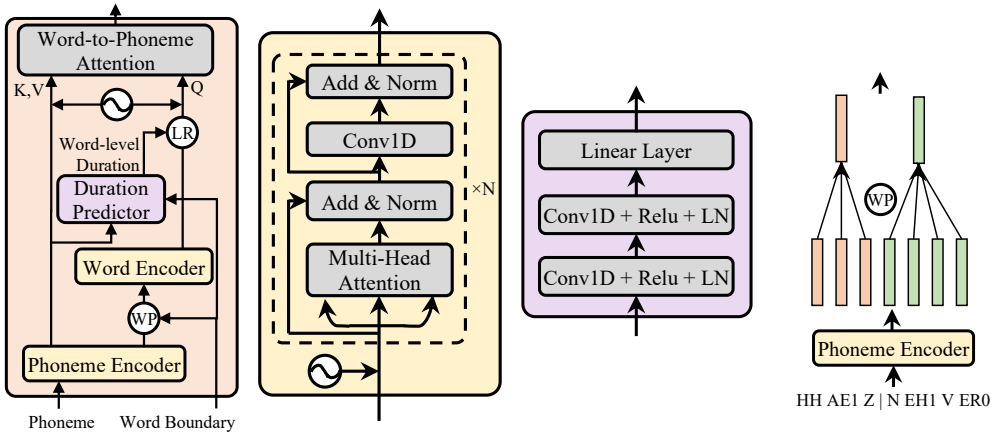
Appendices

A Details of Models

In this section, we describe details in the linguistic encoder, variational generator, post-net and the models we used in Section 4.2.

A.1 Linguistic Encoder



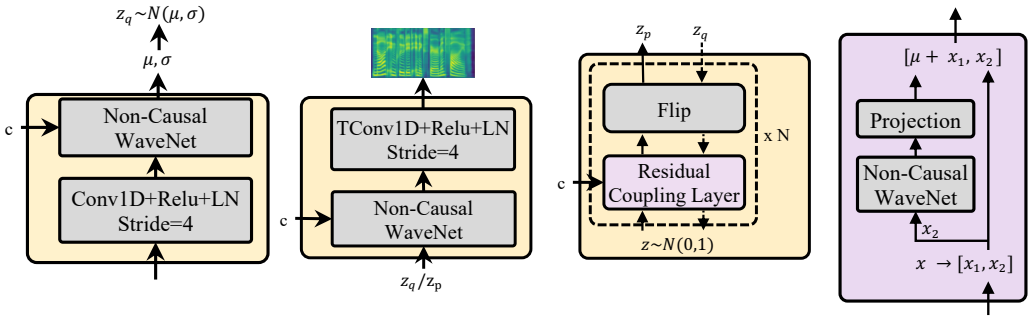
(a) Linguistic Encoder (b) Phoneme/Word Encoder (c) Duration Predictor (d) Word-Level Pooling

Figure 4: The detailed architecture of linguistic encoder.

As shown in Figure 4, our linguistic encoder consists of a phoneme encoder, a word encoder, a duration predictor and a word-to-phoneme attention module. **The phoneme encoder and the word encoder** are both stacks of feed-forward Transformer layers with relative position encoding [28], as shown in Figure 4b. **The duration predictor**, as shown in Figure 4c, consists of two 1D-convolutional layers, each of which is followed by ReLU activation and layer normalization, and a linear layer to project the hidden states in each timestep to a scalar, which is the predicted phoneme duration. **The word-level pooling** averages the phoneme hidden states inside each word according to the word boundary, as shown in Figure 4d. **The word-to-phoneme attention module** is a multi-head attention [34] with 2 heads and we apply a word-to-phoneme mapping mask to the attention weight to force each query (Q) to only attend to the phonemes belongs to the word corresponding to this query. We also add a well-designed **positional encoding** to the inputs of word-to-phoneme attention module: for K and V, the positional encoding is: $\frac{i}{L_w}E_{kv}$, where i is the position of the corresponding phoneme in the word w; L_w is the number of phonemes in word w; E_{kv} is a learnable embedding; and $i \in \{0, 1, ..., L_w - 1\}$. For Q, the positional encoding becomes: $\frac{j}{T_w}E_q$, where j is the position of the corresponding frame in the word w; T_w is the number of frames in word w; E_q is another learnable embedding; and $j \in \{0, 1, ..., T_w - 1\}$.

