EVATok: Adaptive Length Video Tokenization for Efficient Visual Autoregressive Generation

Supplementary Material

Content of the Appendix

This supplementary material includes the following content:

- Sec. A discusses the limitations of EVATok.
- Sec. B provides the plan for the future work.
- Sec. C gives the detailed implementation of the four-stage framework of EVATok and downstream adaptive length AR models.
- Sec. D provides the reconstruction performances of our final tokenizers, including qualitative examples for the adaptive length reconstruction and generation, as well as cases of how VideoMAE discriminator affects video reconstruction perceptually.
- Sec. E provides the compute cost analysis for our four-stage framework and downstream AR generation model training.
- Sec. G analyzes the router's max-proxy-reward assignment predictions in terms of accuracy and proxy reward.
- Sec. F explains the attention mask mechanism in our Q-Former style video adaptive tokenizers.
- Sec. H shows the results for translating the solution of EVATok to image adaptive tokenization.

A. Limitations

In this work, we focus on addressing the key challenge in adaptive length video tokenization: identifying the optimal assignment. Although we have demonstrated the superiority of our method in video reconstruction, as well as downstream autoregressive (AR) class-to-video generation and frame prediction tasks, our experiments were limited to $16 \times 128 \times 128$ video clips. We did not evaluate it on videos with higher resolution or longer duration that align with industry-level requirements [3, 18]. Additionally, due to limited computational resources, we have not validated EVATok on more complex downstream tasks, such as text-to-video generation.

Furthermore, when extending video duration in adaptive length video tokenizers, the number of possible assignment choices can grow exponentially if exhaustive searching is naively applied. Although this issue is not addressed in the current work, we discuss a potential solution in our future work section (Sec. B), which can reduce the complexity of optimal assignment searching from $O(m^t)$ to around $O(t^2)$ with respect to the maximum video duration t.

B. Future Work

Adaptive length video tokenization on longer videos. In the main paper, when identifying the optimal assignment for a video clip with T temporal blocks and m possible token number choices for each temporal block, we search for m^T possible assignments to find the optimal one. This approach will become unaffordable for larger T. To address this, in future work, we will explore a method that searches for optimal assignments approximately in an autoregressive way. For example, for a video with $2T$ temporal blocks, we can first search the m^T possible assignments for the first T blocks, then, based on the optimal assignment for the first T blocks, we continue to search the m^T possible assignments for the T blocks. Therefore, if we assume the reconstruction cost for the proxy tokenizer increases linearly with longer T, then the complexity for optimal assignment searching is estimated to be $O(T^2)$.

Extension to adaptive length video VAE and diffusion models. The idea of adaptive length video tokenization is not limited to discrete tokenizers, and can naturally transfer to adaptive length VAE [8, 9] training. While it is natural for AR models to learn on variable length sequences, the performance of diffusion models on denoising adaptive length sequences can be discussed in future work.

Router improvements. In our current implementation, the preference weights are implicitly fixed during the process of training data curation for routers. In the future work, we may want the preference weights to be able to be input explicitly for the routers for more flexible applications.

C. Implementation Details

Tokenizer training. On WebVid-10M, we train variable length tokenizer on 3 FPS video frames, following the approaches in VidTok [16]. On UCF & K600 dataset, we train tokenizers on video frames with their original FPS, following typical settings. We use a cosine learning rate schedule. The maximum learning rate is 1×10^{-4} and end learning rate is 1×10^{-6}. The batch size is 128, and proxy tokenizers are all trained for only 400k iterations before being used for proxy reward calculation. The final video adaptive tokenizers are trained for 1000k iterations, whose training cost is aligned with previous work [10, 19].

Proxy reward calculation. For proxy reward calculation from the main paper:

$$R_{\text{proxy}} = w_q Q(\mathcal{E}_{\text{proxy}}, x, a) - w_l L(a) \tag{1}$$

Specifically, we calculate $Q(\mathcal{E}_{\text{proxy}}, x, a)$ as:

$$Q(\mathcal{E}_{\text{proxy}}, x, a) = \frac{\text{LPIPS}(\mathcal{E}_{\text{proxy}}(x, a), x) - \text{MEAN}_{\text{LPIPS}}}{\text{STD}_{\text{LPIPS}}} \tag{2}$$

where $\text{LPIPS}(\mathcal{E}_{\text{proxy}}(x, a), x)$ is the LPIPS value between original video x and the reconstruction result $\mathcal{E}_{\text{proxy}}(x, a)$ using proxy tokenizer $\mathcal{E}_{\text{proxy}}$ under assignment a. $\text{MEAN}_{\text{LPIPS}}$ denotes the expectation of $\mathcal{E}_{\text{proxy}}(x, a)$ for randomly sampled x from all the training videos and randomly sampled a from all candidate assignments, and $\text{STD}_{\text{LPIPS}}$ represents the standard deviation of $\mathcal{E}_{\text{proxy}}(x, a)$ for random x and a. We choose LPIPS for per-video reconstruction quality measurement, as it is a metric designed to better align with human perception. We calculate $L(a)$ as:

$$L(a) = \frac{\sum_{k=1}^{T} a[k] - \text{MEAN}_L}{\text{STD}_L} \tag{3}$$

where $\sum_{k=1}^{T} a[k]$ is the sum of the allocated tokens across all T temporal blocks. MEAN_L and STD_L are the expectation of $\sum_{k=1}^{T} a[k]$ for randomly sampled a.

