MULTIMODAL LATENT LANGUAGE MODELING WITH NEXT-TOKEN DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal generative models require a unified approach to handle both discrete data (e.g., text and code) and continuous data (e.g., image, audio, video). In this work, we propose Latent Language Modeling (LatentLM), which seamlessly integrates continuous and discrete data using causal Transformers. Specifically, we employ a variational autoencoder (VAE) to represent continuous data as latent vectors and introduce next-token diffusion for autoregressive generation of these vectors. Additionally, we develop σ-VAE to address the challenges of error accumulation and collapsed variance, which is crucial for autoregressive modeling. Extensive experiments demonstrate the effectiveness of LatentLM across various modalities. In image generation, LatentLM surpasses Diffusion Transformers in both performance and scalability. When integrated into multimodal large language models, LatentLM provides a general-purpose interface that unifies multimodal generation and understanding. Experimental results show that LatentLM achieves favorable performance compared to Transfusion and vector quantized models in the setting of scaling up training tokens. In text-to-speech synthesis, LatentLM outperforms the state-of-the-art VALL-E 2 model in speaker similarity and robustness, while requiring 10$\times$ fewer decoding steps. The results establish LatentLM as a highly effective and scalable approach to advance large multimodal models.

1 INTRODUCTION

Multimodal generative models require a unified method to process both discrete data (e.g., text, code) and continuous data (e.g., video, audio, robot actions). While pipeline-based systems exist, they lose information and cannot be optimized end-to-end. Research on natively handling both data types in multimodal large language models (MLLMs) has followed three main paths. First, continuous data can be quantized into discrete codes using VQ-VAE [77; 17], making them compatible with autoregressive LLMs [56; 80; 72]. Although simple, this approach creates a restrictive bottleneck, where lossy tokenization degrades both generation and understanding quality. Second, one can use diffusion models [25; 58; 4; 71] for MLLMs. However, these models typically rely on bidirectional attention, making them fundamentally incompatible with the efficient, causal, autoregressive paradigm of LLMs that leverages KV cache. Frameworks like Transfusion [86] attempt to bridge this gap, but still require separate objectives and cannot perform joint understanding and generation within a single forward pass, as the noisy inputs required for diffusion training interfere with the understanding tasks. The third path, masked autoregressive [39] (MAR) models, offers strong generation by also leveraging bidirectional attention, but consequently shares the same incompatibility with causal LLMs and incurs high computational costs.

In this work, we propose latent language modeling (LatentLM) to transcend the above trade-offs. LatentLM provides a single, causal, autoregressive architecture that supports both high-fidelity continuous generation and deep multimodal understanding without compromise. Specifically, we represent continuous data as high-fidelity latent vectors using a variational autoencoder (VAE). We then introduce next-token diffusion, where a lightweight diffusion head autoregressively predicts these latent vectors, conditioned on the causal Transformer's hidden states. For discrete data, the same Transformer backbone performs standard next-token prediction. Furthermore, to address the problem of error accumulation in continuous autoregressive generation, we propose σ-VAE, which maintains the variance of the latent space to mitigate exposure bias and ensure robust, long-sequence generation.

As shown in Figure 1, LatentLM unifies the generation of discrete and continuous tokens under a single language modeling paradigm. The proposed method simplifies implementation by reusing the existing distributed training infrastructure of large language models. Another advantage is that LatentLM unifies generation and understanding with a general-purpose interface, which perceives and produces any combination of multimodal data, e.g., text, image, audio, video, and robot action data. Compared to quantizing continuous data, LatentLM achieves a higher token reduction ratio while preserving reconstruction quality.

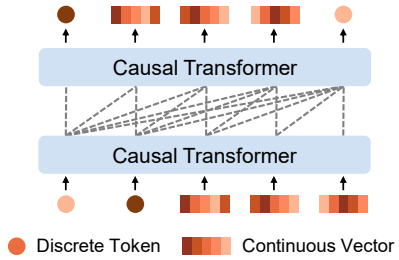


Figure 1: LatentLM seamlessly handles continuous (e.g., image, audio, video) and discrete (e.g., text and code) data.

We conduct experiments on image generation, multimodal large language models, and text-to-speech synthesis to show the flexibility and effectiveness of LatentLM across modalities. First, image generation on ImageNet [15] shows that LatentLM achieves competitive performance with the models based on diffusion (e.g., DiT [50]) or discrete tokens. Second, we train multimodal large language models with text, image-text pairs, and interleaved data. LatentLM outperforms Transfusion [86] and the model with vector-quantized image tokenizers, in terms of language modeling, text-to-image generation, and vision-language understanding metrics. We also scale up the number of training tokens and find that LatentLM has favorable scaling properties. Third, experimental results on text-to-speech synthesis show that LatentLM achieves better performance than previous systems. Because our tokenizer uses continuous representations, the token reduction ratio is much larger than previous vector-quantized tokenizers, which improves both the training and inference efficiency.

2 Related Work

Continuous Autoregressive Generation Using discrete tokens for autoregressive generation has been a common approach [17; 83; 68]. In the context of continuous token generation, recent work has explored alternative mechanisms. For example, GIVT [74] introduces the use of a VAE head or adapter to predict continuous tokens, while MAR [39] proposes the use of a diffusion head for masked generation. MAR's primary contribution is its masked bidirectional attention model; its causal attention baseline was shown to underperform, a finding which our work revisits. LatentLM combines the strengths of diffusion as a powerful predictive model with error-tolerant σ-VAE, demonstrating that a purely causal autoregressive approach can be highly effective and MLLM-compatible.

Unified Multimodal LLM Previous unified multimodal LLM approaches can be broadly classified into two types. The first, based on discrete token models [72; 43], usually sacrifices some multimodal capability with vector quantization. These models excel at text tasks, but struggle with complex multimodal interactions. The second approach integrates LLM backbones for image-level diffusion [86; 82], but this is incompatible with autoregressive generation and increases inference complexity. Moreover, noisy image input during pretraining reduces training efficiency. In contrast, LatentLM achieves a better balance by maintaining multimodal understanding while addressing these inference challenges, offering both efficiency and performance.

3 Latent Language Modeling

Latent language modeling (LatentLM) autoregressively perceives and generates multimodal sequences (with discrete and continuous data) in a unified way. As shown in Figure 2, the model is a causal Transformer, where the t-th token is predicted by conditioning on previous $t-1$ tokens. Continuous data are generated by next-token diffusion (Section 3.1), where the diffusion head is used to produce continuous vectors for each position. In addition, discrete tokens are generated by next-token prediction, similar to conventional language modeling.

Our model processes an input sequence $x = x_1, \ldots, x_N$, which can contain both discrete and continuous tokens. We convert discrete tokens into vectors using an embedding lookup table. For continuous data, we use a variational autoencoder (VAE) to encode them into latent vector

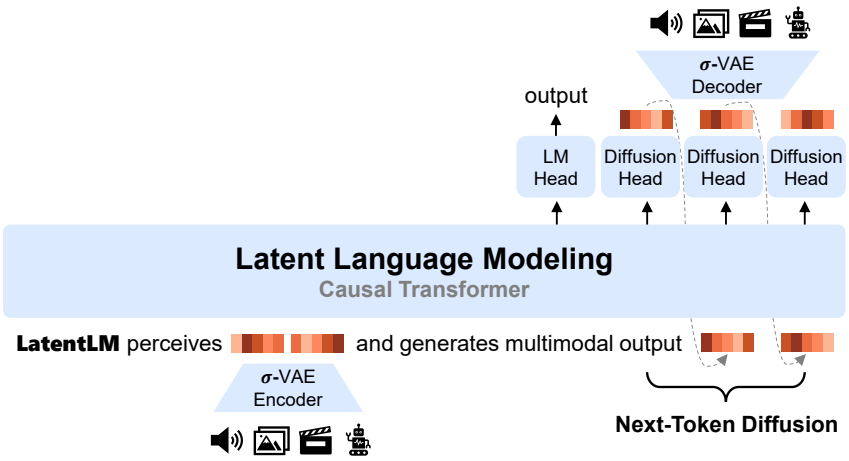


Figure 2: LatentLM unifies the modeling of continuous and discrete data with causal Transformers. We introduce σ-VAE (Section 4) to represent continuous data as latent vectors. We perform next-token diffusion (Section 3.1) to autoregressively predict the latent vectors one by one. The diffusion head generates vectors by conditioning on the output states of Transformer. The predicted vectors can be decoded to produce the final outputs.

representations. Finally, we combine all token vectors into a single matrix $X^0 = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{N \times d}$, where d is the model's hidden dimension. This matrix X^0 is then fed into a causal Transformer.

