

CLAPSpeech: Learning Prosody from Text Context with Contrastive Language-Audio Pre-training

Zhenhui Ye*
zhenhuiye@zju.edu.cn
Zhejiang University

Rongjie Huang*
rongjiehuang@zju.edu.cn
Zhejiang University

Yi Ren
ren.yi@bytedance.com
Bytedance

Ziyue Jiang
jiangziyue@zju.edu.cn
Zhejiang University

Jinglin Liu
liu.jinglin@bytedance.com
ByteDance

Jinzheng He
jinzhenghe@zju.edu.cn
Zhejiang University

Xiang Yin
yixiang.stephen@bytedance.com
Bytedance

Zhou Zhao[†]
zhaozhou@zju.edu.cn
Zhejiang University

Abstract

Improving text representation has attracted much attention to achieve expressive text-to-speech (TTS). However, existing works only implicitly learn the prosody with masked token reconstruction tasks, which leads to low training efficiency and difficulty in prosody modeling. We propose CLAPSpeech, a cross-modal contrastive pre-training framework that explicitly learns the prosody variance of the same text token under different contexts. Specifically, 1) We encourage the model to connect the text context with its corresponding prosody pattern in the joint multi-modal space with the elaborate design of the encoder inputs and contrastive loss; 2) We introduce a multi-scale pre-training pipeline to capture prosody patterns in multiple levels. We show how to incorporate CLAPSpeech into existing TTS models for better prosody. Experiments on three datasets not only show that CLAPSpeech could improve the prosody prediction for existing TTS methods, but also demonstrate its generalization ability to adapt to multiple languages and multi-speaker TTS. We also deeply analyze the principle behind the performance of CLAPSpeech. Ablation studies demonstrate the necessity of each component in our method. Source code and audio samples are available at <https://clapspeech.github.io>.

1 Introduction

With the development of deep learning, the audio quality of modern TTS systems has been improved, yet prosody modeling is still a challenging problem. Previous works on expressive TTS have utilized external variation predictors (prediction-based, PB)

(Ren et al., 2021a) and variational generative models (variation-based, VB) (Kim et al., 2020; Liu et al., 2022) to inject prosody variance into the TTS model. Another popular direction is to learn better text representation for prosody prediction (Tan et al., 2021). However, the existing text representation learning methods for TTS are either based on the masked language model task (Devlin et al., 2019; Jia et al., 2021; Chen et al., 2021) (i.e., learn a BERT-like large language model on a text corpus) or masked acoustic model task (Chen et al., 2020; Bai et al., 2022) (i.e., reconstruct the masked mel-spectrogram based on the input text), which result in two disadvantages. Firstly, they only implicitly learn prosody with reconstruction losses, which distracts the model from improving the prosody modeling. Secondly, they do not decouple the pronunciation space and prosody space, which leads to low training efficiency and a waste of model capacity. We perform a case study in Section 4.3.1, in which we can see that previous text representation used in TTS cannot capture the prosody variance under different text contexts.

Technically, prosody can be regarded as the pitch and duration variance of the same token under different conditions (such as text contexts and speakers) (Tan et al., 2021). This paper mainly studies the prosody correlated to the text context. For instance, for the same word "higher", saying "higher up" or "slightly higher" can lead to different prosodies. Inspired by recent cross-modal contrastive learning works in the text-to-image task (Radford et al., 2021; Elizalde et al., 2022), we propose a contrastive learning method that connects the text context and the high-level prosody pattern in the text-speech joint multi-modal space, namely **Contrastive Language-Audio Pre-Training**

*Equal contribution.

[†]Corresponding author.

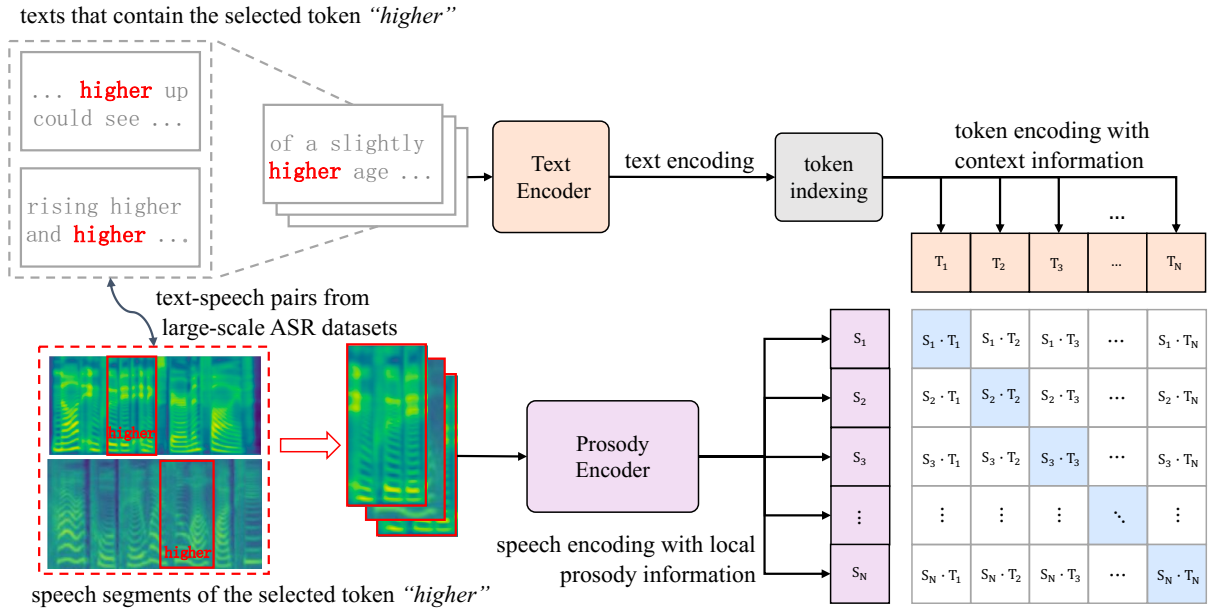


Figure 1: The contrastive pre-training process of CLAPSpeech. For clarity, we only show the word-level pre-training here. Note that we also perform a phoneme-level pre-training.

for Text-to-Speech (CLAPSpeech). Specifically, we learn a text encoder to predict the prosody from the text context and a prosody encoder to extract the ground-truth (GT) prosody from the speech segment of the selected token. During training, we select N text-speech pairs that contain the same pronounceable token (e.g., the word "higher" or phoneme "AEO"). By aligning the text token with its corresponding prosody (extracted from GT speech) and pushing away the prosody representation from other text contexts, the text encoder is encouraged to extract prosody from the text context. An intuitive example of pre-training CLAPSpeech can be found in Figure 1. We also observe that the prosody pattern can be expressed at multiple levels. Therefore, we propose a multi-scale pre-training framework that learns two CLAPSpeech models to capture the prosody information at the phoneme and word levels, respectively. After the pre-training stage, our CLAPSpeech can be regarded as a plug-in text encoder applicable to all TTS models to provide fine-grained prosody representation.

