InstructSpeech: Following Speech Editing Instructions via Large Language Models

Anonymous Authors[1]

Abstract

Instruction-guided speech editing aims to follow the user's natural language instruction to manipulate the semantic and acoustic attributes of a speech. In this work, we construct triplet paired data (instruction, input speech, output speech) to alleviate data scarcity and train a multi-task large language model named InstructSpeech. To mitigate the challenges of accurately executing user's instructions, we 1) introduce the learned task embeddings with a fine-tuned Flan-T5-XL to guide the generation process towards the correct generative task; 2) include an extensive and diverse set of speech editing and speech processing tasks to enhance model capabilities; 3) investigate multi-step reasoning for free-form semantic content editing; and 4) propose a hierarchical adapter that effectively updates a small portion of parameters for generalization to new tasks. To assess instruction speech editing in greater depth, we introduce a benchmark evaluation with contrastive instruction-speech pretraining (CISP) to test the speech quality and instruction-speech alignment faithfulness. Experimental results demonstrate that InstructSpeech achieves state-of-the-art results in eleven tasks, for the first time unlocking the ability to edit the acoustic and semantic attributes of speech following a user's instruction. [1]

1. Introduction

Speech editing (Borsos et al., 2022; Bai et al., 2022) is a widely-used application that millions engage with every day, which allows the user to edit the recorded speech without degrading the quality and naturalness. However, existing efforts (Tan et al., 2021; Yang et al., 2023b) are quite limited due to 1) providing only a predefined set of content operations such as inserting missed words, replacing mispronounced words, and removing unwanted speech, while the acoustic attributes (e.g., timbre, volume, speed, emotion, and style) have been relatively overlooked, and 2) requiring a user-drawn mask, or per-example prompt to ensure that iterative edits are applied only to the target region.

To address the aforementioned issues, instruction-based speech editing aims to follow the user's natural language instruction to manipulate both **semantic and acoustic attributes** in speech, which benefits practicality as such guidance is more aligned with human intuition. For instance, a user can provide a model with a speech sample and instruct it to "Make it sound happy" or "Speed up pronunciation", effortlessly describing editing goals using natural language instructions. Despite the significant benefits, achieving high-quality instruction-guided speech inpainting remains challenging due to 1) data scarcity and 2) the complexity of accurately executing instruction.

In this work, we propose InstructSpeech, introducing the first speech editing model to follow human-written instructions. To mitigate the data scarcity, we generate triplet paired data (instruction, input speech, output speech) by combining large models pretrained on text and speech modality. InstructSpeech casts conditional generation as a sequence-to-sequence modeling task and trains a large language model (LLM) using instruction and input speech as conditions and generating output (edited) speech.

To accurately process a variety of instructions, we 1) include the learned task embeddings to steer the generation process toward the correct generative task and fine-tune a Flan-T5-XL to identify the task given any instruction, and 2) train a LLM on an extensive and diverse set of tasks, including both speech editing and speech processing tasks to enhance its capabilities; 3) investigate the multi-step reasoning in free-form semantic editing to alleviate the difficulties of following human's instruction; 4) propose a hierarchical adapter and show that InstructSpeech can generalize to new tasks by solely updating a small portion of parameters in few-shot adaptation.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]Audio samples are available at https://InstructSpeech.github.io

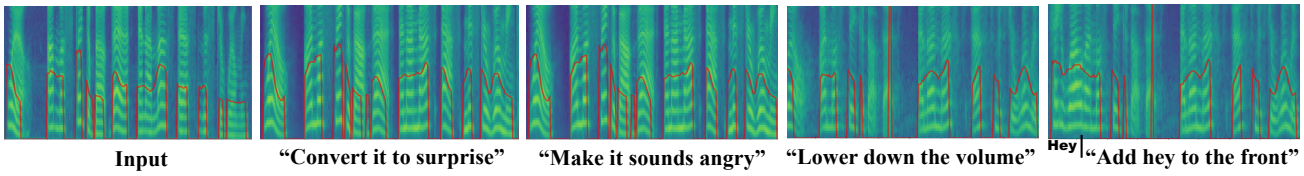


Figure 1: Multi-turn speech editing. Each subsequent speech is derived from the prior one, using its associated instruction.

To assess instruction speech editing in greater depth, a comprehensive benchmark with contrastive instruction-speech pretraining (CISP) is introduced, InstructSpeech exhibits superior speech quality and instruction-speech alignment faithfulness. Experimental results demonstrate that InstructSpeech achieves state-of-the-art results in eleven tasks, for the first time unlocking the ability to edit the acoustic and semantic attributes of speech following a user's instruction. Key contributions of the paper include:

- We collect triplet training data (instruction, input speech, output speech) and propose InstructSpeech - the first speech editing model to follow human instructions.
- We introduce the learned task embeddings and train InstructSpeech on an extensive and diverse set of tasks to enhance its capabilities.
- We investigate multi-step reasoning to eliminate the difficulties in free-form speech editing.
- We propose a hierarchical adapter for efficient adaptation to new tasks, only updating 1% parameters on top.
- We introduce a benchmark evaluation with contrastive instruction-speech pretraining (CISP), and present state-of-the-art quantitative results with qualitative findings.