The model is stacked with L Transformer [78] layers with LLaMA augmentation [73]. Causal masking is used for autoregressive generation. The input X^0 is further contextualized to obtain the output X^L, i.e., $X^l = \text{Decoder}(X^{l-1})$, $l \in [1, L]$. The output states of Transformer $[\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N] = \text{RMSNorm}(X^L)$ are used to decode the predictions:

$$\text{Decode}(x_i|x_{<i}) = \begin{cases} \text{Sample}\left(P_d(x_i|x_{<i})\right) & x_i \in \mathcal{D} \\ \text{Diffusion}(\boldsymbol{h}_i) & x_i \in \mathcal{C} \end{cases} \tag{1}$$
$$P_d(x_i|x_{<i}) = \text{softmax}(\boldsymbol{h}_i W_v)$$

where $W_v \in \mathbb{R}^{d \times |\mathcal{V}|}$ is the softmax weight, $|\mathcal{V}|$ is vocabulary size, $\mathcal{D}$ is discrete token set, $\mathcal{C}$ is continuous token set, and $\text{Sample}(\cdot)$ is a sampling algorithm (e.g., greedy decoding, and top-p sampling). The $\text{Diffusion}(\cdot)$ head (detailed in Section 3.1) decodes continuous vectors by conditioning on the hidden state $\boldsymbol{h}_i$. The latent vectors are generated autoregressively one by one, i.e., next-token diffusion. Then the VAE decoder is used to generate raw data from the predicted latent vectors.

3.1 Next-Token Diffusion

LatentLM autoregressively generates the continuous tokens. We use diffusion as the language model head for each continuous token. The diffusion head progressively refines and generates the latent vector $\boldsymbol{x}_i$ by conditioning on the hidden state $\boldsymbol{h}_i$. Then the predicted $\boldsymbol{x}_i$ is used as input for the next step of Transformer. In our experiments, we use either denoising diffusion probabilistic model (DDPM) [25] or flow matching [42] as our design choice. We use DDPM as an example to describe the details. Diffusion is formulated as two processes, i.e., the forward process gradually adds noise to the input, and the reverse process learns to denoise step by step.

Forward Process Noise is introduced incrementally into the original vector in T steps. Let $\boldsymbol{x}_i^0 = \boldsymbol{x}_i$ denote the original data and $\boldsymbol{x}_i^t$ the noisy version, where $t = 1, \cdots, T$. The Markov noise-addition process is defined as $q(\boldsymbol{x}_i^t|\boldsymbol{x}_i^{t-1}) = \mathcal{N}(\boldsymbol{x}_i^t; \sqrt{1-\beta_t}\boldsymbol{x}_i^{t-1}, \beta_t \boldsymbol{I})$, where Gaussian noise is injected in each step, β_t follows a predefined noise schedule, and $\boldsymbol{I}$ is the identity covariance matrix. A nice property is that we can directly sample $\boldsymbol{x}_i^t$ from the original data $\boldsymbol{x}_i$ through $\boldsymbol{x}_i^t = \sqrt{\overline{\alpha}_t}\boldsymbol{x}_i + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}$, where $\overline{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$.

Reverse Process The diffusion model learns to reverse the noising process by training a network $p_\theta(\boldsymbol{x}_i^{t-1}|\boldsymbol{x}_i^t, \boldsymbol{h}_i)$ to predict the noise added at each step. DDPM learns a model $\epsilon_\theta(\boldsymbol{x}_i^t, t, \boldsymbol{h}_i)$ to estimate the noise $\boldsymbol{\epsilon}$ (as described in the forward pass) of $\boldsymbol{x}_i^t$ in the t-th step, conditioning on the

Transformer state $\boldsymbol{h}_i$. The model parameters are learned by minimizing the following loss:

$$\mathcal{L}_{\text{Diff}}(\boldsymbol{x}_i, \boldsymbol{h}_i) = \mathbb{E}_{\boldsymbol{x}_i,t,\boldsymbol{\epsilon}} \parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_i^t, t, \boldsymbol{h}_i) \parallel^2 \tag{2}$$

where $\boldsymbol{\epsilon}$ is the actual Gaussian noise.

Head Architecture We use a lightweight neural network as $\boldsymbol{\epsilon}_\theta(\cdot)$ in Equation (2), which is a residual architecture incorporating pre-RMSNorm [84] and feedforward networks [39]. The network input is a vector that contains noise. The output is the predicted noise $\boldsymbol{\epsilon}_\theta(\cdot)$. We also utilize AdaLN-Zero [50] which conditions on both the timestep t and the Transformer output $\boldsymbol{h}_i$. This head processes a noised continuous vector and predicts the corresponding noise.

Inference The Transformer state $\boldsymbol{h}_i$ is used as the condition for diffusion head. Starting from pure Gaussian noise $\boldsymbol{x}_T$, the diffusion head iteratively denoises it using the predicted noise $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_i^t, t, \boldsymbol{h}_i)$ to produce $\boldsymbol{x}_{t-1}$. We use DPM-Solver [47; 48] to accelerate the denoising process, significantly reducing the number of inference steps compared to training.

3.2 Model Training and Inference

Training During training, we compute the token-level loss for training sequences. For discrete data, we use the standard language modeling loss, $\mathcal{L}_{\text{LM}} = -\sum_{x,i} \log P_d(x_i|x_{<i})$, where the prediction probability is presented in Equation (1). For continuous data, the loss function $\mathcal{L}_{\text{Diff}}$ described in Equation (2) is used. The training objective is to minimize $\mathcal{L}_{\text{LM}} + \alpha\mathcal{L}_{\text{Diff}}$, where α is a hyperparameter. In practice, we sample multiple diffusion timesteps, typically four, for a single forward pass [39]. As the diffusion head is usually lightweight, reusing the computation of the Transformer backbone improves training efficiency while introducing minimal overhead.

Inference The decoding process is similar to that of standard causal Transformers, i.e., following Equation (1) to predict the next token based on the generation history that has come before it. The Transformer backbone is computed in a single pass, and only the lightweight diffusion head requires multiple denoising steps. In addition, we use special tokens to indicate the switch between the language modeling head and the diffusion head. For instance, we use `<BOD>` to denote the beginning of the diffusion head usage, and `<EOD>` to indicate the switch back to the language modeling head.

4 σ-VAE: The Keystone for Next-Token Diffusion

4.1 Background: Variational Autoencoder (VAE)

The tokenizer compresses continuous data into latent vectors. It is based on a variational autoencoder [33] (VAE), which encodes the input data into a latent space and then decodes it back to the original space. Let x denote the continuous input and z the compressed vector representations. VAEs maximize the evidence lower bound of log-likelihood $\log p(x)$ via: $\max \ \mathbb{E}_{q_\phi(z|x)} [\log p_\psi(x|z)] - \mathcal{D}_{\text{KL}} [q_\phi(z|x) \parallel p(z)]$, where the encoder $q_\phi(z|x)$ encodes input x to latent vectors z, the decoder $p_\psi(x|z)$ reconstructs data by conditioning on z, and the KL term encourages that the latent space follows a Gaussian prior.