To prove the effectiveness and generalizability of our approach, we use two large-scale automatic speech recognition (ASR) datasets (LibriSpeech (Panayotov et al., 2015) for English and WenetSpeech (Zhang et al., 2022) for Chinese) to pre-train the CLAPSpeech model. The pre-trained text encoder of CLAPSpeech is then plugged into prediction/variation-based TTS baselines to demonstrate the improvement of CLAPSpeech to the exist-

ing expressive TTS systems. We then evaluate the performance on three TTS datasets, including one single-speaker English dataset, one single-speaker Chinese corpus, and one multi-speaker English dataset. Experiments on all datasets show that CLAPSpeech improves the prosody of the TTS models and outperforms previous representation learning methods.

To summarize, CLAPSpeech has three prominent advantages: 1) It can provide better prosody representation than previous representation learning methods with a much smaller model scale, thanks to its contrastive objective that explicitly learns the prosody. 2) The text representation of CLAPSpeech can be conveniently used in existing TTS systems, only with a minor modification of the front-end network architecture. 3) We also show its potential applications such as fine-grained prosody transfer in Section 4.3.2.

2 Related Work

2.1 Expressive TTS

In the past few years, modern neural TTS has made significant progress in high practicality and audio quality (Ren et al., 2019; Kim et al., 2020; Huang et al.; Elias et al., 2021; He et al., 2022; Miao et al., 2021; Kim et al., 2021; Donahue et al., 2021; Jiang et al., 2022; Huang et al., 2022c; He et al., 2023; Jiang et al., 2021; Huang et al., 2022b,a). However, modeling expressive prosody given the plain input text is still challenging. To achieve expressive

TTS, one common practice is to use a reference encoder and style tokens (Wang et al., 2018; Jia et al., 2018). But it is difficult to select appropriate reference audios during inference (Tan et al., 2021). Other works seek to improve prosody modeling with advanced network designs, which can be categorized into two classes: (1) the prediction-based (PB) TTS systems (Ren et al., 2021a) learn several external predictors to predict the prosody attributes such as pitch contour, duration, and energy; (2) the variation-based (VB) TTS systems leverage variational auto-encoder (VAE) (Ren et al., 2021b) or normalizing flow (Kim et al., 2020) to model the prosody in the latent space.

There are also some works that explore providing better text presentation with rich prior knowledge to help the prosody prediction. For instance, Liu et al. (2021) and Ye et al. (2022) incorporate syntax information through dedicated modeling methods such as graph networks. Representation learning methods for text pre-training and speech pre-training also show improvements in the prosody of TTS. We will discuss the representation learning works for TTS in the next section.

2.2 Representation Learning for TTS

Self-supervised pre-training methods have been leveraged in TTS to enhance text processing or speech generation capabilities (Chung et al., 2019; Zhang et al., 2019). Some early works (Wang et al., 2015) use pre-trained word embeddings to improve the robustness of TTS systems. Recently, some works explore incorporating pre-trained large masked language models (MLMs) (Devlin et al., 2019; Chen et al., 2021; Jia et al., 2021) to enjoy the rich semantic information learned from the web-scale text corpus. However, the above-mentioned works only focus on the text space, it is challenging for them to model expressive prosody considering the models are unaware of the high variable prosody patterns in the speech space. There are several inspiring speech representation learning methods in ASR. Baevski et al. (2020) and Hsu et al. (2021) utilize masked continuous speech features to predict predetermined cluster assignments. As for TTS, ProsoSpeech (Ren et al., 2022) designs a word-level vector quantization bottleneck to extract discrete prosody representation from speech. Masked acoustic model (MAM) (Chen et al., 2020) proposes to learn a speech encoder that generates continuous speech (prosody) representations.

Specifically, during training they replace a span of speech spectrogram with mask tokens and learn to recover the masked spectrogram without text conditions. A³T (Bai et al., 2022) additionally learns a text encoder as auxiliary information for MAM to reconstruct the masked mel-spectrogram.

The difference between CLAPSpeech and previous representation works in TTS is obvious: While previous works implicitly learn the prosody information with the masked token reconstruction task, CLAPSpeech is the first work that utilizes the cross-modal contrastive learning to explicitly learn the context-correlated prosody, which leads to better prosody prediction and more efficient usage of model capacity.

3 CLAPSpeech

We propose CLAPSpeech, a cross-modal contrastive learning approach to provide better text representation for prosody prediction in TTS. As shown in Figure 1, CLAPSpeech comprises a text encoder and a prosody encoder, whose training objective is to connect the text token and the speech segment in the joint prosody space. In this section, we first design the network structure and input features of these two encoders. These elaborate designs enable the text encoder to effectively process the text context and ensure that the prosody encoder focuses on extracting the high-level prosody pattern from the speech segment while eliminating other variables, such as timbre. Then we introduce the multi-scale contrastive pre-training framework, which enables CLAPSpeech to capture prosody in both phoneme and word levels. Finally, we show how the pre-trained text encoder of CLAPSpeech can be conveniently plugged into modern TTS systems to improve prosody prediction. We describe these designs in detail in the following subsections and provide more technical details in Appendix A.

3.1 Text Encoder and Prosody Encoder

The prosody of the same pronounceable token¹ varies in different text contexts. CLAPSpeech aims to model the correlation between the text context and the high-level prosody pattern. To this end, we design a text encoder and a prosody encoder to construct a text-speech multi-modal prosody embedding space.

As shown in Figure 2(a), the text encoder uses phoneme and byte pair encoding (BPE) (Shibata

¹such as the phoneme "AEO" or the word "higher".

