T2V2: A UNIFIED NON-AUTOREGRESSIVE MODEL FOR SPEECH RECOGNITION AND SYNTHESIS VIA MULTITASK LEARNING

Nabarun Goswami[1], **Hanqin Wang**[1], **Tatsuya Harada**[1, 2]
[1]The University of Tokyo, Japan
[2]RIKEN, Japan
{nabarungoswami, wang, harada}@mi.t.u-tokyo.ac.jp

ABSTRACT

We introduce T2V2 (**T**ext to **V**oice and **V**oice to **T**ext), a unified non-autoregressive model capable of performing both automatic speech recognition (ASR) and text-to-speech (TTS) synthesis within the same framework. T2V2 uses a shared Conformer backbone with rotary positional embeddings to efficiently handle these core tasks, with ASR trained using Connectionist Temporal Classification (CTC) loss and TTS using masked language modeling (MLM) loss. The model operates on discrete tokens, where speech tokens are generated by clustering features from a self-supervised learning model. To further enhance performance, we introduce auxiliary tasks: CTC error correction to refine raw ASR outputs using contextual information from speech embeddings, and unconditional speech MLM, enabling classifier free guidance to improve TTS. Our method is self-contained, leveraging intermediate CTC outputs to align text and speech using Monotonic Alignment Search, without relying on external aligners. We perform extensive experimental evaluation to verify the efficacy of the T2V2 framework, achieving state-of-the-art performance on TTS task and competitive performance in discrete ASR.

1 INTRODUCTION

Speech recognition and synthesis are foundational tasks in human-computer interaction, enabling applications ranging from voice assistants to automated transcription services. Traditionally, both tasks rely heavily on autoregressive models, which generate sequences one token at a time, resulting in higher latency and limited efficiency (Graves, 2012; Sutskever et al., 2014; Chan et al., 2016; Wang et al., 2017; Li et al., 2019; Casanova et al., 2024). Recently, non-autoregressive models have gained traction by reducing inference latency while maintaining robustness in both TTS and ASR(Ren et al., 2020; Kim et al., 2021; Lee et al., 2023; Casanova et al., 2022; Graves et al., 2006; Higuchi et al., 2020; 2021a). Another promising approach is the use of discrete speech tokens, made possible by advances in neural audio codecs(Zeghidour et al., 2021; Kumar et al., 2024) (*acoustic tokens*) and clustering features from large-scale self-supervised speech models(Hsu et al., 2021; Baevski et al., 2020) (*content tokens*). These discrete tokens offer advantages in sequence modeling, particularly when paired with transformer-like architectures. Discrete tokens enable efficient storage and transmission, ideal for large datasets. They also serve as intermediate representations that capture acoustic and linguistic information while being less speaker-specific, aligning well with the unified modality in our framework.

Despite the recent success of non-autoregressive models in both ASR and TTS, integrating these tasks into a unified framework remains a significant challenge. A few works (Rubenstein et al., 2023; Wang et al., 2024; Maiti et al., 2024; Yang et al., 2024a; Toyin et al., 2024) have explored a unified modeling of speech-text related tasks, but they are limited to AR encoder-decoder or decoder only methods. However, existing NAR models typically handle ASR and TTS separately, missing opportunities to share representations and leverage joint training to improve performance across tasks. While multitask learning offers potential for parameter sharing, the performance of individual tasks might suffer due to the inherent complexity of training multiple tasks simultaneously. Authors in Xiujuan & Zhongke (2004), detail these challenges and common optimization approaches like scalarization and Pareto optimality. Another key challenge in NAR CTC-based ASR is the conditional independence assumption of CTC loss(Graves et al., 2006), which can lead to errors in transcription. Several works have tackled this problem with varying degrees of success(Higuchi et al., 2020; 2022; Chi et al., 2021; Nozaki & Komatsu, 2021). Moreover, NAR TTS models rely heavily on accurate alignment between text and speech, making external alignment tools(McAuliffe et al., 2017) necessary in most systems. A unified approach that can address both the independence limitations of CTC in ASR and the alignment challenges in TTS is needed. Last but not least, the use of discrete speech tokens, which have shown success in TTS (Borsos et al., 2023a; Rubenstein et al., 2023; Wang et al., 2023; Casanova et al., 2024) , has not been fully explored in ASR(Chang et al., 2023b; Yang et al., 2024b), leaving a gap in developing efficient models that can handle both tasks with the same set of representations. Addressing these challenges, while reducing reliance on external aligners and maintaining competitive performance, forms the core motivation of our work.

To address these challenges, we propose T2V2, a non-autoregressive model that unifies ASR and TTS using shared representations. T2V2 models *content tokens* with CTC-based training for ASR and a conditional masked language modeling approach for TTS, converting these tokens to *acoustic tokens* and ultimately to speech via the SoundStorm(Borsos et al., 2023b) and codec decoder. The core of T2V2 is a Conformer architecture with rotary positional embeddings (RoPE), capturing both local and long-range dependencies in speech and text. For TTS, Monotonic Alignment Search (MAS) is applied to intermediate CTC outputs, eliminating the need for external alignment tools and ensuring consistency across tasks.

To improve speech generation, we employ classifier-free guidance(Ho & Salimans, 2021), using an unconditional masked speech model to iteratively refine outputs. For ASR, we address CTC's independence limitation through a CTC error correction task, refining outputs with speech embeddings to recover time-step dependencies and enhance transcription accuracy. T2V2 efficiently handles both tasks within a unified framework, with auxiliary tasks like CTC error correction and unconditional speech MLM boosting performance. By leveraging discrete content tokens, our method improves sequence modeling and unifies speech and text processing. Our key contributions are as follows:

1. We propose T2V2, the first, to the best of our knowledge, unified NAR model for ASR and TTS and operates on discrete speech tokens derived from self-supervised models.
2. We leverage Monotonic Alignment Search (MAS) with intermediate CTC outputs for TTS alignment without relying on external tools for self-contained text-speech alignment.
3. We introduce a CTC error correction formulation to refine raw CTC outputs, improving ASR performance within the unified framework.
4. Extensive experimental validation confirms that T2V2 markedly improves synthesized speech robustness achieving state-of-the-art TTS performance and competitive discrete ASR performance.

2 MULTITASK T2V2 FRAMEWORK

The overview of our method is shown in Figure 1. The following sections provide a detailed description of each component and task.

