MELD: Mel-Spectrogram-Based Speech Language Modeling with Discrete Latent Variables

Anonymous ACL submission

Abstract

Recent speech language models rely on encoders that are optimized separately from autoregressive models. Since these encoders are unaware of the downstream objectives, the extracted representations may not be optimal for downstream tasks. To address this limitation, we introduce a discrete latent variable model on mel spectrograms that jointly optimizes the encoder and the speech language model. Joint optimization not only brings improvements over codec-based and other mel-spectrogram-based baselines on zero-shot Text-to-Speech (TTS) and Speech-to-Text (STT) tasks, but also effectively alleviates common issues in autoregressive mel-spectrogram modeling, such as prolonged silence generation and word omissions. Samples are available at https://anonymous/demo.

1 Introduction

Recent advancements in speech language modeling typically rely on a decoupled two-stage training strategy: first, a speech codec (Défossez et al., 2023; Kumar et al., 2023) or a variational autoencoder (VAE) is trained to encode speech signals via intermediate representations; second, an autoregressive model is trained on these representations (Wang et al., 2023; Chen et al., 2025; Défossez et al., 2024; Turetzky et al., 2024; Sun et al., 2024). While this two-stage approach allows each component to be independently optimized and reused across different speech systems, the encoder is typically unaware of the downstream tasks. It is therefore difficult for codec models or autoencoders to know what to represent and what can be discarded. In fact, discretized representations often fail to preserve task-relevant information unless they are jointly optimized with the downstream objectives (Yeh and Tang, 2024; Onda et al., 2025).

A compelling alternative is to jointly train the encoder and the autoregressive model on mel-spectrograms, motivated by the success of mel-spectrograms in speaker verification (Desplanques et al., 2020), speech recognition (Gulati et al., 2020) and speech generation (Ren et al., 2020). Autoregressive modeling of mel-spectrograms, however, has its difficulties. Generating speech autoregressively often gets trapped in stretched silence or produces constant artifacts (Tu et al., 2025; Song et al., 2025; Chen et al., 2024), especially with mel-spectrograms (Chen et al., 2023; Battenberg et al., 2025).

In this work, we introduce a Mel-Spectrogram-Based Discrete Latent Language Model (MELD) for autoregressive text-speech modeling. We distinguish our framework from "end-to-end" speech-to-speech systems, emphasizing the **joint optimization** of the autoregressive process that directly models spectrograms and text sequences. Specifically, we extend the generative process to a discrete latent space and a continuous mel-spectrogram space. With the discrete latent space, we make use of sampling methods over discrete distributions that have been shown to effectively suppress stretched silence (Chen et al., 2024, 2023; Song et al., 2025). At the same time, modeling the continuous mel-spectrogram space avoids potential information loss that can degrade STT performance.

We evaluate MELD against mel-spectrogram-based models like MELLE (Meng et al., 2025), which proposes a continuous Gaussian sampling space. We show that such sampling space is limited with MELLE's objective compared to a discrete latent space in MELD. Empirically, our reproduction of MELLE frequently produces prolonged silence, while the discrete sampling in MELD effectively suppresses this issue. On zero-shot TTS continuation benchmarks, we observe consistent improvements over MELLE and other codec-based baselines, including VALL-E (Wang et al., 2023). The advantage of joint optimization is even more pronounced for STT. Unlike input representations

such as dMel that are discretized independently of the STT objective, MELD is optimized directly from mel-spectrograms w.r.t. the STT objective, leading to lower word error rates.

Finally, we show that MELD opens new opportunities to improve joint TTS–STT modeling. Training the two tasks together is known to be challenging (Battenberg et al., 2025; Bai et al., 2024) for several reasons. For instance, an STT-only model does not necessarily need to preserve prosodic information that is crucial for zero-shot TTS. We show that MELD can, by design, learn both STT and TTS tasks within a single autoregressive model. In particular, the strength of joint optimization over mel-spectrograms results in significant improvements in STT compared to dMel (Bai et al., 2024).

2 Autoregressive learning objectives

We briefly review autoregressive modeling of mel-spectrograms conditioned on its transcription, and derive the learning objective of our discrete latent variable model. Let $y = (y_1, \ldots, y_M)$ be a byte-pair-encoding (BPE) token sequence (Sennrich et al., 2016) with each token from a text vocabulary $y_m \in \mathcal{V}_{\text{text}}$, $x = (x_1, \ldots, x_T)$ be a sequence of mel-frames where $x_t \in \mathbb{R}^{d_{\text{Mel}}}$. Consider an autoregressive model, the log likelihood of generating acoustic frames given the text factorizes

$$-\log p(x|y) = \sum_{t=1}^{T} -\log p(x_t|x_{<t}, y). \quad (1)$$

The conditional probability is recently modeled by a decoder-only Transformer. Because the model only learns to predict the next frame without knowing when to terminate generation, prior work introduces a jointly trained stop predictor (Wang et al., 2017; Meng et al., 2025). We will show that the stop predictor is unnecessary in our model.

A limitation of such models is that they typically predict deterministic frame estimates and lack a well-defined distribution for sampling. Generation without sampling tends to get stuck in infinite silence (Chen et al., 2024; Song et al., 2025), as commonly observed in sequence-to-sequence mel-spectrogram TTS models (Wang et al., 2017; Shen et al., 2018; Chen et al., 2023). There is also no finer control over the output distributions as in discrete sampling (Wang et al., 2023; Chen et al., 2024).

2.1 Generation with discrete latent variables

Inspired by the success of discrete sampling in speech language modeling (Wang et al., 2023), MELD extends the autoregressive process across a discrete latent space and a continuous mel-spectrogram space. Let $z = (z_1, \ldots, z_T)$ be the discrete latent variables with $z_t \in \mathcal{V}_{\text{latent}}$ and $\mathcal{V}_{\text{latent}} = \{1, \ldots, K\}$. The generative factorization over the next Mel frame x_t and latent variable z_t is

$$p(x_t, z_t|x_{<t}, y) = p(x_t|z_t, x_{<t}, y)p(z_t|x_{<t}, y), \quad (2)$$

where $x_{<t}$ and y represent historical frames and text tokens, respectively.