A.2 Variational Generator

As shown in Figure 5, our variational generator consists of an encoder, a decoder and a volume-preserving (VP) flow-based prior model. **The encoder**, as shown in Figure 5a, is composed of a 1D-convolution with stride 4 followed by ReLU activation and layer normalization, and a non-causal WaveNet. **The decoder**, as shown in Figure 5b, consists of a non-causal WaveNet and a 1D transposed convolution with stride 4, also followed by ReLU and layer normalization. **The prior model**, as shown in Figure 5c, is a volume-preserving normalizing flow, which is composed of a residual coupling layer (Figure 5d) and a channel-wise flip operation.



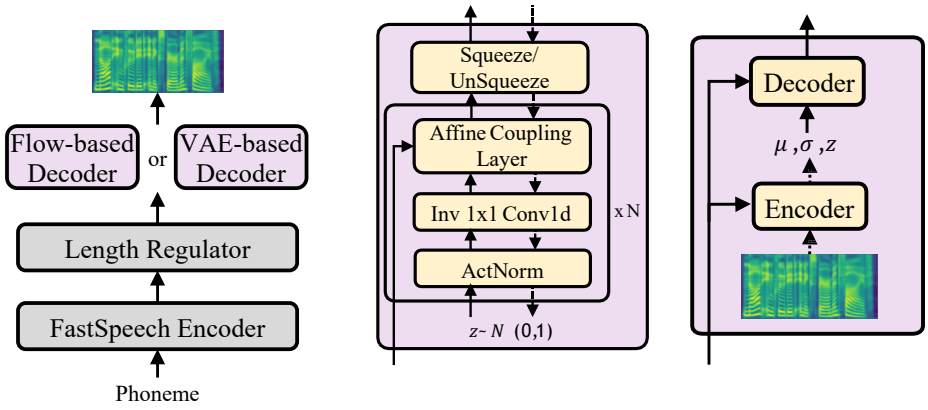
(a) Encoder (b) Decoder (c) VP Flow (d) Coupling Layer

Figure 5: The detailed architecture of variational generator.

A.3 Post-Net

We use non-causal WaveNet as the main architecture of NN in the affine coupling layer. We introduce the number of shared groups N_g, for example, when $N_g = 2$, NNs in flow steps ($\mathbf{f}_1$, $\mathbf{f}_2$, ..., $\mathbf{f}_{K/2}$) and ($\mathbf{f}_{K/2+1}$, $\mathbf{f}_{K/2+2}$, ..., $\mathbf{f}_K$) share the parameters separately. In inference, we can sample z from $N(0, T^2)$, where T is the temperature and use $T = 0.8$ by default.

A.4 Models Used in Section 4.2



(a) TTS with VAE/flow-based Decoder (b) Flow-based Decoder (c) VAE-based Decoder

Figure 6: The detailed architecture of NAR-TTS models with VAE and flow-based decoders.

We use FastSpeech [25] as the backbone for preliminary analyses in Section 4.2. We replace the decoder of FastSpeech with flow-based decoder and VAE-based decoder to explore the characteristics of them. The flow-based decoder is mainly adopted from Glow [9] and WaveGlow [23], which uses the expanded encoder outputs as the condition, as shown in Figure 6a. The VAE-based decoder is similar to the variational generator in our proposed PortaSpeech, except that it does not use the flow-based prior. The model hyperparameters of different model configurations are listed in Table 5.

B Detailed Experimental Settings

In this section, we describe more model configurations and details in subjective evaluation.

Table 5: Hyperparameters of VAE and flow-based TTS models.

