Attention-Constrained Inference for Robust Decoder-Only Text-to-Speech

Hankun Wang[1], *Chenpeng Du*[1], *Yiwei Guo*[1], *Shuai Wang*[2], *Xie Chen*[1], *Kai Yu*[1]

[1]MoE Key Lab of Artificial Intelligence, AI Institute
X-LANCE Lab, Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
[2]Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong, Shenzhen, China
{wanghankun, kai.yu}@sjtu.edu.cn

Abstract

Recent popular decoder-only text-to-speech models are known for their ability of generating natural-sounding speech. However, such models sometimes suffer from word skipping and repeating due to the lack of explicit monotonic alignment constraints. In this paper, we notice from the attention maps that some particular attention heads of the decoder-only model indicate the alignments between speech and text. We call the attention maps of those heads Alignment-Emerged Attention Maps (AEAMs). Based on this discovery, we propose a novel inference method without altering the training process, named Attention-Constrained Inference (ACI), to facilitate monotonic synthesis. It first identifies AEAMs using the Attention Sweeping algorithm and then applies constraining masks on AEAMs. Our experimental results on decoder-only TTS model VALL-E show that the WER of synthesized speech is reduced by up to 20.5% relatively with ACI while the naturalness and speaker similarity are comparable.

Index Terms: Decoder-only Text-to-Speech, Self-Attention Mechanisms, Training-Free Optimization

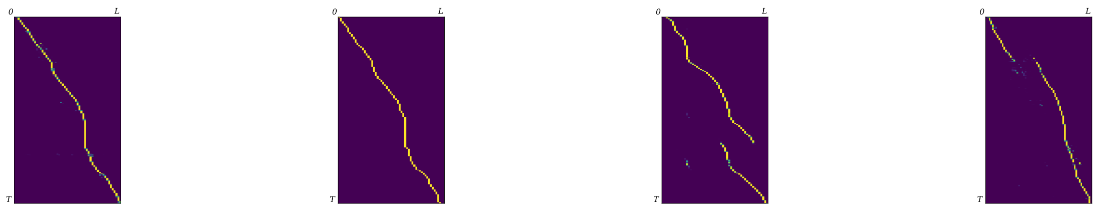
1. Introduction

In the field of text-to-speech (TTS), numerous breakthroughs have been achieved based on deep neural networks [1, 2, 3]. TTS is a sequence-to-sequence task, where the text and speech possess a monotonic aligned nature, and the audio frame sequence is usually much longer than the text sequence. Prior methods typically integrate an explicit duration module to fill this modality gap. These models obtain the target durations via conducting traditional alignment algorithms either externally [2, 4, 5] or internally [6, 7], and feed them as model input for training. At the training stage, using the target durations, the encoded text sequence is expanded to the length of speech frames and then serves as the decoder inputs. Meanwhile, the duration module is trained to predict the target durations. At the inference stage, the predicted durations are employed to expand the encoded text sequence for further decoding. In summary, monotonic alignment constraints are explicitly involved in this framework to avoid robustness issues like word skipping, repeating, or mispronouncing.

With the widespread use of discrete audio tokens [8, 9, 10], the research paradigm in language models [11, 12] has shown a profound impact on speech modeling and synthesis [13, 14]. Motivated by recent advancements in auto-regressive (AR) models employing decoder-only architectures for text generation [12, 15], several studies, such as VALL-E [3] and BASE TTS [16] apply similar architectures to TTS task. These studies demonstrate the remarkable capacity of decoder-only architectures in producing natural-sounding speech. Unfortunately, robustness is a challenging issue for such models, due to the lack of explicit monotonic alignment constraints and the gap between teacher-forced training and AR inference [17]. To mitigate this, VALL-T [18] proposes a decoder-only transducer for imposing monotonic alignment constraints. However, this method requires training a new model under a different criterion. Drawing from experience with prior TTS systems, we believe that existing decoder-only TTS models should also implicitly learn some form of alignment via the internal attention black box during training. If we can identify the internal representation of alignment within the model and apply forced alignment constraints during inference, then building a lightweight method to enhance model robustness becomes feasible. A key insight is that in the neural TTS model Tacotron [1, 19] with an encoder-decoder architecture, researchers discovered diagonal speech-text alignments within its sole attention module's attention map. This suggests that alignments in decoder-only models might also manifest within their attention maps. However, in decoder-only TTS models that employ massive attentions across various layers and heads [3, 20], different attention maps are responsible for different functionalities and are not always diagonal. This complicates both awareness and control of the alignment from the attention maps.