In (2), there are two key conditional independence assumptions. First, we assume the past frames $x_{<t}$ and text y provide sufficient information for the next latent z_t, making z_t conditionally independent of $z_{<t}$. Second, given z_t and $x_{<t}$, the next frame x_t is conditionally independent of $z_{<t}$. Importantly, conditioning the prediction of x_t solely on z_t is usually insufficient to capture the acoustic details for autoregressive zero-shot TTS, unless one expands the latent space K via residual vector quantization (RVQ) or introduces speaker embeddings. Conditioning on $x_{<t}$ and y alongside z_t enables accurate next-frame prediction while maintaining a compact discrete latent space. We visualize the generative process in Figure 1.

Directly maximizing the marginal log likelihood, i.e., $\log \sum_{z_t=1}^{K} p(x_t|z_t, x_{<t}, y)p(z_t|x_{<t}, y)$, is computationally intractable for large K. We thus introduce a variational framework to optimize its lower bound in the following section.

2.2 Variational next-frame prediction

To efficiently approximate the log likelihood, we introduce a variational distribution to draw quantized samples from the discrete latent space. We define $q(z|x, y) = \prod_t q(z_t|x_t)$, assuming frame-wise conditional independence to match the autoregressive generative factorization. The optimization objective under such assumption can be computed linear in T. The Variational Lower Bound (VLB) is derived as:

$$\mathcal{L}_{\text{VLB}} = \sum_{t=1}^{T} \Big[\text{KL}\big[q(z_t|x_t) \,\|\, p(z_t|x_{<t}, y)\big] - \mathbb{E}_{z_t \sim q}\big[\log p(x_t|z_t, x_{<t}, y)\big]\Big], \quad (3)$$

where the first term is the KL divergence between the frame-wise quantization network $q(z_t|x_t)$ and

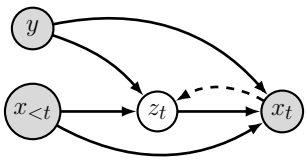

Figure 1: A graphical model of the generative process. We adopt the style of Kingma and Welling (2014), denoting the solid lines as the generative model $p(x_t|z_t, x_{<t}, y)p(z_t|x_{<t}, y)$, while the dashed line as the proposal distribution $q(z_t|x_t)$.

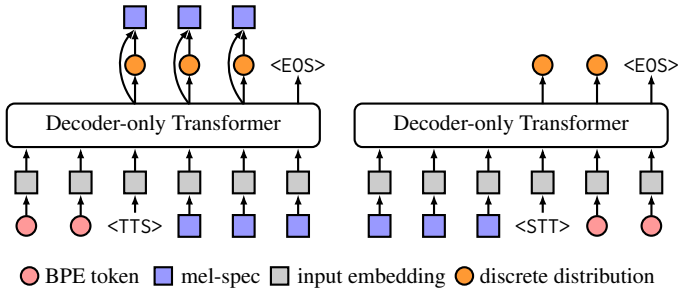

Figure 2: An illustration of two sequence prediction tasks controlled by two special tokens <TTS> and <STT>, respectively.

an autoregressive network $p(z_t|x_{<t}, y)$, the second term is the reconstruction loss of a reconstruction network $p(x_t|z_t, x_{<t}, y)$. The step-by-step derivation can be found in Appendix A.1.

2.3 Extension to speech-to-text

The connection between our variational objective and standard next-token prediction becomes clear by expanding the KL term in Equation (3) into cross-entropy and entropy terms:

$$\mathbb{E}_{z_t \sim q}\big[-\log p(z_t|x_{<t}, y) + \log q(z_t|x_t)\big]. \quad (4)$$

The cross-entropy term over the discrete latent variable z_t allows us to integrate STT into the same sequence modeling framework. To achieve this, we extend the target discrete space to a union token set $\mathcal{V} = \mathcal{V}_{\text{text}} \cup \mathcal{V}_{\text{latent}}$, where $\mathcal{V}_{\text{text}}$ represents the BPE tokens and $\mathcal{V}_{\text{latent}} = \{1, \ldots, K\}$ represents discrete speech latents.

The STT objective minimizes the cross entropy

$$\sum_{t=1}^{M} \mathbb{E}_{z_t \sim q}\big[-\log p(z_t|y_{<t}, x)\big], \quad (5)$$

where we sample z_t from $q(z_t|y_t) = \mathbb{1}_{z_t = y_t}$ and $y = (y_1, \ldots, y_M)$. We model the next-token distributions, $p(z_t|y_{<t}, x)$ in STT and $p(z_t|x_{<t}, y)$ in TTS, with the same autoregressive network.

Together, there are in total 3 special tokens as presented in Figure 2, alongside BPE tokens and K discrete latents: <TTS> for speech synthesis conditioning, <STT> for transcribing conditioning, and <EOS> for the end of sequence. <EOS> is merged into the discrete space, and jointly predicted with BPE tokens and discrete latents.

3 MELD parameterization and training

The feedforward process of MELD for TTS is visualized in Figure 3: A quantization network $q(z_t|x_t)$ first quantizes the next frame x_t. Next, an autoregressive network $p(z_t|x_{<t}, y)$ is learned to predict a sample z_t drawn from $q(z_t|x_t)$. Finally, a reconstruction network $p(x_t|z_t, x_{<t}, y)$ reconstructs the next mel frame. We now describe the parameterization of each component step by step.

3.1 Quantizing the next frame and sampling

As Figure 3 shown, we draw samples from $q(z_t|x_t)$ during training to serve both as targets for computing the KL term and as the latent embeddings for reconstructing mel frames described in Eq (3). To draw a sample that represents x_t well, we initialize the codebook with k-means centroids trained on mel-spectrograms. The variational distribution assigns a frame based on the Euclidean distance between x_t and each codeword entry,

$$q(z_t|x_t) = \frac{\exp\left(-\|x_t - c_{z_t}\|^2/\tau\right)}{\sum_{k=1}^{K} \exp\left(-\|x_t - c_k\|^2/\tau\right)}, \quad (6)$$

where a codeword c_k is the k-th column of a codebook $C \in \mathbb{R}^{d_{\text{mel}} \times K}$, and τ is the temperature. We set $\tau = 1$ in this work.

The distribution can be interpreted as the assignment probability under an isotropic Gaussian mixture model (GMM) with an isotropic covariance τI. When $\tau \to 0$, the distribution becomes minimization or hard assignment in k-means (Murphy, 2012; Kulis and Jordan, 2011). This parameterization is also known as *soft* vector quantization (Agustsson et al., 2017; Sønderby et al., 2017).

With the codebook being properly initialized, each quantized frame represents the original frame well with small distortion. We freeze the codebook throughout training and let the reconstruction network (Section 3.3) refine the codewords. This avoids potential challenges of training neural networks with vector quantization (Huh et al., 2023). The quantization network is never used during inference because the future frames are not accessible.