2. Related Work

2.1. Speech editing

Speech editing systems expect to correct mispronunciation and improve fluency. EdiTTS (Tae et al., 2021) is an off-the-shelf speech editing methodology based on score-based generative modeling for pitch control and content replacement. VoiceBox (Le et al., 2024) supports region-based content editing, which is created by filling frames mapped to unreplaced phones with the original frames and leaving those for new phones with zeros. AudioBox (Vyas et al., 2023) takes as input a masked speech with an accompanying transcript and an optional description and infills the masked portion. EditSpeech (Tan et al., 2021) allows a user to perform deletion, insertion, and replacement of words in a given speech utterance, where partial inference and bidirectional fusion are proposed to achieve smooth transition at both left and right boundaries. Another line of works operates on acoustic attributes (e.g., timbre, emotion, and prosody) and keeps the semantic content representation unchanged: voice conversion (Qian et al., 2019; Chan et al., 2022) aims to alter the voice of a person to suit different styles while conserving the linguistic content. However, these methods still require a user-drawn mask or per-example prompt and are constrained by a predefined set of operations. In this work, we train the speech editing model to follow human-written instructions on an extensive and diverse set of tasks.

2.2. Learning to follow instructions

Several recent studies propose to control audio style through instruction-guided generative models. Prompt-TTS (Guo et al., 2023) takes a prompt with both style and content descriptions as input to synthesize the corresponding speech. Prompt-TTS 2 (Leng et al., 2023) adopts a variation network to provide variability information of voice not captured by text prompts. Instruct-TTS (Yang et al., 2023a) takes advantage of cross-modal learning and captures semantic information from the style prompts to control the speaking style. AUDIT (Wang et al., 2023b) proposes to edit background sound effects, and InstructME (Han et al., 2023) offers a latent diffusion model for instruction-guided music editing and remixing. However, previous works focus on TTS or instruction-guided music/sound editing, following instructions to edit human speech is relatively overlooked.

2.3. Adapting speech generative models

Speech generative models have achieved remarkable advances in recent years, opening up a wide array of applications that leverage their power by adapting models. UniAudio (Yang et al., 2023b) is designed to continuously support new generation tasks (e.g., audio editing) through fine-tuning the whole parameters. Liu et al. (2023) achieve better performance by finetuning low-rank adaptation (LoRA) which adds the linear input projection to each self-attention layer. Vyas et al. (2023) include two-stage full fine-tuning to improve our model fidelity and quality where all parameters are optimized together. Chen et al. (2021) introduce conditional layer normalization and fine-tune this part in addition to speaker embedding for new speaker adaptation. In this work, we present a hierarchical adapter to efficiently fine-tune the cross-attention mechanism in the global transformer and bias/norm for the local transformer, which updates only 1% of the parameters on top.

3. Multi-Task Dataset for Instruction Speech Editing

Region-Based Semantic Content Editing
Add: Insert a new word into the speech given mask.
Remove: Erase a word from the speech given mask.
Replace: Replace the word by another given mask.
Free-Form Semantic Content Editing
Add: Insert a new word following instruction.
Remove: Erase a word following instruction.
Replace: Replace the word following instruction.
Acoustic Editing
Style: Change the style of speech.
Emotion: Change the emotion of speech.
Speed: Change the speaking speed of speech.
Volume: Change the volume of speech.
Gender: Change the gender of speech.
Speech tasks
TTS: Convert phone into corresponding audio.
Frame-level TTS: Convert frame-level phone into audio.
VC: Transform the timbre of a speech.
ASR: Transcribe speech into corresponding text.
Duration: Predict the frame-level phone alignment.

Table 1: Description of the tasks forming the InstructSpeech dataset.

Training a robust and accurate speech editing model to follow human-written instructions typically requires a highly diverse dataset on an extensive and diverse set of tasks, while there are very few resources providing triplet paired data (instruction, input speech, output speech) due to the heavy workload.

Table 1 includes the complete list of tasks. To mitigate the data scarcity, we combine the abilities of two large-scale pretrained models that operate on different modalities: a text large language model (i.e., GPT-3.5-Turbo) and a speech generative model to generate triplet paired data.

3.1. Generating paired speech

We use several different pre-trained speech generative models to synthesize speech samples (after editing) that align well with human instruction.