4.2 The Ineffectiveness of Conventional VAEs for Autoregressive Modeling

Figure 5 shows that conventional VAEs perform poorly for autoregressive modeling. We explain why conventional VAEs are not suitable for this purpose as follows.

Training-Inference Mismatch The issue relates to exposure bias in autoregressive models, which becomes more severe in continuous latent spaces. Consider a latent token x whose VAE posterior is modeled as $\mathcal{N}(x, \sigma^2)$. During training with teacher forcing, the model always observes samples from $\mathcal{N}(x, \sigma^2)$. However, during inference, it uses its own generated outputs as inputs. These generated latent vectors come from a different distribution, call it $\mathcal{N}(x, \sigma^2_{\text{gen}})$, and in practice σ^2_{gen} can be larger, due to compounding generation errors. This mismatch between the training and inference distributions can lead to degradation over time, especially in long-form autoregressive generation. In traditional VAEs, where the variance may vary across tokens (and in some cases be extremely small), this divergence can push the model into out-of-distribution regions during inference.

Collapsed Variance The VAE models used in previous diffusion models typically perform more like an autoencoder, i.e., with collapsed σ, as shown in the statistics of Figure 5. This leads to inference degradation when diffusion is combined with autoregressive decoding, as explained above. Although the variance is not directly fed to the model, the variance affects the teacher-forcing inputs fed into the autoregressive training. Larger σ of VAE makes the model more robust to exposure bias.

4.3 Solution: σ-VAE

As shown in Figure 3, we propose σ-VAE to prevent variance collapse by enforcing a fixed variance in the latent space. The reconstruction pass is computed as:

$$\mu = \text{Encoder}_\phi(x)$$
$$z = \mu + |\sigma| \odot \boldsymbol{\epsilon}, \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0,1),\ \sigma \sim \mathcal{N}(0, C_\sigma)$$
$$\hat{x} = \text{Decoder}_\psi(z)$$

where σ is a scalar, C_σ is a hyperparameter, $\text{Encoder}_\phi(\cdot)$ and $\text{Decoder}_\psi(\cdot)$ are learnable models. The input x is fed into the encoder to obtain μ. The re-parameterization trick is used to make z follow the Gaussian distribution $\mathcal{N}(\mu, \sigma I)$. The variance σ is fixed across channels, and is sampled from $\mathcal{N}(0, C_\sigma)$ for each example. Then z is fed into the decoder for reconstruction. The training objective of σ-VAE is:

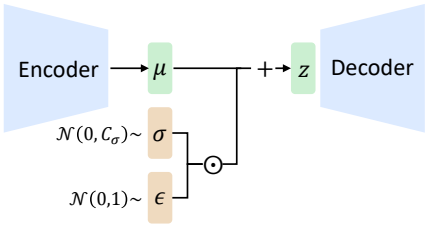


Figure 3: σ-VAE uses a fixed variance for the latent space. It avoids variance collapse and makes LatentLM more robust to exposure bias during autoregressive generation. σ is a scalar that is sampled from $\mathcal{N}(0, C_\sigma)$ for each example.

$$\text{minimize} \quad \|\hat{x} - x\|_2^2 + \beta \|\mu\|_2^2$$

where the first term is the reconstruction error, and β controls the trade-off between reconstruction quality and adherence to the prior distribution [23].

Advantages The proposed σ-VAE addresses the above issues (described in Section 4.2) by explicitly lower-bounding the latent variance. This makes the model robust to generation error by ensuring the inference-time distribution stays within the distributional support seen during training. In practice, we observe that after training, the condition $\sigma_{\text{gen}} < \sigma_{\text{train}}$ tends to hold in most tokens, effectively converting an out-of-distribution generalization problem into an in-distribution one, thereby significantly improving stability in autoregressive inference.

5 Experiments

We evaluate LatentLM through multiple dimensions to thoroughly assess its effectiveness and scalability. We conduct experiments on various tasks, i.e., image generation, multimodal large language models, and text-to-speech synthesis.

5.1 Image Generation: Scalable Autoregressive Modeling

The image generation experiments are conducted on ImageNet [15]. Given a category, the goal is to generate the corresponding images. First, we systematically benchmark our model against state-of-the-art baselines to demonstrate the advantages of next-token diffusion. We also investigate the scalability of our approach by evaluating it with larger model sizes and higher resolutions. Furthermore, we compare the design choices of σ-VAE tokenizers. Finally, we assess the inference efficiency to highlight the practical deployment benefits of our method.

5.1.1 System Evaluation

Setup We scale up model size and number of training steps. We set the Transformer's hidden size to 1024 and the number of layers to 32. The intermediate dimension of feedforward networks is 2730. The diffusion head has six layers. We use the AdamW [46] optimizer with $\beta = (0.9, 0.98)$. We use a cosine learning rate schedule with the maximal value of 5e-4 and 100 warmup steps. The weight decay is set to 0.1. We train models with 250,000 steps with batch size of 2048. The number of training epochs is about 400. Classifier-free guidance [24] is set to 1.65. As shown in Table 1, the

Type	Model	#Params	#Epochs	FID↓	IS↑
Non-Causal-Masking Generation					
Diffusion	LDM-4 [58]	400M	—	3.60	247.7
	DiT-XL/2 [50]	675M	400	2.27	278.2
	U-ViT-H/2 [3]	501M	400	2.29	263.9
Masked Generative	MaskGIT [8]	227M	300	4.02	355.6
	MAR-L [39]	479M	800	1.78	296.0
Causal-Masking Generation					
Causal-Discrete	VQGAN [17]	1.4B	240	5.20	280.3
	ViT-VQGAN [83]	1.7B	240	3.04	227.4
	LlamaGen-XL [68]	775M	300	2.62	244.1
	LlamaGen-XXL [68]	1.4B	300	2.34	253.9
Causal-Continuous	GIVT-Causal-L+A [74]	1.67B	500	2.59	—
	LatentLM-L (This Work)	479M	400	2.24	253.8

Table 1: Image generation results on ImageNet [15]. We evaluate FID [22] and IS [61]. LatentLM achieves competitive performance, especially compared with other causal-masking generation models.

model configurations have been aligned with those of previous models to ensure fair comparisons. The training details of σ-VAE are presented in Section H.2.

Table 1 presents a comprehensive comparison between LatentLM and various image generation methods. These methods can be categorized into two main groups: (1) non-causal-masking models, including image-level diffusion models (LDM [58], DiT [50], U-ViT [3]) and masked generative models (MaskGIT [8], MAR [39]); and (2) causal-masking models, comprising discrete-token generation approaches (VQGAN [17], ViT-VQGAN [83], LlamaGen [68]) and continuous autoregressive generation methods [74] (GIVT-Causal).

Results As shown in Table 1, LatentLM achieves competitive performance compared to previous work. Notice that non-causal-masking models typically require iterative forward computation during inference. Consequently, the inference FLOPs of non-causal-masking models tend to be larger due to multiple forward passes. Moreover, models using continuous representations typically outperform those using discrete code, even though LatentLM-L has fewer parameters. Among the methods, MAR [39] and GIVT [74] are the most relevant. In comparison, MAR uses a bidirectional Transformer to implement masked autoregressive modeling, instead of causal Transformer, which renders MAR unable to reuse key-value caches for multiple forward passes. Furthermore, unifying MAR and language modeling in multimodal models remains challenging due to their distinct modeling approaches. In contrast, Section 5.2 shows that our approach can be naturally applied to multimodal large language models. In addition, GIVT directly predicts latent vectors of VAEs with Gaussian mixture models. The main difference is that LatentLM integrates diffusion into causal Transformers, which tends to offer more powerful modeling expressivity. The results also indicate that our approach outperforms GIVT with a smaller model size and fewer training epochs.

5.1.2 Scalability

We compare the scalability properties of Diffusion Transformer [50] (DiT) and LatentLM, in terms of model size, and image resolution.

