizers for enhancing text-to-video generation with autoregressive large language models, 2025.

[69] Xianwei Zhuang, Yuxin Xie, Yufan Deng, Dongchao Yang, Liming Liang, Jinghan Ru, Yuguo Yin, and Yuexian Zou. Vargpt-v1.1: Improve visual autoregressive large unified model via iterative instruction tuning and reinforcement learning, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Reflected in Section1 contribution.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation in conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper do not propose new theory.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The reproduce details are already included in the experiment section and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code in supplementary.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details are reported in the experiment section and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We did not include error bars as the experiments are computationally expensive, and existing related literature do not have the convention to report error bras.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: We discuss the details in the first subsection of experiment and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed and followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impact in appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We expect our model to have similar risk as the Lumina-mgpt[24] or other models. We discuss potential mitigation strategies in appendix.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cited all code, data, and models used in this paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Included code in supplementary material.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work do not include human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Only used LLMs for editing (e.g., grammar, spelling, and word choice).

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix:

A Additional Methodological Details about VaporTok

A.1 Computation of video spatio&temporal complexity

Given a video tensor $V \in \mathbb{R}^{T\times H\times W\times 3}$, each frame is converted to grayscale via

$$g_t(x,y) = 0.299\, V_{t,x,y,1} + 0.587\, V_{t,x,y,2} + 0.114\, V_{t,x,y,3}, \tag{20}$$

,where $t = 1,\dots,T$ and $(x,y) \in \{1,\dots,H\} \times \{1,\dots,W\}$.

Spatial Complexity. Define the empirical pixel-value distribution of frame t as

$$p_t(v) = \frac{1}{HW}\big|\{(x,y) \mid g_t(x,y) = v\}\big|, \quad v = 0,1,\dots,255. \tag{21}$$

The Shannon entropy of frame t is

$$H_t = -\sum_{v=0}^{255} p_t(v)\log_2 p_t(v), \tag{22}$$

and the spatial complexity is the average frame entropy:

$$\mathrm{SC} = \frac{1}{T}\sum_{t=1}^{T} H_t. \tag{23}$$

Temporal Complexity. The temporal complexity is defined as the mean absolute difference between consecutive frames:

$$\mathrm{TC} = \frac{1}{(T-1)\,H\,W}\sum_{t=1}^{T-1}\sum_{x=1}^{H}\sum_{y=1}^{W}\big|g_{t+1}(x,y) - g_t(x,y)\big|. \tag{24}$$

A.2 AR prior in VaporTok

Inspired by the AR prior model introduced in LARP [48], we integrate a similar lightweight autoregressive model into VaporTok as shown in Figure 6. This AR model is designed to make latent tokens more compatible with downstream AR generation tasks, and thus its implementation and associated evaluation metrics can serve as a proxy for downstream generation performance.

Similar to LARP, our lightweight AR model is trained jointly with the VaporTok tokenizer in an end-to-end manner. Specifically, the model takes the quantized latent token embeddings as input, and uses the corresponding codebook IDs as labels. To address the instability caused by the training-inference discrepancy inherent to AR models, Scheduled Sampling Mixing as proposed in [3, 30] is employed.

The key difference is that the prior model used in VaporTok is trained **only on the retained latent tokens after truncation** rather than the whole latent space. Moreover, to enable efficient batchwise training, we adopt the attention masking scheme described in Section A.3 within the transformer blocks of the AR prior model.

A.3 Attention mask in reconstruction&generation pipeline

Attention mask for the VaporTok decoder. During the training of VaporTok, the spatiotemporal complexity of each video sample varies, which results in different taildrop probabilities. Additionally, since sampling is performed over the entire probability distribution during training. These two factors lead to varying truncation positions across samples. Consequently, it becomes infeasible to reconstruct all samples within a batch using a shared decoder input length.To address this issue, we design an adaptive attention mask for the VaporTok decoder to accommodate the variable token lengths caused by the probabilistic taildrop. Specifically, as illustrated in Figure 7(a): For each sample, we construct an individual attention mask: all tokens from decoder queries M are granted

