Master's Project

Image Reconstruction and Generation using 1D Tokens

Sejal Mutakekar

Examiner: Prof. Dr.-Ing Thomas Brox
Supervisor: Dr. Sudhanshu Mittal

Albert-Ludwigs-University Freiburg
Faculty of Engineering
Department of Computer Science

August 29th, 2025

Abstract

Recent progress in generative models emphasizes the importance of image tokenization for producing efficient high-resolution images. Using a latent representation of an image for image reconstruction and generation reduces computational complexity compared to processing raw pixels and improves overall generation performance. Traditional approaches, like VQGAN, rely on 2D latent grids with fixed downsampling, which struggle to address the redundancies found in images, where neighboring regions often represent similar information. To tackle this, we use the 'Transformer-based 1-Dimensional Tokenizer (TiTok)' [1], that reconstructs the image using quantized 1-dimensional learnable latent tokens. This offers an efficient and effective representation with a smaller number of latent tokens, enabling a substantial reduction in token count without sacrificing performance. The advantages become even more pronounced with a larger dataset, significantly improving reconstruction and generation quality. Additionally, to enhance the quality of images reconstructed and generated, we leverage transfer learning (initialization) [2], [3], where we use pretrained model weights from Masked Autoencoders (MAE), DINO, DINO-V2, CLIP and Depth-Anything V2 models. We make use of these learned latent tokens from the Vision Transformer (ViT) [4] encoder for further image generation using the Maskgit framework [5]. Our approach of using the 1d-tokenizer for image reconstruction and further using those tokens in our 1d-Maskgit for image generation performs relatively better than the baseline VQGAN tokenizer, with Maskgit setting for image generation. Code available at this https URL.

Contents

List of Figures

List of Tables

1 Introduction

Recent breakthroughs in image generation have been driven by the rise of powerful architectures such as transformers and diffusion models. A key component in many of these systems is the image tokenizer, which compresses high-dimensional pixel data into a compact latent space, facilitating efficient training and generation. Traditionally, tokenizers adopt a 2D structure to preserve spatial alignment with the original image. However, this design choice limits their ability to exploit redundancy across image regions. Inspired by the tasks like classification [4], object detection [6], [7] and segmentation [8], [9] - where high-level information is often distilled into compact 1D representations - we use 'TiTok' [1] to explore whether such a structure is useful for image generation. TiTok is a Transformer-based 1-Dimensional tokenizer that maps images into a discrete 1D latent sequence. It further uses Maskgit framework [5] to analyze model's ability for image generation based on the output latent tokens from the ViT encoder.

1.1 Motivation

TiTok is trained on the ImageNet dataset [10], which contains images that predominantly capture one or two salient objects against relatively simple and uncluttered backgrounds. Its performance for more complex real-world scenes that involve multiple interacting objects and richer contextual relationships, such as those found in the BDD100K dataset [11], is still unknown. Evaluating TiTok on such a dataset enables an assessment of its applicability on more challenging visual domains. In particular, examining its reconstruction performance across varying numbers of 1D latent tokens is essential, as the token configurations that yield strong results on ImageNet may not translate directly to datasets with substantially higher scene complexity. Additionally, it's beneficial to assess whether leveraging richer or more diverse pretraining sources can improve the quality of latent representations for complex scenes, particularly in terms of reconstruction performance across varying numbers of 1D tokens. Further exploring how these encoded learned latent tokens can be utilized for the image generation using Maskgit is important. Such experiments provide insight into both the domain robustness of TiTok and the potential benefits of adapting different pretrained weights for more challenging visual environments.

Through our experiments, we aim to systematically investigate the applicability and robustness of the TiTok tokenizer in more complex, real-world visual settings. Specifically, we focus on the following research objectives:

1. **Evaluating TiTok in complex driving scenarios :** Assess the performance of the

TiTok architecture on BDD100K dataset, which represents a more realistic and challenging driving scenarios compared to ImageNet.

2. **Analyzing token efficiency :** Experiment with the varied number of 1D latent tokens to determine the optimal token length that balances representational compactness with reconstruction and generation quality.

3. **Exploring transfer learning :** Leverage different pretrained model weights to examine whether diverse initializations improve the quality of learned latent representations for complex scenes.

4. **Image generation :** Utilize the encoded 1D token sequences, from TiTok tokenizer, with the Maskgit framework to generate images, and analyze the fidelity and realism of the reconstructions.

1.2 Insights

Based on the experiments described in the previous section (section 1.1), we derive the following insights:

- **Effect of Latent Token Count :** Increasing the number of latent tokens consistently leads to better reconstruction performance, as reflected in Figure 1. This supports the claim that richer token sequences, such as 256 latent tokens, allow the model to capture finer details in complex scenes.

- **Impact of Transfer Learning :** Incorporating transfer learning through pretrained weights significantly enhances the reconstruction quality (Figure 2), particularly when training data is limited. From these results, we observe that the use of TiTok with 64 latent tokens (TiTok-64) offers a strong trade-off between model compactness and reconstruction fidelity.

- **Effects of Initialization Strategies :** Different tokenizer initialization strategies, such as MAE, CLIP, DINO, DINO v2 and Depth-Anything v2, lead to varying behaviors in reconstruction quality and representational properties, as seen in the visual outputs (Figure 12) and t-SNE visualizations (Figure 14). Notably, MAE and CLIP exhibit distinct clustering behavior, which correlates with difference in the downstream reconstruction quality and rFID scores.

- **Role of Tokenizer Initialization in Image Generation :** The different tokenizer initializations significantly impacts the quality of generated images. In particular, tokenizations derived from MAE-initialized encoders yield visually coherent image generations. Increasing the input dataset size for reconstruction, according to subsection 4.1.1, further improves the quality of image reconstructed and generated, as shown in Figure 3.

- **Performance relative to Baseline VQGAN Tokenizer :** The MAE-initialized TiTok-64 tokenizer achieves superior results in image generation tasks, outperforming baseline VQGAN tokenizer on evaluation metrics, including generation-based FID (gFID) scores.

- **Contrast Between Reconstruction and Generation Metrics :** A lower reconstruction FID (rFID) score does not necessarily indicate superior image generation performance (Figure 4). Although the MAE-initialized TiTok-64 yields average reconstruction quality (as measured by rFID) relative to DINO-initialized TiTok-64 tokenizer models, it consistently outperforms the latter in image generation quality, as reflected by its superior generation FID (gFID) scores.

With these findings, our experiments systematically explore both image reconstruction and image generation performance (Figure 5) by varying the number of learnable 1D latent tokens and comparing multiple tokenizer initializations.

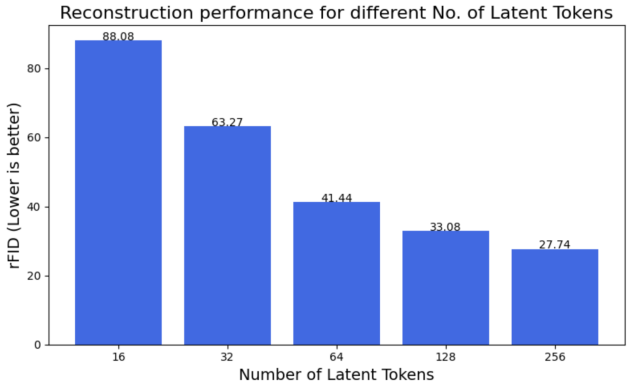


Figure 1: Reconstruction performance for MAE-initialized TiTok across different numbers of input latent tokens after training for 150 epochs. Increasing the number of latent tokens consistently improves reconstruction performance.