Router training. We train 19.9M ViT-S size routers with a batch size of 128 for 50k iterations. We optionally use frozen V-JEPA2 [1] to patchify raw frames into video embeddings. Otherwise, we use the typical learnable linear projection for patch embeddings. In practice, we find that there is no obvious performance gap between these two visual embedding strategies.

AR model training. For the adaptive length token sequences produced by EVATok, before each temporal block, a special token indicating the number of tokens for the upcoming temporal block will be inserted for AR training. Therefore, for AR inference, before generating the tokens of the next temporal block, the AR model first predicts the special tokens indicating the length of the next block. On UCF-101, AR models are trained for 3000 epochs using WSD [6] learning rate schedule, where the learning rate is kept constant for the first 80% of the training and quickly annealed to 0 in the rest 20% training iterations. On K600, the AR models are trained for 75 epochs with the same learning rate scheduler.

AR inference for adaptive length video generation. In adaptive length AR generation, we observe that even with a special token preceding each temporal block to indicate the number of tokens in the upcoming block, the AR model may still occasionally produce unexpected tokens during inference (*e.g.*, sampling a special token when a visual token is expected, or vice versa). To ensure the model generates precisely the number of tokens specified by the preceding special token for each temporal block, we employ a logit-masking strategy. For instance, when sampling the first token of a variable length sequence, which is expected to be a special token denoting the token count for the initial block, all logit entries corresponding to visual tokens are masked to $-\texttt{inf}$ before the softmax operation. This guarantees that only a special token is sampled. Subsequently, for the next k tokens (as indicated by the special token), the logits for special tokens are masked, ensuring only visual tokens are generated. This process continues until m special tokens and their corresponding temporal blocks are generated. This approach incurs nearly no additional computational overhead and guarantees the generated variable length sequence maintains the correct structure. We use constant classifier free guidance (CFG) schedules for AR model inference during class-to-video sampling. For GPT-B models, the CFG value is 2.5, for larger GPT models, we use 1.75 CFG value.

Settings / Dataset	PSNR↑	LPIPS↓	rFVD↓	#rTokens↓
Final w/ VideoMAE Disc. (WebVid)	27.37	0.1063	7.3	721
Final w/o VideoMAE Disc. (WebVid)	28.18	0.0983	32	721
Final w/ VideoMAE Disc. (UCF&K600)	25.75	0.1140	9.7	774

Table 1. **The detailed performances of final tokenizers reconstruction results.** All models are trained for the full 1000k iterations. The tokenizers trained on WebVid-10M are evaluated on the WebVid validation set, and the tokenizers trained on UCF-101 and K600 are evaluated on UCF-101 training set.

D. More Results and Qualitative Analysis

In this section, we present the full metrics of our final adaptive tokenizers on reconstruction in Tab. 1, and their reconstruction examples on samples from WebVid and the UCF-101 dataset. We also qualitatively present the effect of using VideoMAE [17] as part of the video discriminator. Besides, for video generation, we present generated samples on the UCF-101 class-to-video and K600 frame prediction task.

D.1. Adaptive Length Reconstruction Examples

We present the reconstruction results of our final video adaptive tokenizer, along with their token assignments decided by the router. In Fig. 1, we present the reconstruction results of the final adaptive tokenizer trained on WebVid-10M [2]. And in Fig. 2 are the reconstruction results on UCF-101 dataset, using another final video adaptive tokenizer trained on UCF-101 and K600 datasets. The patterns of video and assignment pairs given by the router correspond to intuitions. The router typically assigns more tokens to the initial temporal block, which helps the reconstruction of subsequent frames to also benefit from more precise information encoded for the initial block. Content with simple layouts or largely repeats previous frames receives fewer tokens. In contrast, videos that vary intensely are assigned more.