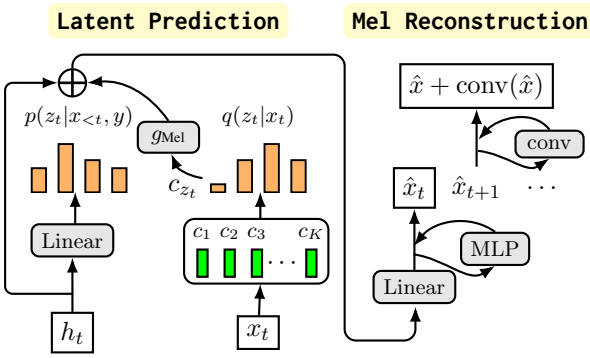

Figure 3: Variational autoregressive training process, where h_t summarizes $x_{<t}$ and y. The model is trained to minimize the KL divergence between q and p, and the MSE between x_t and $\hat{x}_t$, $\hat{x}_t + \text{conv}(\hat{x})_t$.

3.2 Autoregressive latent prediction

As shown in Figure 2, we apply a decoder-only Transformer to predict the next discrete latent variable z_t drawn from the variational distribution q, or the subsequent text token, i.e., the two prediction distributions $p(z_t|x_{<t}, y)$ and $p(z_t|y_{<t}, x)$. Text tokens and mel-spectrograms are mapped to d_{model}-dimensional embeddings before the Transformer. A lookup function $g_{\text{text}} : \mathcal{V}_{\text{text}} \to \mathbb{R}^{d_{\text{model}}}$ maps a text token y_m into a text embedding. We follow (Meng et al., 2025), using an encoder of 3-layer multilayer perceptron (MLP), $g_{\text{Mel}} : \mathbb{R}^{d_{\text{Mel}}} \to \mathbb{R}^{d_{\text{model}}}$ to encode each mel frame. Figure 3 (left) presents the feedforward process of latent prediction.

When modeling TTS and STT tasks together, the two distributions are modeled by a softmax over the discrete space, the disjoint vocabulary $\mathcal{V} = \mathcal{V}_{\text{text}} \cup \mathcal{V}_{\text{latent}}$, followed by a linear layer.

3.3 Spectrogram reconstruction

The reconstruction network is similar to that of Tacotron 2 (Shen et al., 2018), consisting of a linear layer with residual connections of a 3-layer multilayer perceptron and a convolutional module. The prediction consists of two stages. In Figure 3, an output embedding h_t and a codeword c_{z_t} sampled from q are mapped to a mel-spectrogram space in an autoregressive manner. An MLP layer is used to predict the residual and added the output of a linear layer to produce $\hat{x}_t$. We denote this process as

$$\begin{aligned} \hat{x}_t &= \text{SpecNet}(h_t + g_{\text{Mel}}(c_{z_t})) \text{ where,} \\ z_t &\sim q(z_t|x_t) \text{ during training,} \\ z_t &\sim p(z_t|x_{<t}, y) \text{ during inference.} \end{aligned} \tag{7}$$

During inference, we sample z_t from the predicted discrete distributions. To ensure the discrete signals z_t will not get overlooked, we embed the codevector c_{z_t} with g_{Mel} that is also used to embed input mel-spectrograms.

After autoregressive decoding, the predicted spectrogram is refined by a convolutional network as shown in Figure 3 similar to Shen et al. (2018). By assuming the reconstruction likelihood as a product of two isotropic Gaussian distributions, i.e., $\mathcal{N}(x_t; \hat{x}_t, I)\mathcal{N}(x_t; \hat{x}_t + \text{conv}(\hat{x})_t, I)$, the reconstruction term $\mathbb{E}_{z_t \sim q}\big[\log p(x_t|z_t, x_{<t}, y)\big]$ in Eq (3) is the mean squared error (MSE):

$$\begin{aligned} \text{MSE}(x, \hat{x}) = \frac{1}{T}\sum_{t=1}^{T}\big[\|x_t - \hat{x}_t\|_2^2 + \\ \|x_t - (\text{conv}(\hat{x})_t + \hat{x}_t)\|_2^2\big], \end{aligned} \tag{8}$$

where $\text{conv}(\cdot)$ stands for convolutional network.

3.4 Slowness penalty

To promote diverse generations, prior work has considered penalizing similarly synthesized acoustic frames (Meng et al., 2025). We introduce a slowness penalty to penalize slow changes in the predicted acoustic frames $\hat{x}$,

$$\mathcal{L}_{\text{slow}} = -\frac{1}{T-1}\sum_{t=1}^{T-1}\|\hat{x}_t - \hat{x}_{t+1}\|^2. \tag{9}$$

4 Related work

The proposed approach differs from others in terms of (1) the dependency on a pre-trained encoder (or a discretization step) to derive input representations, and (2) how the next prediction is sampled. We contrast MELD to prior work on autoregressive speech modeling in this section.

Codec-based LMs. Recent advancements using discrete codecs derived from residual vector quantization (RVQ) have demonstrated success in zero-shot TTS (Chen et al., 2025, 2024). The proposed approach does not apply RVQ to quantize mel-frames, since there are complications of using RVQ codecs. In RVQ, each frame consists of multiple levels of codes that follow a fixed hierarchical order (Zeghidour et al., 2021; Kumar et al., 2023). Previous approaches predict this hierarchy using non-autoregressive models (Chen et al., 2025) or rearranging codes at each level with a delay pattern (Kharitonov et al., 2022; Copet et al., 2023; Défossez et al., 2024). Our approach does not have such complications. Moreover, codec-based LMs predict N codebooks per time step. This increases

output activation memory by a factor of N, resulting in higher GPU memory usage than MELD, which predicts a single latent token. The limitation is also noted in Chou et al. (2025).

Discretized mel-spectrograms. To apply mel-spectrograms for next-token prediction, `dMel` (Bai et al., 2024) discretizes mel-spectrograms with finite scaler quantization (Mentzer et al., 2023). We show that discretizing mel-spectrograms before autoregressive modeling of speech is not necessary. In fact, such discretization limits the performance of models on STT and joint TTS-STT modeling, as we will demonstrate in Sec 5.5 and 5.6.

