RALL-E: ROBUST CODEC LANGUAGE MODELING WITH CHAIN-OF-THOUGHT PROMPTING FOR TEXT-TO-SPEECH SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present RALL-E, a robust language modeling method for text-to-speech (TTS) synthesis. While previous codec language modeling methods have demonstrated impressive performance in zero-shot TTS, they often struggle with robustness issues, such as unstable prosody (irregular pitch and rhythm/duration) and high word error rates (WER), largely due to their autoregressive prediction style. RALL-E addresses these issues through chain-of-thought (CoT) prompting, which breaks the task into simpler steps to improve the stability of TTS. First, RALL-E predicts prosody tokens (pitch and duration) from the input text and uses them as intermediate conditions to guide the prediction of speech tokens in a CoT manner. Second, RALL-E utilizes the predicted duration prompt to guide the computing of self-attention weights in Transformer, enforcing the model to focus on the corresponding phonemes and prosody tokens during speech token prediction. Comprehensive objective and subjective evaluations show that RALL-E significantly improves robustness in zero-shot TTS compared to the baseline method VALL-E, reducing WER from 5.6% to 2.5% without reranking, and from 1.7% to 1.0% with reranking. Furthermore, RALL-E outperforms several prior approaches aimed at improving the robustness of codec language models, and successfully synthesizes challenging sentences that VALL-E struggles with, lowering the error rate from 68% to 4%.

1 INTRODUCTION

Language models (LMs) have made significant advancements in natural language generation (Radford et al., 2019; Brown et al., 2020). With sufficiently large model sizes, these models demonstrate powerful in-context learning abilities, enabling them to handle unseen tasks with a text prompt in a zero-shot or few-shot manner (Wei et al., 2022a). Additionally, the simple yet effective next-token prediction framework allows language models to be applied to other domains, such as vision (Dehghani et al., 2023) and speech synthesis (Wang et al., 2023a), as long as the data can be converted into discrete tokens. This work focuses on language modeling for text-to-speech (TTS) synthesis. Recent studies (Wang et al., 2023a; Kharitonov et al., 2023) have demonstrated that TTS can be effectively modeled using a decoder-only language model by employing a neural codec (Zeghidour et al., 2021; Défossez et al., 2022) to convert continuous waveforms into discrete tokens. These methods, typically leveraging tens of thousands of hours of speech data, exhibit in-context learning abilities that allow the model to clone a speaker's voice using only a short audio prompt, achieving remarkable performance in zero-shot TTS.

Table 1: Performance of RALL-E and the baseline method VALL-E (Wang et al., 2023a) on 50 particularly hard sentences obtained from Ren et al. (2019). The result of NaturalSpeech 2 is from Shen et al. (2023).

Model	Mispronunciation	Omission	Repetition	Hallucination	Error rate
NaturalSpeech 2	0	0	0	0	0%
VALL-E	10	19	8	7	68%
RALL-E	2	0	0	0	4%

However, due to the sequential nature of language model generation, codec LMs often struggle with robustness issues. While the autoregressive (AR) prediction style allows for generating speech with

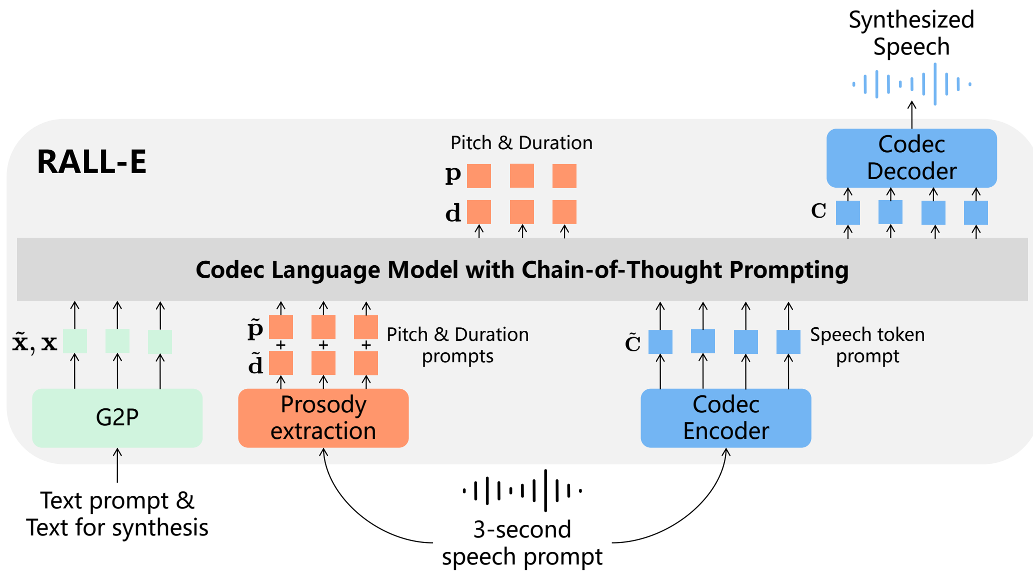

Figure 1: Overview of RALL-E with CoT prompting. Symbols are all defined in Section 3.1. The proposed CoT prompting of prosody tokens and duration-guided masking are introduced in Section 3.2 and 3.3, respectively.

diverse prosody patterns, it can also lead to unnatural prosody in some cases. Additionally, because there is no strict alignment between text and speech, these models may omit or repeat words from the input text. In contrast, non-autoregressive (NAR) TTS methods (Shen et al., 2023; Le et al., 2024; Ju et al., 2024) generate all tokens simultaneously, resulting in higher robustness but lower prosodic diversity. As noted in previous studies (Yang et al., 2023; Ju et al., 2024), codec AR TTS systems tend to have a higher word error rate (WER) compared to NAR TTS, despite showing similar performance on other metrics. One straightforward yet effective approach to mitigate this issue is to sample the same input text multiple times and then select the best result through reranking (Kharitonov et al., 2023; Yang et al., 2023). However, this reranking process significantly increases inference time.