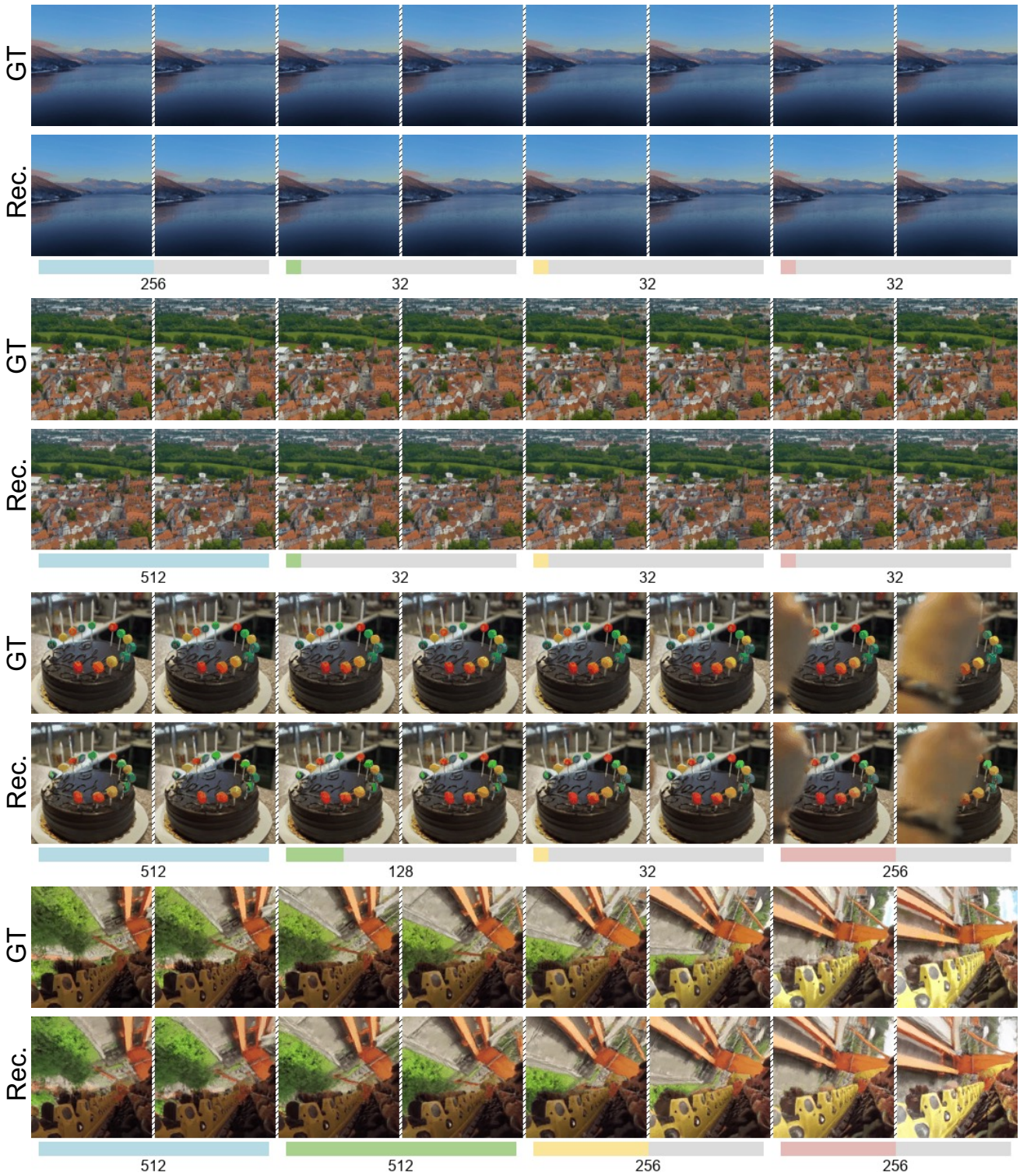

Figure 1. **Adaptive reconstruction results on WebVid.** We downsample 16 frames into 8 frames for visualization, and each two frames represent a 4-frame temporal block. The router typically assigns more tokens to the initial temporal block, allowing the reconstruction of subsequent frames to also benefit from more precise information encoded for the initial block. Content with simple layouts receives fewer tokens (first example *vs*. other examples). If later frames largely repeat previous ones, they are assigned the minimum number of tokens. Video clips that vary constantly and intensely are allocated more tokens.