	Hyperparameter	Flow-based			VAE-based		
		big	middle	small	big	middle	small
Encoder	Phoneme Embedding	256	192	128	256	192	128
	Layers	4	4	3	4	4	3
	Hidden Size	256	192	128	256	192	128
	Conv1D Kernel	9	5	3	9	5	3
	Conv1D Filter Size	1024	768	512	1024	768	512
VAE Decoder	VAE Encoder Layers	/			8		
	VAE Conv1D Kernel	/			5		
	Latent Size	/			16		
	WaveNet Channel Size	/			300	128	128
	VAE Decoder Layers	/			16	12	12
Flow Decoder	WaveNet Layers	4			/		
	WaveNet Kernel	5			/		
	WaveNet Channel Size	128	112	112	/		
	Flow Steps	22	6	4	/		
Total Number of Parameters		41.2M	10.2M	4.5M	43.2M	9.3M	4.4M

B.1 Model Configurations

We list the model hyper-parameters of PortaSpeech (normal) and PortaSpeech (small) in Table 6 and total number of parameters of each module in Table 7.

Table 6: Hyperparameters of PortaSpeech (normal) and PortaSpeech (small) models.

Hyperparameter		PortaSpeech (normal)	PortaSpeech (small)
Linguistic Encoder	Phoneme Embedding	192	128
	Word/Phoneme Encoder Layers	4	3
	Hidden Size	192	128
	Conv1D Kernel	5	3
	Conv1D Filter Size	768	512
Varational Generator	Encoder Layers	8	8
	Encoder Kernel	5	3
	Decoder Layers	4	3
	Encoder/Decoder Kernel	5	3
	Encoder/Decoder Channel Size	192	128
	Latent Size	16	16
	VP-Flow Steps	4	3
	VP-Flow Layers	4	4
	VP-Flow Channel Size	64	32
	VP-Flow Conv1D Kernel	3	3
Post-Net	WaveNet Layers	3	3
	WaveNet Kernel	3	3
	WaveNet Channel Size	192	128
	Flow Steps	12	8
	Shared Groups	3	2
Total Number of Parameters		21.8M	6.7M

B.2 Details in Subjective Evaluation

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. For audio quality evaluation (MOS-Q and CMOS-Q), we tell listeners to "*focus*

Table 7: Total number of parameters of each module in PortaSpeech (normal) and PortaSpeech (small).

Modules	PortaSpeech (normal)	PortaSpeech (small)
Linguistic Encoder	7.2M	2.0M
Duration predictor	0.3M	0.2M
Post-Net	10.8M	3.6M
Decoder in VG	2.5M	0.6M
VP-Flow in VG	1.0M	0.3M
Total	21.8M	6.7M

on examining the naturalness of prosody and rhythm, and ignore the differences of audio quality (e.g., environmental noise, timbre)". For prosody evaluations (MOS-P and CMOS-P), we tell listeners to "*focus on examining the naturalness of prosody and rhythm, and ignore the differences of audio quality (e.g., environmental noise, timbre)*". The screenshots of instructions for testers are shown in Figure 7. We paid $8 to participants hourly and totally spent about $750 on participant compensation.

C Results on Multi-Speaker Dataset

We conduct the MOS evaluation on the multi-speaker dataset: LibriTTS. The results are shown in Table 8 (we use a pre-trained Parallel WaveGAN [36] for LibriTTS as the vocoder). We can draw similar conclusions as that on LJSpeech that PortaSpeech can achieve good prosody and audio quality in terms of MOS-P and MOS-Q, even in more complicated (multi-speaker) scenarios.

Table 8: The audio performance (MOS-Q and MOS-P) comparisons on LibriTTS dataset.

Method	MOS-P	MOS-Q
GT	4.24±0.08	4.36±0.09
GT (vocoder)	4.21±0.09	4.01±0.10
Tacotron 2	3.81±0.10	3.71±0.11
TransformerTTS	3.79±0.09	3.72±0.12
FastSpeech	3.59±0.11	3.61±0.14
FastSpeech 2	3.64±0.11	3.70±0.11
Glow-TTS	3.76±0.15	3.78±0.10
PortaSpeech (normal)	**3.84±0.13**	**3.83±0.13**
PortaSpeech (small)	3.80±0.12	3.81±0.11