In this paper, we present RALL-E (short for robust VALL-E), a method designed to improve the robustness of TTS based on codec LMs. The core idea behind RALL-E is inspired by chain-of-thought (CoT) prompting (Wei et al., 2022b). In CoT prompting, the language model generates an intermediate result, which serves as a condition for predicting the final outcome. This approach breaks down complex tasks into simpler steps, improving the robustness of language models, especially in challenging tasks like arithmetic (Wei et al., 2022b). To adapt CoT prompting to codec LMs, RALL-E first predicts prosody tokens (pitch and duration) before generating speech tokens, stabilizing the prosody. Given an input sentence, RALL-E initially predicts phoneme-level pitch and duration, then conditions the generation of speech tokens on both the input phonemes and the predicted prosody tokens. Furthermore, RALL-E leverages the predicted duration to mask irrelevant phonemes and prosody tokens during the computation of self-attention weights, ensuring the codec LM focuses on the relevant phonemes and prosody when predicting each speech token. We use VALL-E (Wang et al., 2023a), a recent powerful AR TTS method based on codec LMs, as the base model for applying our method, and conduct experiments to compare RALL-E with VALL-E and previous approaches aimed at improving the robustness of codec LMs. Comprehensive objective and subjective evaluations demonstrate that RALL-E significantly enhances the robustness of AR TTS based on codec LMs, reducing the WER on the LibriSpeech (Panayotov et al., 2015) test-clean set from 5.6% (w/o reranking) and 1.7% (with reranking) to 2.5% and 1.0%, respectively. Furthermore, we evaluate RALL-E on 50 particularly challenging sentences. As shown in Table 1, compared to VALL-E, RALL-E dramatically reduces the error rate from 68% to 4% by eliminating almost all types of errors, demonstrating its superior robustness (see Section 4.4 for more details). The contributions of this work are summarized as follows:

- We present RALL-E, a robust codec language modeling method with chain-of-thought prompting for TTS. RALL-E improves the robustness of codec LMs by (1) incorporating prosody tokens as chain-of-thought prompts to stabilize speech token generation, and (2) using duration-guided masking to enhance the alignment between phoneme and speech tokens.
- We conduct comprehensive objective and subjective evaluations. Experimental results demonstrate that RALL-E achieves significantly better robustness compared to the baseline VALL-E and two prior methods.
- We further evaluate RALL-E on sentences that are particularly difficult to synthesize for TTS based on codec LMs. The results show that RALL-E correctly synthesizes these challenging sentences, reducing the error rate from 68% to 4% compared to VALL-E, approaching the performance of non-autoregressive TTS.

Audio samples can be found at `https://ralle-demo.github.io/RALL-E`

2 Related work

TTS based on codec LMs Several recent works have adopted codec LMs to model TTS (Wang et al., 2023a; Yang et al., 2023; Kharitonov et al., 2023), utilizing decoder-only architecture based on Transformer (Vaswani et al., 2017). In these models, text and speech tokens are concatenated and fed into a single Transformer, with the entire model trained on a next-token prediction task, similar to a language model. TTS systems based on codec LMs are typically trained on tens of thousands of hours of speech data and consist of hundreds of millions of parameters. This allows them to leverage the emergent capabilities of large language models (LLMs), such as in-context learning (Wei et al., 2022a), enabling zero-shot TTS (Wang et al., 2023a). Additionally, recent works (Rubenstein et al., 2023; Wang et al., 2023b; Yang et al., 2023) have demonstrated that the decoder-only architecture can be extended to learn multiple tasks.

Robust autoregressive TTS The robustness of AR TTS is a popular topic extensively studied in the literature. For encoder-decoder AR TTS, several prior works have improved robustness by enforcing monotonicity in the attention weights (Zhang et al., 2018; He et al., 2019; Chen et al., 2020). This approach effectively stabilizes the alignment between text and speech. Additionally, Shen et al. (2020) introduced Non-Attentive Tacotron, which replaces the attention mechanism with a duration predictor to pre-determine the alignment before decoding. In decoder-only TTS, the attention mechanism differs in that the attention weights are computed simultaneously for text and context, meaning the attention weights are not required to be monotonic. Song et al. (2024) proposed ELLA-V, which interleaves phonemes and speech tokens by inserting a phoneme token and a special `EndOfPhone` (EOP) token at the beginning and end of the speech tokens corresponding to each phoneme. While these tokens indicate the duration of each phoneme, this implicit approach entangles the prediction of speech tokens and duration. In contrast, RALL-E disentangles the prediction of duration and speech tokens by first predicting the duration for all phonemes before generating the speech tokens, offering better controllability over the generation process. Du et al. (2024) proposed VALL-T, which uses an unsupervised transducer loss (Graves, 2012) to implicitly model phoneme duration. Although VALL-T does not rely on external alignment tools during training, its training process is significantly slower, as the transducer loss requires a forward pass for each phoneme. Furthermore, like ELLA-V, VALL-T also entangles the predictions of duration and speech tokens, resulting in less controllability compared to RALL-E.

3 RALL-E

The overview of RALL-E is illustrated in Figure 1. The core idea of RALL-E is to use CoT prompting, which generates intermediate results to assist and stabilize the generation of speech tokens, thereby improving the robustness of codec LMs. To achieve this, we first propose predicting two types of phoneme-level prosody tokens: pitch and duration, before generating the speech tokens. These prosody tokens are modeled together with the speech tokens within a single Transformer, allowing them to directly influence the predicted speech tokens' duration and pitch. To further leverage the predicted duration and improve robustness, we introduce duration-guided masking, which enhances the alignment between speech tokens, phonemes, and prosody tokens learned by the language model. At each step of decoding speech tokens, RALL-E masks irrelevant phonemes and prosody tokens based on the duration information, ensuring that the model focuses on the most relevant inputs for synthesizing the current speech token.

In the following sections, we first briefly introduce VALL-E, as RALL-E is implemented on top of it in our experiments. We then provide a detailed formulation and explanation of RALL-E. It is important to note that while we use VALL-E to demonstrate our method, the proposed approach can be applied to any decoder-only AR TTS model.

3.1 Preliminary: VALL-E

We adopt most of the symbols and notation from the original VALL-E paper (Wang et al., 2023a) for ease of reading. Readers are encouraged to refer to the original paper for additional details.