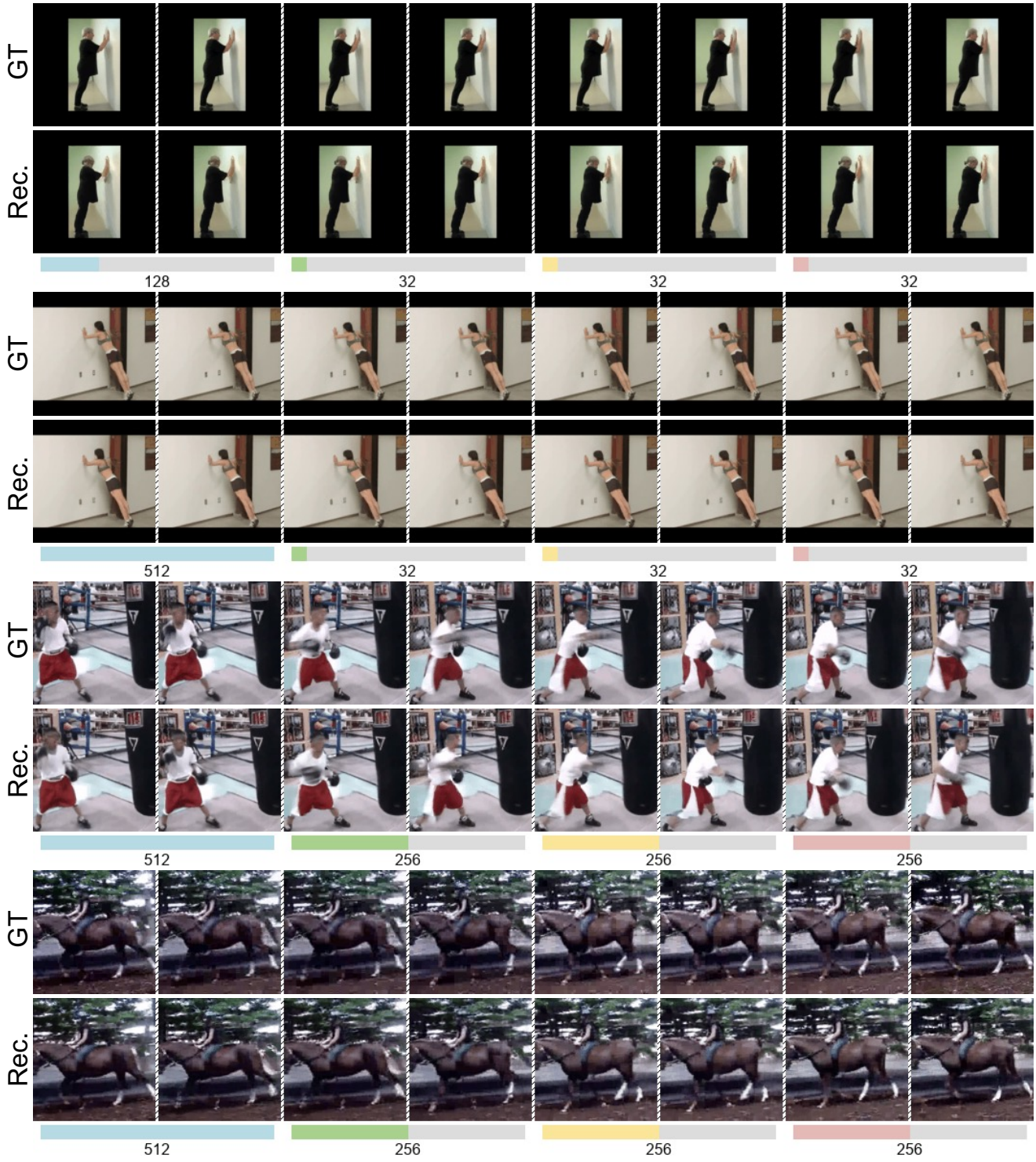

Figure 2. **Adaptive reconstruction results on UCF-101.** We downsample 16 frames into 8 frames for visualization, and each two frames represent a 4-frame temporal block.