Emotion, gender and style editing. We train a large-scale, in-context learning Speech LLM and use the speech prompt on the ESD (Zhou et al., 2022), LibriTTS (Zen et al., 2019), and LibriTTS-style datasets to respectively control the emotion, gender, and style. The models are trained in wild data at the scale of around 100K hours (e.g., Librilight (Kahn et al., 2020)), which leads to better generalization for synthesizing unseen speech styles in a zero-shot fashion. To construct LibriTTS-style, we use texts from LibriTTS and 19 provided styles in Microsoft Azure TTS API [6] to synthesize corresponding speech.More details have been included in Appendix A.

Content editing with adding, removing, replacing words. We train a speech editing model to generate target-edited speech samples, which require a user-drawn mask to ensure that editions are applied only to the target region. In practice, we train a TTS model and randomly choose some phonemes to mask during the training stage, where we expect it to recover the whole speech based on the phoneme sequence. In the inference stage, we can mask the region we want to update in the speech and input the new words to obtain the speech samples after the edit.

Speed editing. We train a multi-speaker non-autoregressive TTS model (Popov et al., 2021) with duration predictor to tell us how many frames each element of text input lasts. Following common practice, we control speech tempo by multiplying predicted durations by some factor λ. We set $\lambda = 0.5, 1.0, 1.3$ to generate speech respectively with "slow", "normal", and "fast" pronouncing speed while keeping the speaker/content unchanged.

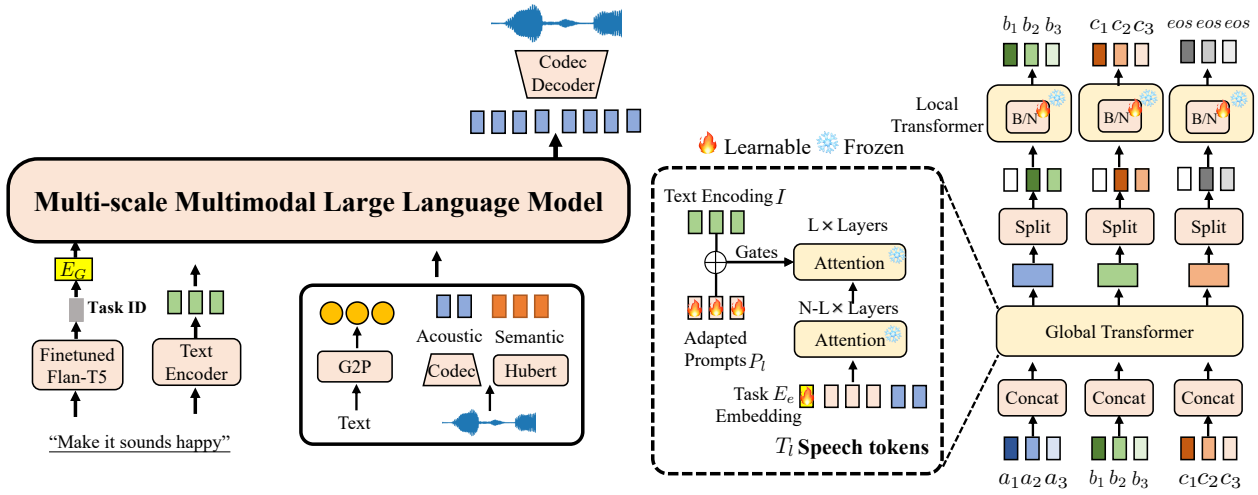
Energy editing. We build three categories of "low", "medium", and "high", indicating the amplitude root mean square (RMS) ranges of $[0.02, 0.04]$, $[0.07, 0.10]$ and $[0.16, 0.20]$, respectively. To construct the dataset, we rescale audio into different ranges dynamically.

3.2. Generating text instruction

For each task, we leverage GPT-3.5-Turbo to generate diverse instructions, where we provide the LLM with a task description and a few task-specific exemplars. For acoustic editing (e.g., timbre, emotion, and prosody), we expect the LLM to output diverse instructions such as "Make this speech sound happy", or "change the style to broadcasting". For semantic content editing, we randomly choose operations (i.e., add, remove, and replace) and indicate the edited words, such as "Add sunny between good and day", "Delete the word sunny", "Replace the word today by tomorrow".

4. InstructSpeech

We train a large language model (LLM) using instruction and input speech as conditions and generating output (edited) speech. In this section, we overview the discrete speech tokens and text representation, and then introduce the decoder-only architecture with differentiable multiscale transformers. Next, we introduce the designs of hierarchical adapter for few-shot adaptation in Section 4.4, as well as the chain-of-thought reasoning in Section 4.5.