Setup In order to be consistent with LatentLM, we also augment DiT with RMSNorm [84] and SwiGLU [55; 66]. All models were trained with 75k steps, i.e., approximately 120 epochs, for scaling experiments. Classifier-free guidance [24] is set to 1.75 during inference. Detailed hyperparameters are presented in Section H.1.

Scaling Image Resolution As shown in Table 2, we conduct experiments at a resolution of 384, training a 1.82B model for 100k steps. The results of 384-pixel resolution show significant improvements over the 256 when using classifier-free guidance [24]. The improvement stems from the richer details and additional information captured in the tokenizer

Resolution	FID↓
256×256	3.19
384×384	**2.51**

Table 2: FID-50k [22] results of scaling up image resolution.

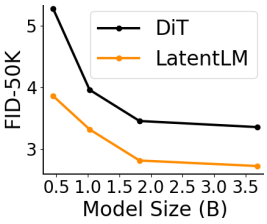
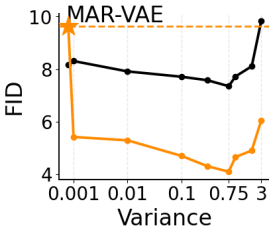


Figure 4: Scaling curves of DiT and LatentLM. FID results improve with larger model size. LatentLM shows favorable scaling properties.



VAE	Variance	FID↓ DiT	FID↓ LatentLM
MAR [39]	8e-4	**8.16**	9.64
σ-VAE	0.75	7.35	**4.10**

Figure 5: FID [22] scores of Diffusion Transformer (DiT) and LatentLM on ImageNet. The "star" (MAR-VAE) represents the tokenizer tuned for previous diffusion models [39], which are ineffective for LatentLM. The results indicate that LatentLM favors tokenizers with larger variances, highlighting the importance of σ-VAE.

with higher resolutions. Moreover, increasing resolution leads to longer sequences, which scales the decoding computation up.

Scaling Model Size As shown in Figure 4, we trained models of varying sizes, i.e., 455M, 1.03B, 1.82B, 3.68B. LatentLM consistently outperforms DiT models. The results demonstrate our approach's effective scaling properties in terms of model size.

5.1.3 Effects of Tokenizer

As shown in Figure 5, we analyze the effects of σ-VAE tokenizers with various configurations. We evaluate their performance in both the DiT and LatentLM frameworks. Specifically, we train the σ-VAE tokenizers with different variance. To simplify the analysis, we use fixed variance values σ, rather than sampling them from $\mathcal{N}(0, C_\sigma)$. More comparison details are presented in Section H.3.

Figure 5 presents the FID-50K scores of DiT and LatentLM using tokenizers with different variance. The "stars" in the figure represent tokenizers that were tuned for previous latent diffusion models [58], which usually have a small variance, i.e., being more like an autoencoder instead of VAE. The other "dots" are σ-VAE with fixed variance. We summarize the findings as follows:

The tokenizers tuned for previous image-level diffusion models are ineffective for LatentLM. For LatentLM, the "stars" (in Figure 5) perform significantly worse than the others that have larger tokenizer variances. The results indicate that directly adopting tokenizer configurations from previous diffusion models is suboptimal for LatentLM. The tokenizers with small variances are not robust to autoregressive error [74].

LatentLM favors tokenizers with larger variances. For the example without classifier-free guidance (i.e., CFG=1.0), LatentLM improves monotonically with increased variance. In contrast, the choice of variance is not critical for DiT models. The analysis highlights the advantage of σ-VAE, whose variance is easily controllable. So we recommend to use re-trained σ-VAE as tokenizers for LatentLM, rather than directly using previous ones.

5.2 Multimodal LLMs: Unified Understanding and Generation

We train multimodal large language models with LatentLM for unified understanding and generation. In this section, we focus on vision-language models. By unifying next-token prediction and diffusion, the model can seamlessly handle interleaved image-text data, text-only data, and image-text pairs. The proposed method simplifies the multimodal training and inference processes, allowing to learn in context (e.g., few-shot), follow multimodal instructions, and perform multimodal dialogue. Moreover, unified modeling enables new capabilities. For example, we can edit or generate images by conditioning on text and multiple images.

We use three types of data in the training stage: text-only data, image-text pair data, and interleaved text-image data. The mix-up ratio is 2:1:1. We train a 1.3B-size Transformer as the backbone. The training sequence length is 4096. The batch size is 4M tokens. We train the model with 50,000 steps (i.e., 200B tokens) for comparison. More training details are described in Section H.4.

Model	Text	Text-to-Image		Image-to-Text	
	Valid PPL↓	FID↓	CLIP↑	MS-COCO↑	VQAv2↑
VQ-MLLM	2.79	16.92	**29.33**	37.4	30.19
Transfusion [86]	2.74	16.10	28.66	43.4	35.36
LatentLM	**2.73**	**14.54**	28.75	**54.5**	**38.72**

Table 3: Results of multimodal large language models on text language modeling, image-to-text, and text-to-image generation. We compare with Transfusion [86] and vector quantized models (VQ-MLLM; i.e., using discrete code to represent images). "PPL" is perplexity. CLIP [53] score measures the similarity. We report CIDEr [79] score for MS-COCO [41] and accuracy for VQAv2 [20].

We compare LatentLM with Transfusion [86], and vector quantized models (VQ-MLLM; i.e., the models using vector quantized image tokenizers). Specifically, Transfusion shares Transformer weights for autoregressive language modeling and image-level diffusion, which uses bidirectional iterative denoising for images and causal masking for text. Moreover, VQ-MLLM uses VQ-VAE [77; 17] as the tokenizer for images, where images are compressed to discrete code. We use the VQ-VAE tokenizer open-sourced by LlamaGen [68] in VQ-MLLM. We use the same training configuration and tokenizer settings for comparison. To align the number of parameters, we use a 6-layer ViT as the image head of Transfusion.

Language Modeling Table 3 presents the evaluation results on language modeling, text-to-image generation, and multimodal understanding. First, LatentLM achieves a better perplexity in language modeling. The results indicate that our method tends to better share knowledge between modalities with less conflicts. The similarity between next-token prediction and next-token diffusion also benefits the unified modeling. We further evaluate language-only tasks in Section D showing the advantage of LatentLM.

Text-to-Image Generation Then we evaluate text-to-image generation on MS-COCO [40]. Table 3 shows that LatentLM achieves lower FID scores, i.e., better generation quality. The trend is also consistent with Table 1, where Transfusion is aligned with DiT, and VQ-MLLM with LlamaGen. In addition, Figure 6 presents the scaling curves in terms of the number of training tokens, where LatentLM consistently achieves better FID scores. It is worth noting that the performance of VQ-MLLM seems saturated compared to the other methods. Figure 12 also shows several text-to-image samples of LatentLM.

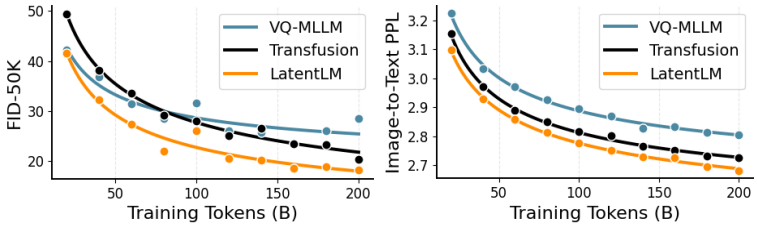


Figure 6: Left: Text-to-image FID [22]. Right: Image-to-text perplexity.

Image-to-Text Generation Table 3 reports image captioning on MS-COCO [40] and visual question answering on VQAv2 [20]. LatentLM's superior performance on understanding tasks can be attributed to two key factors. First, unlike VQ-MLLM, our continuous latent representation is relatively lossless, preserving fine-grained visual detail. Second, unlike Transfusion, our training objective is consistent for both understanding and generation, as we do not corrupt the image input with noise, leading to better feature learning. Figure 6 presents text perplexity on the image-to-text validation data. The results are also consistent as in Table 3.