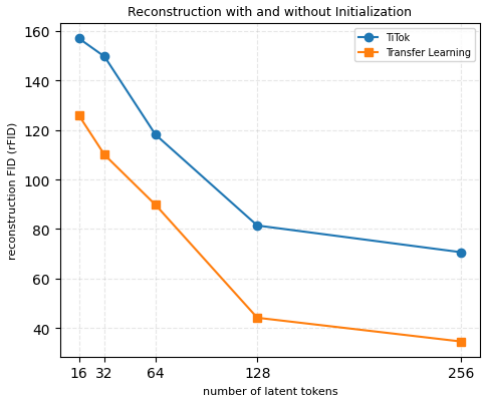


Figure 2: Positive impact of Transfer Learning after training the model for 100 epochs. Transfer learning significantly improves the reconstruction performance dropping the rFID scores in comparison to the performance of TiTok without initialization.

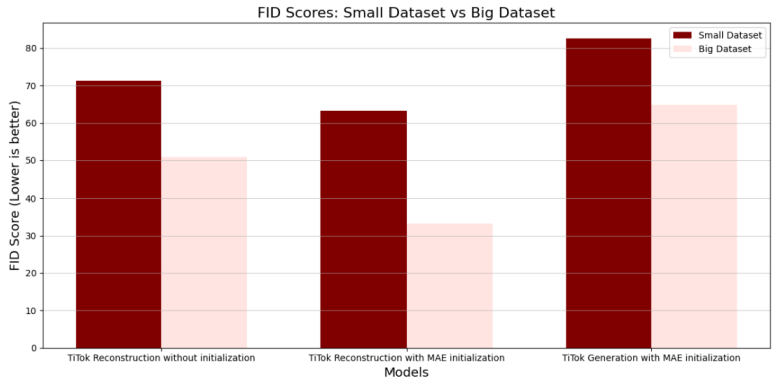


Figure 3: Impact of Training Data Scale and Diversity on FID scores. Referring to dataset configuration in subsection 4.1.1, 'Small Dataset' is the first configuration and 'Big Dataset' is the second configuration.

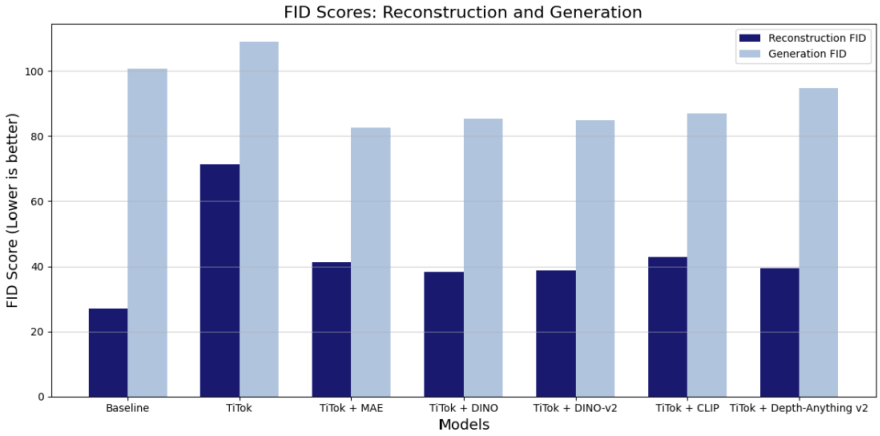


Figure 4: Image Reconstruction and Generation scores. Reconstruction and generation scores for the baseline and TiTok-64 variants, with lower scores indicating better performance. TiTok-64 is a 'Transformer based 1-Dimensional tokenizer trained using 64 tokens', while TiTok + MAE, TiTok + CLIP, etc., denote different initialization strategies for TiTok-64.

Figure 5: Reconstructed and Generated Images. Evaluation of TiTok-64 in complex driving scenarios, comparing its performance with a baseline VQGAN tokenizer for both image reconstruction and generation. In reconstruction, the model reproduces a given input image, whereas in generation, the image is synthesized entirely from scratch, with the input consisting solely of a blank placeholder.

2 Background

2.1 Image Tokenization

Researchers have been using autoencoders [12], [13] to compress images. These models follow an idea: an encoder reduces the image into a smaller, lower-dimensional representation (latents), and a decoder tries to rebuild the original image from latent representations. Over time, this approach has proven effective.

Variational Autoencoders (VAEs) [14] improved on this idea by learning to map the input image to a distribution rather than a single point. Later, VQ-VAEs [15], [16] took it further by representing images using discrete (categorical) tokens instead of continuous values. VQGAN [17] improved the training of these models using adversarial learning (similar to GANs) [18].

Several models like ViT-VQGAN [19] and Efficient-VQGAN [20] explored how transformers can be used in this setup. Other models, such as RQ-VAE [21] and MoVQ [22], looked into using multiple quantization steps, while MAGVIT-v2 [23] and FSQ [24] proposed ways to do quantization without relying on a lookup table.

Most of these methods convert images into a 2D grid of tokens, keeping the spatial structure of patches. In our work, we take a different approach - we represent images as a 1D sequence of tokens like TiTok method, and explore how well this works for image reconstruction and generation on complex driving dataset. TiTok reconstructs both the high-level and low-level details of an image, similar to typical VQ-VAE tokenizers [14], [25], [17].

2.2 Image Generation

Several methods are used for generating images using deep learning, including VAEs [14], GANs [18], Diffusion Models [26], [27], [28], [29], [30], and autoregressive models [31], [32], [15]. Many recent approaches build on top of VQ-VAE, which turns images into discrete tokens using a codebook.

Some methods, like autoregressive transformers [17], [19], [20], [21] (similar to language models), generate images one token at a time - predicting each small part (or patch) step-by-step. This can be slow since they need as many steps as there are tokens (e.g., 256 or 1024 steps). In contrast, non-autoregressive transformers [22], [19], like Maskgit [5], can predict multiple tokens at once, making the generation process much faster.

In our work, we use the Maskgit framework - a common non-autoregressive method - to generate a sequence of tokens that is finally decoded back into an image.

2.3 Initialization Models

We investigate how different pretrained models used for initializing the tokenizer encoder impact downstream image reconstruction and generation. The kind of features each model learns influences how well it performs when transferred to the image tokenization and reconstruction. Below, we briefly describe each of the initialization methods explored in our experiments:

1. **MAE :** Masked Autoencoder (MAE) [33] is a self-supervised learning method for pretraining Vision Transformers (ViTs) on unlabeled image data. The objective is to learn rich visual representation by reconstructing missing parts of an image from a small visible subset. Inspired by masked language modeling (MLM) (14), (15) in Natural Language Processing (NLP), MAE divides an image into fixed nonoverlapping patches and randomly masks a large portion (approx 75%), and feeds only the unmasked visible patches into a ViT encoder. Since, the encoder doesnot process entire image, it functions efficiently. A lightweight decoder then reconstructs the original image using both the encoded visible patches and masked tokens. The model is trained using a reconstruction loss, typically Mean Squared Error (MSE) (13) over the masked patches. This pretraining objective encourages the model to capture global structure and contextual relationships in the image. By focusing on reconstructing unseen regions, the model learns to generate meaningful representations that support image understanding. This setup enables the model to understand the semantic and structural context of images, resulting in efficient and scalable pretraining that transfers well to downstream tasks like detection and classification.