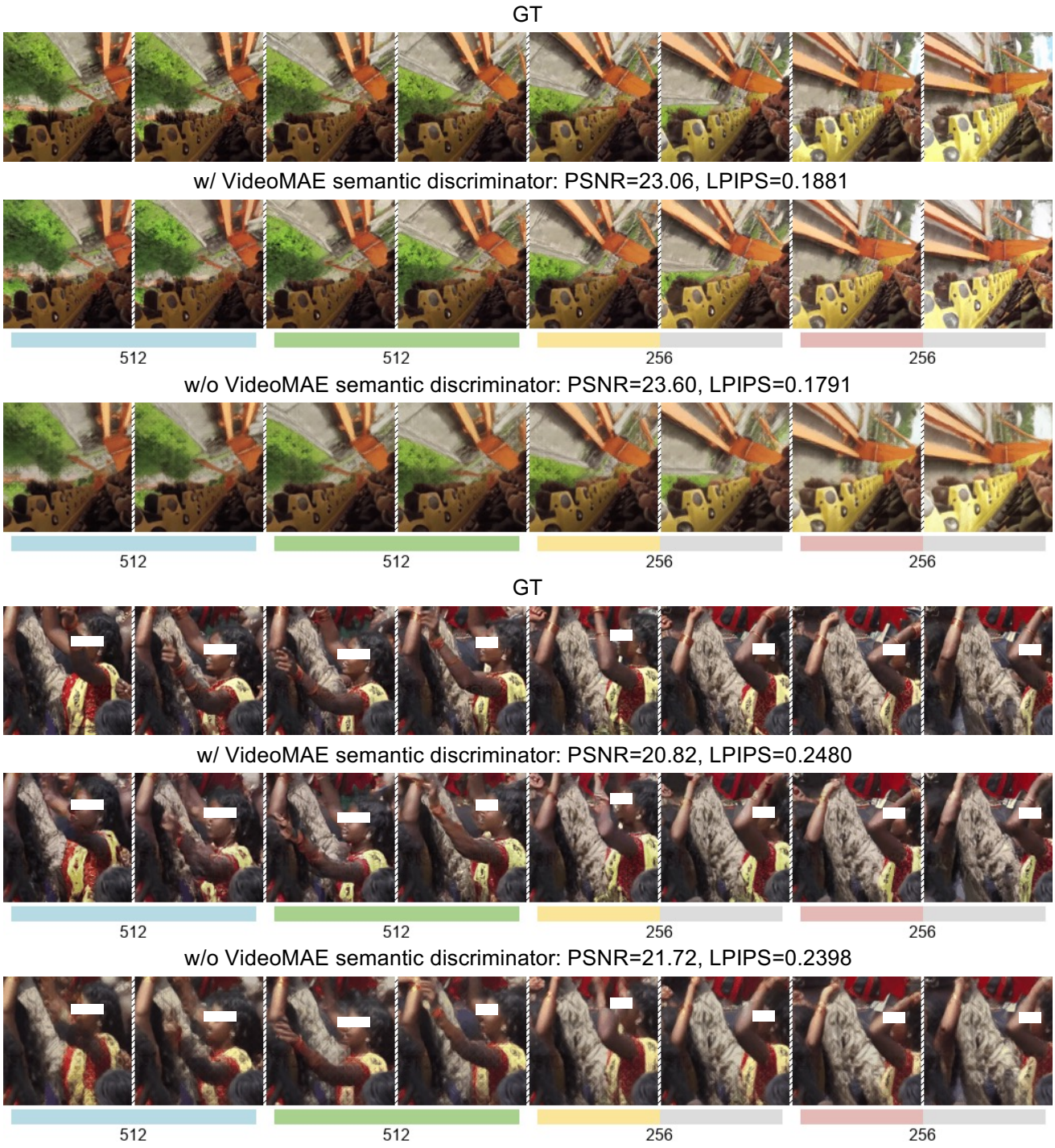

Figure 3. **Qualitative comparison for using VideoMAE discriminator or not.** Using VideoMAE discriminator can degrade the PSNR/LPIPS, but in actual perceptual checking, we find that this degradation is traded for alleviated blurriness and artifact patterns. Zoom in to check the visual details.

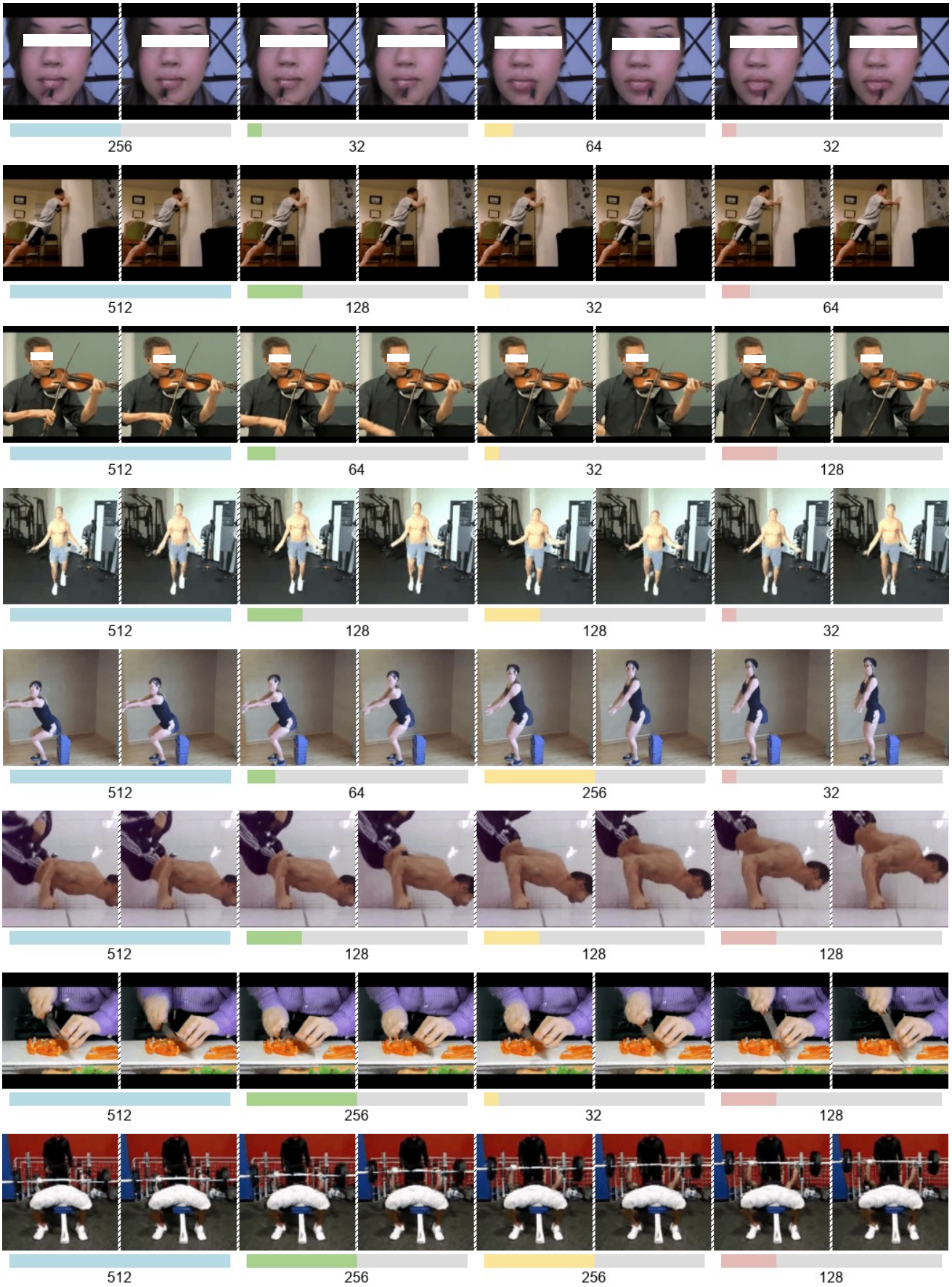

Figure 4. **Adaptive generation results on UCF-101.** We use the 633M GPT model trained on EVATok. We use a constant 3.0 CFG for sampling.

Figure 5. **Adaptive generation results on K600 frame prediction.** We use the 633M GPT model trained on EVATok. We don't use CFG for sampling, following typical approaches. The 1, 3, 5 frames from the 5 condition frames are plotted as the condition part, and the rest 11 frames are downsampled into 6 frames for visualization.