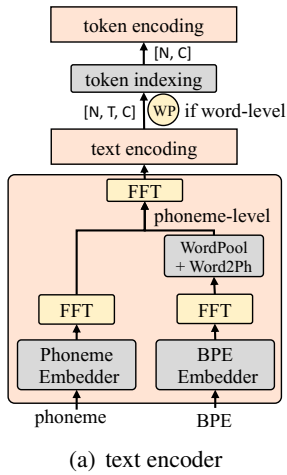


Figure 2: The text / prosody encoder of CLAPSpeech. In subfigure (a), "WP" and "Word2Ph" denotes word pooling and Word2Ph expanding operation, which are illustrated in Figure 3.

et al., 1999) of the input text as the input. The phoneme and BPE sequence help the model extract the prosody pattern related to phonological habits (such as the linking phenomenon in English) and semantic information (which may imply different emotional overtones), respectively. The network structure of the text encoder is composed of several Feed Forward Transformers (FFT) (Vaswani et al., 2017), which have proven the robustness in processing long text sequences in TTS models. Specifically, we learn two independent FFT blocks to process the phoneme and BPE sequences, respectively. This way, the phoneme FFT block could model the phonological habits in phonetic space, and the BPE FFT block could extract the semantic information. One difficulty is fusing the phoneme and BPE sequence of mismatched length. Instead of concatenating these two sequences in the time axis, we use word-level pooling (WP) from Ren et al. (2021b) to process the BPE encoding to the word level, then expand it to the phoneme level (namely the *word2ph* operation). To be specific, as shown in Figure 3(a), the WP operation averages the phoneme hidden states inside each word according to the word boundary, and the word2ph operation repeats the word hidden states for each phoneme inside the word boundary as illustrated in Figure 3(b).

Once the phoneme sequence and BPE sequence is fused, we then use an additional FFT block to fuse the aligned phoneme and BPE encoding to get the final phoneme-level text encoding. During the

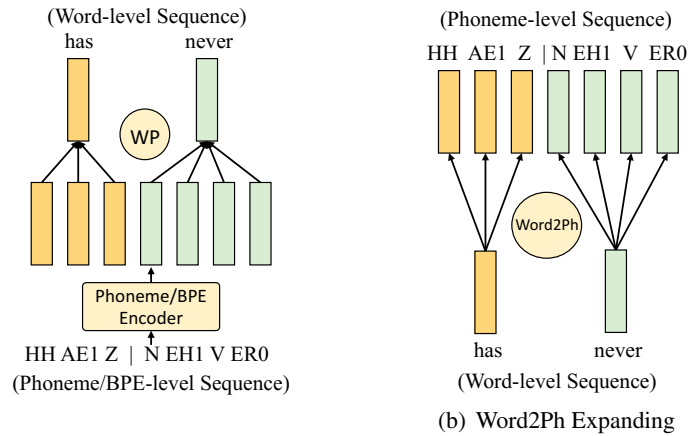


Figure 3: The word pooling and word2ph expanding operation.

pre-training phase, since only one selected token is analyzed, we index from the phoneme-level text encoding to obtain the encoding of the selected token (namely the *token encoding* in Figure 2(a)) and then linearly project it into the multi-modal embedding space. During the TTS phase, the phoneme-level output of the text encoder can be conveniently utilized as auxiliary features for TTS systems, which we will discuss in Section 3.3.

The prosody encoder aims to extract prosody patterns from the GT speech segment of the selected token. Therefore, we clip the mel-spectrogram with the word boundary² as the input speech feature. Then the prosody encoder processes the input mel-spectrogram into a global encoding to be connected with the token encoding. Note that the clipped speech segment only contains the local prosody information for the selected token without leaking any contextual information. Thanks to the contrastive learning setting, the extracted global prosody encoding is disentangled from phonetic and speaker space: 1) since the positive sample and negative samples belong to the same pronounceable token, the phonetic information is eliminated; 2) as the speaker information is not provided to the text encoder³, the prosody encoder will filter out speaker information to maximize the prosody information in the output features during training. This way, by connecting the context-aware text encoding with the context-unaware mel encoding, on the one hand, the prosody encoder learns to extract the high-level prosody information from the

²We extract word boundary with a forced alignment tool.

³We assume that text and speaker are independent of each other (no correlation between them) in our dataset.

speech segment; on the other hand, the text encoder is encouraged to utilize the text context to predict the prosody extracted by the prosody encoder. As shown in Figure 2(b), we use ResNet-50 (He et al., 2016) as the backbone of the prosody encoder due to its robustness. We make several modifications to the original version: 1) to better process the mel-spectrogram, we use 1D convolution with layer normalization to build the fundamental residual block; 2) to handle the speech segment of dynamic lengths, we use an attentive pooling layer from Radford et al. (2021) to aggregate the output feature map of the ResNet.

3.2 Multi-scale Contrastive Pre-training

The key idea of CLAPSpeech is to model the prosody variance of the same text token under different contexts. Therefore, to construct a mini-batch for contrastive pre-training, we randomly select a text token, then sample a batch of N text-speech pairs that contain the selected token (one intuitive sample is shown in Figure 1, where we sample the text-speech pairs that contain the word "higher"). To better extract prosody variance at the phoneme and word level, we introduce a multi-scale contrastive training framework. To be specific, we learn two CLAPSpeech models for phoneme-level and word-level text tokens, respectively.

For clarity, we first illustrate the training process of phoneme-level CLAPSpeech. Let the text context that contains the selected phoneme token (e.g., "AEO") be represented by X_{text} . Let the processed speech segment of the phoneme token be X_{speech} s.t. $X_{speech} \in \mathbb{R}^{F \times T}$, where F is the number of Mel bins and T is the number of time bins. For simplicity, we use X_{text} and X_{speech} to represent a batch of N text-speech pairs.

The text and speech are passed through the text encoder $f_{text}(\cdot)$ and prosody encoder $f_{speech}(\cdot)$, respectively. As can be seen in Figure 2(a), the output of the text encoder $f_{text}(X_{text})$ is the phoneme-level encoding of the input text, hence we index from it to obtain the encoding of the phoneme token $f_{text}(X_{text})_{i_{ph}}$, where i_{ph} denotes the index of the phoneme token in the phoneme-level text sequence. As can be seen in Figure 2(b), the output speech encoding $f_{speech}(X_{speech})$ is a global representation of the input speech segment. The output representations are normalized and then linearly

projected into the multi-modal embedding space:

$$\begin{aligned} T_{ph} &= L_{text}(LN(f_{text}(X_{text})_{i_{ph}})) \\ S &= L_{speech}(LN(f_{speech}(X_{speech}))), \end{aligned} \quad (1)$$

where $T_{ph} \in \mathbb{R}^{N \times C}$ is the phoneme token representation and $S \in \mathbb{R}^{N \times C}$ is the speech representation of channel size C . LN means layer normalization, L_{text} and L_{speech} are linear projections.