2. **CLIP :** Contrastive Language-Image Pre-training (CLIP) [34] learns a shared embedding space for images and natural language by aligning corresponding image-text pairs through contrastive learning. To maximize the similarity between matching image and text embeddings an image encoder (eg. a Vision Transformer) and a text encoder (eg. a Transformer language model) are trained jointly while attempting to minimize similarity with the mismatched pairs. Using a large dataset of image-caption pairs, CLIP computes cosine similarity with scores across all the combinations in a batch and applies a symmetric contrastive loss to encourage correct alignment. This contrastive objective pushes the model to learn rich, semantically grounded representations that connect visual and textual modalities. By learning to distinguish between matched and mismatched pairs, CLIP develops a strong understanding of both visual content and linguistic context. CLIP can understand the visual concepts in the context of natural language and perform zero-shot tasks like image classification, retrieval and visual reasoning, without any need for task-specific fine-tuning. The ability of CLIP to associate images with natural language makes it highly scalable and generalizable to a wide range of vision-language applications.

3. **DINO :** Self-Distillation with No Labels (DINO) [35] is a self-supervised learning method that trains Vision Transformers (ViTs) to produce high level image representations without using labeled data. It uses a teacher-student setup, where both have the same architecture,

while teacher being updated as an exponential moving average of the student. Multiple augmented views of an image are generated and passed through both the networks, with the student being trained to match the teacher's output distribution using a cross-entropy loss. This training objective encourages the model to learn consistent and semantically meaningful features across diverse image augmentations. By aligning the representations of different views, DINO captures high-level patterns and object-centric features without supervision. Using DINO semantic, invariant features are learned purely from the data structure, enabling strong performance on tasks like classification, retrieval and segmentation- without labeled data. The concept also highlights the synergy between DINO and ViT.

4. **DINO v2 :** The objective of DINO-v2 [36] is to train self-supervised ViTs that learns rich, general-purpose visual representations without using any labeled data. It builds on the top of DINO by improving training scale, stability, feature quality and versatility across tasks. DINO-v2 uses student-teacher architecture where both the networks process multiple augmented views of the same image, with the student model trained to match the teacher's output distribution. The teacher model is updated by the momentum-based moving average of the student, and the loss function encourages the model to produce consistent, semantically meaningful features across different views. This pretraining objective helps the model develop view-invariant, high-level features that generalize well across tasks. By aligning representations across augmentations and scaling training effectively, DINOv2 learns robust visual semantics without supervision. DINOv2 builds upon DINO by using better training strategies and diverse data, significantly scales up model size, resulting in a single vision encoder that performs strongly across a wide range of tasks including classification, segmentation, retrieval and does not require task-specific fine-tuning or labels.

5. **Depth-Anything v2 :** Depth-Anything v2 [37] is a selfsupervised method for monocular depth estimation that produces depth-maps from RGB images without relying on a labeled dataset. It uses a student-teacher architecture, where a teacher model is trained on high quality synthetic images with accurate depth and then further used to generate depth labels for real-world images, used to train student models. This approach leverages the precision of synthetic data and the diversity of real imagery to create scalable models that generalize well across diverse scenes. This training setup enables the model to learn accurate geometric understanding by bridging synthetic supervision with real-world visual complexity. By aligning predictions from real RGB images with pseudo-labels generated by the teacher, the model learns robust and transferable depth representations. Depth-Anything v2 achieves state-of-the-art performance on benchmarks like NYUv2 (16) and KITTI (24), offering fine-grained, sharp depth maps and faster inference than diffusion-based alternatives. It's designed to be a drop-in depth encoder for realworld computer vision applications like AR/VR, 3D reconstruction, robotics, and more—without needing expensive ground-truth depth data.

3 Method

3.1 VQGAN Architecture

The Vector Quantized Generative Adversarial Network (VQGAN) [38] builds upon the VQ-VAE framework by integrating adversarial training to improve the perceptual quality of reconstructions. We use VQGAN model (Figure 6) as a baseline to compare against our proposed 1D tokenizer. The scores for this baseline model are reported in Table 1.

VQGAN consists of three main components : an Encoder(Enc), a Vector Quantizer($Quant$) and a Decoder(Dec), similar to VQ-VAE, along with the Discriminator(D) for adversarial learning. An input image $I \in \mathbb{R}^{H \times W \times 3}$ is processed through an Encoder, typically built with a convolutional neural network (CNN), which compresses the image into a lower dimensional latent representation:

$$Z_{2D} = Enc(I),\ Z_{2D} \in \mathbb{R}^{\frac{H}{f} \times \frac{W}{f} \times D}$$

,where f is the spatial downsampling factor and D is the latent embedding dimension.

Each spatial embedding in Z_{2D} is quantized via a vector quantizer, which replaces each vector with the closest code from a learned codebook $C = c1, c2, ...cN$, producing a discrete latent map:

$$\widehat{Z}_{2D} = Quant(Z_{2D})$$

The quantized latent representations $\widehat{Z}_{2D}$ are then passed to the decoder, which reconstructs the high resolution image:

$$\widehat{I} = Dec(\widehat{Z}_{2D})$$

To improve the sharpness of the reconstructions, VQGAN incorporates a discriminator. The generator (Encoder + Quantizer + Decoder) and the discriminator are trained adversarially. By introducing the discriminator, VQGAN encourages the decoder to produce visually plausible and perceptually sharper and more detailed reconstructions - making it especially useful for generative tasks.

In addition to standard VQGAN losses - reconstruction loss, quantization loss, perceptual loss, and adversarial loss as in [39] - we adopt the entropy regularization loss from [40] to promote

both diverse token usage within samples and balanced code utilization across the dataset, given by:

$$L_{\text{entropy}} = \mathbb{E}_r \left[H(p(q(z))) \right] - H \left(\mathbb{E}_r [p(q(z))] \right),$$

,where $H(\cdot)$ denotes the entropy function, $p(q(z))$ is the token distribution for a given input, and $\mathbb{E}_r[\cdot]$ denotes the expectation over the dataset.
This combination encourages faithful reconstruction, expressive discrete representations, and perceptually sharp outputs while maintaining stable training.

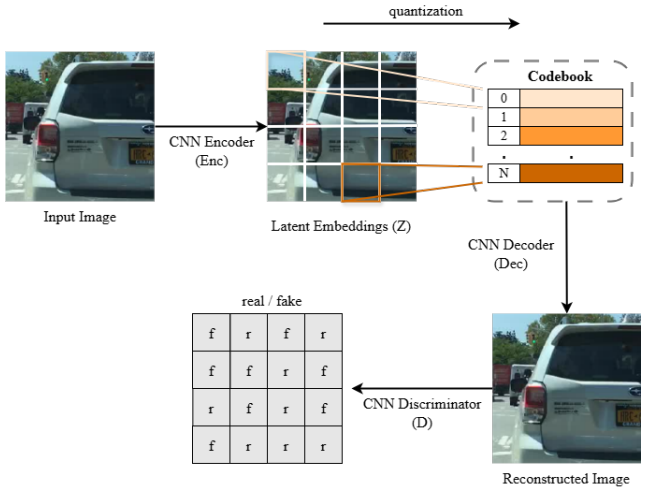


Figure 6: VQGAN Architecture. The diagram illustrates the VQGAN architecture, where an input image is encoded into latent embeddings, quantized into discrete codebook indices, and decoded back into a reconstructed image, with a discriminator evaluating reconstruction realism.

3.2 TiTok Architecture

3.2.1 Image Reconstruction using TiTok

We leverage TiTok (Transformer based 1-Dimensional Tokenizer) that compresses high-resolution images into a compact 1D sequence of latent tokens and reconstructs the original images from these tokens (Figure 7). Both the Encoder *Enc* and Decoder *Dec* in TiTok are implemented as Vision Transformers (ViTs), enabling a unified transformer-based design for tokenization and de-tokenization.