VALL-E is a decoder-only TTS system that utilizes two Transformers (Vaswani et al., 2017) to predict speech tokens from text input. The speech tokens are extracted using EnCodec (Défossez et al., 2022), a neural audio codec based on residual vector quantization (RVQ) (Zeghidour et al., 2021), which converts continuous speech signals into discrete tokens. Once the discrete tokens are predicted, the corresponding waveforms can be reconstructed by feeding them into the EnCodec decoder. An RVQ typically consists of N quantization layers ($N = 8$ in VALL-E), meaning that at each time step, the encoded speech has N tokens. Formally, for a given speech signal $\mathbf{y}$ and its transcription $\mathbf{x}$, the discrete speech token matrix $\mathbf{C}$ encoded by the codec has a shape of $T \times N$, where T is the total number of time steps. In addition to $\mathbf{x}$, to clone a speaker's voice and utilize the in-context learning ability of LMs, VALL-E receives a short prompt $\tilde{\mathbf{C}}^{T' \times N}$ as input before predicting $\mathbf{C}$. Hence, VALL-E models the following distribution:

$$\mathbb{P}(\mathbf{C} \mid \mathbf{x}, \tilde{\mathbf{C}}). \tag{1}$$

VALL-E predicts speech tokens hierarchically, with the tokens from the first layer of the RVQ predicted by an AR Transformer, and the tokens from the remaining layers predicted by a non-autoregressive (NAR) Transformer. This approach is motivated by the structure of RVQ, where higher layers encode residual information not captured by the lower layers. Consequently, the tokens from the first layer contain the majority of the waveform's information, while the amount of information encoded by the higher layers decreases progressively. The AR Transformer takes as input the phoneme sequence $\mathbf{x}$ and the speech tokens from the first layer of the prompt $\tilde{\mathbf{c}}_{:,1}$, and sequentially predicts the target speech tokens of the first layer $\mathbf{c}_{:,1}$. Specifically, it models the following distribution:

$$\mathbb{P}(\mathbf{c}_{:,1} \mid \mathbf{x}, \tilde{\mathbf{c}}_{:,1}; \theta_{AR}) = \prod_{t=1}^{T} \mathbb{P}(\mathbf{c}_{t,1} \mid \mathbf{x}, \mathbf{c}_{<t,1}, \tilde{\mathbf{c}}_{:,1}; \theta_{AR}), \tag{2}$$

where θ_{AR} is the trainable parameters of the AR Transformer. The NAR Transformer predicts all target speech tokens $\mathbf{c}_{:,j}$ of the jth layer simultaneously, conditioned on the phoneme sequence $\mathbf{x}$, the prompt $\tilde{\mathbf{C}}$, and the target speech tokens $\mathbf{c}_{:,<j}$ from all layers lower than j, i.e. models the following distribution:

$$\mathbb{P}(\mathbf{c}_{:,2:N} \mid \mathbf{x}, \tilde{\mathbf{C}}; \theta_{NAR}) = \prod_{j=2}^{N} \mathbb{P}(\mathbf{c}_{:,j} \mid \mathbf{x}, \mathbf{c}_{:,<j}, \tilde{\mathbf{C}}; \theta_{NAR}), \tag{3}$$

where θ_{NAR} is the trainable parameters of the NAR Transformer. By combining Eq. 2 and 3, VALL-E decomposes Eq. 1 into the following form:

$$\mathbb{P}(\mathbf{C} \mid \mathbf{x}, \tilde{\mathbf{C}}) = \mathbb{P}(\mathbf{c}_{:,1} \mid \mathbf{x}, \tilde{\mathbf{c}}_{:,1}; \theta_{AR})\mathbb{P}(\mathbf{c}_{:,2:N} \mid \mathbf{x}, \tilde{\mathbf{C}}; \theta_{NAR}). \tag{4}$$

It is noteworthy that both Transformers share the same architecture but differ in their attention masks. Specifically, the AR Transformers uses a unidirectional mask so that $\mathbf{c}_{t,1}$ can only attend to previous tokens $\mathbf{c}_{<t,1}$, while the NAR Transformer uses a bidirectional mask.

3.2 Prosody tokens as chain-of-thought prompts

One challenge with TTS based on codec LMs is that it directly generates speech from phonemes without controlling prosody features such as pitch and duration, often resulting in unstable prosody. A similar issue was observed in Wei et al. (2022b), where the authors found that LLMs struggle to solve complex tasks like arithmetic without guidance and proposed CoT prompting as a solution. The core idea of CoT prompting is to break down a complex task into simpler steps, allowing the LLM to leverage intermediate results to arrive at the final answer. As demonstrated in Wei et al. (2022b), CoT prompting significantly improves the accuracy of LLMs on complex tasks. Inspired by this, we

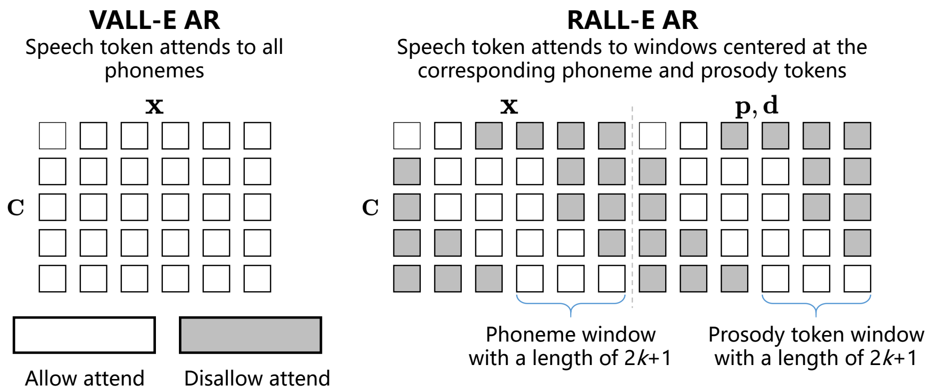

Figure 2: A comparison between how speech token attends to phonemes in the AR Transformer of VALL-E and RALL-E. Here k is set to 1.