5.3 Text-to-Speech Synthesis: Higher Token Reduction Ratio, Fewer Decoding Steps

We apply LatentLM to text-to-speech synthesis (TTS). Due to continuous representations, σ-VAE achieves superior reconstruction results with a significantly higher token reduction ratio and lower frame rate than previous speech tokenizers [13; 36; 67; 29; 14]. LatentLM outperforms the state-of-the-art VALL-E 2 [11] model on both speaker similarity score and robustness while requiring 10$\times$ fewer decoding steps. We follow the settings of VALL-E 2. The training data are Libriheavy [32], which includes 50k hours of speech from approximately 7k different speakers. The Transformer backbone has about 300M parameters. The diffusion head contains three layers of feedforward networks. σ-VAE employ a convolutional architecture that supports streaming encoding and decoding.

System	Frame Rate Length/s ↓	Ref Utterance as Prompt			3s Prefix as Prompt		
		SIM↑	WER-C↓	WER-H↓	SIM↑	WER-C↓	WER-H↓
Ground Truth	-	0.779	1.6	2.2	0.668	1.6	2.2
VALL-E 2 [11]	75	0.643	1.5	2.4	0.504	1.6	2.3
Voicebox [37]	100	0.662	-	1.9	**0.593**	-	**2.0**
MELLE [49]	62	0.625	1.5	2.1	0.508	1.5	**2.0**
LatentLM	15	**0.697**	**1.2**	1.8	0.571	**1.4**	**2.0**
LatentLM	7.5	0.656	**1.2**	**1.7**	0.532	1.6	2.3
LatentLM	3.75	0.598	1.7	2.3	0.467	3.1	4.5

Table 4: LatentLM outperforms previous systems on zero-shot speech synthesis in both settings. Frame rate indicates how many autoregressive steps to generate one second of speech, i.e., fewer is faster. Moreover, the number of decoding steps is much less than others, achieving faster inference speed. The results are reported on LibriSpeech test-clean set. The WER-H and SIM results of VALL-E 2 using 3s prefix as prompt are from [49]. The evaluation metrics are described in Section H.5.3.

We train variants with token reduction ratios of 1600, 3200, and 6400. Section H.5 contains more details.

5.3.1 TTS Evaluation

Table 4 shows zero-shot TTS results on the LibriSpeech test-clean set. We evaluate the synthesis quality under two distinct settings: (1) using a reference utterance from the same speaker as the prompt, and (2) evaluating speech continuation by using the first 3 seconds of speech as the prompt.

Our model, operating at a frame rate of 15 (i.e., generating 1 second of speech in 15 autoregressive steps), surpasses previous state-of-the-art methods when using a same-speaker reference utterance as the prompt. LatentLM with a frame rate of 7.5 achieves superior performance compared to the neural codec language model VALL-E 2 [11], while requiring an order of magnitude ($10\times$) fewer autoregressive inference steps. Moreover, LatentLM eliminates the need for the non-autoregressive (NAR) model employed in VALL-E 2, resulting in improved computational efficiency. Even at a lower frame rate of 3.75, LatentLM maintains competitive performance. The higher token reduction ratio reduces the sequence length, which in turn greatly accelerates the decoding speed.

5.3.2 σ-VAE Tokenizers Have Higher Token Reduction Ratio and Good Quality

Table 5 compares σ-VAE and other codec models on the LibriTTS test-other set. σ-VAE achieves better reconstruction quality in a token reduction ratio of $1600\times$ compared to Encodec [13] ($40\times$), DAC [67] ($160\times$), WavTokenizer [29] ($320\times$), and Mimi [14] ($480\times$). Notably, as we further increase the token reduction ratio, the reconstruction quality does not deteriorate significantly. At a token reduction ratio of 6400, the resulting sequence length when used in a language model is already comparable to BPE tokenization [64], approaching a 1:1 ratio.

Tokenizer	N_q↓	Frame Rate ↓	Comp. Ratio ↑	Mel Dist.↓	PESQ↑	STOI↑	VISQOL↑	UTMOS↑
DAC [67]	2	75	160	0.916	2.269	0.896	3.981	3.297
WavTokenizer [29]	1	75	320	0.871	2.266	0.891	4.120	3.432
Mimi [14]	8	12.5	240	0.987	3.217	0.946	4.332	3.375
Mimi [14]	4	12.5	480	1.458	1.568	0.826	3.390	2.652
WavTokenizer [29]	1	40	600	1.037	1.670	0.834	3.782	3.053
σ-VAE$_{32}$	1	15	1600	0.813	2.724	0.926	4.268	3.491
σ-VAE$_{64}$	1	7.5	3200	**0.798**	**2.756**	**0.929**	**4.289**	**3.505**
σ-VAE$_{128}$	1	3.75	6400	0.852	2.533	0.916	4.165	3.460

Table 5: The σ-VAE tokenizers obtain competitive reconstruction quality while having high token reduction ratio. We report results on the LibriTTS test-other set and compare with the tokenizers whose token reduction ratio is larger than 100. "N_q" represents the number of quantizers. We define the token reduction ratio as the audio sample rate divided by N_q and the frame rate. "σ-VAE$_{32}$" denotes that the latent dimension of the tokenizer is 32. The evaluation metrics are described in Section H.5.3.

6 CONCLUSION

We introduce Latent Language Modeling (LatentLM) for unified multimodal generation and understanding that seamlessly integrates continuous and discrete data using causal Transformers. By leveraging next-token diffusion, LatentLM achieves competitive performance across image generation, text-to-speech synthesis, and multimodal large language models. The method is scalable and practical for real-world applications. Future work will delve into more advanced multimodal-native reasoning (e.g., self-reflection for automatically refining generated images or tracking search states via latent vectors), extend to long video generation with interleaved script creation, explore cross-modal transfer strategies that leverage high token reduction ratios for seamless integration of speech and text, and further investigate embodied AI for end-to-end robot action and planning in continuous spaces.

REFERENCES

[1] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[2] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pp. 4211–4215, 2020.

[3] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22669–22679, 2023.

[4] Fan Bao, Shen Nie, Kaiwen Xue, Chongxuan Li, Shi Pu, Yaole Wang, Gang Yue, Yue Cao, Hang Su, and Jun Zhu. One transformer fits all distributions in multi-modal diffusion at scale. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 1692–1717, 2023.

[5] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image Transformers. In *International Conference on Learning Representations*, 2022.

[6] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. The mtg-jamendo dataset for automatic music tagging. In *ICML*, 2019.

[7] Minwoo Byeon, Beomhee Park, Haecheon Kim, Sungjun Lee, Woonhyuk Baek, and Saehoon Kim. Coyo-700m: Image-text pair dataset, 2022.

[8] 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*, pp. 11315–11325, 2022.

[9] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3558–3568, 2021.

[10] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[11] Sanyuan Chen, Shujie Liu, Long Zhou, Yanqing Liu, Xu Tan, Jinyu Li, Sheng Zhao, Yao Qian, and Furu Wei. VALL-E 2: Neural codec language models are human parity zero-shot text to speech synthesizers. *CoRR*, abs/2406.05370, 2024.

[12] Michael Chinen, Felicia SC Lim, Jan Skoglund, Nikita Gureev, Feargus O'Gorman, and Andrew Hines. Visqol v3: An open source production ready objective speech and audio metric. In *2020 twelfth international conference on quality of multimedia experience (QoMEX)*, pp. 1–6. IEEE, 2020.

[13] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

[14] Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. Technical report, Kyutai, September 2024.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

[16] Harishchandra Dubey, Vishak Gopal, Ross Cutler, Sergiy Matusevych, Sebastian Braun, Emre Sefik Eskimez, Manthan Thakker, Takuya Yoshioka, Hannes Gamper, and Robert Aichner. Icassp 2022 deep noise suppression challenge. In *ICASSP*, 2022.