An image $I \in \mathbb{R}^{H \times W \times 3}$ is divided into non-overlapping patches of size fxf producing D dimensional patch embeddings :

$$P \in \mathbb{R}^{\frac{H}{f} \times \frac{W}{f} \times D}$$

A set of K learnable latent tokens $L \in \mathbb{R}^{K \times D}$ is concatenated with the patch embeddings :

$$P \oplus L$$

The combined sequence of patch tokens and latent tokens is processed by the ViT Encoder *Enc*. From the encoder's output, only the latent tokens are retained as the compact image representation, resulting in a 1D latent sequence of length K.

$$Z_{1D} = Enc(P \oplus L)$$

The latent sequence Z_{1D} is vector quantized via Quantizer ($Quant$) to map each latent token to the nearest code in a learned codebook.

A set of mask tokens $M \in \mathbb{R}^{\frac{H}{f} \times \frac{W}{f} \times D}$ is created by replicating a single learned masked token $\frac{H}{f} \times \frac{W}{f}$ times. The quantized latent tokens are concatenated with these mask tokens and passed through a ViT decoder:

$$\widehat{I} = Dec(Quant(Z_{1D}) \oplus M)$$

Unlike simply flattening a 2D grid of latents into a 1D sequence - which still implicitly preserves the original 2D spatial mapping -TiTok produces a 1D latent representation whose length is independent of the image resolution.

Similar to VQGAN, TiTok is trained using a combination of reconstruction loss, quantization loss, perceptual loss, entropy regularization loss for improved token utilization, and adversarial loss via a discriminator, which together produces semantically meaningful reconstructions.

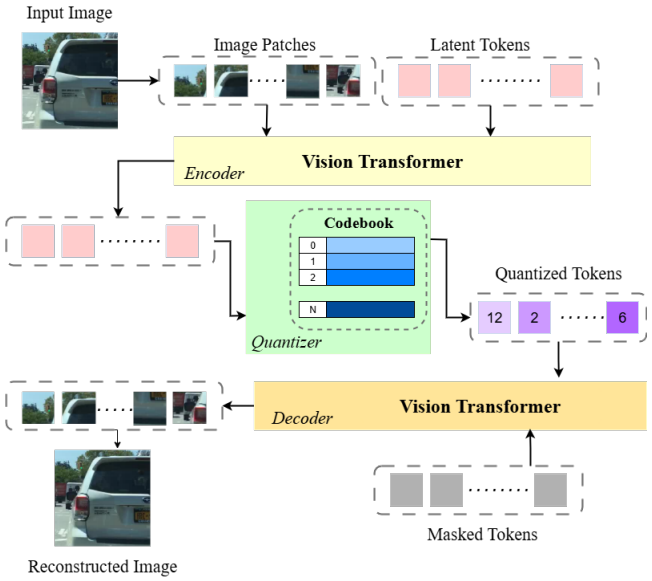


Figure 7: TiTok Tokenizer Architecture. The diagram shows the TiTok tokenizer architecture, where a Vision Transformer encoder converts image patches into embeddings, which are quantized via a learned codebook into discrete tokens, and then decoded by a Vision Transformer decoder to reconstruct the image.

3.2.2 Image Generation using TiTok

We evaluate TiTok's effectiveness for image generation. We adopt Maskgit [5] as the generation framework due to its simplicity and efficiency, with our TiTok tokenizer.

In this setup, the image is first tokenized into a 1D sequence of discrete latent tokens. During training, a random proportion of these tokens is replaced with the special masked tokens and passed to a transformer. A bidirectional transformer then processes this masked sequence, predicting the discrete token IDs for the masked positions as shown in Figure 8(a) and Figure 8(c).

During inference, generation proceeds in multiple steps: at each step, the model predicts the masked tokens, which are then sampled based on prediction confidence and used to update the sequence. This iterative process progressively fills in the masked sequence until all tokens are generated. Finally, the sequence of generated tokens is detokenized back into the pixel space using TiTok's decoder as depicted in Figure 8(b).
Maskgit (a non-autoregressive model) enables faster generation compared to autoregressive models, while maintaining high-quality outputs.

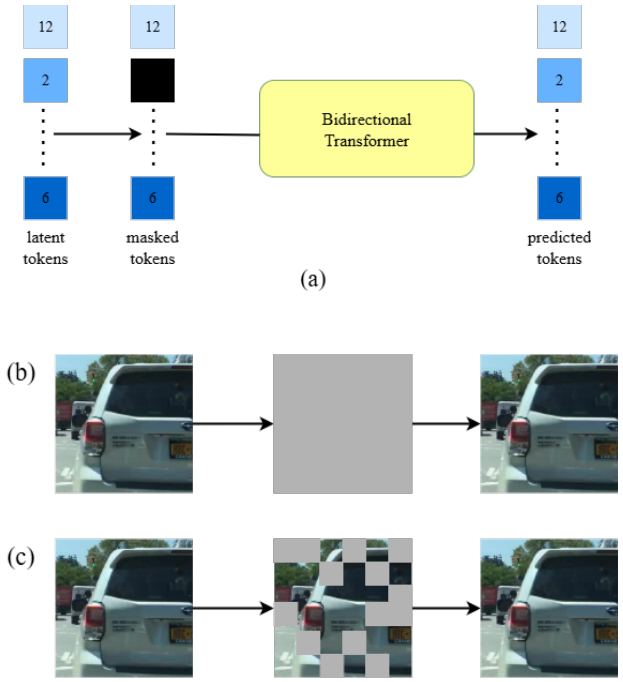


Figure 8: Maskgit Architecture 1(a) Image Generation in Maskgit with the latent tokens generated by a 1D tokenizer (TiTok). Note: The baseline VQGAN model uses the same Maskgit architecture, while the tokens used for generation are from VQGAN tokenizer, (b) Image generation using all the masked tokens, (c) Image reconstruction, by progressively filling in the masked tokens.

4 Experimental Setup

4.1 Dataset Description

We employ the Berkeley DeepDrive 100K (BDD100K) dataset [11] for both training and evaluation. The dataset consists of 100,000 high-resolution video clips captured across various cities in United States under different lighting, season and road conditions. This dataset covers a broad spectrum of real-world driving scenarios - including urban streets, residential areas and highways, with complex scenes including multiple interacting objects within a single frame.

4.1.1 Image Reconstruction Dataset

For the image reconstruction experiments, we extract still frames from the driving videos in the BDD100K dataset. Two dataset configurations are defined in order to evaluate the model's performance under varying data availability conditions.

1. In the first configuration, a subset containing only daytime images is utilized, yielding a training set of 12,454 images and a validation set of 1,764 images.

2. In the second configuration, we employ an expanded variant of the dataset that encompasses a wider range of conditions, including both daytime and nighttime scenes across multiple seasons. This extended dataset consists of 35,000 training images, while retaining the same 1,764 validation images.

Maintaining these two configurations enables a comparative assessment of the proposed model's reconstruction performance when trained on limited versus more diverse and extensive datasets. The corresponding experimental results are presented in Table 2 and Table 3, with the extended variant referred to as the Big Dataset.

4.1.2 Image Generation Dataset

For Image Generation, we utilized the same BDD100K dataset, with the still frames from driving videos including the images from varied lighting, season and road conditions. The training subset comprises 158,903 images, while the validation set contains 39,818 images. This configuration ensures that the generative model is evaluated on a broad spectrum of real-world driving scenarios. The corresponding experimental results are presented in Table 5.