In our initial experiments on the popular decoder-only TTS model VALL-E, we observed from the attention maps that certain attention heads within a specific layer exhibit diagonal patterns, indicating the alignment between speech and text. The attention maps here specifically represent the attention from speech to text, where speech tokens function as queries while text tokens act as keys. We call the attention maps of those heads Alignment-Emerged Attention Maps (AEAMs, cf. Fig. 1a). In the circumstances of repeating and skip generation, the alignments on AEAMs also exhibit corresponding patterns (cf. Fig. 1c and 1d). Based on this discovery, we introduce a novel inference method named Attention-Constrained Inference (ACI), which facilitates monotonic synthesis without changing the model structure and the training process. It first identifies all AEAMs among the attention maps of all layers and heads using the Attention Sweeping algorithm and then applies constraining masks (CMasks) on AEAMs to guide the monotonic generation process. The main contributions of our work are listed below:

1. We discover that among the massive self-attention modules in the whole decoder, some particular heads in a particular early layer are responsible for emerging alignments in their attention maps (i.e., AEAMs).
2. We propose a training-free method, Attention-Constrained Inference (ACI), which seamlessly works on top of existing decoder-only TTS models, to detect AEAMs of a model and apply CMasks on AEAMs at the inference stage to guide the alignment to be monotonic and realize robust synthesis.



(a) *AEAM values for paired data.* (b) *Viterbi-produced alignment.* (c) *AEAM in a repeat generation.* (d) *AEAM in a skipping generation.*

Figure 1: *Illustration of real attention alignments. (a), (c) and (d) are attention heatmaps from the same AEAM of the same model, with different inputs. (a) is for a text-speech pair directly picked from the test set. With the same inputs, (b) shows the reference alignment produced by the Viterbi algorithm. (c) and (d) illustrate the AEAM when some text tokens are repeated or skipped in synthesis.*

3. The word error rate (WER) of synthesized speech is reduced by up to 20.5% relatively with ACI on VALL-E models of various configurations, while the naturalness and speaker similarity are comparable.

2. Attention-Constrained Inference

In this section, we introduce our Attention-Constrained Inference (ACI) for robust decoder-only TTS, using the Attention Sweeping algorithm to pinpoint AEAMs among all attention maps, and the attention constraining strategies to apply CMasks to AEAMs, helping the model to synthesize monotonically without changing or retraining the model.

2.1. Notations

We denote the discrete tokens of a speech utterance as $\boldsymbol{y} = \{y_1, y_2, \ldots, y_T\}$ and the transcription text sequence (usually phonemes) as $\boldsymbol{x} = \{x_1, x_2, \ldots, x_L\}$. Let the number of layers and heads of VALL-E AR decoder be N_{L} and N_{H}. The AR decoder essentially receives a speech sequence and a text sequence to predict the next speech token. We further define $\mathcal{E}(\boldsymbol{a}, \boldsymbol{b}, t) = \frac{1}{t}\sum_{i=1}^{t}(a_i - b_i)^2$, where the numerical sequences $\boldsymbol{a}$ and $\boldsymbol{b}$ have equal lengths T and $T \geq t$.

2.2. Attention Sweeping

We introduce the Attention Sweeping algorithm designed to identify the heads responsible for the emergence of alignments. This method is initiated by subjecting the AR decoder to conduct a forward pass with several ground-truth text-speech pairs as inputs, aiming to obtain attention maps for every head within all layers. We then analyze these maps to identify AEAMs and record the corresponding heads for subsequent inference.