D.2. VideoMAE Discriminator for Visual Quality

In our ablation study in the main paper, the application of the VideoMAE [17, 20] discriminator significantly improves the rFVD and downstream gFVD but leads to degradation in PSNR and LPIPS. In this part, we aim to qualitatively examine the perceptual effect for improved rFVD and degraded PSNR/LPIPS. We compare two video adaptive length tokenizers on WebVid-10M, one is trained with the pretrained VideoMAE discriminator, while another is trained with the PatchGAN discriminator. As shown in Fig. 3, although the reconstructed videos of the tokenizer trained with VideoMAE discriminator show worse PSNR and LPIPS, they are actually more perceptually preferable as they largely alleviate the blurriness or artifact patterns, especially for highly dynamic and challenging examples. Therefore, we conclude that despite the degradation in PNSR/LPIPS, VideoMAE discriminator still largely en-

hances the reconstruction quality perceptually.

D.3. Adaptive Length Video Generation Examples

We present UCF-101 class-to-video generation results in Fig. 4 and K600 frame prediction results in Fig. 5. As in Fig. 4, the AR generation model learns an intuitive way for adaptive length generation. First, the model tends to pay more efforts for the generation of the first temporal block, which lays the foundation for the later generation. For later blocks, content with more variation tends to take more tokens to generate, while small-motion content takes fewer tokens.

E. Computational Overhead Analysis

We present the computation cost for the model training of our four-stage framework and the downstream AR models, as shown in Tab. 2. Compared to the previous fixed-length methods, the extra training cost of our four-stage framework comes from the first three stages. However, the first three stages only take around 27.8% of the total four-stage training cost. This percentage can be further reduced in real-world applications. The size of the proxy tokenizer and its training duration can be decreased for faster training, as we only need the proxy tokenizer to compare assignments, instead of performing well. The data curation can be processed by parallel and independent processes, without any GPU communication bottlenecks. And ultimately, the extra cost in adaptive tokenizer training is a one-time investment, but the savings for downstream deployment will consistently take effect. Therefore, the extra cost of the four-stage training is controllable and worthwhile.

F. Attention Mask for EVATok

In this section, we illustrate the specific details of the attention mask mechanism in our Q-Former style tokenizer, which ensures the temporal causal structure of our 1D token sequences. Each Q-Former layer consists of one self-attention module and one cross-attention module. The queries are first passed through the self-attention module, and then in the cross-attention module, the queries will attend to the reference embeddings. Next, we use an example to show what attention masks look like in the Q-Former encoder and Q-Former decoder. Assume a video clip is patchified into a $4 \times 4 \times 4$ shape tensor, where the first 4 corresponds to the number of temporal blocks. And let the assignment of tokens across the 4 blocks for this video be $(16, 8, 2, 2)$. Then, the attention masks for the Q-Former encoder and Q-Former decoder will be the ones shown in Fig. 6. The query embeddings of each temporal block can only attend to query embeddings or reference embeddings that are no later than this temporal block.

G. Accuracy *vs*. Proxy Reward for Routers

In this part, we examine the accuracy of the router on the validation sets. We find that although the accuracy is relatively slow, the assignments given by the router still obtain decent proxy reward. We use preference weights $w_q = 1.2, w_l = 0.8$ for proxy reward calculation, which are the same as the weights used for the evaluated router training data curation. To evaluate relatively how good the predicted assignments are among all candidate assignments, we use a new metric, proxy reward percentile, defined as:

$$\mathcal{P} = \frac{\mathbb{E}_x(R_{\text{proxy}}(a_{\text{eval}}, x)) - \mathbb{E}_x(R_{\text{proxy}}(a_{\text{worst}}, x))}{\mathbb{E}_x(R_{\text{proxy}}(a_{\text{best}}, x)) - \mathbb{E}_x(R_{\text{proxy}}(a_{\text{worst}}, x))} \times 100\% \tag{4}$$

where a_{eval} is the assignment to be evaluated for video x, a_{best} is the searched max-proxy-reward assignment for x, and a_{worst} is the min-proxy-reward assignment. In practice, a_{eval} can be given by the router according to x or by some other manually designed strategy. $\mathbb{E}_x(R_{\text{proxy}}(a_{\text{eval}}, x))$ is the expectation for the proxy reward based on a_{eval} and x. The range of $\mathcal{P}$ is $[0, 1]$ and the larger $\mathcal{P}$, the better the assignment strategy is. We design a best-uniform baseline, which chooses the max-proxy-reward uniform assignment for x, to compare with the router assignment. As shown in Tab. 3, on WebVid validation set, the top1 accuracy of the router is relatively low, but the proxy reward percentile of the router is high. Moreover, when tested on the unseen UCF-101 dataset, although the top1 accuracy of the router significantly drops, its proxy reward percentile is largely maintained. This phenomenon indicates that the router does not need to be very precise to achieve good performance, implying that the optimal assignment prediction task is not demanding, and some deviation from the best choice won't result in a large performance drop.

H. Image Adaptive Tokenization

Different from videos, images don't have a temporal dimension, so intuitively images are much less redundant than videos. Our experiments on ImageNet [12] 256×256 show that in this setting, the improvement in overall reconstruction quality that can be brought by assigning different token lengths to different images could be limited. However, for downstream generation, adaptive image tokenization can still help produce better generation FID [7] with fewer tokens generated. This result highlights that training generative models on adaptive length sequences is not only efficient but also beneficial to their generation capability.