4.2 Evaluation Metrics

The quality of reconstructed image and the effectiveness of the tokenizer is assessed using Fréchet Inception Distance (FID) metric. This metric evaluates the similarity of the original images with reconstructed and generated images by comparing their feature distributions as extracted by a pretrained Inception network.
The FID is computed using the Fréchet distance between the Gaussian-distributed features of the original images and those of the reconstructed images and generated images, according to the formula:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{1/2}\right)$$

where:

- μ_r, Σ_r denote the mean and covariance of features from the original (real) images,
- μ_g, Σ_g denote the mean and covariance of features from the reconstructed images,
- Features are typically extracted from the penultimate layer of a pretrained Inception-V3 model.

This formulation allows us to measure how well the latent codes retain semantically relevant information, offering a reliable indication of reconstruction quality. We use terms 'rFID' for reconstruction Fréchet distance and 'gFID' for generation Fréchet distance. (*Note : Even if the terms rFID and gFID differ, the base formula and evaluation goal remains the same - assessing the output reconstructions and generations.*)

Additionally, a comprehensive set of evaluation metrics is employed to assess model behavior throughout the training and validation phases. These metrics encompass entropy loss, quantization loss, reconstruction loss, perceptual loss, accuracy, and other relevant measures, each providing complementary insights into the model's learning dynamics and the quality of its learned representations. Furthermore, codebook usage is monitored to evaluate the efficiency of token utilization within the latent space, while t-SNE visualizations are utilized to examine reconstruction patterns across training iterations. Alongside these, additional loss components and qualitative visual analyses are incorporated to enable a deeper understanding of the model's convergence characteristics. All metrics and visualizations are systematically logged and tracked using TensorBoard, facilitating real-time analysis and ensuring thorough evaluation of the model's convergence and output quality.

4.3 Training Setup

We conducted model training using PyTorch Lightning with a multi-GPU setup, employing two NVIDIA GPUs and a batch size of 4 per device, resulting in an effective batch size of 128. Optimization was performed using Adam and AdamW optimizers with a fixed learning rate of

1e-4. The learning rate schedule incorporated a linear warm-up phase over the initial 5000 steps, during which the learning rate increased from 0 to 1e-4, and remained constant thereafter. All experiments were conducted with the same random seed and a large model configuration featuring a model width of 768. The latent token size was set to 16, and the quantizer employed a codebook size of 4096 with entropy-based regularization.

Depending on the hardware configuration, training for image reconstruction required approximately 48 hours to complete 150 epochs, while the image generation training spanned roughly 55 hours over 30 epochs.

5 Results

5.1 Baseline VQGAN Results

To establish a strong baseline, we employed the VQGAN framework for image reconstruction. The tokenizer was trained by minimizing the reconstruction loss, thereby ensuring that the latent codes effectively preserve critical visual features.
Following training, the tokenizer was frozen and utilized as the foundational component of our image generation pipeline. Specifically, we used the pretrained tokenizer checkpoints within the Maskgit framework, which generates images through iterative masked token prediction in the latent space. The FID scores corresponding to this baseline method are reported in Table 1.

Method	FID Scores
VQGAN (Image Reconstruction)	27.09
MaskGIT (Image Generation)	100.70

Table 1: Baseline scores. The Baseline score for Image Reconstruction and Image Generation using VQGAN tokenizer and Maskgit framework, respectively.

5.2 TiTok Image Reconstruction Results

A primary goal of this study is to attain reconstruction quality comparable to that of traditional 2D tokenizers such as VQGAN, while markedly reducing the number of latent tokens required. As presented in Table 2, TiTok exhibits the ability to recover visually coherent images even from highly compressed latent representations. Notably, reconstructions utilizing only 64 tokens successfully preserved structural and semantic information, although the overall quality remains somewhat inferior to the VQGAN baseline. Nonetheless, the outputs are visually interpretable and retain essential scene details, thereby supporting the hypothesis that a compact and well-optimized one-dimensional latent space can achieve an advantageous balance between reconstruction fidelity and computational efficiency [1]. Moreover, increasing the number of latent tokens at the encoding stage, followed by decoding, consistently enhances perceptual quality and reduces the performance gap relative to the baseline. Finally, the size of the training dataset is shown to have a significant impact on model performance.

Latent tokens	↓ rFID Score	CB Usage
16	156.98	42.35%
32	149.67	52.03%
64	111.89	58.49%
128	81.41	75.95%
Big Dataset	60.93	28.19%

Table 2: TiTok Results for varied number of learnable latent tokens. 'BIG DATASET' refers to the second configuration in the dataset description (subsection 4.1.1), using 64 number of latent tokens. The model is trained without initialization for 100 epochs. Lower rFID scores imply better reconstruction. The codebook usage decreases as the number of epochs increases.

5.3 TiTok with Initialization Results

5.3.1 MAE Initialization

We performed a comprehensive investigation of transfer learning by initializing the TiTok encoder with weights from the pretrained Masked Autoencoder (MAE) model [33]. Incorporating MAE initialization significantly improved reconstruction quality compared to the TiTok model without initialization, as shown by lower rFID scores. The better pronounced gains were observed in configuration utilizing 64 latent tokens, while access to more diverse training data further enhanced reconstruction performance.

Latent tokens	↓ rFID Score	CB Usage
16	88.08	30.66%
32	63.27	36.47%
64	41.44	40.40%
128	33.08	43.45%
256	27.74	54.86%
Big Dataset	33.19	34.62%

Table 3: MAE initialized TiTok Results. 'BIG DATASET' refers to the second configuration in the dataset description (subsection 4.1.1), using 64 number of latent tokens. The model is trained for 150 Epochs. Lower rFID scores imply better reconstruction and the codebook usage decreases as the number of epochs increases.

Discussion : Impact of Training Data Scale and Diversity

Results from Table 2 and Table 3 for 'Big Dataset' show that TiTok-64's performance significantly improves with larger, more diverse datasets. Larger training sets correlate with better reconstruction and generation quality, reflected in lower rFID scores. Qualitative analysis (Figure 9) confirms that models trained on bigger datasets preserve finer details and coherent scene structures using only 64 tokens, yielding sharper boundaries and better semantic alignment. This is attributed to increased exposure to varied patterns during training, which enhances generalization and reduces

overfitting, underscoring the importance of dataset scale for TiTok's performance.

(a) Image Reconstruction for MAE initialized TiTok

(b) Image Generation for MAE initialized TiTok

Figure 9: Image Reconstruction and Generation for MAE initialized TiTok-64 on Small Dataset and Big Dataset. Referring to the dataset description in subsection 4.1.1, 'Small Dataset' refers to the first configuration, while 'Big Dataset' refers to the second configuration.

5.3.2 Use of varied Initializations

After observing significant improvements in reconstruction and generation quality using the MAE-initialized TiTok, we extended our study by experimenting with other strong pretrained vision models. Specifically, we tested the impact of initializing the encoder with weights from CLIP [34], DINO [35], DINO-v2 [36], and Depth Anything v2 [37]. To ensure a fair comparison, we fixed the number of latent tokens to 64 across all experiments. This setup allows us to isolate the effect of encoder initialization while maintaining a balanced token number for representation.