D Robustness Evaluation

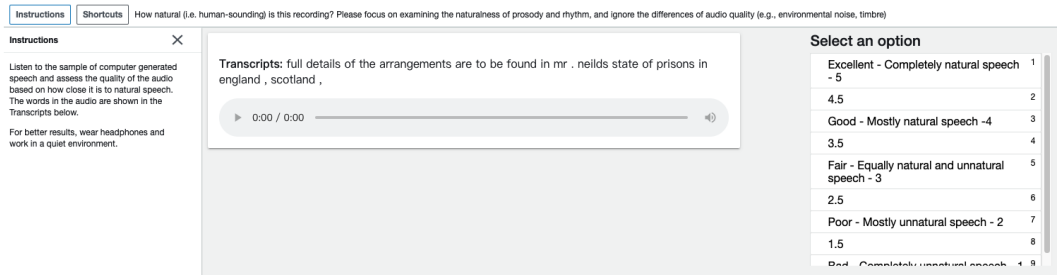
We conduct the robustness evaluation on LJSpeech and LibriTTS datasets. We select 50 sentences that are particularly hard for TTS systems following FastSpeech [25]. The results are shown in Tables 9 and 10. We can see that PortaSpeech achieves comparable robustness performance with state-of-the-art non-autoregressive TTS models.

E Visualization of Attention Weights

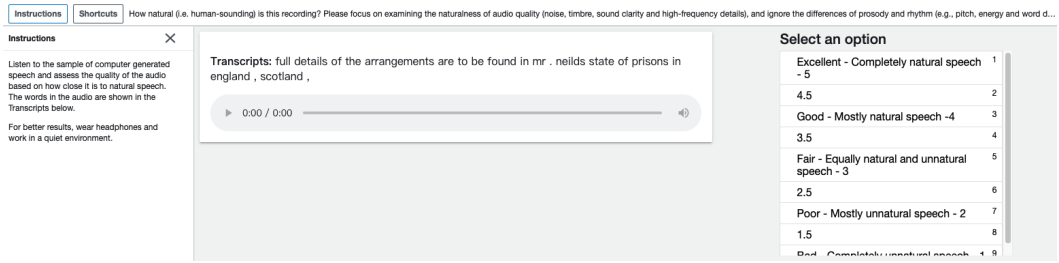
We put some word-to-phoneme attention visualizations in Figure 8. We can see that PortaSpeech can create reasonable phoneme-to-spectrogram alignments which are close to the diagonal, which helps the end-to-end training.

F More Visualizations of Mel-Spectrograms

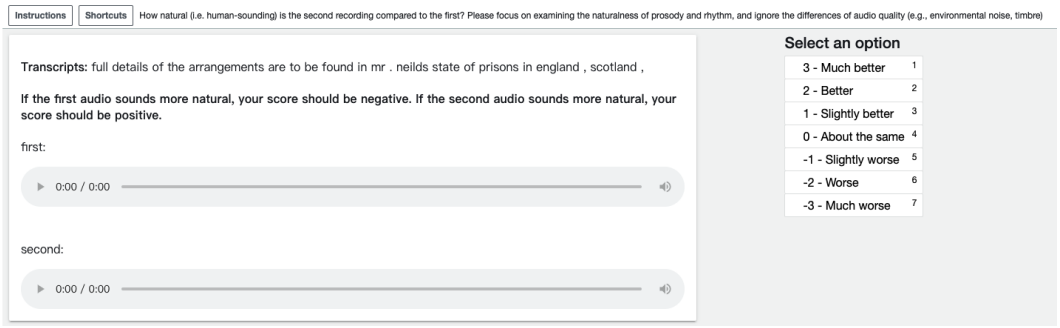
We put more visualizations of mel-spectrograms with different sampling temperatures of post-net and different random seeds on PortaSpeech (normal) in Figure 9 and Figure 10. We have several



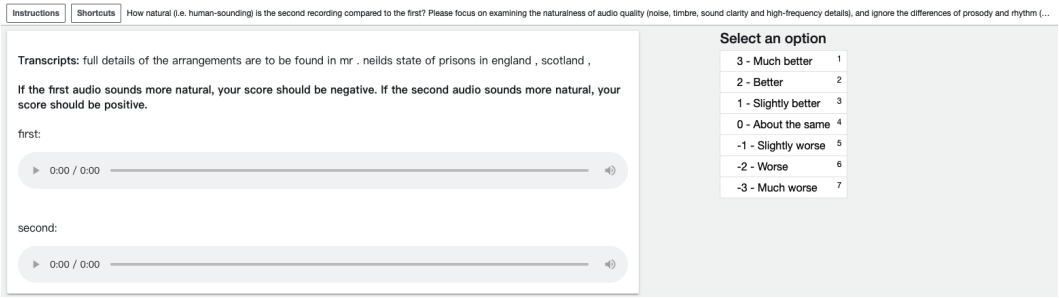
(a) Screenshot of MOS-P testing.