MELLE. MELLE (Meng et al., 2025) is a mel-spectrogram-based model that draws samples from a single Gaussian autoregressively. In contrast, we predict the next frame by sampling over a categorical distribution mapped to K isotropic Gaussians in a GMM (Sec 3.1). We will show that this discrete latent space effectively suppresses stretched silence in synthesized speech in Sec 5.3. In addition, the proposed objective forms a variational lower bound, but the objective of MELLE is designed in an ad-hoc manner (Appendix A.6). MELD does not need a stop predictor and can be easily extended to STT task. MELLE, however, focuses on speech generation and relies on voice activity detection (VAD) to avoid infinite silence generation. We do not depend on such pre-processing step.

End-to-end speech language modeling. Although VoxCPM (Zhou et al., 2025) is characterized as an "end-to-end" speech language model, it remains dependent on a pre-trained VAE to extract input representations. In contrast, MELD directly models mel-spectrograms without disconnected gradient flow, the encoder g_{Mel} is jointly optimized with the autoregressive model.

5 Experiments

We conduct experiments to study: (1) whether discrete latent variables improve autoregressive mel-spectrogram modeling (Sec 5.3); (2) if optimizing directly over mel-spectrograms improves STT compared to two-stage approaches (Sec 5.5); and (3) whether the strength of joint optimization transfers effectively to joint TTS-STT modeling (Sec 5.6).

5.1 Experimental Settings

We train all autoregressive models on the 960-hour subset of LibriSpeech (Panayotov et al., 2015), referred to as LS960. If not otherwise mentioned, we train all models using 12-layer Transformers with 16 attention heads, an embedding dimension (i.e., d_{model}) of 1024, a feed-forward network dimension of 4096, and a dropout rate of 0.2. The resulting model parameters are roughly 200M. We include more training details in Appendix A.2.

The choice of text tokens is critical to extend speech language models from TTS to STT. Unlike prior speech language models on TTS that heavily trained on phonemic transcriptions (Chen et al., 2025, 2024; Song et al., 2025; Meng et al., 2025) on LS960, we adopt BPE tokenization with a vocabulary size $|\mathcal{V}_{\text{text}}|$ of 4096 for both TTS and STT modeling. We compare two types of speech representations for speech language models — discrete speech codecs, due to their prominence in speech language modeling, and mel-spectrograms.

MELD implementation details. Log mel-spectrogram configurations are detailed in Appendix A.3. We encode mel-spectrograms and codevectors with g_{Mel}, a 3-layer MLP encoder with a network dimension of 1024. Each layer is followed by a GELU activation function then a dropout of 0.5. We follow the common practice, applying test-time dropout in g_{Mel} during the inference of zero-shot TTS (Wang et al., 2017; Meng et al., 2025).[1]

We set the codebook size K to 8192 and hold it fixed after initializing it with k-means. The convolutional layers are based on the postnet of Tacotron 2 (Shen et al., 2018), detailed in Appendix A.5. Mel-spectrograms are converted into waveforms using a HiFi-GAN vocoder (Kong et al., 2020) pre-trained on 585-hour LibriTTS.[2] Since our models are trained on normalized mel-spectrograms, the predicted frames are rescaled before the vocoder.

5.2 A summary of model variants

We summarize main model variants we are going to compare in Table 1. We apply top-p sampling with $p = 0.9$ and top-k sampling with $k = 60$ to sample from discrete distributions. Moreover, we impose a score of -1 to the codes if they were sampled in the previous top-p candidates, denoted as repetition penalty.

Codec-based baselines. We implement a variant of our model based on speech codecs, labeled as

[1] We find this necessary to mitigate the training-inference mismatch.

[2] https://huggingface.co/mechanicalsea/speecht5-tts.

Model	Main Loss	Aux. Loss	Text	Speech	Vocoder	Sampling
Codec-LM	cross entropy	-	BPE	DAC code	DAC decoder	Top-k / Top-p
Mel-LM	MSE (8) $+\mathcal{L}_{\text{stop}}$	$\mathcal{L}_{\text{slow}}$ (9)	BPE	Mel	HiFi-GAN	No sampling
MELLE	MSE (8) + $\mathcal{L}_{\text{stop}}$	$\mathcal{L}_{\text{KL}} + \mathcal{L}_{\text{slow}}$ (9)	BPE	Mel	HiFi-GAN	Gaussian
MELD (proposed)	$\mathcal{L}_{\text{VLB}}$ (3)	$\mathcal{L}_{\text{slow}}$ (9)	BPE	Mel	HiFi-GAN	Top-k / Top-p

Table 1: Summary of models for zero-shot TTS task w.r.t. their losses, the type of speech input and output, and the sampling methods, where $\mathcal{L}_{\text{stop}}$ is the binary cross-entropy loss of the stop predictor (Wang et al., 2017).

Codec-LM in Table 1. We train Codec-LM by extracting speech codes using a DAC encoder consists of 12 codebooks with a codebook size of 1024 (Kumar et al., 2023).[3] Codebook embeddings are randomly initialized. The DAC encoder encodes waveforms at 16kHz to a sequence of 12-level codecs at 50 Hz after RVQ (Zeghidour et al., 2021). To capture the dependency of residuals, we introduce a delay of one code to each level of the RVQ codecs (Kharitonov et al., 2022; Copet et al., 2023; Défossez et al., 2024). The delay pattern introduces a total delay of $(12-1) \times 20$ ms. We use 12 linear heads to predict 12 levels of RVQ codes each time step for TTS. The generation is terminated once <EOS> is predicted in one of the heads. The predicted codes are passed to the DAC decoder to synthesize waveforms. To perform STT task, another linear head is used to predict BPE tokens.

Mel-spectrogram-based baselines. Two variants based on mel-spectrograms are implemented by holding most of the architectures fixed. First, we remove the latent space and consider a decoder-only variant of Tacotron 2 (Shen et al., 2018), denoted as Mel-LM. The model is trained with MSE loss (8). No sampling is used for this variant. Second, we reproduce MELLE (Meng et al., 2025) following their settings. MELLE differs from Mel-LM in terms of an additional KL term that regularizes Transformer outputs to a Gaussian prior, and samples predictions from a Gaussian distribution. Both Mel-LM and MELLE rely on another linear layer to predict when to terminate, while in MELD, <EOS> is part of the discrete code vocabulary. We apply a weight of 0.2 for slowness penalty $\mathcal{L}_{\text{slow}}$ to all mel-spectrogram based approaches. We detail the re-production of MELLE in Appendix A.6.