adapt CoT prompting to codec LMs by generating intermediate prosody tokens before generating speech tokens to improve robustness. In RALL-E, we incorporate pitch and duration into the AR Transformer of VALL-E. First, we obtain the alignment between phonemes and speech tokens and extract the pitch value for each speech token. Next, we compute the phoneme-level pitch based on the duration and linearly quantize it into M_p buckets. For duration, we define a maximum value M_d, with all duration values exceeding M_d truncated to this maximum. RALL-E predicts both prosody tokens before predicting the speech tokens in a CoT style. Formally, let $\mathbf{p}$ and $\mathbf{d}$ represent the discrete pitch and duration sequences of the target speech tokens $\mathbf{C}$, and $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{d}}$ denote those of the prompt $\tilde{\mathbf{C}}$, we model the following distribution:

$$\mathbb{P}(\mathbf{p}, \mathbf{d} \mid \mathbf{x}, \tilde{\mathbf{p}}, \tilde{\mathbf{d}}; \theta_{AR}) = \prod_{t=1}^{L} \mathbb{P}(p_t, d_t \mid \mathbf{x}, \mathbf{p}_{<t}, \mathbf{d}_{<t}, \tilde{\mathbf{p}}, \tilde{\mathbf{d}}; \theta_{AR}), \tag{5}$$

where L is the length of $\mathbf{x}$. In practice, p_t and d_t are predicted by two separate heads, and their embeddings are summed and fed into the model for the next step's prediction. RALL-E then uses $\mathbf{p}$ and $\mathbf{d}$ as conditions for predicting speech tokens, modifying Eq. 2 as follows:

$$\mathbb{P}(\mathbf{c}_{:,1} \mid \mathbf{x}, \tilde{\mathbf{c}}_{:,1}, \mathbf{p}, \tilde{\mathbf{p}}, \mathbf{d}, \tilde{\mathbf{d}}; \theta_{AR}) = \prod_{t=1}^{T} \mathbb{P}(\mathbf{c}_{t,1} \mid \mathbf{x}, \mathbf{c}_{<t,1}, \tilde{\mathbf{c}}_{:,1}, \mathbf{p}, \tilde{\mathbf{p}}, \mathbf{d}, \tilde{\mathbf{d}}; \theta_{AR}). \tag{6}$$

The log-likelihood of these two distributions is jointly optimized by the AR Transformer. Although the proposed method introduces L additional decoding steps, since $L \ll T$, the impact on efficiency is minimal. See Section 4.5 for more details.

For the NAR Transformer, we simply sum the embeddings of the phoneme, pitch, and duration as the input. This modifies Eq. 3 to:

$$\mathbb{P}(\mathbf{c}_{:,2:N} \mid \mathbf{x}, \tilde{\mathbf{C}}, \mathbf{p}, \tilde{\mathbf{p}}, \mathbf{d}, \tilde{\mathbf{d}}; \theta_{NAR}) = \prod_{j=2}^{N} \mathbb{P}(\mathbf{c}_{:,j} \mid \mathbf{x}, \mathbf{c}_{:,<j}, \tilde{\mathbf{C}}, \mathbf{p}, \tilde{\mathbf{p}}, \mathbf{d}, \tilde{\mathbf{d}}; \theta_{NAR}). \tag{7}$$

3.3 Enhancing alignment with duration-guided masking

As illustrated on the left side of Figure 2, in the AR Transformer of VALL-E, each speech token attends to all phonemes, meaning the alignment between phonemes and speech tokens is implicitly modeled by the self-attention mechanism. This can lead to imprecise alignment, resulting in errors such as word omissions or hallucinations. While RALL-E introduces prosody CoT prompting to guide and stabilize generation, we still observe alignment issues in the experiments. To address this, we propose duration-guided masking, which fully leverages the intermediate duration predictions to enhance alignment and further improve robustness.

As illustrated on the right side of Figure 2, in the proposed duration-guided masking, each speech token is restricted to attend only to a phoneme (or prosody token) window centered around the corresponding phoneme (or prosody token). The window size is defined as k, meaning each speech token can attend to $2k+1$ phonemes and $2k+1$ prosody tokens. All phonemes and prosody tokens outside this window are masked out, with their attention weights set to zero. When $k=0$, each speech token strictly attends to its corresponding phoneme. In an ideal scenario with perfect alignment, this would suffice. However, our preliminary experiments revealed that the alignment tool usually made errors. Therefore, we relax this restriction by allowing speech tokens to also attend to neighboring phonemes. This design is further justified by the fact that the pronunciation of a phoneme often depends on adjacent phonemes. As demonstrated in Section 4.3 and Appendix A, the experimental results confirm the effectiveness of this design. For the NAR Transformer, however, we observed minimal improvement when applying the masking strategy in preliminary experiments. Thus, we apply the masking strategy only to the AR Transformer.

The general inference procedure follows VALL-E (Wang et al., 2023a) with two differences. First, before sampling the speech tokens $\mathbf{c}_{:,1}$, the prosody tokens $\mathbf{p}$ and $\mathbf{d}$ are sampled, conditioned on the phoneme sequence $\mathbf{x}$ and the acoustic prompt $\tilde{\mathbf{p}}, \tilde{\mathbf{d}}$. Second, while typical LMs rely on a special token <eos> to signal the end of generation, since the total duration $D = \sum_{t=1}^{L} d_t$ is known, we propose a duration-guided inference method that forces the inference to stop at the D-th step. This approach ensures no phonemes are omitted or repeated, as the inference continues even if the <eos> token is predicted before the D-th step, and stops at the right step as guided by the predicted duration. In addition, we use KV caching to accelerate the inference efficiency of the AR Transformer.

4 Experiments

4.1 Setup

Data We use the English subset of the multilingual LibriSpeech (MLS) corpus (Pratap et al., 2020), which contains approximately 44K hours of speech data from 5,490 speakers, as training data. The test-clean set from the LibriSpeech corpus Panayotov et al. (2015) is used for evaluation. Following Wang et al. (2023a), we select only utterances with lengths between 4 and 10 seconds, resulting in 1,205 utterances for testing. For each test utterance, we randomly select another utterance from the same speaker, using the first 3 seconds as the prompt. All speech data is sampled at 16 kHz. Transcriptions are converted into phonemes using a grapheme-to-phoneme tool (Sun et al., 2019), and frame-level pitch values are extracted using the WORLD vocoder (Morise et al., 2016). Alignments between phoneme sequences and speech tokens are obtained using our internal alignment tool. The maximum duration value M_d is set to 32. The phoneme-level pitch values are calculated based on the alignments. The number of quantization buckets M_p for pitch is set to 256.

Model configuration We use SoundStream (Zeghidour et al., 2021) as the speech codec to extract speech tokens and decode waveforms from the tokens. The architecture follows the original design, with the number of quantization layers N in the RVQ set to 16. The codec language model is based on VALL-E (Wang et al., 2023a), where both the AR and NAR models consist of a 12-layer Transformer (Vaswani et al., 2017). The Transformer uses 1024-dimensional token embeddings, sinusoidal positional embeddings, 4096-dimensional feed-forward layers, and a dropout rate of 0.1. The window size k is set to 1 unless otherwise stated (see Appendix A for a detailed explanation of how this value was chosen).