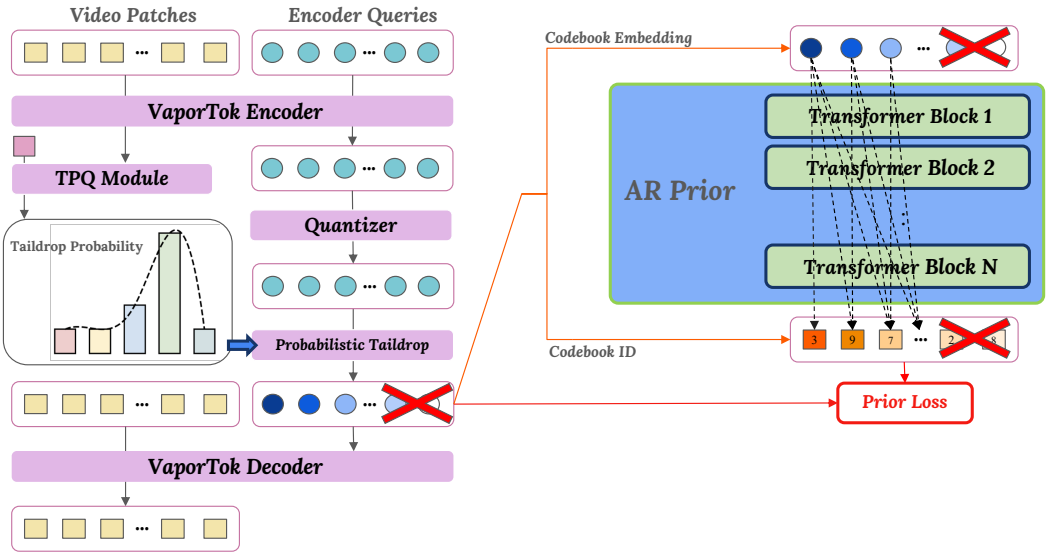


Figure 6: AR prior model in VaporTok.

full visibility, while for latent tokens, positions **beyond the truncation point** are masked out to ensure that dropped tokens do not participate in attention computation. This prevents non-informative tokens from interfering with the training process and allows batchwise training of VaporTok.

Attention mask for the AR prior model. The AR prior model originally adopts a causal attention mask, which ensures that later tokens do not affect the prediction of earlier tokens. However, if we apply a standard causal mask without modification, tokens before the truncation point can still attend to and influence those after the truncation, which is undesirable. Due to the introduction of taildrop, tokens after the truncation point should not be supervised and influenced by prior tokens during training. To resolve this, we propose a modified attention mask as shown in Figure 7(b).

Attention mask for the downstream AR generative model. Since the AR prior model serves as a compact abstraction of the downstream AR generative model, the attention mask used in the downstream AR model is identical to that of the AR prior model, which is also illustrated as Figure 7(b).

A.4 The complete loss function of VaporTok

$$\mathcal{L}_{\text{rec}} = \lambda_{\text{L1}} \cdot \mathcal{L}_{\text{L1}} + \lambda_{\text{perc}} \cdot \mathcal{L}_{\text{perc}} + \lambda_{\text{GAN}} \cdot \mathcal{L}_{\text{GAN}} + \lambda_{\text{commit}} \cdot \mathcal{L}_{\text{commit}} \tag{25}$$

$$\mathcal{L}_{\text{representation prior}} = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i \mid x_i) \tag{26}$$

$$\mathcal{L}_{\text{probability prior}} = \text{KL}(P \,||\, GaussianPrior) \tag{27}$$

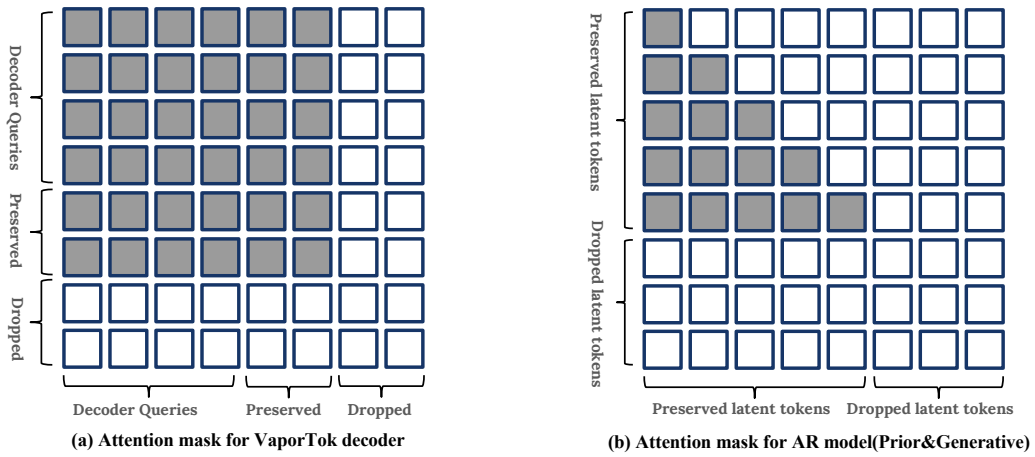
$\mathcal{L}_{\text{rec}}$ is comprised of L1 loss, perceive loss, GAN loss, and commitment loss as traditional VQ tokenizer. In Equation 26, x denotes codebook embedding, y denotes codebookid. In Equation 27, P denotes taildrop probability and $GaussianPrior$ denotes prior probability. The complete loss of VaporTok is:

$$\mathcal{L}_{\text{complete}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{rep}} \cdot \mathcal{L}_{\text{rep\_prior}} + \lambda_{\text{prob}} \cdot \mathcal{L}_{\text{prob\_prior}} \tag{28}$$