5.3 Zero-shot text-to-speech

We begin with zero-shot TTS task. Given the text transcription and the first 3 seconds of an utterance as the prompt, we ask the model to continue the utterance with the hope of preserving speaker characteristics. We follow the evaluation protocol in previous work (Wang et al., 2023; Borsos et al., 2023), using audio samples with the duration between 4 seconds and 10 seconds from the LibriSpeech test-clean subset (2.2 hours). We repeat speech generation 3 times and report the averaged scores.

Model	Text	Speech	Freq	WER↓	SIM↑
Ground truth	–	–	–	2.2 / 1.6	0.925
DAC	–	–	50.0	2.2 / 1.6	0.922
HiFi-Gan	–	–	62.5	2.2 / 1.6	0.903
VALL-E ♣	Phn	Encodec	75.0	- / 5.0	0.868
Codec-LM	Phn	DAC	50.0	5.7 / 4.7	0.872
Codec-LM	BPE	DAC	50.0	5.3 / 4.8	0.864
MELD	BPE	Mel	62.5	2.4 / 1.9	0.872
MELD	BPE	Mel	31.3	2.5 / 1.9	0.855

Table 2: Model comparisons between codec-based approaches and ours. All models are trained on LS960. ♣ denotes the VALL-E results quoted from Song et al. (2025), which is trained on Encodec codecs derived from 24kHz audio samples.

We evaluate the generated samples with subjective and objective metrics. For subjective metrics, two mean opinion scores (MOS) are evaluated — Similarity MOS (SMOS) and Comparison MOS (CMOS) (ITU; Cooper et al., 2024). We detail the subjective assessment in Appendix A.4. Content fidelity is evaluated by passing synthesized speech through a Conformer-Transducer and a Whisper-large model(Radford et al., 2023) to compute word error rates (WERs) against the ground truth. We also evaluate speaker similarity between the given speech prompt (the first 3 seconds) and the generated speech. We follow Song et al. (2025), computing the cosine similarity (SIM) over speaker embeddings extracted from a WavLM-finetuned model,[4] in which values above 0.86 are recognized as the same speaker. Importantly, we use it solely as a proxy for accessing speaker similarity.

[3] https://github.com/descriptinc/descript-audio-codec

[4] https://huggingface.co/microsoft/wavlm-base-sv.

Our Codec-LM is on par with VALL-E. We first compare our Codec-LM with VALL-E (Song et al., 2025), both are trained on LS960. For a closer comparison, we provide a Codec-LM baseline with phoneme tokens extracted from Park and Kim (2019). In Table 2, our Codec-LM is on par with VALL-E in both content and speaker evaluations, despite VALL-E's two Transformers and higher frequency codecs derived from Encodec (Kumar et al., 2023). We observe only a small decrease in speaker similarity when switching from phoneme tokens to BPE tokens in our Codec-LM.

Joint optimization over mel-spectrograms v.s. two-stage approaches. We then compare MELD with Codec-LM. The two share the same sampling strategies (Table 1). In particular, we find the repetition penalty effectively reduces the occurrence of long pauses and silence. MELD obtains up to 2.3% lower WERs compared to codec-based approaches, while maintaining comparable speaker similarity. The improvements over Codec-LM also reflect on subjective evaluation in Table 5. When we reduce the frequency to 31.3 by concatenating every 2 frames, we observe a decrease in speaker similarity, showing the importance of frame rates to preserve speaker similarity.

Sampling from a discrete latent space is preferred over a single Gaussian. Regarding other approaches based on mel-spectrograms in Table 3, they fall behind in both WERs and speaker similarity. Although Mel-LM and MELLE have better WERs than Codec-LM in Table 2, the similarity scores fall behind by 0.047. We consistently observe long pauses in samples generated by Mel-LM and even MELLE despite the application of a slowness penalty during training. While MELLE samples the next frame from a Gaussian latent space (Meng et al., 2025), the latent space at t is constrained to the choice of the prior $\mathcal{N}(x_t, I)$, due to the nature of the reverse KL divergence (Kingma et al., 2019). Consequently, when a Gaussian is predicted to represent silence, it becomes unlikely to sample anything other than silence. In contrast, the discrete latent space includes codes corresponding to both silent and non-silent frames, enabling MELD to escape the silence loop.

Model	$\mathcal{L}_{\text{slow}}$	Freq	WER↓	SIM↑
Mel-LM	✓	62.5	4.7 / 4.2	0.825
MELLE	✓	62.5	4.8 / 4.2	0.826
MELD	✓	62.5	2.4 / 1.9	0.872
MELD	✗	62.5	6.0 / 3.7	0.862
MELD	✓	31.3	2.5 / 1.9	0.855

Table 3: Model comparisons of mel-spectrogram based approaches. Predicted mel-spectrograms are converted to waveforms with the same HiFi-GAN.

	WER↓	SIM↑	Mins	S / D / I
Ground truth	2.2 / 1.6	0.925	131.8	0 / 0 / 0
MELD	2.4 / 1.9	0.872	129.3	1231 / 254 / 263
w/o rep penalty	3.1 / 2.6	0.869	137.4	1238 / 405 / 255
w/o z_t	52.3 / 51.7	0.520	>200	

Table 4: The effectiveness of the discrete latent space during inference stage. **Mins** shows the total duration of synthesized samples. **S / D / I** is the WER breakdown, representing substitution, deletion and insertion errors.

5.4 Effectiveness of the discrete latent space

Our objective admits collapsed solutions, e.g., constant z_t. However, we empirically observe that the method does not converge to such cases. To verify this, we set c_{z_t} to zero when generating x_t in (7). As shown in Table 4, we observe a substantial degradation of both WERs and speaker similarity when information from the discrete latents is removed, indicating that the discrete samples are essential to infer the next frames. Unlike sampling from a Gaussian (Meng et al., 2025), we can enhance sample diversity through repetition penalty that discourages repeated generations. Removing the repetition penalty increases WER by 0.7% with a slight decrease of speaker similarity, while the synthesized samples are noticeably longer (∼6 mins) than the ground truth, suggesting excessive silence generation (Table 4). Repetition penalty also effectively reduces deletion errors, which suggests it helps suppress word omissions.