2.1 NOTATION AND TERMINOLOGY

Here we introduce the notation used throughout the paper. The inputs to our model are discrete speech content tokens, denoted by $\boldsymbol{S} = \{s_1, s_2, \ldots, s_N\}$, where N is the number of tokens in the speech sequence, and corresponding text tokens, represented as $\boldsymbol{T} = \{x_1, x_2, \ldots, x_L\}$, with L being the text sequence length. These tokens are first converted into embeddings through their respective embedding layers, resulting in $\boldsymbol{Z}_S$ for speech embeddings and $\boldsymbol{Z}_T$ for text embeddings.

During training, subsets of tokens are masked depending on the task, and the masked tokens, along with their corresponding embeddings, are denoted with a superscript M, such as $\boldsymbol{S}^M$ and $\boldsymbol{Z}_S^M$. The task-specific masking schemes are detailed in Section 2.5.

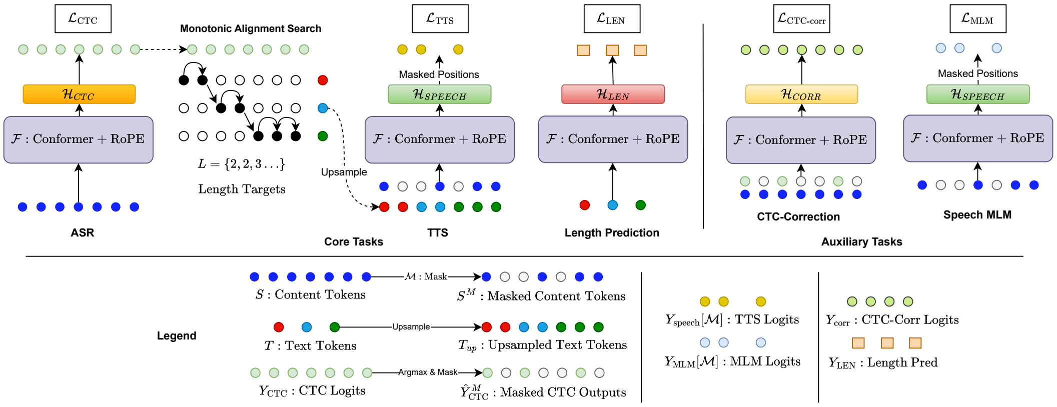
$\mathcal{L}_{\text{CTC}}$
Monotonic Alignment Search
$\mathcal{L}_{\text{TTS}}$
$\mathcal{L}_{\text{LEN}}$
$\mathcal{L}_{\text{CTC-corr}}$
$\mathcal{L}_{\text{MLM}}$
$\mathcal{H}_{CTC}$
Masked Positions
$\mathcal{H}_{SPEECH}$
$\mathcal{H}_{LEN}$
$\mathcal{H}_{CORR}$
Masked Positions
$\mathcal{H}_{SPEECH}$
$\mathcal{F}$: Conformer + RoPE
$L = \{2, 2, 3 \dots\}$
Length Targets
Upsample
ASR
Core Tasks
TTS
Length Prediction
CTC-Correction
Auxiliary Tasks
Speech MLM
Legend
S : Content Tokens
$\mathcal{M}$: Mask
S^M : Masked Content Tokens
T : Text Tokens
Upsample
T_{up} : Upsampled Text Tokens
Y_{CTC} : CTC Logits
Argmax & Mask
$\hat{Y}^M_{\text{CTC}}$: Masked CTC Outputs
$Y_{\text{speech}}[\mathcal{M}]$: TTS Logits
$Y_{\text{MLM}}[\mathcal{M}]$: MLM Logits
Y_{corr} : CTC-Corr Logits
Y_{LEN} : Length Pred

Figure 1: Overview of T2V2 architecture, best viewed in color. The parameters of the Conformer are shared across all tasks. The task specific heads with same color also share their parameters.

The shared Conformer encoder backbone is represented as $\mathcal{F}$, and the task-specific heads include the CTC prediction head, $\mathcal{H}_{\text{CTC}}$, CTC correction head, $\mathcal{H}_{\text{CORR}}$, speech prediction head, $\mathcal{H}_{\text{SPEECH}}$, and the length prediction head, $\mathcal{H}_{\text{LEN}}$.

2.2 Conformer with RoPE Backbone

In our proposed method, all tasks share a single backbone, $\mathcal{F}$, based on the Conformer (Gulati et al., 2020) architecture with rotary positional embeddings (RoPE) (Su et al., 2024). The Conformer combines convolutional layers, which capture local dependencies in the input sequences, with self-attention layers, which model long-range dependencies in both speech and text sequences.

Rotary Positional Embeddings (RoPE) further enhance the model's ability to process variable-length sequences by encoding positional information directly within the self-attention mechanism. Unlike traditional positional encodings, RoPE handles long sequences more effectively, making it especially beneficial for speech and text tasks that require both fine-grained temporal modeling and global context understanding. This combination has been demonstrated to be particularly effective for speech recognition (Li et al., 2021) and synthesis (Borsos et al., 2023b) tasks. Thus we utilize this combination as the backbone to efficiently share parameters across multiple tasks, improving both performance and flexibility in multitask learning.

2.3 Core Tasks

2.3.1 Speech Recognition with CTC

For the speech recognition task, the speech embeddings $\boldsymbol{Z}_S$, are passed through the shared Conformer backbone $\mathcal{F}$, and then through the CTC text prediction head $\mathcal{H}_{\text{CTC}}$. The Connectionist Temporal Classification (CTC) framework (Graves et al., 2006) is used to align the input speech with the target transcription without requiring explicit frame-level alignment. The predicted output logits over the vocabulary at each time step are given by:

$$\boldsymbol{Y}_{\text{CTC}} = \mathcal{H}_{\text{CTC}}(\mathcal{F}(\boldsymbol{Z}_S)), \tag{1}$$

where $\boldsymbol{Y}_{\text{CTC}}$ represents the CTC logits.