Training and inference The SoundStream codec is trained on 8 NVIDIA V100 GPUs with a batch size of 200 per GPU. We use AdamW (Loshchilov & Hutter, 2018) as the optimizer with a learning rate of $2e$-4. The model converges after approximately 440K steps. The AR and NAR Transformers are trained separately on 16 AMD MI200 GPUs with a batch size of 7,000 speech tokens per GPU. AdamW is also used as the optimizer, and the scheduled inverse square root learning rate is applied, with 30K warm-up steps and a peak learning rate of $5e$-4. Both Transformers converge after approximately 500K steps.

We adopt nucleus sampling (Holtzman et al., 2019) as the sampling method for the AR Transformer. For the predicted probability distribution, nucleus sampling selects a token set with the highest probabilities whose cumulative probability exceeds a hyperparameter ρ, and randomly samples from this set. Note that ρ can differ for the sampling of pitch, duration, and speech tokens, resulting in three hyperparameters: ρ_p, ρ_d, and ρ_c for pitch, duration, and speech tokens, respectively. Unless otherwise specified, we set $\rho_p = \rho_d = \rho_c = 0.9$ in the following experiments. For the NAR Transformer, we select the token with the highest probability without sampling.

Table 2: Main results of RALL-E on the LibriSpeech test set with 1,205 utterances. **Bold** indicates the best score. The WER in parentheses is obtained from the HuBERT model used in the original VALL-E paper (Wang et al., 2023a). † indicates the results of VALL-E trained on LibriLight, while ‡ indicates the results of VALL-E trained on the English subset of MLS. We get the results of VALL-T (500 samples) (Du et al., 2024) and ELLA-V (912 samples) (Song et al., 2024) from the authors. RALL-E (912) refers to results computed on the same test set as ELLA-V including 912 samples.

	WER% (↓)	WER-R% (↓)	UTMOS (↑)	SIM (↑)	Sub (↓)	Del (↓)	Ins (↓)
GT	1.8 (2.1)	-	4.1	0.69	1.4	0.2	0.2
VALL-E†	- (5.9)	-	-	0.58	-	-	-
VALL-E‡	5.6 (6.3)	1.7	3.9	0.49	2.8 (3.6)	1.5 (1.4)	1.3 (1.3)
ELLA-V (912)	2.8 (4.1)	0.8	3.7	0.42	2.2 (3.4)	0.4 (0.4)	**0.2** (0.3)
VALL-T (500)	3.9 (5.4)	-	**4.0**	0.46	2.4 (3.6)	1.3 (1.6)	**0.2** (0.2)
RALL-E (912)	2.3 (2.6)	0.8	4.0	0.49	1.4 (2.0)	0.6 (0.3)	0.3 (0.3)
RALL-E	**2.5** (**2.8**)	**1.0**	**4.0**	0.49	**1.7** (2.2)	0.6 (0.3)	0.3 (0.3)

Baseline methods We use VALL-E (Wang et al., 2023a) as the baseline method, implementing and training it on our dataset. Additionally, we compare RALL-E with two previous works: VALL-T (Du et al., 2024) and ELLA-V (Song et al., 2024). VALL-T was trained on the LibriTTS (Zen et al., 2019) corpus, which contains 520 hours of speech data, and we obtained 500 synthesized samples selected from the test-clean set of LibriTTS from the authors. ELLA-V (Song et al., 2024) was trained on the LibriSpeech Panayotov et al. (2015) corpus, which consists of 960 hours of speech data. We requested the authors to run ELLA-V on our test set and received 912 samples. We do not use the results from the original ELLA-V paper (Song et al., 2024), as the continual generation method used in ELLA-V can yield significantly better WER compared to non-continual generation, as noted by Wang et al. (2023a).

Objective metrics We use the following objective metrics:

- **Word error rate (WER)**. We transcribe the synthesized samples by a large Conformer-based (Gulati et al., 2020) ASR model[1], which is trained on a large collection of speech corpora including LibriSpeech (Panayotov et al., 2015). The WER is then computed by comparing the recognized transcriptions with the ground truth (GT) transcriptions. Additionally, we report WERs computed using transcriptions recognized by a HuBERT model[2] (Hsu et al., 2021), which is trained on Libri-Light (Kahn et al., 2020) and fine-tuned on LibriSpeech (Panayotov et al., 2015). We regard the WERs from the Conformer-based model as the primary scores, as it provides better performance than the HuBERT model, although the HuBERT model is used in the original VALL-E paper (Wang et al., 2023a).
- **Reranked WER (WER-R)**. For each test utterance, we generate 5 samples and select the one with the lowest edit distance to the GT transcription to compute WER. This metric serves as an upper bound for performance, while regular WER reflects the average performance.
- **Substitution (Sub)**, **Deletion (Del)**, and **Insertion (Ins)** computed by the edit distance algorithm. These three metrics are by-products of WER calculation. They provide insights into specific error types made by the TTS model. Typically, Sub refers to mispronunciations, Del indicates word omissions, and Ins refers to word repetitions or hallucinations.
- **UTMOS** (Saeki et al., 2022), which is a powerful automatic speech quality assessment model used to evaluate speech naturalness.
- **Speaker similarity (SIM)** defined as the cosine similarity between the speaker embeddings of the prompt and the synthesized utterance. Following VALL-E (Wang et al., 2023a), we use the `wavlm_large_finetune` checkpoint from WavLM-TDNN[3], a speaker verification model based on WavLM (Chen et al., 2022), to extract the speaker embeddings.

[1] `https://huggingface.co/nvidia/stt_en_conformer_transducer_xlarge`

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

[3] `https://github.com/microsoft/UniSpeech/tree/main/downstreams/speaker_verification`

Table 4: Results of ablation studies. **Bold** indicates the best score.

	WER%(↓)	UTMOS (↑)	Sub (↓)	Del (↓)	Ins (↓)
RALL-E	**2.5**	**4.00**	**1.7**	**0.5**	**0.3**
w/o pitch	2.6	3.96	1.8	0.5	0.3
w/o window masking	2.7	3.84	1.8	0.6	0.3
w/o duration-guided masking	3.2	3.88	2.0	0.8	0.5
w/o duration CoT prompting	13.4	3.52	7.8	4.1	1.5

Subjective metrics We use two common subjective metrics: comparative mean opinion score (CMOS) and similarity mean opinion score (SMOS) to evaluate speech naturalness and speaker similarity, respectively. CMOS measures the performance difference between two systems on a scale from -3 (the new system is much worse than the old system) to 3 (the new system is much better than the old system). SMOS is rated on a 5-point scale, where higher values indicate better speaker similarity between the synthesized and GT samples.