5.5 Speech-to-text with joint optimization

Prior work on mel-spectrogram language modeling only considers zero-shot TTS task (Meng et al., 2025). We extend MELD to STT task by putting speech input ahead of the BPE tokens as shown in Figure 2 (right). We use the same decoder-only architecture in TTS for STT task but with two changes. First, we deactivate dropout in g_{Mel} since we find it leads to unstable STT training. Second, we use SpecAugment (Park et al., 2019), applying two masks on frequency subbands with maximum frequency bands of 30 and 10 masks at frame level with maximum consecutive frames of 50. The number of frames per mask at frame level is clipped

	SMOS↑	CMOS↑
Ground Truth	4.11 ± 0.10	0.27
Codec-LM	3.72 ± 0.15	-0.31
MELD (joint)	3.81 ± 0.12	-0.20
MELD	3.89 ± 0.06	0.0

Table 5: Subjective evaluation was carried out on 43 samples with 40 speakers from the LibriSpeech test-clean set. CMOS is computed w.r.t. MELD.

			dev		test	
	Size	Hrs	clean	other	clean	other
Moshi ♣	7B	7M	-	-	5.8	-
dMel (ASR) ♣	258M	960	3.8	10.3	4.2	10.4
Codec-LM	200M	960	6.1	16.5	6.4	16.4
w/o codebook init	200M	960	>100	>100	>100	>100
MELD	200M	960	4.0	9.8	4.2	10.0
w/o SpecAug	200M	960	4.3	12.5	4.5	12.5
MELD	260M	960	3.6	9.0	3.5	9.2

Table 6: Performance of decoder-only Transformers on LibriSpeech dev/test sets using LS960 hours. ♣ numbers quoted from Défossez et al. (2024) and Bai et al. (2024), respectively.

by 0.1 times the number of frames. Note that, our MELD baseline on STT is equivalent to adding a linear head to MELLE to predicts BPE tokens.

Discretized representations fail to preserve task-specific information. Table 6 reports decoder-only STT results using beam search (a beam size of 5). We first note that, initializing codebooks for codec-based STT is critical for better convergence and WERs as noted in Dhawan et al. (2024). We therefore initialize and freeze the codebooks from a pre-trained TTS-only Codec-LM. As shown in Table 6, using mel-spectrograms consistently improves over Codec-LM for up to 6% in test other, showing the strength of end-to-end training in our framework. We further compare MELD with dMel (ASR) that is trained on discretized mel-spectrograms with a 18-layer Transformer decoder and the same SpecAugment settings (Bai et al., 2024). Our approach outperforms dMel on decoder-only ASR model in dev-other (0.5%) and both test-other (0.4%). The gap is more pronounced with a larger Transformer (260M).

5.6 TTS-STT modeling with MELD

Finally, we explore the joint modeling of both STT and TTS tasks with a 12-layer Transformer decoder, which enables us to perform either task given a speech or text prompt. We apply a linear layer after the Transformer to jointly predict the discrete latent variables and BPE tokens and <EOS>.

		TTS		STT	
	Model	WER↓	SIM↑	clean	other
Moshi	joint	-	-	5.8	-
dMel (ASR)	separate	-	-	4.2	10.4
dMel (ASR-TTS)	joint	-	-	7.5	15.3
Codec-LM	separate	5.3 / 4.8	0.864	6.4	16.4
MELD	separate	2.4 / 1.9	0.872	4.2	10.0
MELD	joint	2.8 / 2.2	0.870	4.9	12.1

Table 7: Results of MELD on joint TTS and STT modeling (joint), where we present STT results on test-clean/other. We compare the joint model with models trained on the separate tasks (separate).

Joint optimization reduces the gap between TTS-STT modeling and task-specific modeling. To enable both tasks, we adopt a simple multitask training strategy. We initialize the training with TTS task for 80k steps, which is sufficient for a TTS model to converge. Then we continue the training with both tasks by mixing speech-text sequences for two modes equally in a training batch. Also, we use a dropout of 0.5 on TTS during training and inference, while deactivating the dropout in STT mode. We apply SpecAugment to STT input stream with two masks on frame level and two on frequency subbands, since we find the original configuration too aggressive and leads to worse WERs. Table 7 reports the comparisons of joint models (joint) and task-specific models (separate). MELD reduces the WER of dMel (ASR-TTS) on STT by 3.2% on test-other. It also improves Codec-LM trained on separate tasks in both objective (Table 7) and subjective (Table 5) metrics. The results show that MELD not only supports high-quality zero-shot TTS but it can also perform STT task.

6 Conclusion

In this work, we introduce MELD that predicts mel-spectrogram frames through discrete latent variables. MELD presents a jointly optimized framework capable of TTS, STT, and TTS-STT modeling within a single model. Our results show the effectiveness of joint optimization in MELD, which improves STT performance over codec-based and discretized mel-spectrogram baselines while maintaining strong zero-shot TTS quality. These findings show that mel-spectrograms, combined with discrete latent modeling, provide an effective alternative to two-stage speech language modeling.

7 Limitations

We discuss several limitations of the proposed framework. First, we acknowledge that it is hard to have fully fair comparisons between codec-based and mel-spectrogram-based methods. Although all models share the same Transformer decoder architecture, they differ in how speech representations are mapped to waveforms, using a DAC decoder for codecs or a HiFi-GAN for mel-spectrograms.

Second, although we present several advantages relative to MELLE such as sampling discrete latents, we are not able to fully reproduce the results reported in MELLE (Meng et al., 2025) with LS960, while closely following their training configuration. We offer several possible explanations when compared to MELLE. We also detail our reproduction in Appendix A.6. We acknowledge that certain implementation details may have been overlooked during reproduction. For example, we suspect that the use of a VAD pre-processing step may be critical for achieving higher-quality samples, but not emphasized in the original work.

Last, we mainly focus on speech language models that model the conditional distributions of output speech (for TTS) and text (for STT). We acknowledge that there are other speech tasks of interest to the community, such as question answering and speech translation (Arora et al., 2025). An immediate next step is to explore the proposed model on more speech tasks.

7.1 Ethical considerations

This work is intended to advance fundamental research in speech language modeling using mel-spectrograms and discrete latent variables. All experiments are conducted on LibriSpeech, a publicly available dataset, used under the license of CC by 4.0. Speakers are anonymized using numeric IDs. Their use complies with the original dataset license and intended research purposes.

Potential ethical risks associated with this work include the misuse of speech generation methods, such as voice cloning. While the proposed models can perform zero-shot TTS, they are trained only on read speech. However, we acknowledge that the models can be potentially adapted to unseen speakers for realistic speech generation. It is important for such systems to incorporate protocols to ensure speaker approval, regarding what data will be collected, how it will be used.

References

ITU-T recomendation P.808: Subjective evaluation of speech quality with a crowdsourcing approach.

Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. 2017. Soft-to-hard vector quantization for end-to-end learning compressible representations. *NIPS*.