H.1. Implementation Details

We train image tokenizers on 256×256 ImageNet [12] dataset using a similar CNN + Q-Former hybrid architecture of GigaTok-S-B [21]. The basic training recipe is

Stage / Task	Model Size	Dataset	Bsz	Iters / Epochs	GPUs	Time
Stage 1: Proxy tokenizer training	145M	WebVid (or UCF& K600)	128	400k iters	64×V100	116 h
Stage 2: Data curation	–	WebVid 100k-Subset	–	–	4×64×V100	12.5 h
Stage 3: Router training	20M	WebVid 100k-Subset	128	50k iters	32×V100	5 h
Stage 4: Final adaptive tokenizer training	145M	WebVid (or UCF& K600)	128	1000k iters	64×V100	347 h
AR training: UCF-101 class-to-video	633M	UCF-101	128	3000 epochs	64×V100	88 h
AR training: K600 frame prediction	633M	Kinetics-600	128	75 epochs	64×V100	140 h

Table 2. Summary of compute and time for the four-stage tokenizer pipeline and subsequent AR model training.

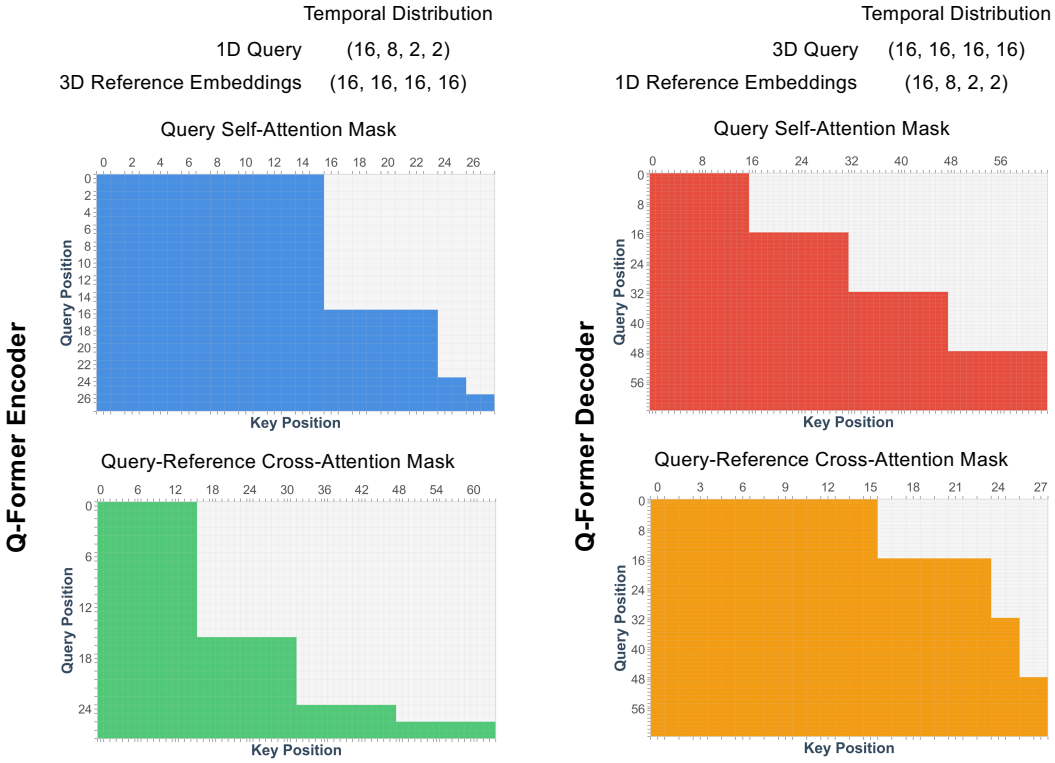

Figure 6. **Example for the attention masks in our Q-Former style adaptive tokenizer.**

Dataset	Method	Val top1/top5 acc.	Proxy Reward Percentile
WebVid	Best-Uniform	-	84.88%
	Router	11.72% / 35.03%	96.96%
UCf-101	Best-Uniform	-	88.46%
	Router	5.77% / 23.68%	96.19%

Table 3. **Accuracy *vs*. proxy reward percentile for the router assignment.** In terms of accuracy. the assignments predicted by the router do not usually hit the top1 or top5 highest proxy reward assignments. However, in terms of proxy reward percentile, the router assignments achieve good results, and generalize to unseen dataset (UCF-101) well.

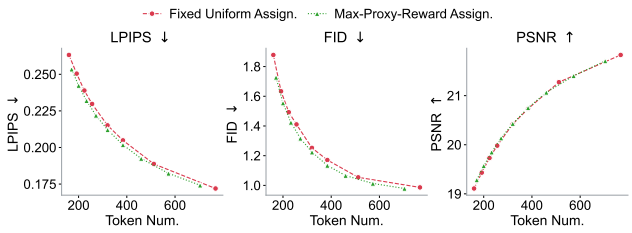

Figure 7. **Image tokenization quality-cost trade-off curve.** On ImageNet 256 × 256 reconstruction, the improvements of max-proxy-reward assignment can be marginal compared to uniform assignment.

also largely aligned to GigaTok except that we utilize DINOv3 [14] to provide semantic alignment. We train all image tokenizers with 256 batch size with only 400k iterations, as we only target to validate the gain of adaptive tokenization on images compared to the fixed-length baseline.