Now that the text and speech embeddings are comparable, CLAPSpeech is trained to predict which of the $N \times N$ possible text-speech pairings across a batch actually occurred. Specifically, the text encoder and prosody encoder are encouraged to maximize the cosine similarity of the text and speech encoding of the N real pairs in the batch while minimizing the cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings. Following Radford et al. (2021), we optimize a symmetric cross-entropy loss over these similarity scores:

$$\mathcal{L}_{ph} = 0.5 \times (l_{text}(\tau \cdot C_{ph}) + l_{speech}(\tau \cdot C_{ph})) \quad (2)$$

where $C_{ph} \in \mathbb{R}^{N \times N}$ is the cosine similarity matrix between the phoneme token encoding T_{ph} and the speech encoding S , measured by $C_{ph} = T_{ph} \cdot S^T$; τ is a learnable temperature parameter to scale the range of logits; and $l_k = \frac{1}{N} \sum_{i=0}^N \log \text{diag}(\text{softmax}(C))$ is the cross entropy function along the text and speech axis in C .

The word-level CLAPSpeech can be trained similarly. As shown in Figure 2(a), for the word-level CLAPSpeech, we use word pooling to process the phoneme-level text encoding into word level, then index from it to obtain the word token encoding T_{word} . Similar to Equation 2, the training loss for word-level CLAPSpeech is formulated as:

$$\mathcal{L}_{word} = 0.5 \times (l_{text}(\tau \cdot C_{word}) + l_{speech}(\tau \cdot C_{word})) \quad (3)$$

where C_{word} is the cosine similarity matrix between the word token encoding T_{word} and the speech encoding S .

3.3 CLAPSpeech Plugged in TTS Systems

The text encoder of CLAPSpeech could provide text representation with rich prosody information for the TTS task. Since the generated text representation is at the phoneme level, which is in line with the majority of current TTS models that also utilize phoneme sequence as the text input, CLAPSpeech can be a convenient plugin unit for TTS systems to improve prosody prediction. Specifically, we

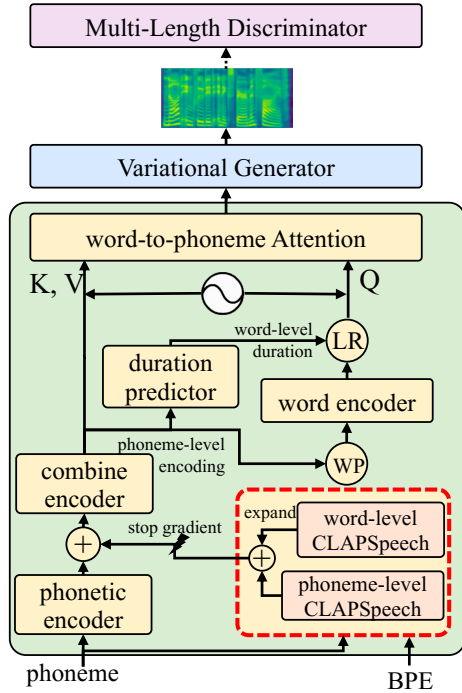


Figure 4: PortaSpeech with CLAPSpeech plugged in.

take a state-of-the-art variation-based TTS system, *PortaSpeech*, as an example. As shown in Figure 4, the pre-trained text encoders of CLAPSpeech (marked with a red dashed rectangle) perform as an auxiliary encoder to the original phonetic encoder of *PortaSpeech*. The phoneme-level outputs of the phonetic encoder and CLAPSpeech text encoder are fused and processed by the following encoder. Note that we fix the parameters of CLAPSpeech text encoders during the training of the TTS system to avoid overfitting. CLAPSpeech can be easily plugged into other TTS systems in a similar way. To demonstrate the universality, we illustrate how to combine CLAPSpeech with a widely-used prediction-based TTS system, *FastSpeech 2*, in Appendix A.1. We additionally adopt multi-length adversarial training in TTS models to improve audio quality. More details about the adversarial training can be found in Appendix A.2.

4 Experiments

4.1 Experimental Setup

Datasets and Baselines We pre-train CLAPSpeech on two ASR datasets: 1) LibriSpeech (Panayotov et al., 2015), an English database that contains 982 hours of speech from 2484 speakers; 2) WenetSpeech (Zhang et al., 2022), a Chinese speech corpus consisting of 10,000 hours of

speech⁴. Then we evaluate the pre-trained CLAPSpeech on three TTS datasets: 1) LJSpeech (Ito and Johnson, 2017), a single-speaker database that contains 13,100 English audio clips with a total of nearly 24 hours of speech; 2) Biaobei⁵, a Chinese speech corpus consisting of 10,000 sentences (about 12 hours) from a Chinese speaker; 3) LibriTTS (Zen et al., 2019), an English dataset with 149,736 audio clips (about 245 hours) from 1,151 speakers (We only use *train clean360* and *train clean100*). The raw text is transformed into phoneme and BPE sequences using open-sourced tools. The GT mel-spectrograms are generated from the raw waveform with a frame size of 1024 and the hop size of 256. We compare CLAPSpeech against two pre-training baselines (BERT (Devlin et al., 2019) and A³T (Bai et al., 2022)) in a prediction-based (PB) TTS model, *FastSpeech 2*, and a variation-based (VB) TTS model, *PortaSpeech*.

Model Configuration CLAPSpeech consists of a text encoder and a prosody encoder, whose structures are shown in Figure 2 and discussed in Section 3.2. As for the PB and VB TTS models, we use the same structure in the original papers with an additional multi-length discriminator to improve audio quality. The multi-length discriminator consists of multiple stacked convolutional layers with batch normalization and treats the input spectrogram as images. We put more detailed model configurations in Appendix B.1.

Training and Evaluation Our approach is implemented with Pytorch. We pre-train CLAPSpeech on 4 Nvidia 3090Ti GPUs with a batch size of 1,024 text-speech pairs (256 pairs per GPU). We use the Adam optimizer with an initial learning rate of 0.0005. We train the CLAPSpeech model for 640,000 iterations (which takes about 1 week) and follow the cosine learning rate schedule in CLIP. Then we train the TTS models on 1 Nvidia 2080Ti GPU with a batch size of 64 sentences, following the learning rate schedule in Vaswani et al. (2017). We use HiFi-GAN (Kong et al., 2020) as the vocoder. We conduct the mean opinion score (MOS) and comparative mean opinion score (CMOS) evaluation to measure the prosody and audio quality. Details about the subjective evaluation

⁴We filter samples with a correctness confidence level above 0.95, finally get a subset of 1000 hours.