Siddhant Arora, Kai-Wei Chang, Chung-Ming Chien, Yifan Peng, Haibin Wu, Yossi Adi, Emmanuel Dupoux, Hung-Yi Lee, Karen Livescu, and Shinji Watanabe. 2025. On the landscape of spoken language models: A comprehensive survey. *arXiv preprint arXiv:2504.08528*.

Richard He Bai, Tatiana Likhomanenko, Ruixiang Zhang, Zijin Gu, Zakaria Aldeneh, and Navdeep Jaitly. 2024. dMel: Speech tokenization made simple. *arXiv preprint arXiv:2407.15835*.

Eric Battenberg, RJ Skerry-Ryan, Daisy Stanton, Soroosh Mariooryad, Matt Shannon, Julian Salazar, and David Teh-Hwa Kao. 2025. Robust and unbounded length generalization in autoregressive transformer-based text-to-speech. In *NAACL*.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and 1 others. 2023. Audiolm: a language modeling approach to audio generation. *IEEE/ACM transactions on audio, speech, and language processing*.

Li-Wei Chen, Shinji Watanabe, and Alexander Rudnicky. 2023. A vector quantized approach for text to speech synthesis on real-world spontaneous speech. In *AAAI*.

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

Sanyuan Chen, Chengyi Wang, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, and 1 others. 2025. Neural codec language models are zero-shot text to speech synthesizers. *IEEE Transactions on Audio, Speech and Language Processing*.

Ju-Chieh Chou, Jiawei Zhou, and Karen Livescu. 2025. Flow-slm: Joint learning of linguistic and acoustic information for spoken language modeling. In *ASRU*.

Erica Cooper, Wen-Chin Huang, Yu Tsao, Hsin-Min Wang, Tomoki Toda, and Junichi Yamagishi. 2024. A review on subjective and objective evaluation of synthetic speech. *Acoustical Science and Technology*.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2023. Simple and controllable music generation. *NeurIPS*.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. 2024. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*.

Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. 2020. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. *Interspeech*.

Kunal Dhawan, Nithin Rao Koluguri, Ante Jukić, Ryan Langman, Jagadeesh Balam, and Boris Ginsburg. 2024. Codec-asr: Training performant automatic speech recognition systems with discrete speech representations. In *Interspeech*.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2023. High fidelity neural audio compression. *TMLR*.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and 1 others. 2020. Conformer: Convolution-augmented transformer for speech recognition.

Bing Han, Long Zhou, Shujie Liu, Sanyuan Chen, Lingwei Meng, Yanming Qian, Yanqing Liu, Sheng Zhao, Jinyu Li, and Furu Wei. 2024. Vall-E R: Robust and efficient zero-shot text-to-speech synthesis via monotonic alignment. In *NeurIPS*.

Minyoung Huh, Brian Cheung, Pulkit Agrawal, and Phillip Isola. 2023. Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *ICML*. PMLR.

Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu Anh Nguyen, Morgane Riviere, Abdelrahman Mohamed, Emmanuel Dupoux, and Wei-Ning Hsu. 2022. Text-free prosody-aware generative spoken language modeling. In *ACL*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

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

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *NeurIPS*.

Brian Kulis and Michael I. Jordan. 2011. Revisiting k-means: New algorithms via bayesian nonparametrics. In *ICML*.

Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. 2023. High-fidelity audio compression with improved rvqgan. *NeurIPS*.

Lingwei Meng, Long Zhou, Shujie Liu, Sanyuan Chen, Bing Han, Shujie Hu, Yanqing Liu, Jinyu Li, Sheng Zhao, Xixin Wu, Helen M. Meng, and Furu Wei. 2025. Autoregressive speech synthesis without vector quantization. In *ACL*.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. 2023. Finite scalar quantization: Vq-vae made simple. In *ICLR*.

Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.

Kentaro Onda, Yosuke Kashiwagi, Emiru Tsunoo, Hayato Futami, and Shinji Watanabe. 2025. Differentiable k-means for fully-optimized discrete token-based asr. In *Interspeech*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Interspeech*.

Kyubyong Park and Jongseok Kim. 2019. g2pe.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *ICML*.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2020. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *ICLR*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, and 1 others. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *ICASSP*.

Casper Kaae Sønderby, Ben Poole, and Andriy Mnih. 2017. Continuous relaxation training of discrete latent variable image models. In *Beysian DeepLearning workshop, NIPS*.

Yakun Song, Zhuo Chen, Xiaofei Wang, Ziyang Ma, and Xie Chen. 2025. Ella-v: Stable neural codec language modeling with alignment-guided sequence reordering. In *AAAI*.

Yutao Sun, Hangbo Bao, Wenhui Wang, Zhiliang Peng, Li Dong, Shaohan Huang, Jianyong Wang, and Furu Wei. 2024. Multimodal latent language modeling with next-token diffusion. *arXiv preprint arXiv:2412.08635*.

Zehai Tu, Guangyan Zhang, Yiting Lu, Adaeze Adigwe, Simon King, and Yiwen Guo. 2025. Enabling beam search for language model-based text-to-speech synthesis. In *ICASSP*.

Arnon Turetzky, Nimrod Shabtay, Slava Shechtman, Hagai Aronowitz, David Haws, Ron Hoory, and Avihu Dekel. 2024. Continuous speech synthesis using per-token latent diffusion. *arXiv preprint arXiv:2410.16048*.

Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, and 1 others. 2023. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, and 1 others. 2017. Tacotron: Towards end-to-end speech synthesis. In *Interspeech*.

Sung-Lin Yeh and Hao Tang. 2024. Estimating the completeness of discrete speech units. In *2024 IEEE Spoken Language Technology Workshop (SLT)*.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Yixuan Zhou, Guoyang Zeng, Xin Liu, Xiang Li, Renjie Yu, Ziyang Wang, Runchuan Ye, Weiyue Sun, Jiancheng Gui, Kehan Li, and 1 others. 2025. Voxcpm: Tokenizer-free tts for context-aware speech generation and true-to-life voice cloning. *arXiv preprint arXiv:2509.24650*.

A Appendix

A.1 Variational Lower Bound

To derive the variational lower bound we require the following two factorizations

$$q(z|x,y) = \prod_t q(z_t|x_t)$$

$$p(x,z|y) = \prod_t p(x_t, z_t|x_{<t}, z_{<t}, y)$$

$$= \prod_t p(x_t|x_{<t}, z_t, y)p(z_t|x_{<t}, y),$$

the first equation is the proposal distribution with frame-wise dependency, while last line is due to the same assumptions we make in Section 2.1, $z_t \perp z_{<t} \mid x_{<t}, y$ and $x_t \perp z_{<t} \mid z_t, x_{<t}, y$.