An AEAM is expected to manifest the following patterns: (1) the attention values of all rows (speech tokens) are concentrated on very few columns (text tokens), and (2) as speech tokens are generated, focused text tokens shift towards the sentence end, with the focus positions strongly correlated with the real alignment. We respectively design the entropy cost and the alignment cost to quantify the extent of conformity an attention map has with the two patterns mentioned above. Note that calculating the alignment cost requires a reference alignment of the input ground-truth utterance, obtainable via a forced alignment algorithm such as Viterbi. However, such reference alignments are solely requisite for the Attention Sweeping stage and are unnecessary for subsequent inference employing attention constraining. Additionally, before calculating the costs, all attention maps should be normalized such that each row sums up to 1. The normalized attention map is denoted as $\mathbf{M}^{T\times L}$.

Entropy Cost. Since the sum of each row in the normalized matrix $\mathbf{M}$ equals 1, we directly use an entropy-like cost to measure the concentration of attention weights in rows of $\mathbf{M}$:

$$\mathcal{C}_{\mathrm{E}}(\mathbf{M}) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\ell=1}^{L} M_{t,\ell} \cdot \log M_{t,\ell}. \qquad (1)$$

Based on the properties of entropy, the more concentrated the attention values are in each row, the lower the entropy cost.

Alignment Cost. To further verify whether an attention map conforms to the alignment, we first calculate the average attention position for each row $m_t = \sum_{\ell=1}^{L} l \cdot M_{t,\ell}$. Next, we aim to find a sequence $\boldsymbol{a}$ that closely matches $\boldsymbol{m}$ and satisfies the alignment properties: $\boldsymbol{a} = \arg\min_{\boldsymbol{a'}} \mathcal{E}(\boldsymbol{m}, \boldsymbol{a'}, T)$, where $\boldsymbol{a'}$ is a monotonic integer sequence such that $0 \leq a'_{t+1} - a'_t \leq 1$ for $t = 1, 2, \ldots, T-1$, and all its elements fall in range $[1, L]$ (specifically, $a'_1 = 1$ and $a'_T = L$). We utilize a dynamic programming (DP) algorithm to determine the sequence $\boldsymbol{a}$ that minimizes the cost $\mathcal{E}(\boldsymbol{m}, \boldsymbol{a}, T)$. Each potential value for every element of $\boldsymbol{a'}$ is treated as a distinct state, so the DP state space contains $T \times L$ states. We denote $d_{t,\ell}$ as the minimum total cost up to the first t positions (i.e., $\mathcal{E}(\boldsymbol{m}, \boldsymbol{a'}, t)$) when at state $a'_t = \ell$. When $t, l \geq 2$, state $a_t = \ell$ can only be inherited from two prior states: $a'_{t-1} = \ell$ or $a'_{t-1} = \ell - 1$, so we reach the DP equations (when $t \geq 2$):

$$d_{t,\ell} = \begin{cases} d_{t-1,1} + (m_t - 1)^2 & \ell = 1, \\ \min\{d_{t-1,\ell}, d_{t-1,\ell-1}\} + (m_t - \ell)^2 & l \geq 2. \end{cases} \qquad (2)$$