⁵<https://www.data-baker.com/opensource.html>

can be found in Appendix B.2. As for the objective evaluation, following Ren et al. (2021b), we evaluate the prosody from the aspects of pitch and duration: 1) we compute the average dynamic time warping (DTW) (Muller, 2007) distances between the pitch contours of GT speech and synthesized speech to measure the pitch accuracy; 2) we calculate the average absolute duration error (DE) in micro-seconds⁶ to measure the duration accuracy.

4.2 Performance

We compare the performance of our CLAPSpeech against BERT and A³T in PB/VB TTS models. GT (the ground-truth audio) and GT (voc.) (the audio waveform generated by the vocoder using the GT mel-spectrogram) are also included in the experiment. We perform the TTS experiments on three datasets as mentioned in Section 4.1. The results are shown in Table 1. We can see that CLAPSpeech outperforms other representation learning methods in both PB and VB TTS baselines in terms of MOS, pitch accuracy, and duration accuracy, which proves that CLAPSpeech could effectively improve the prosody prediction in current expressive TTS models (no matter prediction-based or variation-based). Besides, we observe that CLAPSpeech achieves better performance than BERT and A³T with much fewer model parameters. We suspect it is due to the fact that the MLM-based method (i.e., BERT) require a large model capacity to store the semantic information and MAM-based method (i.e., A³T) have to jointly learn the phonetic information to reconstruct the masked mel-spectrogram. By contrast, our CLAPSpeech eliminates the phonetic space and only focus on the prosody space during pre-training, which is parameter-efficient.

We then visualize the mel-spectrograms generated by different methods in Figure 5. We can see that CLAPSpeech can generate results with more realistic pitch contours, which result in expressive prosody. In conclusion, our experiments demonstrate that CLAPSpeech could help TTS systems synthesize more expressive and prosodic audio.

4.3 Deeper Analysis

4.3.1 Token Representation Self-similarity

To better understand the performance superiority of CLAPSpeech over existing representation learning methods for TTS, we analyze the token represen-

tation learned by CLAPSpeech and other methods. Following Su et al. (2021), we define the averaged similarity on the selected token under different contexts $T = [T_1, \dots, T_N]$ as,

$$s(T) = \frac{1}{N(N-1)} \sum_{i=1}^N \sum_{j=1, j \neq i}^N \text{cosine}(T_i, T_j) \quad (4)$$

where T_i and T_j are the selected token’s encoding extracted by the model from different text contexts. Intuitively, a lower $s(T)$ indicates that the selected token itself plays a smaller role in generating its representation, which means that the model captures more context-related information from the input text sequence, and thus predicts better prosody.

Quantitative Evaluation We sample 10,000 batches (each batch consists of 256 sentences that contain the same selected token) from the ASR validation datasets and compute the averaged self-similarity. The result is shown in Table 2. We observe that our CLAPSpeech learned with the contrastive objective (in Equation 2) achieves the lowest similarity in the off-diagonal entries of the similarity matrix, which denotes that the model has made use of the text context to capture the prosody variance of the same token, thus achieve the best prosody performance in Table 1. Besides, we can see that BERT also achieves a relatively low off-diagonal similarity, which is due to its MLM task during pre-training, in which the model needs to extract semantic information from context to predict the masked token. By contrast, the vanilla TTS text encoder and A³T fail to achieve a low off-diagonal similarity, which means that both models cannot extract discriminative information from different contexts. We suspect the failure of A³T is due to the fact that its MAM objective encourages the model to predict the masked mel-spectrogram patch based on the input unmasked text sequence, which increases the model’s demand for phonetic information of the selected token.

Qualitative Evaluation We sample 8 sentences⁷ that contain the word "higher" from LibriSpeech and visualize the self-similarity matrix M (where $M_{i,j} = \text{cosine}(T_i, T_j)$) produced by CLAPSpeech and vanilla TTS text encoder. The results are shown in Figure 6, where a darker color denotes a higher self-similarity score. We also provide the self-similarity matrix of BERT and A³T in Figure 9

⁶In our PB/VB TTS baseline, the duration is predicted in phoneme/word level, respectively.

⁷We list these sentences in Table 5 of Appendix C.

Table 1: Performance comparison of different methods. *PB* and *VB* denote *prediction-based* and *variation-based* TTS baselines, respectively. DTW denotes the dynamic time warping distance of pitch contours in the Mel-spectrogram. DE means the averaged absolute duration error in micro-seconds.

Method	LJSpeech			Biaobei			LibriTTS			#Params
	MOS↑	DTW↓	DE↓	MOS↑	DTW↓	DE↓	MOS↑	DTW↓	DE↓	
<i>GT</i>	4.81	0	0	4.59	0	0	4.40	0	0	/
<i>GT(voc.)</i>	4.63	0	0	4.43	0	0	4.26	0	0	/
<i>PB</i>	3.77	29.09	25.77	3.37	18.01	28.79	3.43	14.26	27.42	11.99M
<i>PB + BERT</i>	4.04	27.43	24.97	3.43	16.79	28.06	3.60	13.82	26.70	109.48M
<i>PB + A³T</i>	3.92	28.18	25.63	3.51	17.18	28.44	3.54	13.67	27.03	48.25M
<i>PB + CLAPSpeech</i>	4.11	27.16	24.19	3.62	16.04	27.60	3.71	13.37	26.46	30.51M
<i>VB</i>	3.96	27.58	53.23	3.75	14.22	40.31	3.81	11.96	52.51	23.02M
<i>VB + BERT</i>	4.13	26.97	52.01	3.91	13.63	38.41	3.95	11.51	51.27	132.69M
<i>VB + A³T</i>	4.05	26.37	52.17	4.04	13.97	39.15	3.82	11.71	51.98	59.73M
<i>VB + CLAPSpeech</i>	4.28	25.94	51.34	4.22	13.48	37.07	4.06	10.93	50.89	41.54M

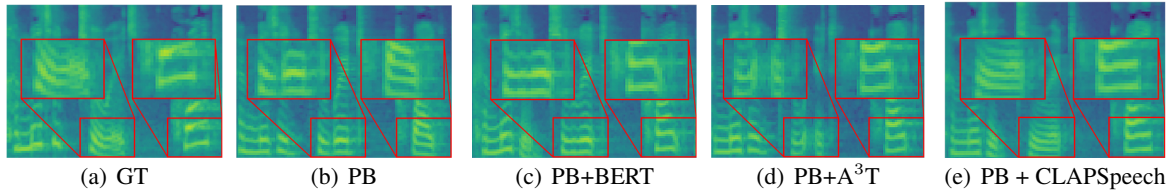


Figure 5: Visualizations of the mel-spectrograms generated by different TTS systems.