4.2 Main results

We first evaluate the overall performance of RALL-E on the full LibriSpeech test set with 1,205 utterances. The results are shown in Table 2. It can be observed that RALL-E outperforms all other methods in terms of WER. Notably, the reranked WER (WER-R) of RALL-E is even lower than the WER of GT. Compared to the baseline VALL-E method, RALL-E achieves a 55% relative improvement in WER and a 41% relative improvement in WER-R, showing the superior robustness of the proposed method. This is further supported by the reduction in all three error types, where RALL-E consistently reduces substitution, deletion, and insertion errors from $2.8/1.5/1.3$ to $1.7/0.6/0.3$, respectively. In addition, the higher UTMOS score for RALL-E compared to VALL-E indicates that RALL-E synthesizes speech with better naturalness, highlighting its effectiveness in stabilizing speech prosody. Regarding speaker similarity, both RALL-E and VALL-E outperform previous methods, possibly due to the larger training dataset we used. However, the original VALL-E reports a significantly higher SIM score (0.58) than other methods. One possible reason is that the SIM score in the original VALL-E is computed between the synthesized utterance and the prompt resynthesized by the codec, rather than the GT prompt. We also note that VALL-T shows slightly fewer insertion errors (0.2) compared to RALL-E (0.3). However, this may be attributed to the smaller test set used by VALL-T, which contains only 500 samples. As suggested by the result of RALL-E (912), computed on the 912 samples used by ELLA-V, fewer test samples often lead to a better WER.

Next, we conduct subjective tests to evaluate the performance of RALL-E. For the CMOS tests, we randomly select 20 samples from the test set, and for the SMOS test, we select 10 samples from distinct speakers. Two CMOS tests, each with 6 workers, are conducted on two pairs: (GT vs. RALL-E) and (VALL-E vs. RALL-E), with each utterance receiving 6 responses. Similarly, an SMOS test is conducted with 6 workers. The results are shown in Table 3. RALL-E achieves a higher CMOS score than VALL-E and even slightly outperforms the GT utterances, demonstrating its effectiveness in stabilizing prosody by incorporating prosody tokens as CoT prompts. In terms of SMOS, both methods perform similarly, which aligns with the SIM scores presented in Table 2.

Table 3: Results of subjective CMOS (v.s. RALL-E) and SMOS tests. **Bold** indicates the best score.

	CMOS	SMOS
GT	-0.02	4.23
VALL-E	-0.17	3.50
RALL-E	**0.00**	3.57

4.3 Ablation study

We conduct ablation experiments to analyze the contributions of each component in RALL-E. Specifically, we evaluate the following four settings: (1) w/o pitch that removes the pitch prompt; (2) w/o window masking where the window size k is set to 0; (3) w/o duration-guided masking that uses normal unidirectional autoregressive attention masks; (4) w/o duration CoT prompting that removes duration from the CoT and model it separately. In the w/o duration CoT prompting setting we use a separate 8-layer Transformer with 256-dimensional token embeddings and 8 self-attention heads to predict duration, while the masking strategy is still applied based on the duration.

The results are shown in Table 4. First, the result of w/o pitch demonstrates that including the pitch token helps to reduce mispronunciation. Second, the results of w/o window masking show that model performance degrades when the window size is set to 0, confirming the effectiveness of the window masking strategy. For a detailed study on the impact of window size k, refer to Appendix A. Third, w/o duration-guided masking shows consistently worse performance across all metrics, highlighting the value of the proposed duration-guided masking strategy. Finally, w/o duration CoT prompting exhibits the worst performance, despite using duration-guided masking. This is because the predicted duration from the independent Transformer fails to effectively guide the synthesized speech, and the masking strategy based on this predicted duration further disrupts inference by forcing the model to focus on possibly misaligned phonemes. This highlights the importance of incorporating duration into the CoT prompting. In summary, each component of RALL-E contributes to its robustness improvements, with CoT prompting emerging as the most critical element.

4.4 Evaluations on hard sentences

To further evaluate the robustness of RALL-E, we synthesize 50 particularly challenging sentences (see Appendix B for the transcripts of these sentences) using RALL-E and VALL-E. We manually evaluate the results since the WER computed on these sentences with a lot of numbers and symbols is imprecise. We categorize the possible errors into four types: mispronunciation, omission, repetition, and hallucination. Each utterance is synthesized 5 times, and the best version is selected. We count the frequency of each error type and calculate the overall sentence error rate, where each error type is counted only once per utterance. The results are shown in Table 1, with a powerful non-autoregressive TTS method, NaturalSpeech2 (Shen et al., 2023) included for reference. RALL-E significantly reduces the error rate from 68% to 4%, with only 2 mispronunciation errors, achieving performance close to the error-free NaturalSpeech2. This further highlights RALL-E's effectiveness in enhancing the robustness of TTS based on codec LMs. In particular, for very short sentences (e.g. a single letter like "A"), VALL-E often generates words not present in the input, leading to hallucination issues. Additionally, for sentences where words are repeated many times (e.g. "22222222"), VALL-E frequently makes errors by omitting or repeating the word. These issues demonstrate that codec LMs like VALL-E struggle with controlling the duration of synthesized speech and exhibit poor alignment between phonemes and speech tokens. In contrast, RALL-E improves controllability by introducing prosody tokens through CoT prompting and further enhances alignment with duration-guided masking. This effectively mitigates the common errors made by codec LMs. All in all, RALL-E demonstrates superior robustness in all evaluations. We strongly encourage readers to listen to the audio samples for a firsthand impression.

4.5 Efficiency analysis

We finally analyze the inference efficiency of RALL-E, as our method introduces L additional decoding steps in the AR Transformer. We randomly select 128 samples from the test set and approximate the real time factor (RTF) of the codec LM for both VALL-E and RALL-E on an NVIDIA V100 GPU. An RTF greater than one indicates that the model can process data in real time. The RTFs for VALL-E and RALL-E are $2.15\times$ and $1.94\times$, respectively. Both models achieve real-time data processing, and the slightly lower RTF of RALL-E is expected, as it reflects the trade-off for its improved robustness.