The CTC loss $\mathcal{L}_{\text{CTC}}$, is computed by marginalizing over all possible alignments between the input sequence and the target transcription $\boldsymbol{T}$:

$$\mathcal{L}_{\text{CTC}} = -\log \sum_{\mathbf{a} \in \mathcal{A}(\boldsymbol{T})} P(\mathbf{a} | \boldsymbol{Y}_{\text{CTC}}), \tag{2}$$

where $\mathcal{A}(\boldsymbol{T})$ is the set of all valid alignments of the target sequence $\boldsymbol{T}$. This formulation allows the model to predict the text sequence while accounting for varying lengths between the input speech and the target transcription.

2.3.2 Text to Speech and Length Prediction

For the text-to-speech (TTS) task, we compute the alignment score matrix $\boldsymbol{M} \in \mathbb{R}^{N \times L}$ by extracting the log probabilities of the text tokens $\boldsymbol{T}$ from the CTC logits $\boldsymbol{Y}_{\text{CTC}} \in \mathbb{R}^{N \times V}$ as:

$$\boldsymbol{M}_{n,\ell} = \boldsymbol{Y}_{\text{CTC},n,x_\ell}, \quad \forall n \in \{1, \ldots, N\}, \forall \ell \in \{1, \ldots, L\}. \tag{3}$$

Next, we apply the Monotonic Alignment Search (MAS) (Kim et al., 2020) to $\boldsymbol{M}$ to obtain the alignment matrix $\boldsymbol{A} \in \{0, 1\}^{N \times L}$. The MAS establishes a monotonic mapping between the text tokens and the speech time steps. For details, we refer the reader to Kim et al. (2020).

The input to the Conformer backbone is the point-wise addition of the embeddings of the upsampled text tokens $\boldsymbol{T}_{\text{up}} = \text{Upsample}(\boldsymbol{T}, \boldsymbol{A})$, $\boldsymbol{Z}_T^{\text{up}}$ and the masked speech embeddings $\boldsymbol{Z}_S^M$: $\boldsymbol{X}_{\text{TTS}} = \boldsymbol{Z}_T^{\text{up}} + \boldsymbol{Z}_S^M$. The backbone output is passed through the speech prediction head $\mathcal{H}_{\text{SPEECH}}$, and the loss is computed using cross-entropy focused on the masked positions:

$$\mathcal{L}_{\text{TTS}} = -\sum_{i \in \mathcal{M}} \boldsymbol{S}_i \log P(\mathcal{H}_{\text{SPEECH}}(\mathcal{F}(\boldsymbol{X}_{\text{TTS}}))_i), \tag{4}$$

where $\mathcal{M}$ is the set of masked positions, $\boldsymbol{S}$ represents the target speech tokens, and $P(\cdot)$ is the predicted probability.

For the length prediction task, the target token lengths $\boldsymbol{L}_i$ are derived from the MAS alignment matrix $\boldsymbol{A}$. Text tokens are passed through the length prediction head $\mathcal{H}_{\text{LEN}}$, and the loss is computed using an L1 objective:

$$\mathcal{L}_{\text{LEN}} = \sum_i \left| \mathcal{H}_{\text{LEN}}(\mathcal{F}(\boldsymbol{T}))_i - \log(\boldsymbol{L}_i) \right|. \tag{5}$$

2.4 Auxiliary Tasks

2.4.1 CTC Error Correction

In traditional CTC-based models, outputs at each time step are conditionally independent (Graves et al., 2006), limiting their ability to capture long-term dependencies and recover from early errors. To address this, we introduce a CTC error correction task that refines initial CTC predictions using confidence based masked intermediate CTC outputs and acoustic context $\boldsymbol{Z}_S$. Unlike Mask-CTC based approaches (Higuchi et al., 2020; 2021b), which operate on reduced intermediate CTC outputs and explicitly handle substitutions, insertions, and deletions, our method is more closely related to Nozaki & Komatsu (2021); Chi et al. (2021) and operates on un-reduced outputs, allowing implicit handling of all error types as well as leveraging full acoustic context, thereby relaxing the independence limitations in the correction phase. For a more detailed discussion in relation to the related works, please refer to appendix A.1.

The CTC logits $\boldsymbol{Y}_{\text{CTC}}$, are detached from the computational graph, and the argmax is taken to produce the raw CTC predictions $\hat{\boldsymbol{Y}}_{\text{CTC}}$. A subset of the tokens in $\hat{\boldsymbol{Y}}_{\text{CTC}}$ is then masked to create the masked CTC outputs $\hat{\boldsymbol{Y}}_{\text{CTC}}^M$. These masked CTC outputs are embedded into $\boldsymbol{Z}_{\hat{Y}}^M$ and point-wise added with the unmasked speech embeddings $\boldsymbol{Z}_S$, resulting in $\boldsymbol{X}_{\text{corr}} = \boldsymbol{Z}_{\hat{Y}}^M + \boldsymbol{Z}_S$, where $\boldsymbol{X}_{\text{corr}}$ represents the combined input for CTC error correction, which are passed through the shared Conformer backbone $\mathcal{F}$, and forwarded to the CTC correction head, $\mathcal{H}_{\text{CORR}}$, where the loss is computed using the CTC loss function:

$$\boldsymbol{Y}_{\text{corr}} = \mathcal{H}_{\text{CORR}}(\mathcal{F}(\boldsymbol{X}_{\text{corr}})), \tag{6}$$

$$\mathcal{L}_{\text{CTC-corr}} = -\log \sum_{\mathbf{a} \in \mathcal{A}(\boldsymbol{T})} P(\mathbf{a} | \boldsymbol{Y}_{\text{corr}}). \tag{7}$$

By correcting the CTC predictions in this manner, we expect to mitigate the potential negative impact of multitask learning on ASR performance and improve transcription accuracy.

2.4.2 Unconditional Speech Masked Language Model

Classifier-Free Guidance (CFG) (Ho & Salimans, 2021) has been used in diffusion models to balance conditional and unconditional outputs at each time step. This approach has also been successfully applied to mask-predict models in image generation, such as Muse (Chang et al., 2023a). We adapt CFG to improve the TTS performance by incorporating an unconditional speech only MLM.