B Supplementary Experiments on Vapor Reward

B.1 Penalty reward

In our experiment of argmax sampling of probabilistic taildrop, the truncation index always becomes too small to be enough to reconstruct&generate videos, so a reward to punish such truncation is



(a) Attention mask for VaporTok decoder

(b) Attention mask for AR model(Prior&Generative)

Figure 7: Attention mask for different parts.

introduced in such senario. Specifically, for the i-th path, the penalty reward is defined as:

$$R^{(i)}_{\text{penalty}} = -\sum_{j=1}^{L} \text{Penalty}(I_{i,j}), \quad \text{if } I_{i,j} < N_{\text{threshold}} \tag{29}$$

where $K_{threshold}$ is a manually defined threshold specifying the minimum number of tokens tolerable for reconstructing the video clip and $\text{Penalty}(t_i)$ denotes a penalty function that imposes more punishment when the truncation index becomes smaller. Then we design a five way ablation study in argmax sampling strategy of probabilistic taildrop as shown in Table 5:

Table 5: Ablation of each rewards. Each row "w/o R" indicates the model trained without reward R. Lower is better for rFVD, gFVD and MSE; higher is better for PSNR and prior top-5 accuracy.

Missing Reward	**#Tokens**	**rFVD↓**	**PSNR↑**	**gFVD↓**	**MSE↓**	**ACC↑**
VaporTok-GRPO	299	72.5	24.1	118	4.85×10^{-3}	3.24%
w/o efficiency reward	577	59.4	25.6	88.46	3.89×10^{-3}	3.88%
w/o diversity reward	305	75.4	24.7	105.85	4.80×10^{-3}	3.24%
w/o reconstruction reward	272	91.6	23.3	138.58	5.83×10^{-3}	3.13%
w/o generation reward	286	83.6	23.8	122.21	5.33×10^{-3}	3.17%
w/o penalty reward	58	3267	10.7	3255.59	9.45×10^{-2}	2.73%

B.2 Effect of reconstruction&generation reward under different weight

We adopt a baseline weighting of 1:1:1:1:1 for the efficiency, penalty, diversity, reconstruction, and generation rewards respectively and then increased the proportion of each of reconstruction&generation rewards. As shown in Figure 8, as the weights for these two rewards grow, their numerical values also increase, which demonstrates that the higher the combined weight on these two performance rewards, the more faithfully the original reconstruction and generation quality is preserved. It is worth noting that if both weights are set too high, performance gains come at the expense of efficiency reward, contradicting our original intent that making tokenizer to be more efficient.

(a) Generation Reward

(b) Reconstruction Reward

(c) Efficiency Reward

(d) Diversity Reward

Figure 8: (a) generation reward (b) reconstruction reward (c) efficiency reward (d) diversity reward of differenct reward weight where the order is efficiency, penalty, diversity, reconstruction, generation.

C Analysis about VaporTok

C.1 Mitigating the three core challenges of autoregressive generation

To mitigate the three key limitations of autoregressive (AR) visual generation, we propose a unified solution within the VaporTok framework:

Quadratic complexity with long sequences. We introduce a *sparse but sufficient* token representation by leveraging visual priors to reduce the number of tokens required for generation. Furthermore, we apply GRPO to adaptively compress the token sequence while preserving downstream task performance as much as possible.

Error accumulation during AR inference. We propose a *probabilistic taildrop* training strategy that pushes important tokens toward the beginning of the visual representation. As a result, during inference, the model generates the most critical tokens when the accumulation of prediction errors is still minimal, thereby mitigating the impact of error accumulation.