Table 2: Self-similarity score of different methods. TTS denotes the text encoder of the vanilla TTS baseline.

Text Encoder of	TTS	BERT	A ³ T	CLAPSpeech
Self-Similarity	0.9854	0.5517	0.9390	0.4160

of Appendix C. We can see that the self-similarities of CLAPSpeech are much lower in the off-diagonal entries.

4.3.2 Fine-grained Prosody Transfer

We perform an intuitive case study about prosody transfer to further validate that our CLAPSpeech’s text-speech joint multi-modal space represents high-level prosody patterns (i.e., the pitch contours and duration information). We take s7/8 in Table 5 as the reference/source audio and expect to transfer the word “higher”’s prosody pattern from s7 to s8. Specifically, we use the text encoder of CLAPSpeech to extract the text prosody encoding of s7 and s8, then replace the text token encoding of “higher” in s8 with that in s7. As shown in Figure 7, the prosody pattern of “higher” in s8⁸ in Figure 7(a) has been successfully transferred into s7 in

⁸the pitch contours in reference remain flat in the early stage and then rise in the late stage

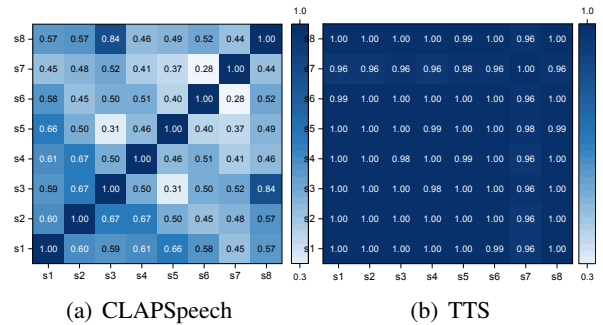
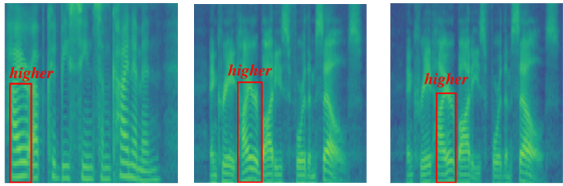


Figure 6: Example: self-similarity matrix visualization of CLAPSpeech and the text encoder of the vanilla TTS model. s_i denotes the i_{th} sentence.

Figure 7(c). We also provide audio samples of this case study on our demo page. The manipulation of the local prosody proves that our CLAPSpeech extract prosody representation effectively influences the prosody prediction of the TTS system.

4.4 Ablation Studies

Use BPE as Auxiliary Features We first analyze the effectiveness of the BPE as an auxiliary feature to help extract prosody information from the text context. During the pre-training phase of CLAPSpeech, we found removing BPE from the



(a) reference (s7) (b) source (s8) (c) transferred (s8)

Figure 7: Visualizations of the mel-spectrograms generated in prosody transfer.

text encoder significantly degrades the validation CLIP loss from 0.3692 to 0.6764. Then in the TTS phase, as can be seen in line 3 in Table 3, the ablated model using the pre-trained text encoder without BPE leads to a performance drop in terms of CMOS, DTW, and DE. This is possibly due to the fact that BPE could better represent the semantic information than the low-level phoneme sequence.

Multi-scale Pre-training To demonstrate the effectiveness of multi-scale pre-training, as can be seen in line 4/5 in Table 3, we tried to remove phoneme-level or word-level CLAPSpeech from the model, which leads to a worse prosody performance. We also tried to use the untrained CLAPSpeech to prove the necessity of the pre-training process, and we found this ablated model (line 6) achieves a slightly worse performance than the TTS baseline (line 3).

Table 3: Performance comparison for ablation studies.

Setting	CMOS	DTW	DE
<i>TTS + CLAPSpeech</i>	0	27.16	24.19
<i>TTS baseline</i>	-1.53	29.09	25.77
<i>w/o BPE</i>	-1.08	28.21	24.93
<i>w/o ph-level</i>	-1.11	27.68	25.01
<i>w/o word-level</i>	-0.46	27.55	24.52
<i>untrained</i>	-1.67	29.45	25.96

5 Conclusion

In this paper, we propose CLAPSpeech, a cross-modal contrastive pre-training framework that provides better text representation with rich prosody information for TTS. With the design of a text encoder and a prosody encoder, CLAPSpeech learns to connect the text context with its corresponding prosody pattern in the speech. We also introduced multi-scale pre-training to extract prosody patterns at multiple levels. We have demonstrated the performance and generalization ability of CLAPSpeech

on three TTS datasets (English, Chinese, and multi-speaker, respectively). We have also deeply analyzed the principle behind the improvement of CLAPSpeech and performed ablation studies to prove the necessity of each component.

6 Limitations

There are majorly two limitations: Firstly, in this work, we only consider the current-sentence text context-related prosody. In future work, we will focus on improving the inter-sentence prosody to achieve coherent, expressive TTS for long-form text. Secondly, other variables are not considered during the contrastive pre-training. One can explore similar approaches that connect prosody to other conditions such as speaker, emotion, etc.

7 Ethics Statement

CLAPSpeech improves the prosody of the synthesized speech, which may cause unemployment for people with related occupations. Besides, the production of fake speeches may cause voice security issues. Further efforts in automatic speaker verification should be made to improve voice security.

8 Acknowledgment

This work was supported in part by the National Key R&D Program of China under Grant No.2022ZD0162000, National Natural Science Foundation of China under Grant No. 62222211 and Grant No.61836002 and Grant No.62072397, and Yiwise.

References

- Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NIPS*.
- He Bai, Renjie Zheng, Junkun Chen, Mingbo Ma, Xintong Li, and Liang Huang. 2022. A3t: Alignment-aware acoustic and text pretraining for speech synthesis and editing. In *ICML*.
- Junkun Chen, Mingbo Ma, Renjie Zheng, and Liang Huang. 2020. Mam: Masked acoustic modeling for end-to-end speech-to-text translation. *arXiv preprint arXiv:2010.11445*.
- Liping Chen, Yan Deng, Xi Wang, Frank K Soong, and Lei He. 2021. Speech bert embedding for improving prosody in neural tts. In *ICASSP*.