For speech MLM training, the masked speech embeddings $\boldsymbol{Z}_S^M$, are passed through the shared Conformer backbone, $\mathcal{F}$. The output is then used to predict the speech tokens at masked positions using a cross-entropy loss, similar to the text-to-speech task but without text conditioning and reusing the same speech head $\mathcal{H}_{\text{SPEECH}}$:

$$\boldsymbol{Y}_{\text{MLM}} = \mathcal{H}_{\text{SPEECH}}(\mathcal{F}(\boldsymbol{Z}_S^M)), \tag{8}$$

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in \mathcal{M}_{\text{speech}}} \boldsymbol{S}_i \log P(\boldsymbol{Y}_{\text{MLM},i}), \tag{9}$$

where $\mathcal{M}_{speech}$ is the set of masked positions $\boldsymbol{S}$ are the target speech tokens, and $P(\boldsymbol{Y}_{\text{MLM},i})$ is the predicted probability at position i. The goal during training is to reconstruct the masked speech tokens from the unmasked speech embeddings.

2.5 Masking Scheme

In our multitask framework, we employ different masking schemes depending on the task, as the requirements for masking vary. For the CTC error correction task, we use a uniform masking strategy, while for the TTS and speech MLM tasks, we utilize a more aggressive cosine-based masking strategy. Below, we explain the rationale for each scheme.

CTC Error Correction Task: Uniform Masking During training for the CTC error correction task, the model refines low-confidence tokens without starting from fully masked tokens. A uniform masking strategy is applied, where the mask is sampled as:

$$\mathcal{M}_{\text{corr}} = \text{Bernoulli}(p), \quad p \sim \mathcal{U}(0, 1), \tag{10}$$

targeting a random number of tokens from the CTC output $\hat{\boldsymbol{Y}}_{\text{CTC}}$ for correction. This approach ensures the model focuses on low-confidence tokens while maintaining enough context for effective refinement.

TTS and Speech MLM Tasks: Cosine Masking Schedule For the TTS and speech MLM tasks, we start with fully masked sequences and progressively unmask tokens using a cosine masking schedule inspired by MaskGIT (Chang et al., 2022). At each step, the mask is determined as:

$$\mathcal{M}_{\text{speech}} = \text{Bernoulli}(p), \quad p = \cos(u), \quad u \sim \mathcal{U}(0, \frac{\pi}{2}), \tag{11}$$

where u is sampled from a uniform distribution over $[0, \frac{\pi}{2}]$, and p is the cosine of u. This schedule mimics the inference iterative refinement where more tokens are masked initially and reduces masking as the model refines the outputs.

2.6 Overall Training Objective

In our framework, we adopted the simplest weighted-sum scalarization approach for multi-objective optimization, assigning equal weights to all task-specific losses from eqs. (2), (4), (5), (7) and (9) to obtain a single objective as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CTC}} + \mathcal{L}_{\text{TTS}} + \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{CTC-corr}} + \mathcal{L}_{\text{LEN}}, \tag{12}$$

This choice was based on the assumption that the tasks are cooperative rather than conflicting and the combined loss ensures that the model learns to balance multiple tasks efficiently, allowing it to leverage shared knowledge across tasks while optimizing performance for each individual task.

2.7 Inference

2.7.1 Speech Recognition

During inference, the speech embeddings $\boldsymbol{Z}_S$ are passed through the Conformer backbone into the CTC head to generate logits. Low-confidence tokens are identified based on a threshold τ, masked and refined iteratively by combining masked outputs $\hat{\boldsymbol{Y}}_{\text{CTC}}^M$ with the speech embeddings $\boldsymbol{Z}_S$. Refinement continues until all token confidence scores exceed τ or for a fixed number of iterations. The final transcription is obtained through CTC decoding of $\boldsymbol{Y}_{\text{corr}}$.

2.7.2 Text to Speech

For TTS, text tokens $\boldsymbol{T}$ are passed through the backbone into the length prediction head to predict log lengths, which are then converted to integer lengths as $\boldsymbol{L}_{\text{pred}} = \lceil \exp(\boldsymbol{Y}_{\text{LEN}}) \rceil$. These predicted lengths are used to upsample the text tokens, $\boldsymbol{T}_{\text{up}} = \text{Upsample}(\boldsymbol{T}, \boldsymbol{L}_{\text{pred}})$, which are combined with fully masked speech embeddings $\boldsymbol{Z}_S^M$, and passed through the backbone into the speech prediction head. During generation, tokens are iteratively unmasked based on their confidence scores, starting with fully masked tokens. The unmasking follows the cosine schedule, where the masking probability at each iteration t is given by $p_t = \cos\left(\frac{t\pi}{2T}\right)$.

Additionally, for Classifier-Free Guidance (CFG), we forward $\boldsymbol{Z}_S^M$ without text conditioning to obtain the unconditional logits $\boldsymbol{Y}_{\text{MLM}}$. The final output logits are then a weighted combination of the conditional logits $\boldsymbol{Y}_{\text{TTS}}$, and the unconditional logits $\boldsymbol{Y}_{\text{MLM}}$, as follows:

$$\boldsymbol{Y}_{\text{pred}} = (1 + \lambda) \cdot \boldsymbol{Y}_{\text{TTS}} - \lambda \cdot \boldsymbol{Y}_{\text{MLM}}, \tag{13}$$

where λ is the guidance weight controlling the balance between conditional and unconditional outputs. This process yields the final speech sequence $\boldsymbol{S}_{\text{pred}}$.

3 Experiments

To verify the effectiveness of our proposed methods, we perform extensive experimental evaluations. In this section, we first describe the training details followed by the various experimental results in the subsequent subsections.

3.1 Model Architecture and Training Details

3.1.1 Training Infrastructure and Settings

All our models were implemented using the Pytorch(Paszke et al., 2019) framework and trained on 4 NVIDIA H100-80G GPUs with *bfloat16*(Burgess et al., 2019) precision. For efficient distributed training, we utilized *deepspeed*(Rajbhandari et al., 2020) ZeRO 2 optimization. For all experiments, we utilize the Adam optimizer(Kingma & Ba, 2015) with a peak learning rate of $2.5e^{-4}$, linearly warmed up over the first 4K iterations and decayed over the remaining iterations with a cosine schedule. The *beta1* and *beta2* of the optimizer are set to $\{0.8, 0.99\}$, with no weight decay, and clip the gradient norm with a maximum of 0.5.