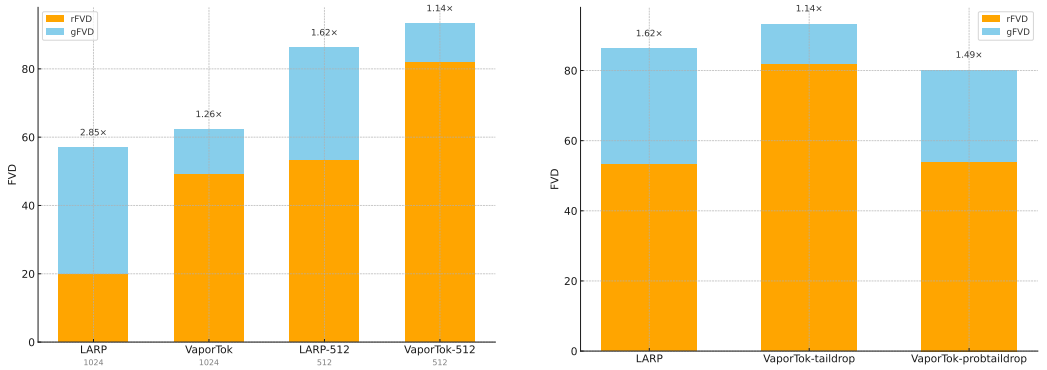
Training gap between the tokenizer and the AR generator. To close this gap, we adopt two complementary strategies. First, we refine the latent space following the approach of LARP [48] to make it more suitable for AR generation. Second, we leverage the same AR-aligned information to guide the training of the taildrop query module via GRPO. This enables our adaptive tokenizer to incorporate downstream task constraints into both the adaptivity mechanism and the latent token representation, thus reducing the mismatch between the tokenizer and the generator.

C.2 Taildrop training strategy for semantic-to-detail representation

The taildrop training strategy is designed to encourage a *semantic-to-detail* organization in the latent token representations. This is primarily achieved through the following mechanism: *for the same video sample, train the model using different numbers of tokens for reconstruction*. As a result, tokens at the beginning of the sequence are exposed to the reconstruction objective more frequently, while later tokens appear less often during training. Furthermore, even when only a small number of tokens are used, the model is still tasked with reconstructing the entire video. This encourages early tokens to encode more global, semantic information that is sufficient for reconstruction.

In addition, it is worth noting the distinction between the two evaluation metrics used in visual reconstruction and generation: rFVD and gFVD. By design, gFVD is consistently worse (higher) than rFVD. This is because rFVD evaluates the reconstruction quality using ground-truth codebook ID directly obtained from the encoder, whereas gFVD evaluates the generation quality using codebook ID predicted by the autoregressive model. *Briefly, the only difference between rFVD and gFVD is that*

reconstruction relies on ground-truth ID, while generation depends on autoregressively predicted ID, making the gap between gFVD and rFVD a direct indicator of the degree of AR error accumulation.



(a) Comparison between LARP and VaporTok with naive taildrop under 1024 and 512 token budgets

(b) Comparison between no technique, naive taildrop, and probabilistic taildrop under 500 token budgets

Figure 9: (a) Naive taildrop reduces the gap between rFVD and gFVD, but it leads to a drop in reconstruction performance, which in turn results in degraded generation performance. (b) Probabilistic taildrop, by incorporating a visual prior, avoids the reconstruction performance degradation caused by taildrop, while preserving its original ability to reduce the gap between rFVD and gFVD.

To quantify this effect, we report the **gFVD/rFVD ratio** as shown in Table 2 of the main paper and Figure 9a to measure the discrepancy between reconstruction and generation. We conduct experiments under both 1024-token and 512-token settings. The results show that with taildrop enabled, the gap between gFVD and rFVD becomes smaller, demonstrating the effectiveness of taildrop in mitigating AR error accumulation; specifically, the gFVD/rFVD ratio decreases from 2.85 to 1.26 under the 1024-token setting, and from 1.62 to 1.14 under the 512-token setting.

C.3 Difference between naive taildrop and probabilistic taildrop

The distinction between naive and probabilistic taildrop primarily manifests in training and inference:

Training: While naive taildrop uses uniform sampling for truncation during training, probabilistic taildrop samples from a learned, prior-informed distribution. (We also investigate three specific sampling strategies under the probabilistic framework.)

- *Truncation by argmax of taildrop probability.* The sequence is truncated at the index corresponding to the maximum value in the taildrop probability distribution. While simple, this approach always uses the same number of tokens for a given input, limiting the semantic-to-detail effect.
- *Sampling from the full taildrop probability.* The truncation index is sampled from the entire taildrop probability distribution. This allows different truncation lengths for the same input and leads to strong reconstruction performance.
- *Sampling indices before the argmax index.* We sample a truncation index from the region before the argmax position, based on the taildrop probability. This also enables varying token counts across samples, though typically results in fewer tokens and slightly degraded reconstruction quality.