[17] 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*, pp. 12873–12883, 2021.

[18] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra. Fsd50k: an open dataset of human-labeled sound events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852, 2021.

[19] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 776–780. IEEE, 2017.

[20] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

[21] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

[22] 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. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6629–6640, 2017.

[23] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2016.

[24] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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

[26] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460, 2021.

[27] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Qiang Liu, Kriti Aggarwal, Zewen Chi, Johan Bjorck, Vishrav Chaudhary, Subhojit Som, Xia Song, and Furu Wei. Language is not all you need: Aligning perception with language models. *ArXiv*, abs/2302.14045, 2023.

[28] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[29] Shengpeng Ji, Ziyue Jiang, Xize Cheng, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Ruiqi Li, Ziang Zhang, Xiaoda Yang, et al. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. *arXiv preprint arXiv:2408.16532*, 2024.

[30] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pp. 694–711. Springer, 2016.

[31] Jacob Kahn, Morgane Rivière, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, Tatiana Likhomanenko, Gabriel Synnaeve, Armand Joulin, Abdelrahman Mohamed, and Emmanuel Dupoux. Libri-light: A benchmark for ASR with limited or no supervision. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pp. 7669–7673. IEEE, 2020.

[32] Wei Kang, Xiaoyu Yang, Zengwei Yao, Fangjun Kuang, Yifan Yang, Liyong Guo, Long Lin, and Daniel Povey. Libriheavy: A 50, 000 hours ASR corpus with punctuation casing and context. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pp. 10991–10995. IEEE, 2024.

[33] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations*, 2014.

[34] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Advances in neural information processing systems*, volume 33, pp. 17022–17033, 2020.

[35] Siqi Kou, Jiachun Jin, Chang Liu, Ye Ma, Jian Jia, Quan Chen, Peng Jiang, and Zhijie Deng. Orthus: Autoregressive interleaved image-text generation with modality-specific heads, 2024.

[36] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. High-fidelity audio compression with improved rvqgan. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 27980–27993, 2023.

[37] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[38] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, and StarCoder Team. StarCoder: may the source be with you! *ArXiv*, abs/2305.06161, 2023.

[39] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *arXiv preprint arXiv:2406.11838*, 2024.

[40] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

[41] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pp. 740–755, 2014.

[42] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[43] Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with ringattention. *arXiv preprint arXiv:2402.08268*, 2024.

[44] Zhijun Liu, Shuai Wang, Sho Inoue, Qibing Bai, and Haizhou Li. Autoregressive diffusion transformer for text-to-speech synthesis. *arXiv preprint arXiv:2406.05551*, 2024.

[45] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[46] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[47] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[48] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[49] Lingwei Meng, Long Zhou, Shujie Liu, Sanyuan Chen, Bing Han, Shujie Hu, Yanqing Liu, Jinyu Li, Sheng Zhao, Xixin Wu, Helen Meng, and Furu Wei. Autoregressive speech synthesis without vector quantization. *CoRR*, abs/2407.08551, 2024.

[50] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

[51] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra-Aimée Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: Outperforming curated corpora with web data, and web data only. *ArXiv*, abs/2306.01116, 2023.

[52] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *ArXiv*, abs/2306.14824, 2023.

[53] 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*, pp. 8748–8763. PMLR, 2021.

[54] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. The musdb18 corpus for music separation, 2017.

[55] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Swish: a self-gated activation function. *arXiv: Neural and Evolutionary Computing*, 2017.

[56] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *ArXiv*, abs/2102.12092, 2021.

[57] Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221)*, volume 2, pp. 749–752. IEEE, 2001.

[58] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

[59] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. Utmos: Utokyo-sarulab system for voicemos challenge 2022. *arXiv preprint arXiv:2204.02152*, 2022.

[60] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[61] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 2234–2242, 2016.

[62] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[63] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[64] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[65] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 2556–2565. Association for Computational Linguistics, 2018.

[66] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[67] Slava Shechtman and Avihu Dekel. Low bitrate high-quality rvqgan-based discrete speech tokenizer. In *Annual Conference of the International Speech Communication Association*, 2024.

[68] 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.

[69] Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[70] Cees H Taal, Richard C Hendriks, Richard Heusdens, and Jesper Jensen. A short-time objective intelligibility measure for time-frequency weighted noisy speech. In *2010 IEEE international conference on acoustics, speech and signal processing*, pp. 4214–4217. IEEE, 2010.

[71] Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. Any-to-any generation via composable diffusion. *arXiv preprint arXiv:2305.11846*, 2023.

[72] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *ArXiv*, abs/2405.09818, 2024.

[73] 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.

[74] Michael Tschannen, Cian Eastwood, and Fabian Mentzer. Givt: Generative infinite-vocabulary transformers. *arXiv preprint arXiv:2312.02116*, 2023.

[75] Michael Tschannen, André Susano Pinto, and Alexander Kolesnikov. JetFormer: An autoregressive generative model of raw images and text, 2024.

[76] Arnon Turetzky, Nimrod Shabtay, Slava Shechtman, Hagai Aronowitz, David Haws, Ron Hoory, and Avihu Dekel. Continuous speech synthesis using per-token latent diffusion, 2024.

[77] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Neural Information Processing Systems*, 2017.

[78] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 6000–6010, 2017.

[79] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, pp. 4566–4575, 2015.

[80] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. Neural codec language models are zero-shot text to speech synthesizers. *CoRR*, abs/2301.02111, 2023.

[81] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: Beit pretraining for vision and vision-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19175–19186, June 2023.

[82] 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.

[83] 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.

[84] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

[85] 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*, pp. 586–595, 2018.

[86] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

A Comparison of Paradigms

Table 6 compares different approaches based on their attention patterns, native compatibility with causal autoregressive large language models (MLLM-Native), support for efficient inference (KV cache), and their ability to be trained for unified understanding and generation. Our proposed method, LatentLM, successfully incorporates the desirable properties.

Paradigm	Attention	MLLM-Native	KV Cache	Unified Understanding-Generation Training
Discrete Token (VQ-MLLM, LlamaGen)	Causal	✓ Yes	✓ Yes	✓ Yes
Image-Level Diffusion (Transfusion)	Vision: Bidirectional Text: Causal	× No	× No	× No (noisy input)
Masked Autoregressive (MAR)	Bidirectional	× No	× No	× No (incompatible with causal LM)
LatentLM (Ours)	Causal	✓ Yes	✓ Yes	✓ Yes

Table 6: A comparative analysis of multimodal LLM paradigms.

B Contrast with Masked Autoregressive (MAR)

MAR [39] highlights its contribution as autoregressive modeling with **masked bidirectional** attention. As shown in the MAR paper's Table 1, its "autoregressive" baseline (causal attention; raster order) is treated as a baseline variant, which underperforms the proposed MAR (i.e., bidirectional attention; randomly masked) method. The key novelty of our method lies in the redesigned sigma-VAE, enabling more effective autoregressive inference. With this key difference, we show that causal-attention autoregressive modeling works very well, which is contradictory to the MAR's findings. We highlight the comparison results in Figure 5. Compared to MAR, our model supports KV cache and is fully compatible with multimodal large language models.

C Comparison between β-VAE and σ-VAE

β-VAE [23] sets different β values to learn disentangled factors. In comparison, the σ-VAE used in this work sets a fixed variance σ. The motivations and implementations are different. The VAE models used in previous diffusion models typically perform more like an autoencoder (AE), i.e., with collapsed σ. This leads to inference degradation when combining diffusion with autoregressive decoding. Our contribution lies in identifying and resolving this specific issue within the diffusion-AR framework using a simple yet effective σ-VAE configuration, without introducing new complexity.