3.1.2 Training Datasets

For training all our models, we utilized the 60K hour LibriLight(Kahn et al., 2020) dataset. Additionally, for punctuated and cased transcriptions, we utilized the transcriptions provided in the LibriHeavy(Kang et al., 2024) dataset. We mainly use the 509 hour *small* subset of the LibriHeavy dataset for our experiments. We filtered out samples shorter than 0.2s and longer than 20s for better GPU utilization. Further, we removed samples which violate the input-target length constraints of the CTC loss. Following this we are left with around 500 hours of speech data that we use for the training. We perform all our experiments on 16Khz speech.

3.1.3 Model Architecture

For all experiments (unless specified otherwise), we used the same conformer backbone with 6 layers with hidden size of 384, 8 attention heads, 1536 linear size, and convolution kernel size of 7. We utilize Rotary Position Embedding (RoPE) (Su et al., 2024). For each of the task-specific heads, we utilized the same structure of a linear layer with the same hidden size, followed by GELU activation and layer normalization, and finally an output linear layer to map to the respective output dimensions. We trained the models on the *small* subset of the LibriHeavy dataset for 30K iterations with a batch size of 64.

To convert the content tokens generated by our proposed method to acoustic tokens from the codec, we utilized the masked iterative generative model, SoundStorm(Borsos et al., 2023b). We reproduced this model by utilizing a conformer backbone with 12 layers with hidden size of 1024, 16 attention heads, 4096 linear size, and convolution kernel size of 5 with RoPE, followed by linear output heads for each RVQ level in the codec. We also trained this model on the LibriLight dataset with 15 second segments for 50K iterations with a batch size of 640.

3.1.4 Tokenizers

For the acoustic tokenizer, we use utilize a 12-level RVQ-based codec following the architecture of Descript Audio Codec (DAC) (Kumar et al., 2024), trained on the LibriLight dataset with a batch size of 144 for 200K iterations using the hyperparameters from[1]. For the content tokenizer, we utilize the publicly available HuBERT large checkpoint trained on the LibriLight dataset[2]. Further we trained a 1024-cluster K-Means tokenizer on the *train-clean-100* subset of the LibriSpeech(Panayotov et al., 2015) dataset. Both the acoustic and content tokenizers produce tokens at the rate of 50Hz, which simplifies the implementation. Finally for the text tokenization, we used a simple *utf-8* byte based representation. Byte representation has the advantage of having a compact output layer, requiring minimal text pre-processing and ability to handle any character and language. We did some basic pre-processing to expand certain common word contractions and numbers as is standard practice in ASR or TTS training.

3.2 Evaluation Datasets and Metrics

To evaluate the performance of our proposed method, we utilized the following datasets and metrics. For evaluation of the ASR task, we utilize the LibriSpeech *test-clean* subset. For the TTS task, we randomly sampled 40 sentences from the LibriSpeech *test-clean* subset with at least 20 words in each sentence as the text input and sampled 3-8 second segments from all 20 speakers of the DAPS((Mysore, 2014) dataset as the speaker prompt for zero-shot TTS evaluation.

For evaluating the ASR performance, we utilized the Word Error Rate (WER) and Character Error Rate (CER). Since, our model is trained on punctuated and cased inputs, for the evaluation, we normalize the predicted sentences be removing all punctuation except the apostrophe and converting to upper-case following the format of the LirbiSpeech transcripts.

We utilized several automatic metrics for zero-shot TTS evaluation, which included UTMOS (Saeki et al., 2022) for speech quality, Speaker Encoder Cosine Similarity (SECS) using the *wavlm-base-plus-sv* model[3] for speaker similarity, Character Error Rate (CER) using the *hubert-large-ls960-ft* model[4] for percieved intelligibility and robustness. To effectively evaluate the zero-shot TTS performance, we finally perform a subjective evaluation. For this evaluation, we utilize one sample per speaker from the evaluation data described above, and present samples from two systems along with the text input and ask the raters to score between $\{+2, -2\}$ (order of methods is randomized), based on *naturalness, acoustic quality, and human likeness*. At least 5 raters rate each sample and this gives us the Comparative mean opinion score (CMOS). Similarly we perform subjective evaluation for Speaker similarity Comparative mean opinion score (SCMOS), the difference being for this evaluation the reference speaker prompt is also presented to the raters and asked to compare which of the two shown methods is more similar to the reference. The rating scale and number of raters per sample are same as the CMOS evaluation. For all scores, we compute the 95% confidence interval by bootstrapping and measure the statistical significance with the Wilcoxon signed-rank test (p-value).

To evaluate latency, we conducted controlled inference runtime (IR) experiments. For TTS, we measured end-to-end runtime (IR-e2e) and text-to-content token runtime (IR-t2c) using a 405-character sentence. For ASR, we measured IR with a 60-second sample. All measurements were averaged over 100 trials on a single H100 GPU.

3.3 Baselines

To test the effectiveness of our method, we utilize several state-of-the-art baselines across various speech synthesis and recognition paradigms.

[1] https://github.com/descriptinc/descript-audio-codec/blob/main/conf/final/16khz.yml

[2] https://huggingface.co/facebook/hubert-large-ll60k

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

[4] https://huggingface.co/facebook/hubert-large-ls960-ft

1. Speech Synthesis
    (a) Non-iterative: HierSpeech++(Lee et al., 2023), YourTTS(Casanova et al., 2022)
    (b) Diffusion: StyleTTS2 (Li et al., 2024)
    (c) Autoregressive: XTTS(Casanova et al., 2024), WhisperSpeech[5] (based on (Borsos et al., 2023a; Kharitonov et al., 2023))
2. Speech Recognition
    (a) Transducer based non-discrete ASR: Zipformer-Transducer(Yao et al., 2023)[6]
    (b) Discrete ASR: Coformer-CTC trained on discrete content tokens.

We utilize publicly released checkpoints of all the above methods except SoundStorm, which we reproduce and train on the same dataset as our method and with similar parameter count.

3.4 Zero-Shot Text to Speech Synthesis

3.4.1 Ablation Study

Table 1: Zero-shot TTS ablation study for different tasks.

Task Setting	UTMOS	CER	SECS
w SMLM, w CORR	4.39 ± 0.04	0.95	0.94 ± 0.01
w SMLM, w/o CORR	4.41 ± 0.04	1.08	0.94 ± 0.01
w/o SMLM, w/o CORR	4.39 ± 0.03	0.82	0.94 ± 0.01

To study the effect of each component of our proposed method for the TTS task, we conduct extensive ablation study. First we check the performance of introducing the auxiliary CTC-correction and Speech MLM tasks. For this evaluation, we predict the content tokens in a single pass through the network. The results in Table 1 indicate that the auxiliary tasks have minimal impact on the core TTS task and the performance is maintained.