Model	↓ rFID Score	CB Usage
TiTok-64	71.22	49.80%
DINO	38.42	36.93%
Depth-Anything v2	39.60	36.32%
DINO-v2	38.94	38.13%
CLIP	42.90	29.63%
MAE	41.44	40.40%

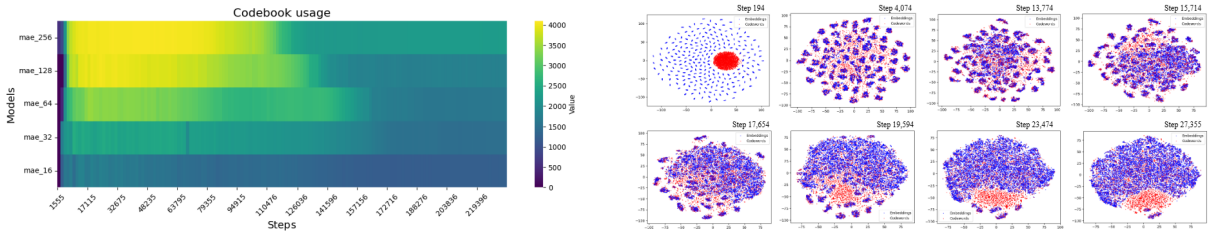
Table 4: Initialization Results for TiTok-64 using different pretrained weights. The model is trained for 150 epochs. TiTok-64 denotes TiTok without initialization. Lower rFID scores imply better reconstruction and the codebook usage decreases as the number of epochs increases.

Utilizing various initialization methods further enhanced reconstruction scores, with DINO achieving the highest performance among the initialization models, albeit with a marginal difference. Moreover, there is a substantial performance gap between models initialized with pretrained

weights and those without initialization.

Discussion :

1. **Codebook utilization and performance dynamics :** Across experiments (Table 2, Table 3, Table 4), codebook usage follows a trajectory of initial narrow activation, mid-training expansion, and late-stage contraction, while reconstruction performance improves. As shown in Figure 10, early training ($\sim 4K$ steps) shows low entropy and tight embedding clusters, indicating limited codeword use. Around $\sim 13K$ steps, codebook usage and entropy peak, reflecting an exploratory phase with broad sampling of the embedding space, as seen in dispersed t-SNE clusters. Later (23K–27K steps), both entropy and usage decrease, with embeddings consolidating into compact clusters, showing selective use of a smaller, more informative subset. This progression, supported by entropy measures, t-SNE visualization, and quantitative usage, highlights the model's shift from broad exploration to efficient, semantically rich encoding without loss of reconstruction fidelity.



(a) Codebook Usage across Training Steps **(b)** Evolution of Embeddings and Codebook vectors

Figure 10: Codebook Usage for Image Reconstruction using MAE initialized TiTok models. (a) Heatmap depicting the usage frequency of codebook embeddings across training steps for various MAE initialized Titok models. Color intensity corresponds to frequency, with brighter regions indicating higher usage. (b) Scatter plots illustrating the evolution of embeddings and codebook vectors in 2D t-SNE space at various training steps. Changes in distribution and clustering reflect model learning dynamics and embedding space organization.

2. **Effect of Token Count on TiTok's Output Quality :** Increasing latent token count enhances reconstruction quality by capturing finer details and making the latent space more expressive, thereby reducing reconstruction error. While 256 tokens achieve performance comparable to the baseline VQGAN (Table 1), fewer tokens lead to higher FID scores and reconstruction losses (Figure 11). However, higher token counts raise computational and memory costs, extending training and inference times - for example, according to our experiments, TiTok-64 trains in 40 hours versus 48 hours for TiTok-128. Thus, selecting an optimal token count is essential for balancing quality and efficiency. In our setting, 64 tokens offer satisfactory performance, with potential for further gains on larger, more diverse datasets.

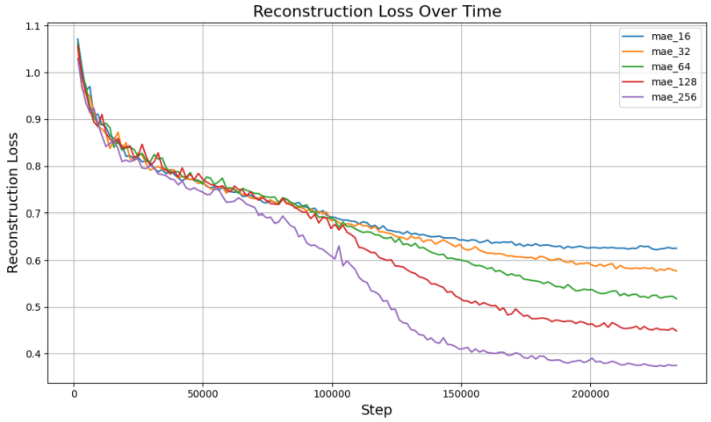


Figure 11: Reconstruction Losses for MAE initialized TiTok. The reconstruction loss decreases as the number of latent tokens used for reconstruction increases, indicating improved reconstruction quality with a higher token count.

5.4 TiTok Image Generation Results

In the image generation experiments, we evaluated TiTok's ability to both reconstruct and synthesize images using the Maskgit framework. The evaluation covered two settings : (i) reconstruction from randomly masked latent tokens and (ii) image synthesis from scratch, where no input image is provided and the model generates novel samples from random latent codes.

Model	↓ gFID Score
TiTok-64	109.07
Depth-Anything v2	94.73
CLIP	87.01
DINO	85.42
DINO-v2	84.92
MAE	82.57

Table 5: Image Generation results using latent tokens from TiTok-64 tokenizer (uninitialized and initialized tokenizers) trained over 30 epochs. TiTok-64 denotes TiTok trained using 64 tokens, without initialization.

gFID scores in Table 5 show that the MAE-initialized TiTok-64 tokenizer consistently outperforms all other variants in image generation quality, with improved preservation of semantic structure and scene-level details, as reflected in enhanced visual fidelity (Figure 5) and objective metrics (Table 4). The MAE-based pipeline also surpasses the baseline configuration using the VQGAN tokenizer with the same Maskgit framework. Notably, its outputs display greater scene coherence and more precise structural alignment than both initialized and uninitialized TiTok-64 models.

Discussion : Effect of initialization Type on Output Quality

Our experiments demonstrate that transfer learning markedly improves TiTok reconstruction (Table 3, Table 4) and generation (Table 5) compared to training TiTok from scratch (Table 2). Pretrained models (MAE, CLIP, DINO variants, Depth-Anything v2) originally trained on large datasets like ImageNet were chosen for their compatibility and transferable features, yielding consistent gains in metrics (e.g., lower rFIDs in Table 4, lower gFIDs in Table 5) and qualitative results (Figure 12). These pretrained weights provide a strong prior on general image features, enabling better generalization, and semantically richer outputs. In contrast, random initialization requires learning all features from scratch, causing poorer results. Overall, pretrained components let the model focus on task-specific learning, enhancing efficiency and output quality.