Inference: During inference, naive taildrop uses the full set of tokens for decoding (which is the default inference mode used for all reported results in this paper. Alternatively, one may adopt a threshold-based token selection strategy, as in [57]). In contrast, probabilistic taildrop performs decoding using all tokens preceding the argmax index of the taildrop probability distribution, achieving an adaptive number of tokens based on input complexity.

It is worth noting that, among the three sampling strategies for probabilistic taildrop during training (from argmax to sample and then to pre-sample), the degree of *semantic-to-detail* structure in the

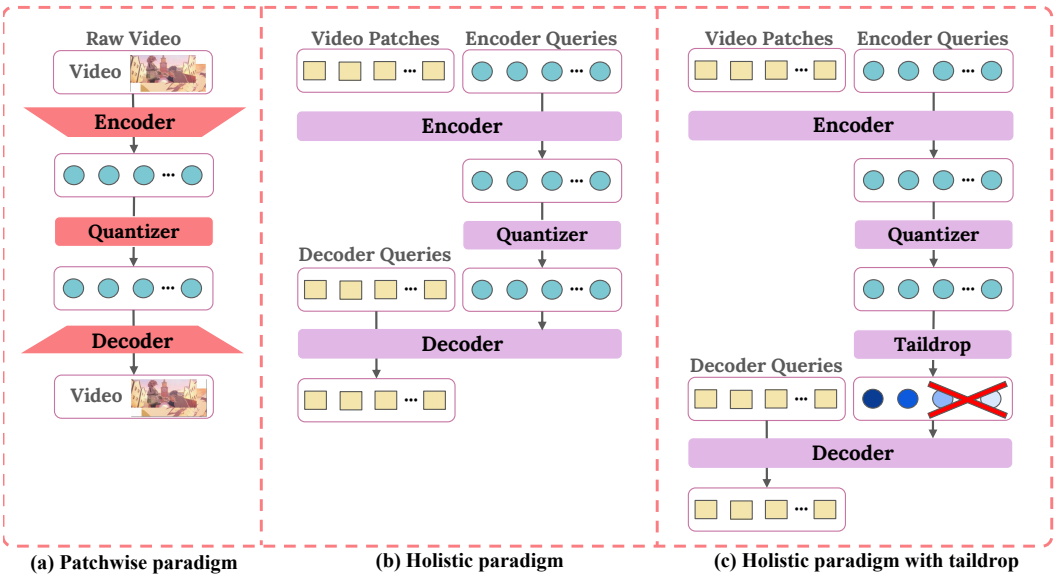


Figure 10: Different paradigm of visual tokenizer: (a) The patchwise-token paradigm typically represents each token as encoding information from a specific spatial region and usually adopts a CNN as the backbone.(b) The holistic-token paradigm is not constrained by fixed spatial positions and can flexibly adjust the information each token represents based on the training strategy. (c) Taildrop is a training technique commonly used in the holistic-token paradigm, enabling the token sequence to exhibit a semantic-to-detail property.

tokens used during inference gradually increases because of the frequency of sampling index before argmax index is gradually frequent. In other words, these strategies increasingly mitigate AR error accumulation. This trend is also validated by the experimental results reported in Table 2 of the main text as the gFVD/rFVD ratio becomes smaller.

Furthermore, while naive taildrop helps narrow the gap between reconstruction and generation (rFVD vs. gFVD), its uninformed dropping during training—without accounting for visual priors—results in compromised reconstruction quality. In contrast, our probabilistic taildrop incorporates a learned prior distribution, maintaining competitive reconstruction performance (rFVD) and achieving superior generation quality by alleviating AR error accumulation.

C.4 Different visual tokenizer paradigm

Visual tokenizers can be classified according to various criteria, and one particularly informative distinction is how they map visual patches to latent tokens, yielding two families: Patchwise-Token and Holistic-Token.

In the Patchwise-Token paradigm as depicted in Figure 10(a) , each learned token corresponds one-to-one with a visual patch, thereby preserving the spatial structure imposed. Briefly, given a video input $V \in \mathbb{R}^{T\times H\times W\times 3}$, the encoder outputs a downsampled feature map

$$Z = \mathrm{Enc}(V) \in \mathbb{R}^{\frac{T}{f_T}\times\frac{H}{f_H}\times\frac{W}{f_W}\times D}, \tag{30}$$

where f_T, f_H, f_W are the temporal and spatial downsampling factors. The reconstructed video is then obtained as

$$\widehat{V} = \mathrm{Dec}(Z) \in \mathbb{R}^{T\times H\times W\times 3}. \tag{31}$$