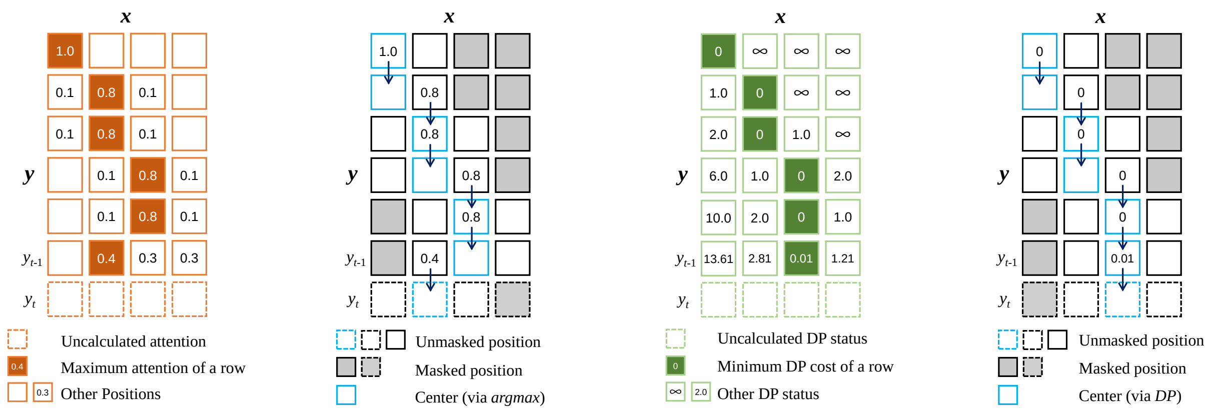
Initial conditions are set at the first row: $d_{1,\ell}$ equals to $(m_1 - 1)^2$ if $\ell = 1$, otherwise ∞. Using Eq. 2 and calculating $d_{t,\ell}$ in the order of $t := 1 \to T$, $\ell := 1 \to L$, we can obtain the final minimum total cost $\min_{\boldsymbol{a'}} \mathcal{E}(\boldsymbol{m}, \boldsymbol{a'}, T) = d_{T,L}$, and the state transition sequence for $d_{T,L}$ is the value assignments for sought sequence $\boldsymbol{a}$. Then, we compute the distance between $\boldsymbol{a}$ and the reference alignment $\boldsymbol{b}$, obtained by a forced alignment algorithm like Viterbi. To enhance the tolerance of the detection, we allow a global deviation of a constant c positions between the two, so the cost becomes $\min_c \mathcal{E}(\boldsymbol{a} + c, \boldsymbol{b}, T)$. Now we reach the expression for the alignment cost $\mathcal{C}_{\mathrm{A}}(\mathbf{M})$:

$$\mathcal{C}_{\mathrm{A}}(\mathbf{M}) = \frac{1}{T}\left(\mathcal{E}(\boldsymbol{m}, \boldsymbol{a}, T) + \min_{c} \mathcal{E}(\boldsymbol{a} + c, \boldsymbol{b}, T)\right) \qquad (3)$$

where $\boldsymbol{a}$ is the previously defined monotonic sequence closest to the average attention positions $\boldsymbol{m}$. Note that the cost calculation pertains to a single input utterance, but to mitigate instability, we can opt to select multiple utterances and compute the average cost, ensuring a more robust assessment of the correlation of each head to the alignment. Finally, AEAMs can be easily filtered out by setting a threshold τ on the average of $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{A}}$:

$$\mathcal{C}_{\mathrm{E}}(\mathbf{M}) + \mathcal{C}_{\mathrm{A}}(\mathbf{M}) < 2\tau. \qquad (4)$$

An attention map is considered an AEAM if and only if Eq. 4 is satisfied. In experiments, we found that for models with different configurations and training datasets, after calculating and ranking the entropy cost and alignment cost for all $N_{\mathrm{L}} \cdot N_{\mathrm{H}}$ attention maps, there is always at least one head in a particular



(a) *AEAM attention pattern.* (b) *Naive argmax CMask.* (c) *AEAM DP values.* (d) *DP CMask.*

Figure 2: *An example of AEAM attention values and DP values, and corresponding CMasks. Here $\boldsymbol{x}$ is the input text tokens, and $\boldsymbol{y}$ is the speech tokens. (a) presents an example of the attention value pattern of a normalized AEAM when generating y_{t+1}, with the maximum value of each row highlighted. The values are normalized so that each row sums up to 1. The constraining mask (CMask) for the newly generated token y_t is now needed. (b) shows the CMask obtained via the trivial argmax strategy, which is prone to causing instability, such as the attention center being falling back. Based on attention values in (a), subfigure (c) illustrates the DP values d of each state. (d) is the CMask obtained via the DP strategy, which shows stability in maintaining the monotonicity.*

early layer (the second or the third layer) whose cost is significantly lower than the others (Section 3.2). An example of heatmap plots for such AEAMs is shown in Fig. 1a, clearly illustrating their strong connection with real alignments (Fig. 1b).