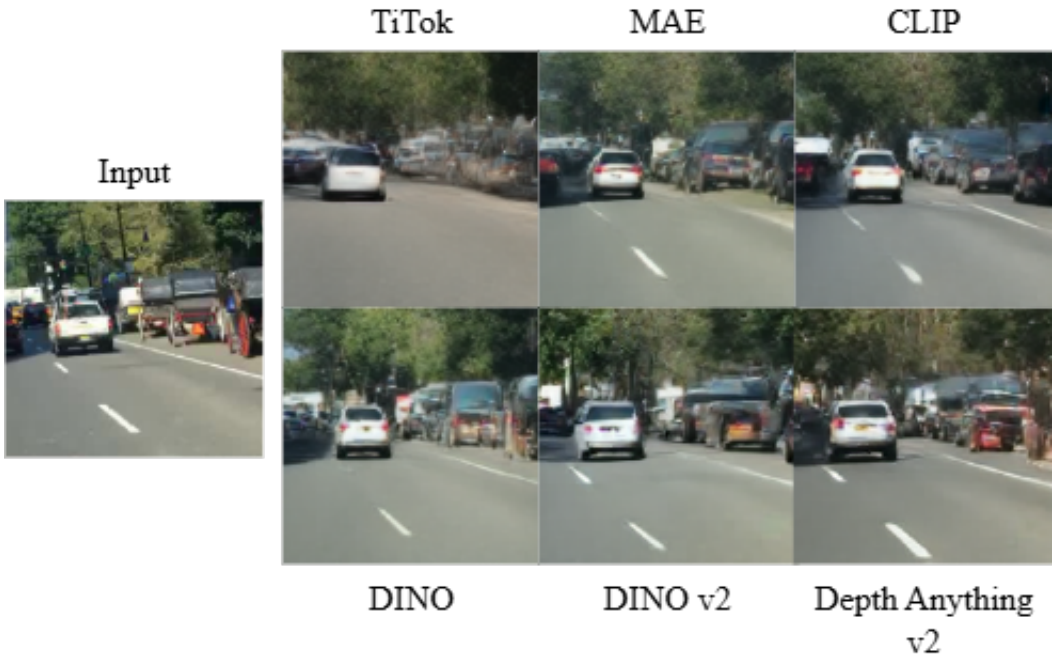


Figure 12: Image Reconstructions for uninitialized TiTok (referred as 'TiTok') and different TiTok initialized models. These images are reconstructed using 64 latent tokens.

6 Additional Analysis

6.1 Effect of initialization Type on Output Quality

Our initialization strategy encompasses diverse models, including MAE, CLIP, DINO, DINO v2, and Depth-Anything v2. Each initialization type exhibits distinct performance levels, resulting in variations in both TiTok image reconstruction and generation quality.

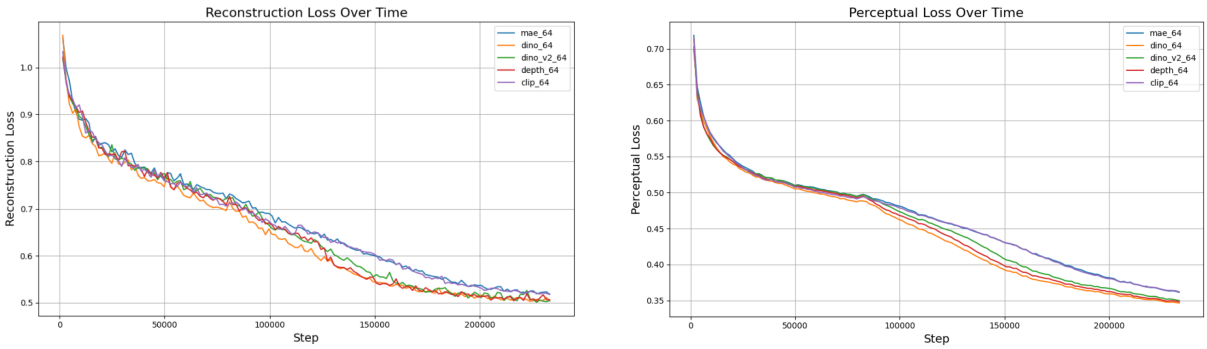
6.1.1 Image Reconstruction

For the image reconstruction task with 64 latent tokens, DINO initialized TiTok performs better than all other models, with a marginal difference compared to DINO-v2 and Depth-Anything v2, and noticeably better than MAE and CLIP, as reflected in the rFID scores (Table 4).

Key Observation : Similar Behavior of MAE and CLIP
MAE and CLIP exhibit similar patterns in t-SNE plots (Figure 14), reconstruction and perceptual losses (Figure 13), and nearly identical rFID scores (Table 4), resulting in visually similar reconstructed outputs. In contrast, DINO, DINO-v2, and Depth-Anything v2 achieve relatively better reconstruction performance.
The exact reasons for the similarity between MAE and CLIP behavior remains unknown; however, for CLIP, this is possibly because it is optimized for image-text alignment, producing primarily global embeddings that lack detailed spatial and local object information. Exploring this further remains an open research question and a limitation of the current study.

6.1.2 Image Generation

The gFID scores reported in Table 5 demonstrate that TiTok models initialized with MAE generate visually more meaningful images compared to those initialized with other pretrained models, including DINO-v2. Notably, MAE - trained on the relatively smaller ImageNet dataset ($\sim$ 1.28 million images) - outperforms DINO-v2, which was trained on the substantially larger and more diverse LDV-142M dataset ($\sim$ 142 million images). Despite the broader training data available to DINO-v2, it does not achieve generative quality comparable to that of MAE. Consequently, it appears that the MAE-initialized TiTok-64 produces more effective latent tokens that facilitate its generative capabilities, whereas DINO-v2 excels primarily in detailed image reconstruction. The exact reasons behind this observation are not yet fully understood and may be explored in future research.



(a) Reconstruction loss

(b) Perceptual loss

Figure 13: Losses across different initializations for TiTok-64 tokenizer. (a) Reconstruction loss for TiTok-64 with all the initialization models. (b) Perceptual loss for TiTok-64 with all the initialization models. MAE and CLIP show nearly the same losses.

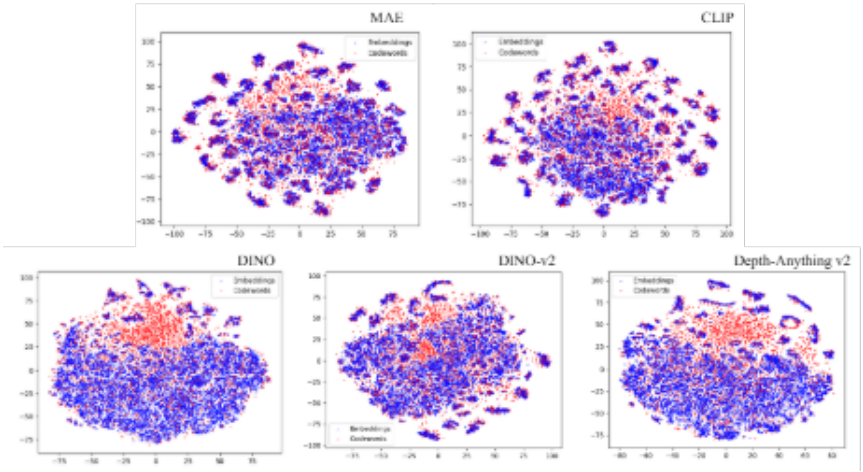


Figure 14: t-SNEs for all the TiTok-64 initialized models at 15,714th step. MAE and CLIP initialized TiTok-64 exhibit similar clustering patterns across training steps.

6.2 Enhanced Image Generation Using 64 Latent Tokens over VQGAN Baseline

The images generated using TiTok-64 from the MAE-initialized TiTok tokenizer outperform the baseline VQGAN model in terms of gFID scores. While the baseline VQGAN tokenizer model exhibits superior performance in image reconstruction - using 256 tokens and focusing on spatial fidelity and low-level texture preservation - it performs worse in image generation compared to the initialized TiTok-64 tokenizer. The initialized TiTok-64 tokenizer, although less precise in reconstructing fine image details, produces semantically rich latent representations that better preserve contextual relationships within the image. Consequently, generation based on the initialized TiTok-64 tokenizer yields more plausible and visually appealing images (Figure 15),

surpassing the baseline VQGAN tokenizer in generative quality despite its comparatively lower reconstruction accuracy.