In the Holistic-Token paradigm as depicted in Figure 10(b), each latent token may assume different semantic roles depending on the training strategy, resulting in a more flexible representational scope. Different from Patchwise-Token paradigm mainly depends on 3D CNN as the backbone, Holistic-Token paradigm usually employs Transformer blocks as the backbone, so a simple patch embedding layer is needed to be conducted to $V \in \mathbb{R}^{T\times H\times W\times 3}$

$$P = \mathrm{Patchify}(V) \in \mathbb{R}^{(\frac{T}{f_T}\times\frac{H}{f_H}\times\frac{W}{f_W})\times D}, \tag{32}$$

and then P will be concated with K learnable query tokens $Q \in \mathbb{R}^{K \times D}$ and the combined sequence will be passed into the encoder:

$$Z_P \oplus Z_Q = \mathrm{Enc}\big(P \oplus Q\big) \in \mathbb{R}^{(\frac{T}{f_T} \times \frac{H}{f_H} \times \frac{W}{f_W} + K) \times D}, \tag{33}$$

where $\oplus$ denotes concatenation and Z_P,Z_Q denotes the represatation of P, Q after encoder respectively. During detokenization, $M \in \mathbb{R}^{(\frac{T}{f_T} \times \frac{H}{f_H} \times \frac{W}{f_W}) \times D}$, the decoder query of the same shape as the video patches P, will be concatenated with Z_Q and passed to the decoder to renconstruct the input video.

$$\widehat{V} = \mathrm{Dec}\big(M \oplus Z_Q\big) \in \mathbb{R}^{T \times H \times W \times 3}. \tag{34}$$

It is worth noting that *the taildrop training strategy is typically employed in holistic-token tokenizers*. This is primarily because the query representation is not constrained by fixed spatial regions, allowing it to flexibly adapt to different training objectives and strategies. Moreover, such flexibility enables the model to achieve effective representation learning with less training data.

D Supplementary Related Work about Fixed-length Visual Tokenizer

Visual tokenizers for understanding tasks typically rely on contrastive learning, for example: CLIP[34] trains paired image and text encoders with an InfoNCE contrastive loss over matched image–caption pairs, enabling strong zero-shot transfer across diverse vision task; SigLIP[65] replaces CLIP's softmax-based InfoNCE loss with an independent pairwise sigmoid loss, removing the need for global normalization and scaling more efficiently to very large or small batch sizes. TULIP[44] augments CLIP-style pretraining with generative data augmentation and unified image–image, text–text, and image–text contrastive objectives plus reconstruction regularization to learn fine-grained visual features without sacrificing semantic alignment.

Whereas visual tokenizers designed for generation usually employ VAE-based architectures: VQ-VAE[47] first introduce vector quantization into VAE, transforming data from continuous spaces into discrete tokens to simplify modeling and circumvent issues of "posterior collapse" in VAE framework; VQ-GAN[12] improves image reconstruction quality by introducing adversarial loss and using a Transformer for autoregressive visual generation; FSQ[29] projects representations into a lower dimensional space for quantization into fixed values, while its variant LFQ[62, 27] further simplifies the process by using binary quantized representations. This new kind of quantization method effectively enhances the AR generation paradigm by dramatically enlarging the vocab size and improving the encoding efficiency. Beyond these patch-to-token VAEs, there are also VAEs that learn holistic tokens, such as TiTok[63] and LARP[48], by compressing visual information into a holistic query, they eliminate the patch-to-token correspondence constraint, yielding a more flexible architecture with inherent compression potential.

Recently, along with the rapid development of the unified model[52, 54, 53, 69], there are also several visual tokenizers designed for both generation and understanding, such as TokenFlow[33], SemHiTok[9] and UniTok[28]. These works focus on unifying visual generation and understanding within a single visual tokenizer by employing discrete representations and hierarchical or multi-codebook strategies, addressing the differing requirements in token granularity and semantic level between generation and understanding tasks.

While the above methods have shown impressive results for static images, extending visual tokenizers to video requires capturing both temporal continuity and spatial detail, presenting new challenges for tokenizer architectures and training paradigms. Recent work addresses this by integrating temporal modeling[49], diffusion-guided reconstruction[16], hierarchical codebooks[68], and coordinate-based patch schemes[20] to efficiently compress and faithfully reconstruct long video sequences.

E Visualization

We present visualizations of VaporTok's reconstruction and final class-based generation results, as shown in the Figure 11 and Figure 12.