Table 2: Zero-shot TTS ablation study for different number of iterations.

Iters	UTMOS	CER	SECS
1	4.39 ± 0.04	**0.95**	0.94 ± 0.01
4	**4.43 ± 0.03**	1.12	0.94 ± 0.01
8	4.41 ± 0.03	1.23	0.94 ± 0.01

Table 3: TTS ablation study for CFG weight λ.

λ	UTMOS	CER	SECS
0.0	**4.43 ± 0.03**	1.12	0.94 ± 0.01
1.0	**4.43 ± 0.02**	**0.55**	0.94 ± 0.01
1.5	4.40 ± 0.04	0.95	0.94 ± 0.01
2.0	4.42 ± 0.02	0.69	0.94 ± 0.01

Next we evaluate the number of iterations for generating the content tokens. For this task we do not use CFG. The results reported in Table 2 show that our method is able to achieve very good performance even with a single step. Increasing the number of iterations slightly improves the UTMOS, while slightly degrading the CER. For subsequent experiments, we utilize 4 iterations, which provides a balance between the UTMOS and CER scores.

Finally we evaluate the effect of CFG weight, λ. From the results in Table 3, we can see that using CFG significantly improves the CER while maintaining the speech quality with $\lambda = 1.0$ giving the best CER score. The degradation observed in single-pass inference with SMLM is marginal (Table 1), and is compensated by the iterative inference with CFG, which fully utilizes SMLM's benefits, significantly improving CER for TTS.

3.4.2 Main Result

Finally, with the best settings from the ablation study, we compare our method with state-of-the-art methods in Table 4. Our method significantly outperforms baselines trained on similar-scale data.

[5] https://github.com/collabora/WhisperSpeech

[6] https://github.com/k2-fsa/icefall/tree/master/egs/libriheavy/ASR

Table 4: Zero-shot TTS performance comparison. Methods with * indicate multilingual models. UD refers to Unpaired Data while PD refers to Paired Data in hours.

	UD	PD	UTMOS	CER	SECS	IR-e2e (s)	IR-t2c (s)
Large scale paired data							
HierSpeech++*	500k	2.8k	**4.46 ± 0.02**	0.88	**0.94 ± 0.01**	0.16 ± 0.00	-
XTTS*	-	27k	4.12 ± 0.07	0.78	0.93 ± 0.01	2.60 ± 0.03	-
WhisperSpeech	60k	60k	3.95 ± 0.11	0.66	0.93 ± 0.01	17.91 ± 0.04	2.84 ± 0.01
Small scale paired data							
YourTTS*	-	689	3.69 ± 0.08	2.02	0.90 ± 0.02	**0.11 ± 0.00**	-
StyleTTS2	94k	245	4.43 ± 0.03	1.59	0.91 ± 0.02	0.27 ± 0.00	-
Ours	60k	500	4.43 ± 0.02	**0.55**	**0.94 ± 0.01**	0.57 ± 0.00	**0.06 ± 0.00**

Additionally, it surpasses systems trained on thousands of hours of paired speech, demonstrating superior robustness in terms of the CER score and data efficiency. A point to note here is that the UTMOS and SECS metrics are more influenced by the SoundStorm and Codec rather than our method. IR-t2c shows that T2V2 is $\sim$ 5 orders of magnitude faster than the AR WhisperSpeech baseline and surpasses other AR methods in IR-e2e.

We present the result of the subjective evaluation in Table 5. From the results we can see that in terms of CMOS, our method is at par with all the state-of-the-art methods since the p-value is greater than 0.05 indicating no statistically significant difference. While for the SCMOS, our method outperforms XTTS and WhisperSpeech while being at par with HierSpeech++ and StyleTTS2. We encourage the readers to listen to the samples provided in the supplementary materials.

Table 5: Comparative MOS for Speech Quality (CMOS) and Speaker Similarity (SCMOS) on a scale $\{-2, +2\}$. p-value ≤ 0.05 indicate statistical significance.

	CMOS (p-value)	**SCMOS (p-value)**
HierSpeech++	**+0.10 ± 0.25 (0.337)**	**+0.12 ± 0.26 (0.287)**
XTTS	**-0.13 ± 0.28 (0.418)**	-0.30 ± 0.22 (0.007)
StyleTTS2	**+0.16 ± 0.25 (0.271)**	**+0.14 ± 0.24 (0.201)**
WhisperSpeech	**-0.11 ± 0.27 (0.490)**	-0.63 ± 0.21 ($1.5e^{-7}$)
Ours	**0.00**	**0.00**

3.5 Automatic Speech Recognition

3.5.1 Ablation Study

To evaluate the impact of various components and inference settings on ASR performance, we conducted an ablation study. In Table 6, we examine the effect of auxiliary tasks on ASR performance, noting that the CTC Correction task notably regulates the model and improves raw ASR performance from the CTC head without correction. While the SMLM task alone slightly degrades performance, the CORR task compensates, allowing iterative correction to significantly enhance ASR results.

Next, we investigated the confidence threshold and number of iterations for CTC error correction (Table 7). A threshold of 0.7 with 16 correction iterations yielded the best results. Increasing the iterations slightly raised the IR, but the process remained efficient.

Finally, we analyzed how iterative correction handles all error types (Table 8). Using the best configuration from Table 7, we observed consistent relative improvements across substitutions, insertions, and deletions, demonstrating the effectiveness of our proposed mechanism.

3.5.2 Main Result

Finally we compare our method with the baselines in Table 9. We show the results of the non-discrete reference model Zipformer-Transducer also trained with casing and punctuation. As we can

Table 6: ASR ablation study for different tasks.

	CER	WER
w SMLM, w CORR	**2.732**	**8.651**
w SMLM, w/o CORR	2.949	9.428
w/o SMLM, w/o CORR	2.886	9.120

Table 8: Individual error type improvements.

	Sub	Ins	Del
w/o CORR	1.300	0.090	0.140
w CORR	**1.255** ↓3.46%	**0.082** ↓8.89%	**0.135** ↓3.57%

Table 7: ASR ablation study for different correction thresholds and iterations.