D MLLM Evaluation on Language Benchmarks

We evaluate the multimodal large language models (MLLMs) trained in Section 5.2 on various language benchmarks.

Method	ARC-C	ARC-E	HellaSwag	OBQA	PIQA	Winogrande	Avg
LatentLM	33.99	64.86	58.30	37.60	74.86	58.96	54.76
Transfusion	32.36	64.86	58.11	35.80	73.88	57.62	53.77
Discrete Token	30.30	63.09	55.11	36.40	73.18	53.39	51.91

Table 7: Accuracy on language benchmarks. The MLLMs are presented in Section 5.2.

E GENERATION EXAMPLES

Figure 7: Samples of LatentLM trained on ImageNet. The resolution is 384×384. The images are generated by models described in Section 5.1.2.

Figure 8: A majestic mountain range covered in snow.

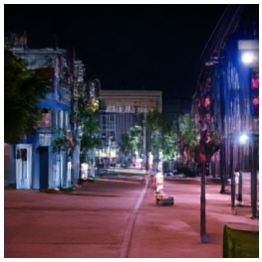


Figure 9: A city street illuminated by lights.

Figure 10: A crystal lake surrounded by autumn trees.

Figure 11: A small house in a wooden at sunset.

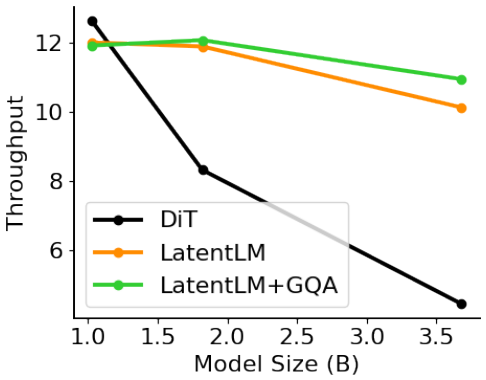
Figure 12: Text-to-image examples of LatentLM. The images are generated by models described in Section 5.2.

F INFERENCE EFFICIENCY WITH DIFFERENT MODEL SIZES

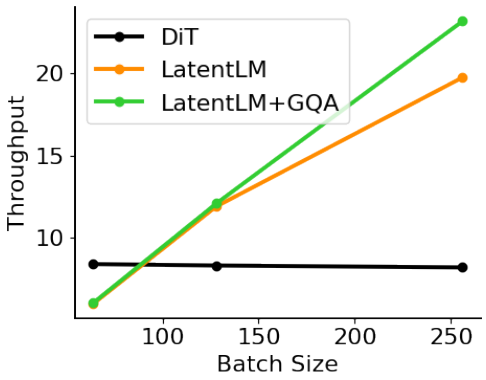
As shown in Figure 13, we investigate the inference capabilities of LatentLM by examining the effects of model size and batch size. We perform efficiency comparisons using 20 diffusion inference steps on a single H100 GPU.

First, we evaluate models ranging from 1B to 3.8B parameters with a fixed batch size of 128. Section F shows that DiT's throughput decreases significantly with larger model size. Because DiT has to iteratively perform multiple forward passes, it incurs higher computational costs. For the largest model with 3.8B parameters, LatentLM achieves a 2.47× increase in throughput, demonstrating its scalability advantages.

As presented in Section F, we then assess the 1.82B models with varying batch sizes. As the batch size increases, the throughput of LatentLM scales favorably with DiT. In addition, group-query attention [1] (GQA) further improves throughput. For a batch size of 256, our approach achieves a 2.84× throughput improvement. The results indicate that LatentLM benefits from significantly reduced FLOPs compared to image-level diffusion models, particularly at larger batch sizes.



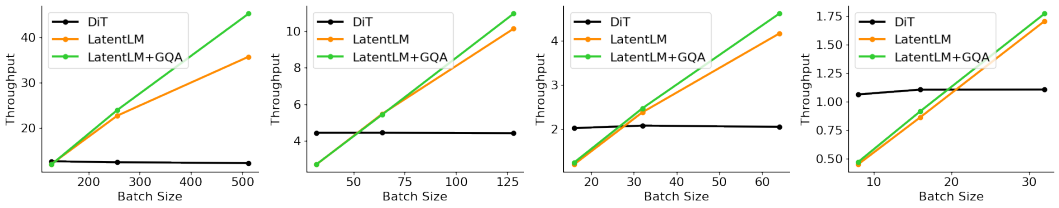
(a) Throughput with increasing model sizes.



(a) Throughput with increasing batch sizes.

Figure 13: We compare the inference throughput of Diffusion Transformer [50] (DiT) and LatentLM in the settings of different model size and batch size. "GQA" stands for group-query attention [1].

As shown in Figure 14, we evaluate the efficiency with various model size and batch size. The results show that LatentLM's throughput increases with a larger batch size. Our approach benefits from key-value caches of causal Transformers, which avoids recomputation of history predictions. In contrast, DiT's throughput remains similar. In addition, group-query attention [1] (GQA) further improves the inference efficiency of LatentLM. Another advantage is that we can directly reuse the inference infrastructure of large language models to deploy LatentLM.



1.03B 3.68B 9.35B 17.96B

Figure 14: Inference throughput of various model size and batch size. "GQA" stands for group-query attention [1].

G Ablation Studies of Text-to-Speech

Token Reduction Ratio and Latent Dimension We find that increasing the latent dimension enables the model to achieve a higher token reduction ratio and a lower frame rate. Table 8 presents the σ-VAE reconstruction and zero-shot speech synthesis results with different token reduction ratios and latent dimensions. We report the in-domain Mel distance performance of σ-VAE, along with the speaker similarity score and WER-C for tokenizer reconstruction and zero-shot speech generation on

Compression Ratio	Frame Rate	Latent Dimension	σ-VAE Reconstruction			Zero-Shot TTS	
			Mel Dist.↓	SIM↑	WER-C↓	SIM↑	WER-C↓
640×	37.5	16	0.929	0.866	1.9	0.655	1.4
1600×	15	16	1.080	0.700	2.7	0.545	1.6
1600×	15	32	0.950	0.870	1.9	0.661	1.5

Table 8: Ablation results of different σ-VAE token reduction ratios and latent dimensions. We report tokenizer reconstruction quality and zero-shot speech synthesis.

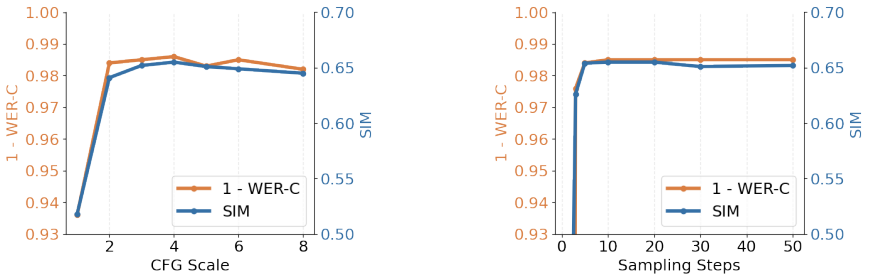


Figure 15: Ablation results of different CFG [24] scales and inference sampling steps. We report zero-shot speech synthesis results.

the LibriSpeech test-clean set. We use a 12-layer Transformer model for the TTS ablation studies. If the latent dimension remains unchanged, a higher token reduction ratio leads to a decrease in reconstruction performance and TTS speaker similarity score. However, by increasing the latent dimension of σ-VAE, we can compensate for this loss, allowing our model to use a higher token reduction ratio and a lower frame rate. Our model can generate 1 second of speech using significantly fewer autoregressive inference steps, compared to VALL-E 2.

CFG Scale Figure 15 illustrates the zero-shot speech synthesis results using classifier-free guidance [24] (CFG). When the CFG scale is set to 1, CFG is not applied. The use of classifier-free guidance significantly enhances the model's performance. Furthermore, we find that setting the CFG scale to 4 yields the best results.