5 Conclusions

This paper presents RALL-E, a robust codec language modeling method for TTS, utilizing CoT prompting. To address the robustness issues in codec LMs, RALL-E (1) incorporates prosody tokens (pitch and duration) as CoT prompts to assist and stabilize the generation of speech tokens, and (2) introduces duration-guided masking, which directs the model's attention to the relevant phonemes and prosody tokens for each speech token. Comprehensive objective and subjective evaluations demonstrate that RALL-E significantly improves the robustness of codec LMs compared to the baseline VALL-E and two prior works. Additionally, RALL-E is able to accurately synthesize particularly challenging sentences for VALL-E, achieving an error rate as low as 4%, approaching the performance of non-autoregressive TTS models.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Mingjian Chen, Xu Tan, Yi Ren, Jin Xu, Hao Sun, Sheng Zhao, Tao Qin, and Tie-Yan Liu. Multi-speech: Multi-speaker text to speech with transformer. *arXiv preprint arXiv:2006.04664*, 2020.

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

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

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *Proc. ICML*, pp. 7480–7512. PMLR, 2023.

Chenpeng Du, Yiwei Guo, Hankun Wang, Yifan Yang, Zhikang Niu, Shuai Wang, Hui Zhang, Xie Chen, and Kai Yu. Vall-t: Decoder-only generative transducer for robust and decoding-controllable text-to-speech. *arXiv preprint arXiv:2401.14321*, 2024.

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

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech*, pp. 5036–5040, 2020. doi: 10.21437/Interspeech.2020-3015.

Mutian He, Yan Deng, and Lei He. Robust sequence-to-sequence acoustic modeling with stepwise monotonic attention for neural tts. *arXiv preprint arXiv:1906.00672*, 2019.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *Proc. ICLR*, 2019.

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.

Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Yanqing Liu, Yichong Leng, Kaitao Song, Siliang Tang, et al. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. *arXiv preprint arXiv:2403.03100*, 2024.

J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *Proc. ICASSP*, pp. 7669–7673, 2020. `https://github.com/facebookresearch/libri-light`.

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. *arXiv preprint arXiv:2302.03540*, 2023.

Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, et al. Voicebox: Text-guided multilingual universal speech generation at scale. *Advances in neural information processing systems*, 36, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. ICLR*, 2018.

Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE TRANSACTIONS on Information and Systems*, 99(7):1877–1884, 2016.

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.

Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. MLS: A Large-Scale Multilingual Dataset for Speech Research. In *Proc. Interspeech*, pp. 2757–2761, 2020. doi: 10.21437/Interspeech.2020-2826.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. *Proc. NeurIPS*, 32, 2019.

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. *arXiv preprint arXiv:2204.02152*, 2022.

Jonathan Shen, Ye Jia, Mike Chrzanowski, Yu Zhang, Isaac Elias, Heiga Zen, and Yonghui Wu. Non-attentive tacotron: Robust and controllable neural tts synthesis including unsupervised duration modeling. *arXiv preprint arXiv:2010.04301*, 2020.

Kai Shen, Zeqian Ju, Xu Tan, Yanqing Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian. Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. *arXiv preprint arXiv:2304.09116*, 2023.

Yakun Song, Zhuo Chen, Xiaofei Wang, Ziyang Ma, and Xie Chen. Ella-v: Stable neural codec language modeling with alignment-guided sequence reordering. *arXiv preprint arXiv:2401.07333*, 2024.

Hao Sun, Xu Tan, Jun-Wei Gan, Hongzhi Liu, Sheng Zhao, Tao Qin, and Tie-Yan Liu. Token-Level Ensemble Distillation for Grapheme-to-Phoneme Conversion. In *Proc. Interspeech 2019*, pp. 2115–2119, 2019. doi: 10.21437/Interspeech.2019-1208.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

Tianrui Wang, Long Zhou, Ziqiang Zhang, Yu Wu, Shujie Liu, Yashesh Gaur, Zhuo Chen, Jinyu Li, and Furu Wei. Viola: Unified codec language models for speech recognition, synthesis, and translation. *arXiv preprint arXiv:2305.16107*, 2023b.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022b.

Dongchao Yang, Jinchuan Tian, Xu Tan, Rongjie Huang, Songxiang Liu, Xuankai Chang, Jiatong Shi, Sheng Zhao, Jiang Bian, Xixin Wu, et al. Uniaudio: An audio foundation model toward universal audio generation. *arXiv preprint arXiv:2310.00704*, 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.

Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. *Proc. Interspeech*, 2019.

Jing-Xuan Zhang, Zhen-Hua Ling, and Li-Rong Dai. Forward attention in sequence-to-sequence acoustic modeling for speech synthesis. In *Proc. ICASSP*, pp. 4789–4793. IEEE, 2018.

A Window size study

We study the window size hyperparameter k used in the proposed duration-guided masking method. Basically, $2k + 1$ is the number of phonemes (prosody features) the model can attend during decoding. As mentioned in Section 3.3, the motivation of the window is to (1) increase context information received by the model during decoding and (2) improve the robustness of the proposed duration-based masking strategy since the extracted duration features can have errors during training and the predicted duration may not strictly correspond to the number of predicted speech tokens for each phoneme during inference. We suppose $k = 0$ will make RALL-E less robust, but large k will also make it difficult to learn the alignment between phonemes and speech tokens. Thus we study the optimal value of k. We train RALL-E and compute WER on the test set with $k = 0, 1, 2, 3, \infty$, in which $k = \infty$ means the window covers the whole phoneme sequence, i.e. no phoneme is masked during decoding. We hypothesize that diverse sampling will make the robustness problem more obvious, thus we perform nucleus sampling on each model with four settings: (1) $\rho_p = \rho_d = \rho_c = 0.9$; (2) $\rho_p = \rho_d = 0.9, \rho_c = 1.0$; (3) $\rho_p = \rho_d = 1.0, \rho_c = 0.9$; and (4) $\rho_p = \rho_d = 1.0, \rho_c = 1.0$. The results are illustrated in Figure 3. First, it can be observed that in every sampling setting the WER can be substantially improved by increasing k from 0 to 1, showing the effectiveness of the proposed window masking strategy. This observation also verifies the hypothesis that the more the sampling becomes diverse, the more the robustness problem becomes obvious. Second, the performance cannot be further improved by increasing k to values larger than 1, which verifies another hypothesis that large k makes it difficult to learn the alignment. When $k = \infty$ the model has to learn the alignment completely by itself, thus resulting in the worst WERs. Combining all results we conclude that the proposed window masking strategy can effectively improve WERs and the best performance is obtained when $k = 1$.