- Yu-An Chung, Yuxuan Wang, Wei-Ning Hsu, Yu Zhang, and RJ Skerry-Ryan. 2019. Semi-supervised training for improving data efficiency in end-to-end speech synthesis. In *ICASSP*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Jeff Donahue, Sander Dieleman, Mikołaj Bińkowski, Erich Elsen, and Karen Simonyan. 2021. End-to-end adversarial text-to-speech. In *ICLR*.
- Isaac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Ye Jia, Ron J Weiss, and Yonghui Wu. 2021. Parallel tacotron: Non-autoregressive and controllable tts. In *ICASSP*.
- Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. 2022. Clap: Learning audio concepts from natural language supervision. *arXiv preprint arXiv:2206.04769*.
- Jinzheng He, Jinglin Liu, Zhenhui Ye, Rongjie Huang, Chenye Cui, Huadai Liu, and Zhou Zhao. 2023. [Rmssinger: Realistic-music-score based singing voice synthesis](#).
- Jinzheng He, Zhou Zhao, Yi Ren, Jinglin Liu, Baoxing Huai, and Nicholas Yuan. 2022. Flow-based unconstrained lip to speech generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 843–851.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.
- Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.
- Rongjie Huang, Chenye Cui, Feiyang Chen, Yi Ren, Jinglin Liu, Zhou Zhao, Baoxing Huai, and Zhefeng Wang. 2022a. Singgan: Generative adversarial network for high-fidelity singing voice generation. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2525–2535.
- Rongjie Huang, Max WY Lam, Jun Wang, Dan Su, Dong Yu, Yi Ren, and Zhou Zhao. 2022b. Fastdiff: A fast conditional diffusion model for high-quality speech synthesis. *arXiv preprint arXiv:2204.09934*.
- Rongjie Huang, Yi Ren, Jinglin Liu, Chenye Cui, and Zhou Zhao. 2022c. Generspeech: Towards style transfer for generalizable out-of-domain text-to-speech. In *Advances in Neural Information Processing Systems*.
- Rongjie Huang, Zhou Zhao, Huadai Liu, Jinglin Liu, Chenye Cui, and Yi Ren. 2022c. Prodiff: Progressive fast diffusion model for high-quality text-to-speech. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2595–2605.
- Keith Ito and Linda Johnson. 2017. The lj speech dataset. <https://keithito.com/LJ-Speech-Dataset/>.
- Ye Jia, Heiga Zen, Jonathan Shen, Yu Zhang, and Yonghui Wu. 2021. Png bert: Augmented bert on phonemes and graphemes for neural tts.
- Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *NIPS*.
- Ziyue Jiang, Yi Ren, Ming Lei, and Zhou Zhao. 2021. Fedspeech: Federated text-to-speech with continual learning. *arXiv preprint arXiv:2110.07216*.
- Ziyue Jiang, Su Zhe, Zhou Zhao, Qian Yang, Yi Ren, Jinglin Liu, and Zhenhui Ye. 2022. Dict-tts: Learning to pronounce with prior dictionary knowledge for text-to-speech. *arXiv preprint arXiv:2206.02147*.
- Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. 2020. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. In *NIPS*.
- Jaehyeon Kim, Jungil Kong, and Juhee Son. 2021. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *ICML*.
- Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *NIPS*.
- Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen, and Zhou Zhao. 2022. Diffsinger: Singing voice synthesis via shallow diffusion mechanism. In *AAAI*.
- Rui Liu, Berrak Sisman, and Haizhou Li. 2021. Graphspeech: Syntax-aware graph attention network for neural speech synthesis. In *ICASSP*.
- Chenfeng Miao, Liang Shuang, Zhengchen Liu, Chen Minchuan, Jun Ma, Shaojun Wang, and Jing Xiao. 2021. Efficienttts: An efficient and high-quality text-to-speech architecture. In *ICML*.
- Meinard Muller. 2007. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84.
- Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2021a. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *ICLR*.

Yi Ren, Ming Lei, Zhiying Huang, Shiliang Zhang, Qian Chen, Zhijie Yan, and Zhou Zhao. 2022. Prosospeech: Enhancing prosody with quantized vector pre-training in text-to-speech. In *ICASSP*.

Yi Ren, Jinglin Liu, and Zhou Zhao. 2021b. Portaspeech: Portable and high-quality generative text-to-speech. In *NIPS*.

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

Yusuxke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, and Setsuo Arikawa. 1999. Byte pair encoding: A text compression scheme that accelerates pattern matching.

Yixuan Su, Fangyu Liu, Zaiqiao Meng, Tian Lan, Lei Shu, Ehsan Shareghi, and Nigel Collier. 2021. Tacotron: Improving bert pre-training with token-aware contrastive learning. *arXiv preprint arXiv:2111.04198*.

Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. 2021. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. 2015. Word embedding for recurrent neural network based tts synthesis. In *ICASSP*.

Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ-Skerry Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A Saurous. 2018. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *ICML*.

Zhenhui Ye, Zhou Zhao, Yi Ren, and Fei Wu. 2022. Syntaspeech: Syntax-aware generative adversarial text-to-speech. *arXiv preprint arXiv:2204.11792*.

Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. 2019. Libritts: A corpus derived from librispeech for text-to-speech. *arXiv preprint arXiv:1904.02882*.

Binbin Zhang, Hang Lv, Pengcheng Guo, Qijie Shao, Chao Yang, Lei Xie, Xin Xu, Hui Bu, Xiaoyu Chen, Chenchen Zeng, et al. 2022. Wenetspeech: A 10000+ hours multi-domain mandarin corpus for speech recognition. In *ICASSP*.

Mingyang Zhang, Xin Wang, Fuming Fang, Haizhou Li, and Junichi Yamagishi. 2019. Joint training framework for text-to-speech and voice conversion using multi-source tacotron and wavenet. In *INTER-SPEECH*.