2.3. Attention Constraining

VALL-E's AR synthesis often encounters issues with word skips, repeats, or pronunciation errors. We notice that when the text and synthesized speech do not perfectly match, the attention alignment in AEAMs show correspoding patterns (cf. Fig. 1c and 1d). Assuming that y_1 to y_t have been input or synthesized and that y_{t+1} is to be generated, a possible AEAM value pattern is shown in Fig. 2a. Note that the key in the last row is the newly generated y_t, for which the alignment position needs to be calculated in this AR iteration. However, the AEAM may mistakenly align y_t with a phoneme identical to the true one but positioned differently, leading to a non-monotonic attention alignment that misguide the subsequent generation.

To address this, we propose attention constraining strategies, i.e., adding constraining masks (CMasks) on AEAMs to guide the inherent alignment to proceed monotonically. We assign potentially different CMasks to each AEAM based on their values. A straightforward approach is to restrict y_t to only attend to the vicinity of the alignment (i.e., the attention center position) of y_{t-1}. The main challenge here is to determine the attention center of y_{t-1} in each AEAM based on the known attention values (i.e., upper $t-1$ rows of the normalized matrix $\mathbf{M}$). A trivial method denoted as the *argmax* strategy (Fig. 2b), considers the column with the maximum attention value as the attention center [21], but this may lead to instability. As long as the attention values of the preceding token are not concentrated, center fallback is likely to occur.

Another method denoted as the *DP* strategy (Fig. 2d), leverages the DP algorithm introduced in Attention Sweeping, selecting the position with the minimum DP cost. Since the global information from the previous $t-1$ rows is taken into account, the *DP* strategy is more stable in maintaining the monotonicity. The two center locating strategies can be summarized as:

$$\begin{aligned} \mathrm{ac}_{t-1} &= \arg\max_{\ell} M_{t-1,\ell} && \text{for } \textit{argmax} \text{ strategies,} \\ \mathrm{ac}_{t-1} &= \arg\min_{\ell} d_{t-1,\ell} && \text{for } \textit{DP} \text{ strategies,} \end{aligned} \tag{5}$$

where ac_t is the attention center position for y_t. The CMask for t-th row is then set as only $[\mathrm{ac}_{t-1}-\rho+1, \mathrm{ac}_{t-1}+\rho-1]$ unmasked, where ρ is the unmasking radius, set to an integer correlated with the entropy cost of the attention map in experiments. Furthermore, to preserve the original attention values and to simulate monotonic alignments, we respectively design two mask strategies to generate the first new token after the prompt: one that applies the CMask only to the t-th row (denoted as the last-row strategy), and another that retains the CMask history for all t rows (denoted as the history-kept strategy). The combination of two center locating strategies and two mask strategies results in four attention constraining strategies, which will be evaluated in the experiments.

3. Experiment

3.1. Setup

We use Encodec [10] tokenizer with a 16 000 Hz sampling rate and a 20 ms frameshift to quantize each speech frame into 8 RVQ indices. We use a widely recognized implementation of VALL-E [1] as the base code. We then train four VALL-E models with three different sizes and on two different multi-speaker datasets (LibriTTS [22] with a total of 585 hours and 2306 speakers, and LibriLight-6k [23] with transcribed text in LibriHeavy [24] (with a total of 5000 hours and 1531 speakers), whose detailed information is shown in Table 1. With each model configuration, both AR and on-AR stages are trained. We use the ScaledAdam [25] optimizer and the Eden [25] scheduler with a 0.05 base learning rate to train all models.