Corr. Thresh	Iters	CER	WER	IR(s)
w/o CORR	-	2.73	8.65	0.32 ± 0.02
0.8	1	2.73	8.44	0.40 ± 0.03
0.8	4	2.72	8.37	0.39 ± 0.03
0.8	8	2.72	8.33	0.42 ± 0.03
0.7	8	2.72	8.29	0.42 ± 0.03
0.7	16	**2.71**	**8.27**	0.47 ± 0.03
0.7	32	**2.71**	**8.27**	0.60 ± 0.03

see discrete ASR still under-performs compared to non-discrete AR zipformer-transducer, however when compared with the discrete baseline (Conformer-CTC), we achieve comparable performance. We further verified how our method scales with scaling the data. For that we trained a larger version of our model from scratch with 10 layers on the LibriHeavy large subset (50K hours) and we see that our method outperforms the CTC baseline with same 10-layer conformer backbone. Thus validating the effectiveness and scaling capabilities of T2V2.

Table 9: ASR results for models trained with punctuation and casing. The publicly released models for Zipformer-Transducer are used for the evaluation, while Conformer-CTC is trained by us.

	Libriheavy Subset	CER	WER	IR (s)
Non-discrete ASR (BPE encoding)				
Zipformer-Transducer	small	2.01	5.33	1.49 ± 0.07
Zipformer-Transducer	large	0.66	1.99	1.51 ± 0.14
Discrete ASR (Byte Encoding)				
Conformer-CTC	small	**2.69**	8.28	0.32 ± 0.02
Ours	small	2.71	**8.27**	0.47 ± 0.03
Conformer-CTC	large	1.53	4.36	0.34 ± 0.02
Ours	large	**1.31**	**4.09**	0.55 ± 0.02

4 Future Perspectives and Limitations

In the current work, we focus on T2V2's capability in the two core tasks of ASR and TTS, however, the multitask training paradigm could potentially allow the model to learn high performing and general purpose representations of speech and text which could be used for other downstream tasks as well as enable training modality agnostic language models. As for the limitations of the T2V2, we see that discrete ASR still underperforms compared to models trained with continuous features, while for TTS, we rely on separate content token to acoustic token translation, which might limit the quality of synthesized speech.

5 Conclusions

In this work, we introduced T2V2, the first unified non-autoregressive model capable of handling both automatic speech recognition and text-to-speech synthesis. By leveraging discrete tokens from self-supervised models and incorporating auxiliary tasks like CTC error correction and unconditional speech MLM, T2V2 effectively improves performance across both ASR and TTS. Additionally, the use of Monotonic Alignment Search ensures a self-contained approach to text-speech alignment without relying on external aligners. Our experimental results demonstrate significant gains in TTS quality while maintaining competitive ASR performance, highlighting the potential of unified models for multitask learning in speech processing.

Ethics Statement

Our proposed method achieves high quality synthetic speech in a target speaker's voice with a short sample, thus it is paramount to address the broader impact of our research. Our method may be potentially misused by malicious parties, towards this end we verified that our speech could still be detected as fake speech by a third party detector[7].

Reproducibility Statement

For reproducibility, we have carefully described the model architecture, optimization tasks, training masking schemes, loss functions, and inference procedure in Section 2. Additionaly, implementation and training hyperparameters, and all pre-trained models have been described in Section 3.1, including their links in the footnotes.

Acknowledgments

This work was partially supported by JST Moonshot R&D Grant Number JPMJPS2011, CREST Grant Number JPMJCR2015 and Basic Research Grant (Super AI) of Institute for AI and Beyond of the University of Tokyo.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023a.

Zalán Borsos, Matt Sharifi, Damien Vincent, Eugene Kharitonov, Neil Zeghidour, and Marco Tagliasacchi. Soundstorm: Efficient parallel audio generation. *arXiv preprint arXiv:2305.09636*, 2023b.

Neil Burgess, Jelena Milanovic, Nigel Stephens, Konstantinos Monachopoulos, and David Mansell. Bfloat16 processing for neural networks. In *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*, pp. 88–91, 2019. doi: 10.1109/ARITH.2019.00022.

Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2709–2720. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/casanova22a.html`.

Edresson Casanova, Kelly Davis, Eren Gölge, Görkem Göknar, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, and Julian Weber. Xtts: a massively multilingual zero-shot text-to-speech model, 2024. URL `https://arxiv.org/abs/2406.04904`.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4960–4964. IEEE, 2016.

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.

[7] https://detect.resemble.ai/

Huiwen Chang, Han Zhang, Jarred Barber, Aaron Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Patrick Murphy, William T. Freeman, Michael Rubinstein, Yuanzhen Li, and Dilip Krishnan. Muse: Text-to-image generation via masked generative transformers. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4055–4075. PMLR, 23–29 Jul 2023a. URL `https://proceedings.mlr.press/v202/chang23b.html`.

Xuankai Chang, Brian Yan, Yuya Fujita, Takashi Maekaku, and Shinji Watanabe. Exploration of efficient end-to-end asr using discretized input from self-supervised learning. In *INTERSPEECH 2023*, pp. 1399–1403, 2023b. doi: 10.21437/Interspeech.2023-2051.

Ethan A. Chi, Julian Salazar, and Katrin Kirchhoff. Align-refine: Non-autoregressive speech recognition via iterative realignment. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1920–1927, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.154. URL `https://aclanthology.org/2021.naacl-main.154`.

Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

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.

Yosuke Higuchi, Shinji Watanabe, Nanxin Chen, Tetsuji Ogawa, and Tetsunori Kobayashi. Mask-ctc: Non-autoregressive end-to-end asr with ctc and mask predict. In *Proceedings of the 2020 International Conference on Spoken Language Processing (INTERSPEECH)*, 2020.

Yosuke Higuchi, Nanxin Chen, Yuya Fujita, Hirofumi Inaguma, Tatsuya Komatsu, Jaesong Lee, Jumon Nozaki, Tianzi Wang, and Shinji Watanabe. A comparative study on non-autoregressive modelings for speech-to-text generation. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 47–54, 2021a. doi: 10.1109/ASRU51503.2021.9688157.

Yosuke Higuchi, Hirofumi Inaguma, Shinji Watanabe, Tetsuji Ogawa, and Tetsunori Kobayashi. Improved mask-ctc for non-autoregressive end-to-end asr. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8363–8367, 2021b. doi: 10.1109/ICASSP39728.2021.9414198.