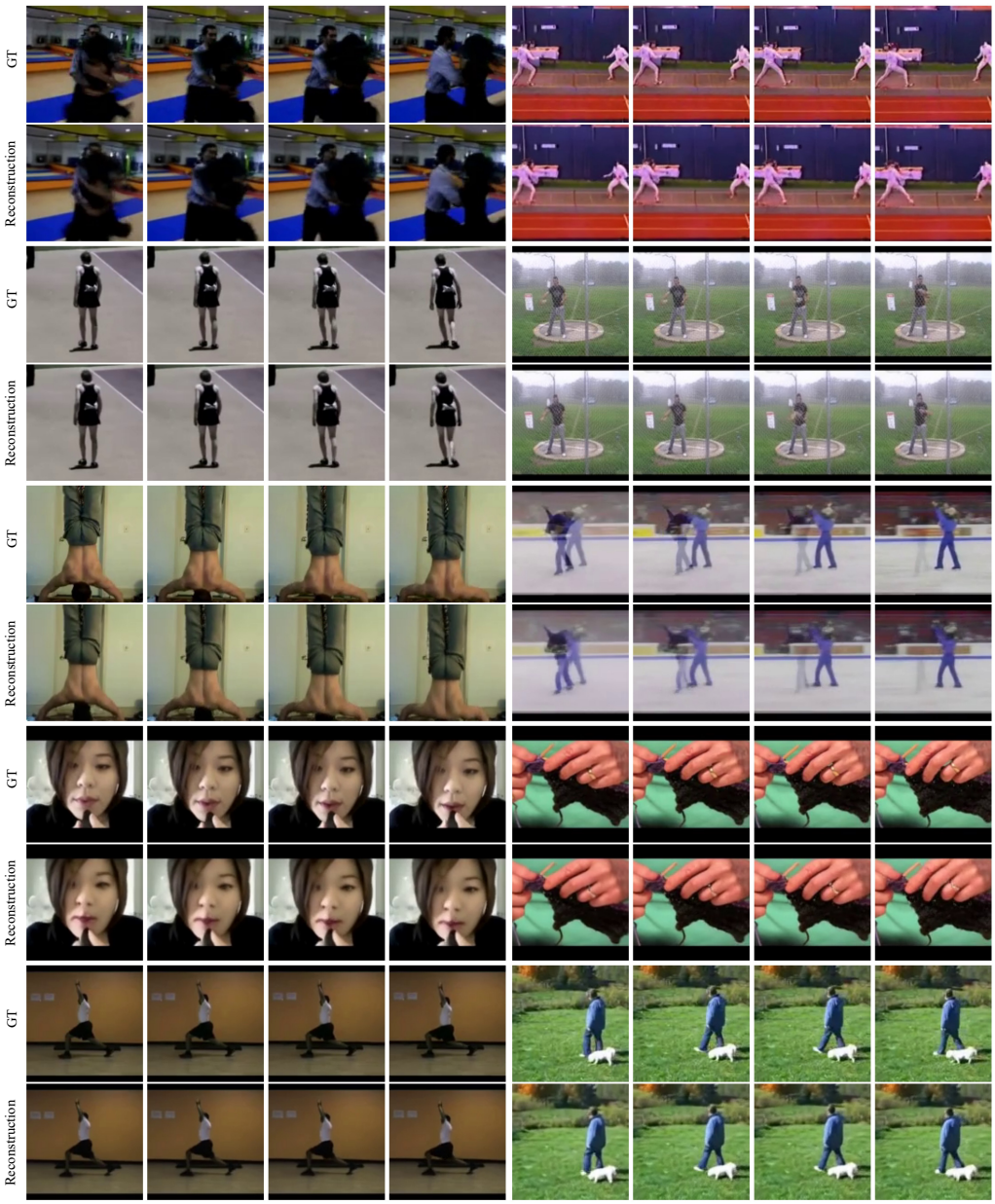


Figure 11: Video reconstruction on UCF101.