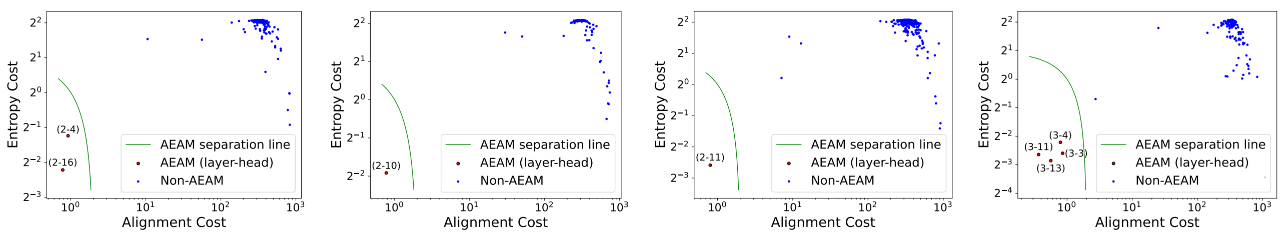
3.2. Attention Sweeping

We first use the Attention Sweeping algorithm to filter out the potential AEAMs. For the four models in Table 1, we respec-

[1]https://github.com/lifeiteng/vall-e

Table 1: *The information on the four VALL-E models used.*

Name	#Params	N_{L}	N_{H}	Hidden Dim.	Dataset	Epochs
STD-585h	367M	12	16	1024	LibriTTS	40
SMALL-585h	159M	9	12	512	LibriTTS	40
LARGE-585h	1.2B	18	24	1536	LibriTTS	40
STD-5000h	367M	12	16	1024	LibriLight-6k	10



(a) STD-*585h* (b) SMALL-*585h* (c) LARGE-*585h* (d) STD-*5000h*

Figure 3: *Calculate $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{A}}$ for each attention map of 4 models, and plot them as 2-D points. The separation line is set at $\tau = 1$.*

Table 2: *WER Results for all four models and four attention constraining strategies.*

Strategy	STD-585h				SMALL-585h	LARGE-585h	STD-5000h
	WER(%) ↓	Del(%)	Ins(%)	Sub(%)	**WER(%)** ↓	**WER(%)** ↓	**WER(%)** ↓
None	4.78	0.74	0.84	3.20	6.65	4.60	4.31
argmax/last-row	4.46	0.65	0.67	3.13	5.63	4.13	3.62
argmax/history-kept	6.68	0.82	1.73	4.13	6.25	4.99	3.88
DP/last-row	4.26	**0.64**	0.79	**2.82**	5.56	4.12	**3.55**
DP/history-kept	**4.14**	0.65	**0.65**	2.84	**5.29**	**3.95**	3.57

Table 3: *MCD, Predicted-MOS (P-MOS), and SECS for* STD-*585h model and DP strategies.*

Strategy (DP)	**MCD** ↓	**P-MOS** ↑	**SECS** ↑
None	3.94	4.40	0.856
last-row	3.93	4.40	0.859
history-kept	**3.92**	**4.41**	**0.860**

tively conduct the algorithm to sweep all $N_{\mathrm{L}} \cdot N_{\mathrm{H}}$ attention maps. We randomly select 5 text-speech pairs from the LibriTTS validation set, feeding them into the models to obtain the attention map values. For each attention map, we follow Eq. 1 and Eq. 3 to calculate its entropy cost $\mathcal{C}_{\mathrm{E}}$ and alignment cost $\mathcal{C}_{\mathrm{A}}$. The reference alignments are obtained via conducting the forced alignment algorithm in Kaldi [26], also employing a frameshift of 20 ms. Then, we average the cost of 5 utterances, so that each attention map can be regarded as a point on a 2-D plot whose y-axis is for $\mathcal{C}_{\mathrm{E}}$ and x-axis is for $\mathcal{C}_{\mathrm{A}}$. Simply setting the AEAM threshold τ (i.e., the average of $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{A}}$, see Eq. 4) as 1, we obtain the results of Attention Sweeping, shown in Fig. 3. The y-axis and x-axis of the plot are log-scaled with bases 2 and 10, respectively. Points located below the AEAM separation line $\tau = 1$ (green line) represent attention maps considered as AEAMs. The four models each have between one to four AEAMs, with their attention value distributions being concentrated and strongly correlated with the true alignment (see Fig. 1a and 1b). Additionally, note that each model's AEAMs are located within the same Transformer layer, specifically in the relatively early second or third layer. This indicates that VALL-E treats the text-speech alignment as a shallow-level feature of the text-speech joint distribution, which implies that by constraining these shallow-layer attentions, we can influence the inference of deeper layers to improve the robustness.