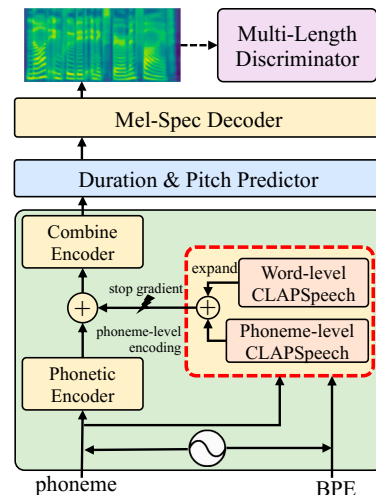


Figure 8: FastSpeech 2 with CLAPSpeech plugged in.

A Details of Models

A.1 CLAPSpeech plugged in FastSpeech 2

We show how to integrate CLAPSpeech into a popular prediction-based TTS system, *FastSpeech 2*. As shown in Figure 8, the pre-trained text encoders of CLAPSpeech (marked with a red dashed rectangle) perform as an auxiliary encoder to the original phonetic encoder of FastSpeech 2. The phoneme-level outputs of the phonetic encoder and CLAPSpeech text encoder are fused and processed by the following encoder. Note that we fix the parameters of CLAPSpeech text encoders during the training of the TTS system to avoid overfitting.

A.2 Multi-length Adversarial Training

For the tested TTS baselines, we adopt an additional multi-length discriminator to provide a least squared GAN loss to improve the audio quality. The multi-length discriminator is an ensemble of multiple CNN-based discriminators which evaluates the mel-spectrogram based on random windows of different lengths. One could refer to Ye et al. (2022) for more details.

B Detailed Experimental Settings

B.1 Model Configurations

We list the hyper-parameters of CLAPSpeech and the tested TTS baselines in Table 4.

B.2 Subjective Evaluation

For each tested dataset, we randomly select 10 texts from the test set and use the TTS systems to generate the audio samples. Each audio has been listened to by at least 20 native listeners, who are

Table 4: The detailed model configurations.

Hyper-parameter		CLAPSpeech	Number of parameters
Text Encoder	Phoneme/BPE embedding hidden size	192	18.517M
	Phoneme/BPE encoder FFT blocks	4	
	Hidden size	192	
	Conv1D kernel	5	
	Conv1D filter size	768	
Prosody Encoder	Residual blocks	4	21.801M
	Number of conv layers per block	12	
	Hidden size	192	
	Input mel-spectrogram length	128	
	Hidden size in pooling layer	768	
	#Attention heads in pooling layer	4	
Prediction-based TTS baseline	Encoder Layers	4	11.993M
	Decoder Layers	4	
	Encoder/Decoder Conv1D Kernel	9	
	Encoder/Decoder Conv1D channel size	256	
Variation-based TTS baseline	Encoder Layers	8	23.020M
	Decoder Layers	4	
	Encoder/Decoder Conv1D Kernel	5	
	Encoder/Decoder Conv1D channel size	192	
	Latent Size	16	
	Prior Flow Layers	4	
	Prior Flow Conv1D Kernel	3	
	Prior Flow Conv1D Channel Size	64	
Multi-Length Discriminator	Number of CNN-based Discriminators	3	0.927M
	Window size	32,64,128	
	Conv2D layers	3	
	Hidden size	192	

recruited on a crowdsourcing platform, Zhengshu Technology. We tell listeners to "*focus on examining the naturalness of prosody (e.g., pitch, energy, and duration) and audio quality (noise, timbre, sound clarity, and high-frequency details)*". For MOS, each tester is asked to evaluate the subjective naturalness of a sentence on a 1-5 Likert scale. For CMOS, listeners are asked to compare pairs of audio generated by systems A and B and indicate which of the two audio they prefer and choose one of the following scores: 0 indicating no difference, 1 indicating small difference, 2 indicating a large difference, and 3 indicating a very large difference.

C More Details in Analysis

C.1 Example Sentences

We list the 8 example sentences in Table 5. These sentences are used as examples in Section 4.3.

C.2 Self-similarity of Other Baselines

The self-similarity visualization of A³T and BERT can be found in Figure 9. We discuss the results in Section 4.3.1.

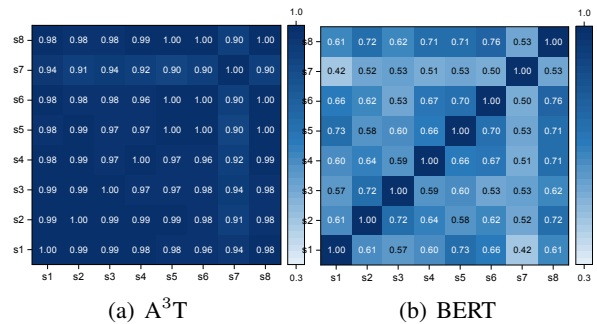


Figure 9: Self-similarity matrix visualization of A³T and BERT.

s1	<i>... for the reputation of the stern judge stands not higher than that of the compassionate ...</i>
s2	<i>As I went on , the precipices rose higher and seemed to overhang. The channel grew narrower ...</i>
s3	<i>Better, and better, and better! Her voice went higher with each better, till it got quite to a squeak at last.</i>
s4	<i>... and the native graduates of our higher institutions have begun to show their strength ...</i>
s5	<i>Innocence is higher than virtue.</i>
s6	<i>Nothing seems more unfit to give a deeper meaning to life and a higher value.</i>
s7	<i>Higher up could be seen some chinamen, but whether they were fishing or washing we could not tell .</i>
s8	<i>May they become convalescents and overcomers, and create higher bodies for themselves !</i>

Table 5: The text sentences used in the intuitive example, the selected word token "higher" is bold.

ACL 2023 Responsible NLP Checklist

A For every submission:

- A1. Did you describe the limitations of your work?
6.Limitations
- A2. Did you discuss any potential risks of your work?
7.Ethics Statement
- A3. Do the abstract and introduction summarize the paper’s main claims?
1.Introduction
- A4. Have you used AI writing assistants when working on this paper?
Left blank.

B Did you use or create scientific artifacts?

Left blank.

- B1. Did you cite the creators of artifacts you used?
No response.
- B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.
- B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.
- B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.
- B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.
- B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C Did you run computational experiments?

4.Experiments

- C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.1 Experimental Setup & Appendix B.1 Model Configurations

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

- ✓ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4.1 Experimental Setup
 - ✓ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4.2 & 4.3 & 4.4
 - ✓ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4.1 Experimental Setup
- D** ✓ **Did you use human annotators (e.g., crowdworkers) or research with human participants?**
Section 4.2
- ✓ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Appendix B.2
 - ✓ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Appendix B.2
 - ✓ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Appendix B.2
 - ✓ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Appendix B.2
 - ✓ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Appendix B.2