Inference Sampling Step Figure 15 presents the results of zero-shot speech synthesis using different inference sampling steps of the diffusion head. We set the CFG scale to 4 for the ablation studies. More sampling steps require more inference time. We find that a sampling step of 3 yields competitive results, and increasing it to 5 leads to further improvement. When the sampling step is increased further, the results improve only slightly. Using a sampling step of 5 allows the model to achieve strong performance while maintaining a fast inference speed.

H Experiment Settings

H.1 Image Generation Scaling

Table 9 details the hyperparameters used for Section 5.1.2, where we compare the scalability properties of Diffusion Transformer [50] (DiT) and LatentLM. We describe the hidden dimension, the number of layers, and the number of heads for the models. Specifically, we follow [50] for the DiT architecture. In addition, we augment DiT with RMSNorm [84] and SwiGLU [55; 66]. To align the number of parameters, the FFN size for DiT is set to $\frac{8}{3}d$, while for LatentLM, it is set to $4d$. We train the models for 75,000 steps, which corresponds to approximately 120 epochs, to facilitate scaling comparisons.

	Size	Hidden Dim.	#Layers	#Heads	Learning Rate
Medium	455M	1024	24	16	8×10^{-4}
Large	1.03B	1536	24	12	3×10^{-4}
XL	1.82B	2048	24	16	2×10^{-4}
3B	3.68B	2560	32	20	1.6×10^{-4}

Table 9: Model size and hyperparameters used for the scaling experiments in Section 5.1.2.

H.2 Tokenizer Used for Image Generation

We train σ-VAE with perceptual loss [85; 30] and GAN loss [28], following [58; 17]. We initialize the encoder from the base-size BEiT-3 [81] checkpoint, and append a randomly initialized decoder.

Both encoder and decoder have 12 Transformer layers, totaling 172 million parameters. The image patch size is 16. We train tokenizers on the ImageNet training set [15] with 200 epochs. The batch size is 256. The optimizer is AdamW [46] with $\beta = (0.0, 0.99)$ and a learning rate of 3e-4. The weight decay is set to 0.01. We apply layer-wise learning rate decay [5] of 0.65 on the encoder.

H.3 Hyperparameters for Tokenizer Analysis

We follow the training recipes of [50] for DiT and LatentLM training. We set the hidden size to 1024. The number of layers is 24. Because LatentLM does not have AdaLN in the Transformer backbone, we adjust the intermediate FFN dimension (i.e., 2730 in DiT, and 4096 in LatentLM) to match their model size. The diffusion head has three layers of feedforward networks.

We use the AdamW [46] optimizer with $\beta = (0.9, 0.98)$. We use the cosine learning rate schedule with a maximal value of 1e-4 and 100 warmup steps. The weight decay is 0.1. We train models using a batch size of 256 for 200,000 steps, which is approximately equivalent to 40 epochs. We use the cosine beta schedule and v-prediction [60] for diffusion. We use DDPM [25] with 1000 steps during training. DPM-Solver [47; 48] with 20 steps is used during inference.

H.4 Multimodal Large Language Model

Training Data We use three types of data in the training stage: text-only data, image-text pair data, and interleaved text-image data. The mix-up ratio is 2:1:1. The data sources are described as follows:

- **Text-Only Data** The text training corpus follows [69], including Common Crawl, RefinedWeb [51], and StarCoder [38].
- **Image-Text Pairs** We follow [27; 52] to construct the paired data, i.e., English LAION-2B [63], LAION-400M [62], COYO-700M [7], and Conceptual Captions [65; 9].
- **Interleaved Image-Text Data** We use the same interleaved multimodal documents as in [27; 52]. The web pages are filtered from Common Crawl archives. The documents are interleaved with text and image.

Configuration We train a 1.3B-size Transformer as the backbone. We set the hidden size to 2048. The number of layers is 24. The training sequence length is 4096. We use `tiktoken-cl100k_base` as the text tokenizer. The batch size is 4M tokens. We use the AdamW [46] optimizer with $\beta = (0.9, 0.98)$. The maximal learning rate is 3e-4 with 500 warmup steps. The total schedule is set to 1T tokens. We train the model with 50k steps (i.e., 200B tokens) for comparison.

Table 10 details the hyperparameters employed for multimodal large language models, as described in Section 5.2.

Params	Values
Layers	24
Hidden size	2048
FFN size	6144
Vocab size	100,288
Heads	16
Adam β	(0.9, 0.98)
LR	3×10^{-4}
Batch size	4M
Warmup steps	500
Weight decay	0.1
Head Layers	6

Table 10: Hyperparameters used for multimodal large language models in Section 5.2.

H.5 TEXT-TO-SPEECH SYNTHESIS

H.5.1 TRAINING SETUP

Considering the variable-length nature of speech data, our speech tokenizer employs a convolutional architecture that supports streaming encoding and decoding. Specifically, σ-VAE for speech consists of a convolutional encoder, a continuous VAE quantizer, and a convolutional decoder. The encoder comprises multiple stages and downsampling layers organized in a hierarchical structure. Each stage includes several ConvNeXt blocks [45], where the original 2D convolution is replaced with 1D causal convolution. For token reduction ratios of 1600, 3200, and 6400, the downsampling layer reduces the input waveform by factors of [2, 4, 5, 5, 8], [4, 4, 5, 5, 8], and [4, 5, 5, 8, 8], respectively. Each time the downsampling layer is applied, the number of channels doubles, starting from 32 and increasing to 1024. The encoder contains around 120 million parameters in total. The decoder is a mirror of the encoder. As for the discriminator, we use the multi-period discriminator [34] and the complex STFT discriminator in DAC [36].

The hidden size of LatentLM is 1024, with 24 layers and 16 attention heads. The intermediate FFN dimension is set to 4096. The diffusion head contains three layers of feedforward networks. We use the same Transformer architecture as VALL-E 2 [11] for comparison. Table 11 lists the hyperparameters utilized for text-to-speech synthesis models, as discussed in Section 5.3.

Params	**Values**
Layers	24
Hidden size	1024
FFN size	4096
Heads	16
Adam β	(0.9, 0.98)
LR	7.5×10^{-4}
LR schedule	cosine
Batch size	5M
Warmup steps	10k
Training steps	100k
Weight decay	0.01
Head Layers	3

Table 11: Hyperparameters used for text-to-speech synthesis in Section 5.3.

H.5.2 TRAINING DATA

Tokenizer We train σ-VAE on a large and diverse corpus that includes speech, audio, and music. For speech, we use the clean speech subset from DNS Challenge 4 [16] and all splits from the Common Voice v7 dataset [2]. For audio, we use the FSD50K dataset [18], along with the balanced and unbalanced splits from AudioSet [19]. For music, we use the MUSDB dataset [54] and the Jamendo dataset [6]. All the data are resampled to 24kHz monophonic format.

TTS Model We utilize Libriheavy corpus [32] as training data following VALL-E 2 [11]. This corpus is a labeled version of the Librilight corpus [31], which features 50,000 hours of speech from approximately 7,000 different speakers, sourced from open-access English audiobooks associated with the LibriVox project[1].

H.5.3 EVALUATION METRICS

We evaluate our speech tokenizer using several automatic metrics, including: **Mel Distance**, which measures the distance between log Mel spectrograms as configured in DAC [36]; **PESQ-WB** [57], an intrusive metric for speech quality by comparing perceptual differences; **STOI** [70], which assesses speech intelligibility through short-time segment correlation; **VISQOL** [12], a perceptual

[1] https://librivox.org

quality metric based on spectral similarity; **UTMOS** [59], a reference-free mean opinion score for audio quality; **Speaker Similarity** (**SIM**), measured using WavLM-TDNN [10]; and **Word Error Rate** (**WER**), calculated using both Conformer-Transducer [21] (WER-C) and HuBERT-Large [26] (WER-H) models.