3.3. Robust Generation

Following the identification of AEAMs and the corresponding layers and heads for each model, we can enhance the robustness of synthesis using attention constraining strategies. Our test set, mirroring the one employed in UniCATS [27], includes 500 utterances from 37 speakers in the LibriTTS test set, with each speaker assigned a distinct speech prompt. We use the word error rate (WER) as our main metric to measure the robustness of zero-shot TTS synthesis. Our evaluation process for each VALL-E model and attention constraining strategy involves performing inference on the test set (applying the strategy during the AR stage and leaving the non-AR stage untouched), then reconstructing the RVQ indices into speech, and finally transcribing the speech into text using the ASR model, Whisper[2] [28]. The WER is then calculated by comparing the ground-truth text and the transcribing text. The CMask radius ρ for attention map $\mathbf{M}$ is set to round $(8\mathcal{C}_{\mathrm{E}}(\mathbf{M})) + 1$, and the baseline is not to employ any attention constraining strategies, i.e., the original VALL-E inference. The results, listed in Table 2, show that while the *argmax* strategies sometimes perform unstably, the *DP* strategies obtain a WER reduction of 13.4% to 20.5% in each model, and the counts of deletions, insertions, and substitutions are all lower than the baseline.

Additionally, we evaluated the mel-cepstral distortion (MCD) generated under the DP strategies to measure the distance between generated speech and ground truth. Pre-trained NISQA[3] [29] and Resemblyzer[4] models are also utilized to assess Predicted Mean Opinion Score (P-MOS) and Speaker Embedding Cosine Similarity (SECS), respectively, evaluating the synthesis naturalness (automatically predicted by the NISQA model) and the speaker similarity compared to the prompt. Results in Table 3 indicate that the speech synthesized using ACI shows no degradation in the aforementioned metrics. Subjective auditory perception also indicates that the naturalness under DP strategies is slightly higher or on par with the baseline.

3.4. Ablation Study

The strategy employed in Section 3.3 involved applying CMasks across all AEAMs. This section conducts an ablation study where, during the inference of the STD-585h model using *DP* with history kept, only the head with the lowest average cost, specifically the 16-th head of the 2-nd layer (denoted as (2-16)), is subjected to a CMask. We also tested the results of applying CMasks to all 2-nd layer attention maps. The results, as shown in Table 4, indicate that only applying CMasks to all AEAMs below the $\tau = 1$ line is the most robust choice.

Table 4: STD-*585h model and DP/history-kept strategy: impact of applying constraining masks on less or more AEAMs*

Attn. Constrained AEAM(s)	**WER(%)** ↓	Del(%)	Ins(%)	Sub(%)
(2-16) and (2-4)	**4.14**	**0.65**	**0.65**	**2.84**
(2-16)	4.28	0.68	0.71	2.89
all 2-nd layer attention maps	4.76	0.89	0.80	3.06

4. Conclusions

Our paper introduced a training-free method, ACI, to improve the decoder-only TTS model VALL-E's synthesis robustness. Utilizing Attention Sweeping, we identify and filter AEAMs, strongly related to the alignment, then employ attention constraining to mitigate synthesis errors. Experimental results

[2]https://huggingface.co/openai/whisper-medium

[3]https://github.com/gabrielmittag/NISQA

[4]https://github.com/resemble-ai/Resemblyzer

demonstrate noticeable improvements in WER without degrading naturalness. Our work demonstrates that decoder-only TTS models can unsupervisedly learn alignment-like structures and shows a way to locate and exploit them, laying the foundation for various potential works, such as streaming generation or integrate additional signals into attention maps during training.

5. References

[1] Y. Wang, R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. V. Le, Y. Agiomyrgiannakis, R. A. J. Clark, and R. A. Saurous, "Tacotron: Towards end-to-end speech synthesis," in *Interspeech*, 2017.

[2] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *International Conference on Learning Representations*, 2021.