(b) Screenshot of MOS-Q testing.



(c) Screenshot of CMOS-P testing.



(d) Screenshot of CMOS-Q testing.

Figure 7: Screenshots of subjective evaluations.

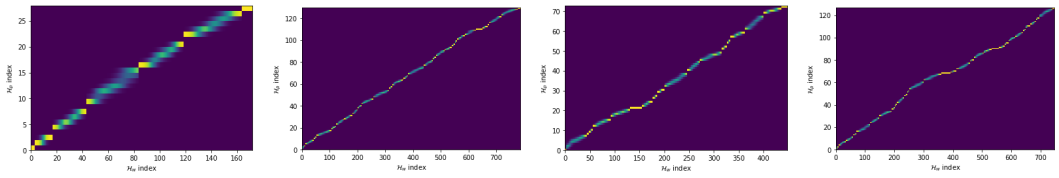
observations: 1) From Figure 9, we can see that when $T = 0.8$, our model can generate natural sound perceptually with reasonable details in mel-spectrograms. 2) From Figure 10, we can see that with different random seeds, PortaSpeech can generate diverse results, which have different prosody and mel-spectrogram details.

Table 9: The robustness evaluation on LJSpeech dataset.

Method	Repeats	Skips	Error Sentences
Tacotron 2	4	5	7
TransformerTTS	7	7	9
FastSpeech	0	1	1
FastSpeech 2	0	1	1
Glow-TTS	0	2	2
PortaSpeech (normal)	1	0	1
PortaSpeech (small)	1	1	1

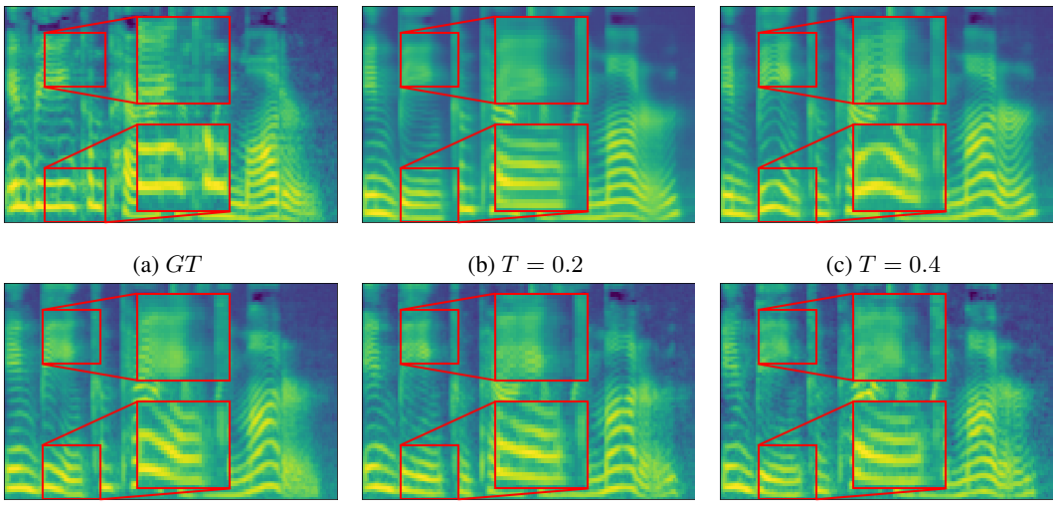
Table 10: The robustness evaluation on LibriTTS dataset.