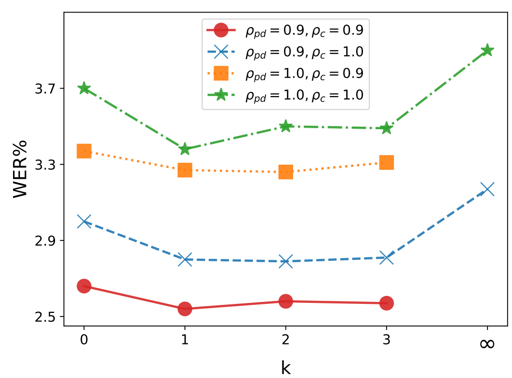

Figure 3: Results of the window size study. For simplicity, we use ρ_{pd} to refer to ρ_p and ρ_d together. $k = \infty$ means the window covers the whole phoneme sequence, i.e. no phoneme is masked. The model with $k = \infty$ fails to generate speech tokens when $\rho_c = 0.9$.

B Transcripts of the 50 hard sentences

We list the 50 hard sentences used in Section 4.4 below:

1. a
2. b
3. c
4. H

5. I
6. J
7. K
8. L
9. 22222222 hello 22222222
10. S D S D Pass zero - zero Fail - zero to zero - zero - zero Cancelled - fifty nine to three - two - sixty four Total - fifty nine to three - two -
11. S D S D Pass - zero - zero - zero - zero Fail - zero - zero - zero - zero Cancelled - four hundred and sixteen - seventy six -
12. zero - one - one - two Cancelled - zero - zero - zero - zero Total - two hundred and eighty six - nineteen - seven -
13. forty one to five three hundred and eleven Fail - one - one to zero two Cancelled - zero - zero to zero zero Total -
14. zero zero one , MS03 - zero twenty five , MS03 - zero thirty two , MS03 - zero thirty nine ,
15. 1b204928 zero zero zero zero zero zero zero zero zero zero zero zero zero one seven ole32 11
16. zero zero zero zero zero zero zero zero two seven nine eight F three forty zero zero zero zero zero six four two eight zero one eight
17. c five eight zero three three nine a zero bf eight FALSE zero zero zero bba3add2 - c229 - 4cdb -
18. Calendaring agent failed with error code 0x80070005 while saving appointment .
19. Exit process - break ld - Load module - output ud - Unload module - ignore ser - System error - ignore ibp - Initial breakpoint -
20. Common DB connectors include the DB - nine , DB - fifteen , DB - nineteen , DB - twenty five , DB - thirty seven , and DB - fifty connectors .
21. To deliver interfaces that are significantly better suited to create and process RFC eight twenty one , RFC eight twenty two , RFC nine seventy seven , and MIME content .
22. int1 , int2 , int3 , int4 , int5 , int6 , int7 , int8 , int9 ,
23. seven _ ctl00 ctl04 ctl01 ctl00 ctl00
24. Http0XX , Http1XX , Http2XX , Http3XX ,
25. config file must contain A , B , C , D , E , F , and G .
26. mondo - debug mondo - ship motif - debug motif - ship sts - debug sts - ship Comparing local files to checkpoint files ...
27. Rusbvts . dll Dsaccessbvts . dll Exchmembvt . dll Draino . dll Im trying to deploy a new topology , and I keep getting this error .
28. You can call me directly at four two five seven zero three seven three four four or my cell four two five four four four seven four seven four or send me a meeting request with all the appropriate information .
29. Failed zero point zero zero percent ¡ one zero zero one zero zero zero zero Internal . Exchange . ContentFilter . BVT ContentFilter . BVT_log . xml Error ! Filename not specified .
30. C colon backslash o one two f c p a r t y backslash d e v one two backslash oasys backslash legacy backslash web backslash HELP
31. src backslash mapi backslash t n e f d e c dot c dot o l d backslash backslash m o z a r t f one backslash e x five
32. copy backslash backslash j o h n f a n four backslash scratch backslash M i c r o s o f t dot S h a r e P o i n t dot
33. Take a look at h t t p colon slash slash w w w dot granite dot a b dot c a slash access slash email dot
34. backslash bin backslash premium backslash forms backslash r e g i o n a l o p t i o n s dot a s p x dot c s Raj , DJ ,
35. Anuraag backslash backslash r a d u r five backslash d e b u g dot one eight zero nine underscore P R two h dot s t s contains
36. p l a t f o r m right bracket backslash left bracket f l a v o r right bracket backslash s e t u p dot e x e
37. backslash x eight six backslash Ship backslash zero backslash A d d r e s s B o o k dot C o n t a c t s A d d r e s
38. Mine is here backslash backslash g a b e h a l l hyphen m o t h r a backslash S v r underscore O f f i c e s v r
39. h t t p colon slash slash teams slash sites slash T A G slash default dot aspx As always , any feedback , comments ,
40. two thousand and five h t t p colon slash slash news dot com dot com slash i slash n e slash f d slash two zero zero three slash f d
41. backslash i n t e r n a l dot e x c h a n g e dot m a n a g e m e n t dot s y s t e m m a n a g e

42. I think Rich's post highlights that we could have been more strategic about how the sum total of XBOX three hundred and sixtys were distributed .
43. 64X64 , 8K , one hundred and eighty four ASSEMBLY , DIGITAL VIDEO DISK DRIVE , INTERNAL , 8X ,
44. So we are back to Extended MAPI and C++ because . Extended MAPI does not have a dual interface VB or VB .Net can read .
45. Thanks , Borge Trongmo Hi gurus , Could you help us E2K ASP guys with the following issue ?
46. Thanks J RGR Are you using the LDDM driver for this system or the in the build XDDM driver ? 12
47. Btw , you might remember me from our discussion about OWA automation and OWA readiness day a year ago .
48. empidtool . exe creates HKEY_CURRENT_USER Software Microsoft Office Common QMPersNum in the registry , queries AD , and the populate the registry with MS employment ID if available else an error code is logged .
49. Thursday, via a joint press release and Microsoft AI Blog, we will announce Microsoft's continued partnership with Shell leveraging cloud, AI, and collaboration technology to drive industry innovation and transformation.
50. Actress Fan Bingbing attends the screening of 'Ash Is Purest White (Jiang Hu Er Nv)' during the 71st annual Cannes Film Festival