[3] C. Wang, S. Chen, Y. Wu, Z.-H. Zhang, L. Zhou, S. Liu, Z. Chen, Y. Liu, H. Wang, J. Li, L. He, S. Zhao, and F. Wei, "Neural codec language models are zero-shot text to speech synthesizers," *ArXiv*, vol. abs/2301.02111, 2023.

[4] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov, "Grad-tts: A diffusion probabilistic model for text-to-speech," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 8599–8608.

[5] Y. Guo, C. Du, Z. Ma, X. Chen, and K. Yu, "Voiceflow: Efficient text-to-speech with rectified flow matching," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.

[6] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-tts: A generative flow for text-to-speech via monotonic alignment search," *Advances in Neural Information Processing Systems*, vol. 33, pp. 8067–8077, 2020.

[7] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 5530–5540.

[8] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 29, p. 3451–3460, oct 2021.

[9] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, "High-fidelity audio compression with improved RVQGAN," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[10] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *Transactions on Machine Learning Research*, 2023.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019.

[12] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, and et al., "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[13] K. Lakhotia, E. Kharitonov, W.-N. Hsu, Y. Adi, A. Polyak, B. Bolte, T.-A. Nguyen, J. Copet, A. Baevski, A. Mohamed, and E. Dupoux, "On generative spoken language modeling from raw audio," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 1336–1354, 2021.

[14] C. Du, Y. Guo, X. Chen, and K. Yu, "VQTTS: high-fidelity text-to-speech synthesis with self-supervised VQ acoustic feature," in *Interspeech*, 2022.

[15] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[16] M. Łajszczak, G. Cámbara, Y. Li, F. Beyhan, A. van Korlaar, F. Yang, A. Joly, Á. Martín-Cortinas, A. Abbas, A. Michalski *et al.*, "Base tts: Lessons from building a billion-parameter text-to-speech model on 100k hours of data," *arXiv preprint arXiv:2402.08093*, 2024.

[17] K. Peng, W. Ping, Z. Song, and K. Zhao, "Non-autoregressive neural text-to-speech," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 7586–7598.

[18] C. Du, Y. Guo, H. Wang, Y. Yang, Z. Niu, S. Wang, H. Zhang, X. Chen, and K. Yu, "Vall-t: Decoder-only generative transducer for robust and decoding-controllable text-to-speech," *arXiv preprint arXiv:2401.14321*, 2024.

[19] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[20] E. Kharitonov, D. Vincent, Z. Borsos, R. Marinier, S. Girgin, O. Pietquin, M. Sharifi, M. Tagliasacchi, and N. Zeghidour, "Speak, read and prompt: High-fidelity text-to-speech with minimal supervision," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 1703–1718, 12 2023.

[21] H. Tachibana, K. Uenoyama, and S. Aihara, "Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[22] H. Zen, V.-T. Dang, R. A. J. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "Libritts: A corpus derived from librispeech for text-to-speech," in *Interspeech*, 2019.

[23] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020.

[24] W. Kang, X. Yang, Z. Yao, F. Kuang, Y. Yang, L. Guo, L. Lin, and D. 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)*. IEEE, 2024, pp. 10 991–10 995.

[25] Z. Yao, L. Guo, X. Yang, W. Kang, F. Kuang, Y. Yang, Z. Jin, L. Lin, and D. Povey, "Zipformer: A faster and better encoder for automatic speech recognition," in *The Twelfth International Conference on Learning Representations*, 2024.

[26] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. K. Goel, M. Hannemann, P. Motlícek, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, "The kaldi speech recognition toolkit," 2011.

[27] C. Du, Y. Guo, F. Shen, Z. Liu, Z. Liang, X. Chen, S. Wang, H. Zhang, and K. Yu, "Unicats: A unified context-aware text-to-speech framework with contextual vq-diffusion and vocoding," in *AAAI Conference on Artificial Intelligence*, 2024.

[28] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.

[29] G. Mittag, B. Naderi, A. Chehadi, and S. Möller, "Nisqa: A deep cnn-self-attention model for multidimensional speech quality prediction with crowdsourced datasets," in *Interspeech*, 2021.