	LPIPS↓	rFID↓	#rTokens↓	gFID↓	#gTokens↓
Uniform (Final)	0.2205	**1.22**	256	4.72	256
Router (Final)	0.2455	1.46	205 (-19.9%)	**4.51**	197 (-23.0%)

Table 4. **Image final tokenizer validation.** For ImageNet 256×256, saving 19.9% tokens by adaptive tokenization inevitably leads to worse rFID, but the performance and efficiency of downstream AR generation models can still benefit from our router. The AR generation models use a constant 1.5 CFG during inference.

Our four-stage framework smoothly translates from videos to image adaptive tokenization, because an image can be equivalent to a one-block video in the tokenization process.

For the proxy tokenizer, we predefine 8 candidate levels of token numbers, $\{512, 384, 256, 192, 128, 96, 64, 32\}$, to be assigned to each image for variable length tokenizer training. For image router training, we train ViT-S [5] size routers on a subset of ImageNet training split of 100k images. They are trained for 25k iterations with a batch size of 256. We use normalized LPIPS as the quality metric in the proxy reward calculation. The reconstruction quality is evaluated on the 50k ImageNet validation set. For downstream AR generation validation, we train Llama-like [15] GPT-B models on each tokenizer for 300 epochs on ImageNet, and evaluate them with generation FID [7] with a constant 1.5 CFG, following the typical approaches [4, 15].

H.2. Results

Quality-cost trade-off curve. We use a similar way as for video proxy tokenizers, to plot the quality-cost trade-off curve under different overall token budgets on an image proxy tokenizer. As shown in Fig. 7, the quality-cost trade-off curve evaluated on image proxy-tokenizers shows that the improvements brought by max-proxy-reward assignment are limited, which is different from the results on videos. This phenomenon corresponds to observations in previous adaptive image tokenization trials [11, 13] on ImageNet, where their adaptive length image tokenizers cannot outperform their fixed-length baselines even with the same overall token budgets.

Final image tokenizer validation. In the final image adaptive tokenizer training, as shown in Tab. 4, we utilize an image router trained with $w_q = 1.3, w_l = 0.7$ to save 19.8% tokens in reconstruction, but it inevitably leads to worse rFID. However, the AR model trained on our adaptive image tokenizer achieves better gFID with 23.0% fewer tokens generated, compared to the fixed-uniform baselines, which assign 256 tokens to all 256×256 images. This indicates that the performance and efficiency of downstream AR image generation can still benefit from image adaptive tokenization using our method.

References

[1] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. 2

[2] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *IEEE International Conference on Computer Vision*, 2021. 2

[3] Yufeng Cui, Honghao Chen, Haoge Deng, Xu Huang, Xinghang Li, Jirong Liu, Yang Liu, Zhuoyan Luo, Jinsheng Wang, Wenxuan Wang, Yueze Wang, Chengyuan Wang, Fan Zhang, Yingli Zhao, Ting Pan, Xianduo Li, Zecheng Hao, Wenxuan Ma, Zhuo Chen, Yulong Ao, Tiejun Huang, Zhongyuan Wang, and Xinlong Wang. Emu3.5: Native multimodal models are world learners, 2025. 1

[4] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 10

[5] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 10

[6] Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro Von Werra, and Martin Jaggi. Scaling laws and compute-optimal training beyond fixed training durations. *arXiv preprint arXiv:2405.18392*, 2024. 2

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 8, 10

[8] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[9] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019. 1

[10] Yan Li, Changyao Tian, Renqiu Xia, Ning Liao, Weiwei Guo, Junchi Yan, Hongsheng Li, Jifeng Dai, Hao Li, and Xue Yang. Learning adaptive and temporally causal video tokenization in a 1d latent space. 2025. 1

[11] Lingjun Mao, Rodolfo Corona, Xin Liang, Wenhao Yan, and Zineng Tang. Images are worth variable length of representations. *arXiv preprint arXiv:2506.03643*, 2025. 10

[12] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 8

[13] Junhong Shen, Kushal Tirumala, Michihiro Yasunaga, Ishan Misra, Luke Zettlemoyer, Lili Yu, and Chunting Zhou. Cat: Content-adaptive image tokenization. 2025. 10

[14] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 9

[15] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. 2024. 10

[16] 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. 1

[17] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022. 2, 7

[18] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 1

[19] Hanyu Wang, Saksham Suri, Yixuan Ren, Hao Chen, and Abhinav Shrivastava. Larp: Tokenizing videos with a learned autoregressive generative prior. In *ICLR*, 2025. 1

[20] Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, et al. Internvideo: General video foundation models via generative and discriminative learning. *arXiv preprint arXiv:2212.03191*, 2022. 7

[21] Tianwei Xiong, Jun Hao Liew, Zilong Huang, Jiashi Feng, and Xihui Liu. Gigatok: Scaling visual tokenizers to 3 billion parameters for autoregressive image generation. 2025. 8