Figure 15: Image Generation, where the initialized TiTok-64 model performs better than the baseline VQGAN model (a) Generated Image using MAE initialized TiTok-64 model, (b) Generated Image using VQGAN model tokens (256 tokens).

6.3 Generalization Challenges of TiTok with Complex Data

TiTok [1] originally reports effective image reconstruction and generation using 32 latent tokens and is trained on the ImageNet dataset, which contains approximately 1.28 million images. This dataset generally features one to two objects against relatively simple backgrounds, facilitating reconstruction tasks. In contrast, our model is trained on the BDD100K dataset, which is smaller in size and characterized by higher visual complexity, often containing multiple objects per image. These differences in dataset size and composition affect reconstruction performance and make direct comparisons with the original TiTok less straightforward.

While our model requires 64 latent tokens to achieve competitive reconstruction quality, as reflected in rFID scores, these scores approach those of our baseline VQGAN model, which is trained using 256 tokens and employs the same reconstruction and generation losses as TiTok. Despite being trained on a smaller and more complex dataset, our reconstruction performance remains close to the baseline, and our model surpasses it in generative performance. This indicates that the 1D-tokenizer's reconstructed tokens effectively support high-quality image generation.

7 Limitations

While this study demonstrates an effective approach for reducing images to 64 tokens with reasonably good-quality image reconstructions and generations, certain limitations remain:

1. The proposed method may incur a loss of fine-grained spatial details and its generalizability to other downstream vision tasks remains unverified.
2. The conclusions presented are based on models trained specifically on the BDD100K driving dataset for image reconstruction and generation. Their performance on more complex or noisy datasets remains an open question.
3. Although our model exhibits continued performance gains with additional training epochs, further exploration was constrained by limited computational resources.
4. The exact reasons behind the observed similarity in behavior between MAE- and CLIP-initialized TiTok-64 models remain unknown, and require systematic investigation.
5. The performance gap in generation quality between different pretrained initializations - such as the weaker generative performance of DINO-v2 compared to MAE - has not been fully explained and examined across a broader range of pretrained models.

Future work could address these by expanding evaluation to discrete-domain datasets across a wider range of vision tasks, conducting in-depth analyses of the effects of different model initializations, and investigating the impact of prolonged training to further enhance performance.

8 Conclusion

We conducted a comprehensive evaluation of the TiTok tokenizer in complex, real-world driving scenarios using the BDD100K dataset. Our findings indicate that token configurations demonstrating strong performance on ImageNet do not necessarily transfer effectively to domains with higher visual complexity, such as urban driving scenes characterized by diverse object categories, dynamic layouts, and varying environmental conditions. To address this, we systematically examined the influence of three key factors: (1) the number of one-dimensional latent tokens, (2) different pretrained initialization strategies, and (3) the integration of TiTok with the Maskgit generative framework. This analysis enabled us to identify configurations that achieve a balanced trade-off between token compactness and high-quality reconstruction and generation performance. In particular, the MAE-initialized TiTok variant consistently produced semantically meaningful tokens that surpassed the baseline in image generation tasks while maintaining competitive reconstruction quality. These results highlight TiTok's applicability to visually demanding domains and underscore the need for domain-aware token design choices and carefully selected initialization strategies to ensure robust performance beyond standard benchmark datasets.

Bibliography

[1] Q. Yu, M. Weber, X. Deng, X. Shen, D. Cremers, and L.-C. Chen, "An image is worth 32 tokens for reconstruction and generation," 2024.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[3] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," 2014.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[5] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, "Maskgit: Masked generative image transformer," 2022.

[6] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," 2020.

[7] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," 2021.

[8] Q. Yu, H. Wang, S. Qiao, M. Collins, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "kmax-deeplab: k-means mask transformer," 2023.

[9] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Max-deeplab: End-to-end panoptic segmentation with mask transformers," 2021.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[11] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," 2020.

[12] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[13] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," *Proceedings of the 25th International Conference on Machine Learning*, 2008.

[14] D. P Kingma and W. Max, "Auto-encoding variational bayes," *arXiv:1312.6114*, 2014.

[15] A. van den Oord, V. Oriol, and K. Koray, "Neural discrete representation learning," *arXiv:1711.00937*, 2018.

[16] R. Ali, v. d. O. Aaron, and V. Oriol, "Generating diverse high-fidelity images with vq-vae-2," *arXiv:1906.00446*, 2019.

[17] E. Patrick, R. Robin, and O. Björn, "Taming transformers for high-resolution image synthesis," *arXiv:2012.09841*, 2021.

[18] G. Ian J., P.-A. Jean, M. Mehdi, X. Bing, W.-F. David, O. Sherjil, C. Aaron, and B. Yoshua, "Generative adversarial networks," *arXiv:1406.2661*, 2014.

[19] J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu, "Vector-quantized image modeling with improved vqgan," *arXiv:2110.04627*, 2022.

[20] C. Shiyue, Y. Yueqin, H. Lianghua, L. Yu, Z. Xin, Z. Deli, and H. Kaiqi, "Efficient-vqgan: Towards high-resolution image generation with efficient vision transformers," *arXiv:2310.05400*, 2023.

[21] L. Doyup, K. Chiheon, K. Saehoon, C. Minsu, and H. Wook-Shin, "Autoregressive image generation using residual quantization," *arXiv:2203.01941*, 2022.

[22] Z. Chuanxia, T. V. Long, C. Jianfei, and P. Dinh, "Movq: Modulating quantized vectors for high-fidelity image generation," *arXiv:2209.09002*, 2022.

[23] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. Minnen, Y. Cheng, V. Birodkar, A. Gupta, X. Gu, A. G. Hauptmann, B. Gong, M.-H. Yang, I. Essa, D. A. Ross, and L. Jiang, "Language model beats diffusion – tokenizer is key to visual generation," *arXiv:2310.05737*, 2024.

[24] M. Fabian, M. David, A. Eirikur, and M. Tschannen, "Finite scalar quantization: Vq-vae made simple," *arXiv:2309.15505*, 2023.

[25] T. Jason and Rolfe, "Discrete variational autoencoders," *arXiv:1609.02200*, 2017.

[26] D. Prafulla and A. Nichol, "Diffusion models beat gans on image synthesis," *arXiv:2105.05233*, 2021.

[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and O. Björn, "High-resolution image synthesis with latent diffusion models," *arXiv:2112.10752*, 2022.

[28] E. Hoogeboom, J. Heek, and T. Salimans, "Simple diffusion: End-to-end diffusion for high resolution images," *arXiv:2301.11093*, 2023.

[29] W. Peebles and S. Xie, "Scalable diffusion models with transformers," *arXiv:2212.09748*, 2021.

[30] S. Gao, P. Zhou, M.-M. Cheng, and S. Yan, "Mdtv2: Masked diffusion transformer is a strong image synthesizer," *arXiv:2303.14389*, 2024.

[31] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, "Conditional image generation with pixelcnn decoders," *arXiv:1606.05328*, 2016.

[32] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, "Generative pretraining from pixels," *PMLR*, 2020.

[33] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," 2021.

[34] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021.

[35] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," 2021.

[36] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "Dinov2: Learning robust visual features without supervision," 2024.

[37] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth anything v2," 2024.

[38] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," 2021.

[39] J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu, "Vector-quantized image modeling with improved vqgan," 2022.

[40] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. Minnen, Y. Cheng, V. Birodkar, A. Gupta, X. Gu, A. G. Hauptmann, B. Gong, M.-H. Yang, I. Essa, D. A. Ross, and L. Jiang, "Language model beats diffusion – tokenizer is key to visual generation," 2024.