Yosuke Higuchi, Brian Yan, Siddhant Arora, Tetsuji Ogawa, Tetsunori Kobayashi, and Shinji Watanabe. Bert meets ctc: New formulation of end-to-end speech recognition with pre-trained masked language model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL `https://openreview.net/forum?id=qw8AKxfYbI`.

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 Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Jacob Kahn, Morgane Rivière, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7669–7673. IEEE, 2020.

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 *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 10991–10995. IEEE, 2024.

Eugene Kharitonov, Damien Vincent, Zalán Borsos, Raphaël Marinier, Sertan Girgin, Olivier Pietquin, Matt Sharifi, Marco Tagliasacchi, and Neil Zeghidour. Speak, read and prompt: High-fidelity text-to-speech with minimal supervision. *Transactions of the Association for Computational Linguistics*, 11:1703–1718, 2023.

Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Advances in Neural Information Processing Systems*, 33:8067–8077, 2020.

Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pp. 5530–5540. PMLR, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. High-fidelity audio compression with improved rvqgan. *Advances in Neural Information Processing Systems*, 36, 2024.

Sang-Hoon Lee, Ha-Yeong Choi, Seung-Bin Kim, and Seong-Whan Lee. Hierspeech++: Bridging the gap between semantic and acoustic representation of speech by hierarchical variational inference for zero-shot speech synthesis. *arXiv preprint arXiv:2311.12454*, 2023.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 6706–6713, 2019.

Shengqiang Li, Menglong Xu, and Xiao-Lei Zhang. Conformer-based end-to-end speech recognition with rotary position embedding. In *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 443–447. IEEE, 2021.

Yinghao Aaron Li, Cong Han, Vinay S. Raghavan, Gavin Mischler, and Nima Mesgarani. Styletts 2: towards human-level text-to-speech through style diffusion and adversarial training with large speech language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2024. Curran Associates Inc.

Soumi Maiti, Yifan Peng, Shukjae Choi, Jee-Weon Jung, Xuankai Chang, and Shinji Watanabe. Voxtlm: Unified decoder-only models for consolidating speech recognition, synthesis and speech, text continuation tasks. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 13326–13330, 2024. doi: 10.1109/ICASSP48485.2024.10447112.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pp. 498–502, 2017. doi: 10.21437/Interspeech.2017-1386.

Gautham J Mysore. Can we automatically transform speech recorded on common consumer devices in real-world environments into professional production quality speech?—a dataset, insights, and challenges. *IEEE Signal Processing Letters*, 22(8):1006–1010, 2014.

Jumon Nozaki and Tatsuya Komatsu. Relaxing the conditional independence assumption of ctc-based asr by conditioning on intermediate predictions. In *Interspeech 2021*, pp. 3735–3739, 2021. doi: 10.21437/Interspeech.2021-911.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20. IEEE Press, 2020. ISBN 9781728199986.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*, 2020.

Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*, 2023.

Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. UTMOS: UTokyo-SaruLab System for VoiceMOS Challenge 2022. In *Proc. Interspeech 2022*, pp. 4521–4525, 2022. doi: 10.21437/Interspeech.2022-439.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL https://www.sciencedirect.com/science/article/pii/S0925231223011864.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 3104–3112, Cambridge, MA, USA, 2014. MIT Press.

Hawau Olamide Toyin, Hao Li, and Hanan Aldarmaki. STTATTS: Unified speech-to-text and text-to-speech model. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6853–6863, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-emnlp.401.

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, 2023. URL https://arxiv.org/abs/2301.02111.

Tianrui Wang, Long Zhou, Ziqiang Zhang, Yu Wu, Shujie Liu, Yashesh Gaur, Zhuo Chen, Jinyu Li, and Furu Wei. Viola: Conditional language models for speech recognition, synthesis, and translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3709–3716, 2024. doi: 10.1109/TASLP.2024.3434425.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*, 2017.

Lei Xiujuan and Shi Zhongke. Overview of multi-objective optimization methods. *Journal of Systems Engineering and Electronics*, 15(2):142–146, 2004.

Runyan Yang, Huibao Yang, Xiqing Zhang, Tiantian Ye, Ying Liu, Yingying Gao, Shilei Zhang, Chao Deng, and Junlan Feng. Polyspeech: Exploring unified multitask speech models for competitiveness with single-task models. *arXiv preprint arXiv:2406.07801*, 2024a.

Yifan Yang, Feiyu Shen, Chenpeng Du, Ziyang Ma, Kai Yu, Daniel Povey, and Xie Chen. Towards universal speech discrete tokens: A case study for asr and tts. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 10401–10405. IEEE, 2024b.

Zengwei Yao, Liyong Guo, Xiaoyu Yang, Wei Kang, Fangjun Kuang, Yifan Yang, Zengrui Jin, Long Lin, and Daniel Povey. Zipformer: A faster and better encoder for automatic speech recognition. In *The Twelfth International Conference on Learning Representations*, 2023.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

A Appendix

A.1 CTC Error Correction and Related Work

A.1.1 Relation to Align-Refine

Align-Refine (Chi et al., 2021) and our method share the conceptual similarity of utilizing an iterative refinement paradigm. However, our approach differs significantly in its formulation:

Separate Decoder vs. Direct Approach: Align-Refine employs a separate non-causal transformer decoder for refinement, integrating encoder features via cross-attention. In contrast, our method formulates refinement as a separate task by reusing the same backbone encoder and directly combining intermediate CTC outputs with speech tokens for refinement. This eliminates the need for additional modules such as decoders or cross-attention layers.

A.1.2 Relation to Self-Conditioned CTC

Our method also shares similarities with Self-Conditioned CTC (Nozaki & Komatsu, 2021) in leveraging intermediate CTC outputs, but there are notable differences:

Intermediate Conditioning: Self-Conditioned CTC applies intermediate CTC logit embeddings via additional linear projection layers at intermediate encoder stages, making it well-suited for dedicated ASR models. However, in our multi-task scenario, where the backbone supports both ASR and TTS tasks, such intermediate conditioning might not generalize well. Our approach avoids additional layers, maintaining a lightweight design.

Iterative Refinement: Unlike Self-Conditioned CTC, which does not support iterative refinement, our method incorporates iterative refinement as a core mechanism.