With these two factorizations,

$$\begin{aligned}
&-\log p(x|y) \\
&= -\log \sum_{z_1,\ldots,z_T} p(x,z|y) \\
&= -\log \sum_{z_1,\ldots,z_T} \prod_t q(z_t|x_t) \frac{p(x,z|y)}{\prod_t q(z_t|x_t)} \\
&= \sum_{t=1}^{T} -\log \mathbb{E}_{z_t\sim q}\left[\frac{p(x_t|x_{<t}, z_t, y)p(z_t|x_{<t}, y)}{q(z_t|x_t)}\right] \\
&\leq \sum_{t=1}^{T} -\mathbb{E}_{z_t\sim q}\left[\log \frac{p(x_t|x_{<t}, z_t, y)p(z_t|x_{<t}, y)}{q(z_t|x_t)}\right] \\
&= \sum_{t=1}^{T} \Big[\mathrm{KL}\big[q(z_t|x_t) \,\|\, p(z_t|x_{<t}, y)\big] \\
&\qquad - \mathbb{E}_{z_t\sim q}\big[\log p(x_t|z_t, x_{<t}, y)\big]\Big], \quad (10)
\end{aligned}$$

we obtain the lower bound by Jensen's inequality. We arrive at the variational lower bound of the conditional log likelihood (3).

A.2 Training configuration

We train all models for 200k steps on 16 NVIDIA V100 (16GB) GPUs with the Adam optimizer, using a maximum of 50k frames per batch and a gradient clip of 10. The learning rate is linearly warmed up for 1k steps to 5×10^{-4}, held constant for 100k steps, and then linearly decayed over the final 100k steps. We also experiment with longer training steps, and we do not see notable improvements.

A.3 Mel-spectrograms

In line with the mel-spectrogram settings from the pre-trained HiFi-GAN (Kong et al., 2020), we extract 80-dimensional log Mel-spectrograms with a frequency of 62.5 Hz (a frame shift of 16 ms), 64 ms frame length, Hann windowing, 1024-point Fourier transform. We apply mel-filterbanks with the frequency range of 80 Hz to 7600 Hz. Mel-spectrograms are then normalized with global mean and variance computed from the training data.

A.4 Subjective evaluation

We use SMOS to evaluate the speaker similarity between the 3-second speech prompt and the generated speech. We use CMOS to evaluate the comparative naturalness of the synthesized speech relative to the synthesized speech from the proposed approach MELD. We recruited listeners via Amazon Mechanical Turk, a crowdsourcing platform. Each

Score	Description
5	Extremely similar
4.5	
4	Very similar
3.5	
3	Moderately similar
2.5	
2	Slightly similar
1.5	
1	Not similar at all

Table 8: Similarity rating scale used in SMOS.

Score	Description
3	A is much more natural than B
2	A is more natural than B
1	A is slightly more natural than B
0	They are about the same
-1	B is slightly more natural than A
-2	B is more natural than A
-3	B is much more natural than A

Table 9: CMOS rating scale for naturalness comparison.

screen contains two subtasks that are evaluated by 5 different listeners.

For each screen, listeners are presented with two samples for comparison and one sample as a reference. We prepare 43 tests across 40 speakers in Librispeech test-clean set. The first task (SMOS) is to rate the similarity of candidates (A and B) to the reference. The scale and the labels is defined in Table 8. Participants are instructed as follows:

- Rate how similar each sample is to the reference (not between A and B).
- Use the first 3 seconds of A and B as the reference and judge the similarity based on voice, speaking style, emotion, and audio quality.

For the second task (CMOS), the rating scale is defined in Table 9 based on the following instruction:

- Compare which candidate sounds more natural (human-like).

A total of 30 listeners participated, with each screen receiving 15–20 scores.

A.5 Postnet

The postnet consists of a stack of 3 convolutional layers each containing 512 filters with shape 5×1 with batch normalization, followed by a $\tanh$ activation on every except the final layer. The receptive filed is therefore 5×16 ms.

A.6 MELLE reproduction

The original objective of MELLE (Meng et al., 2025) consists of four terms, a regression Loss $\mathcal{L}_{\text{reg}}$, KL Divergence Loss $\mathcal{L}_{\text{KL}}$, stop prediction $\mathcal{L}_{\text{stop}}$, and Spectrogram Flux loss $\mathcal{L}_{\text{flux}}$. The losses are combined with different weights.

The regression loss is related to the MSE loss in our objective (8) for reconstruction but with extra L1 terms. Their reconstruction loss only depends on z_t, while we have additional dependency on $x_{<t}$ and y. Note that z_t is "continuous" in MELLE. We experiment with L1 loss following the original work but we do not find notable improvements. The KL loss, $\text{KL}(p(z_t|x_{<t}, y)\|\mathcal{N}(x_t, I))$, is introduced to map the Transformer output to a single Gaussian $\mathcal{N}(x_t, I)$ (the prior). We follow Meng et al. (2025), using a linear layer to predict the mean and log-variance of the Gaussian distribution for variational inference and sampling (Kingma and Welling, 2014), with a weight of 0.1 for $\mathcal{L}_{\text{KL}}$. We vary the weighting from 0.1 to 0.5 in increments of 0.1, but we find no notable improvements.

We experiment with the flux loss in Meng et al. (2025) by varying the flux loss weight from 0.1 to 0.6 in increments of 0.1, but it leads to unstable training. We instead use the proposed slowness penalty with a weight of 0.2 and find the slowness penalty "necessary" for MELLE and Mel-LM to suppress prolonged silence generations. We adopt the convolutional layers described in Tacotron 2 (Shen et al., 2018) to refine predicted mel-spectrograms. Though the exact architectures of the convolutional layers are not presented in Meng et al. (2025). We also experience with extending training MELLE for up to 400k steps, while the proposed approach and Codec-LM converges in 100k steps.

We suspect the VAD step is critical in the original MELLE to prevent models from keep producing silence. Unfortunately, the original work does not provide results for experiments conducted without VAD, nor does it offer comparisons to other codec-based methods trained on audio samples processed with VAD. It is also unclear how they filter "abnormal" silence. Given that prior codec-based TTS studies do not appear to have this pre-processing dependency (Song et al., 2025; Han et al., 2024; Chen et al., 2024), we do not adopt VAD.