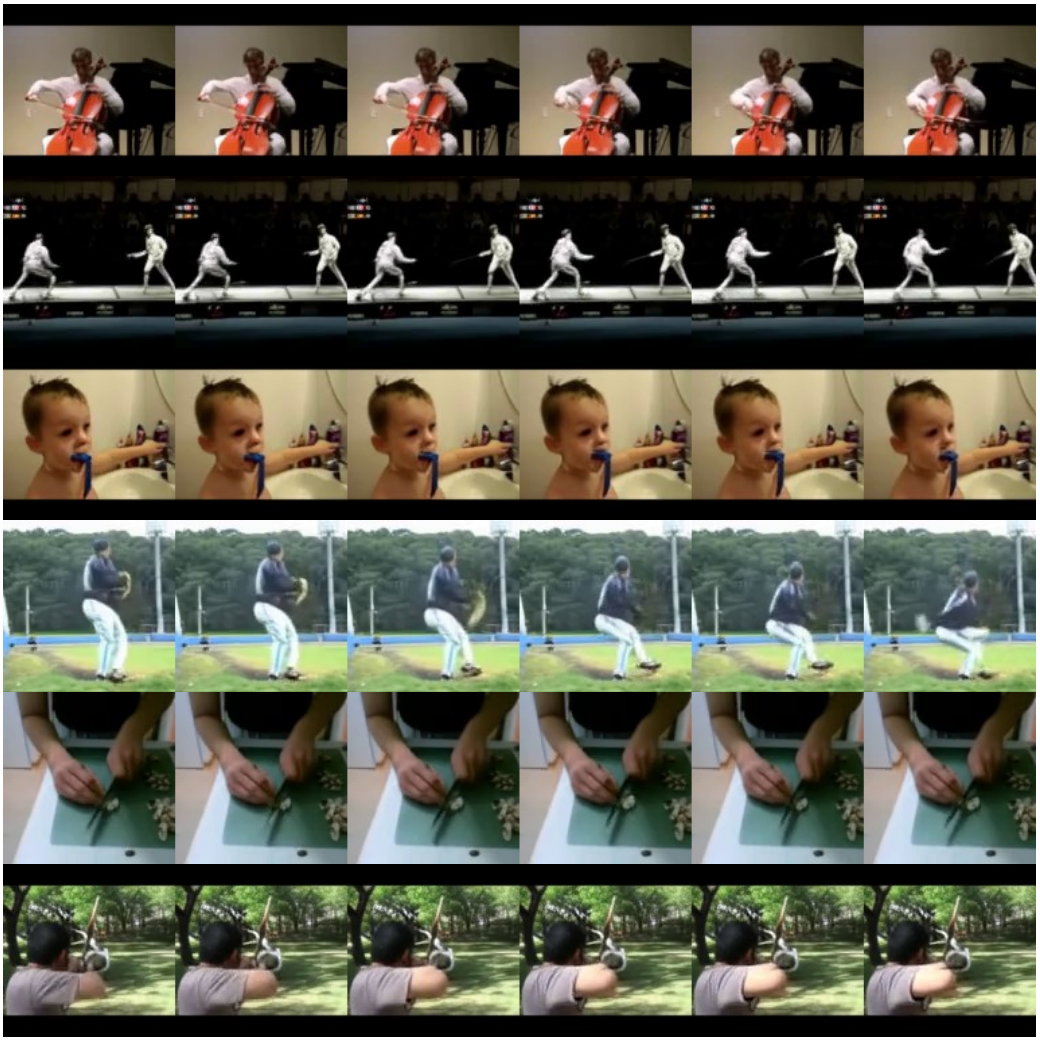


Figure 12: Class-based video generation on UCF101.

F Broader Impacts

Our Adaptive Video Tokenizer is designed exclusively for autoregressive video generation tasks, and is not intended for video understanding or classification; by allocating fewer tokens to low-complexity videos and more to high-detail videos, our method reduces computational and bandwidth costs for real-time generative applications (e.g., interactive video editing, virtual content creation); it enables fast, adaptive video synthesis for artistic tools and educational simulators, lowering barriers for non-expert users to generate high-fidelity video content; it facilitates deployment of generative video models on edge devices (e.g., AR/VR headsets, mobile phones) by reducing token sequence length and inference latency; however, improved efficiency in video generation could be misused to produce highly realistic deepfake videos, exacerbating misinformation campaigns; although not designed for recognition, the underlying tokenizer could be adapted to generate misleading synthetic footage for surveillance evasion or identity spoofing; training on unbalanced datasets may lead the tokenizer to allocate token budgets unevenly, causing generative artifacts that disproportionately affect certain demographics; while inference is more efficient, training the dual-branch model remains GPU-intensive, contributing to carbon emissions.

G Safeguards

To mitigate potential misuse and harms, we will release a detailed model card specifying that use for deceptive or harmful video synthesis (e.g., deepfakes) is prohibited; distribute weights under a non-commercial, no-derivatives license (e.g., CC BY-NC-ND) and/or via an API with rate limits rather than open weight download; embed imperceptible watermarks in generated videos for provenance tracking and provide a companion detection model to flag synthetic content; maintain a public issue tracker for misuse reports and regularly update the model to address discovered biases or vulnerabilities; publish training logs, compute cost estimates, and carbon-emission metrics to inform users of environmental impact.