Method	Repeats	Skips	Error Sentences
Tacotron 2	6	7	12
TransformerTTS	10	12	15
FastSpeech	2	1	2
FastSpeech 2	2	1	2
Glow-TTS	5	4	8
PortaSpeech (normal)	1	2	2
PortaSpeech (small)	2	2	2



(a) *LJ001-0002* (b) *LJ001-0003* (c) *LJ001-0004* (d) *LJ001-0005*

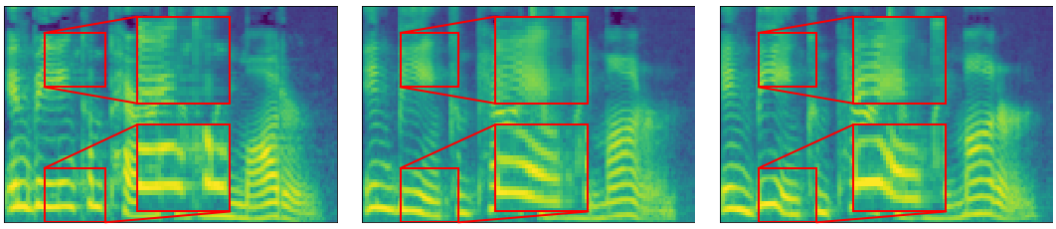
Figure 8: Visualizations of the attention weights.



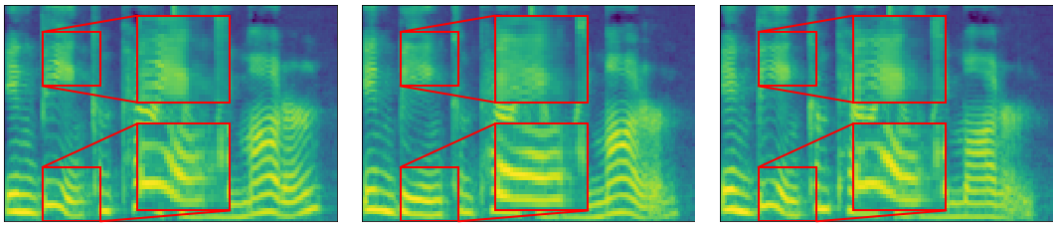
(a) GT (b) $T = 0.2$ (c) $T = 0.4$

(d) $T = 0.6$ (e) $T = 0.8$ (f) $T = 1.0$

Figure 9: Visualizations of the ground-truth and generated mel-spectrograms generated with different sampling temperature T of post-net.



(a) GT (b) $S = 1237$ (c) $S = 1239$



(d) $S = 3237$ (e) $S = 4237$ (f) $S = 9237$

Figure 10: Visualizations of the ground-truth and generated mel-spectrograms generated with different random seeds S.

G Analyses on the Grouped Parameter Sharing Mechanism

In this section, we conduct the subjective evaluation to compare the audio quality with different numbers of shared groups (N_g) for PortaSpeech (normal) and PortaSpeech (small). The results are shown in Table 11. It can be seen that the audio quality drops significantly when sharing parameters among all flow steps, demonstrating the effectiveness of our grouped parameter sharing mechanism.

Table 11: The audio quality (MOS-Q) and number of model parameters (#Params.) comparisons with different number of shared groups (N_g). The evaluation is conducted on a server with 1 NVIDIA 2080Ti GPU and batch size 1. The mel-spectrograms are converted to waveforms using Hifi-GAN (V1) [11].

Method	N_g	MOS-Q	#Params.
GT	/	4.43 ± 0.06	/
GT (voc.)	/	4.12 ± 0.07	/
PortaSpeech (normal)	1	3.86 ± 0.06	19.4M
	3	3.91 ± 0.05	21.8M
	6	3.93 ± 0.07	23.7M
	12	3.92 ± 0.05	28.8M
PortaSpeech (small)	1	3.77 ± 0.06	6.4M
	2	3.87 ± 0.08	6.7M
	4	3.86 ± 0.05	7.5M
	8	3.89 ± 0.06	9.0M

H Potential Negative Societal Impacts

PortaSpeech lowers the requirements for speech synthesis service deployment (memory and CPU performance) and synthesizes high-quality speech voice, which may cause unemployment for people with related occupations such as broadcaster and radio host. In addition, there is the potential for harm from non-consensual voice cloning or the generation of fake media and the voices of the speakers in the recordings might be overused than they expect.