(a) Overview of InstructSpeech architecture (b) Hierarchical adapter

Figure 2: A high-level overview of InstructSpeech. We use B/N to denote the bias tuning and norm tuning.

4.1. Speech Representation

For semantic tokens, we apply Hubert (Hsu et al., 2021) and use k-means to discretize 12th-layer embeddings into semantic tokens with a codebook of size 1000 and a total downsampling rate of 320. For acoustic tokens, We tokenize speech clips with the pre-trained SoundStream tokenizer (Zeghidour et al., 2021; Défossez et al., 2022). The audio encoder E of codec models consists of several convolutional blocks with a total downsampling rate of 320 and generates representations at every 20-ms frame in 16kHz, where we flatten n_q codebooks.

4.2. Text representation

Text-guided synthesis models need powerful text encoders to capture the meaning of arbitrary language inputs. We use pre-trained Flan-T5-XL (Raffel et al., 2020) T_i and freeze the weights to derive text representation, which is trained on text-only corpus significantly larger than multimodal data, thus being exposed to a rich distribution of the text.

As illustrated in Figure 2(a), to guide the generation process toward the correct task, we further signal to the model which task it should perform on a given input by prefixing the information with a tag specifying the task. In the inference stage, we predict the task index by fine-tuning a Flan-T5-XL model T_t to identify the task at hand given the input instruction. As expected, conditioning InstructSpeech on task embedding demonstrates a high accuracy in recognizing task categories from human instruction. We refer the reader to Section 7.3 for our findings.

4.3. Architecture

InstructSpeech θ is built upon end-to-end differentiable multiscale transformers (Yu et al., 2023; Yang et al., 2023b) to predict long sequences with sub-quadratic self-attention.

As illustrated in Figure 2(b): 1) the token embedding matrix E_G maps integer-valued tokens $a_1, a_2, ..., c_2, c_3$ to m dimensional embeddings, and concatenate with continuous T5 representation in time axis, following which 2) we chunk it into patches of size P of length $K = \frac{T}{P}$, 3) a large global transformer $\theta_{AR}^{\text{global}}$ module outputs patch representations $\mathbf{G_o^{1:K}} = \theta_{AR}^{\text{global}}(\mathbf{G_i^{0:K-1}})$, and 4) a small local transformer module operates on a single patch containing P elements, each of which is the sum of an output from the global model and an embedding of the previous tokens, and autoregressively predict the next patch $\mathbf{L_o^{1:K}} = \theta_{AR}^{\text{local}}(\mathbf{L_i^{0:K-1}} + \mathbf{G_o^{1:K}})$.

InstructSpeech presents the improvements from scaling the models' size in depth and width without the requirement of scattered model-specific methodologies. As expected, scaling the model size (160M (base), 520M (medium), and 1.2B (large) parameter) results in better scores. We refer the reader to Section7.3 for our findings.

4.4. Few-shot adaptation

To enable few-shot learning for generalization to new tasks, we propose a hierarchical adapter to finetune the LLM given a few examples of a new task. For the global transformer, we adopt a set of learnable adaption prompts P and learn a new task embedding E_e. E_e denotes a new learnable randomly-initialized embedding vector for task adaptation, which is learned by the language modeling loss objective given a few examples of a new task. For the local transformer, we include a normalization tuning bias tuning strategy, where all parameters in normalization layers as well as the bias/scale in linear layers are set to be updated.

Suppose we have instruction representation $I \in \mathbb{R}^{K \times C}$ of

new tasks with length K and feature dimension C, we initialize learnable adaption prompt $\{P_l\}_{l=1}^{L}$ for L transformer layers, where we have each layer's prompt $P_l \in \mathbb{R}^{K\times C}$ and speech tokens $T_l \in \mathbb{R}^{M\times C}$. Then, the refined tokens is conducted an element-wise addition with instruction tokens: $P_l' = P_l + I \in \mathbb{R}^{K\times C}$.

Suppose the model is processing with the speech tokens T_l and refined tokens P_l', The attention score related to the learnable prompt is calculated as $S_l^p = \text{Attention}(T_l, P_l', P_l') = \text{Softmax}(T_l {P_l'}^T/\sqrt{C})P_l'$, and we have S_l^t self-attention score for original speech tokens.

We consider a learnable gating factor (Zhang et al., 2023a) and inject the encoded adaptation prompts to different Transformer layers, gradually providing instruction semantics to avoid disturbing the speech tokens at the beginning of training. A learnable gating factor g_l is adapted to adaptively control the importance of S_l^p in the attention with $S_l = S_l^p g_l + S_l^t$, which represents how much information the learnable prompt contributes. Initialized by zero, g_l can first eliminate the influence of under-fitted prompts and then increase its magnitude to provide more instruction semantics. We compare different adaptation methods in Section 7.3.

4.5. Multi-step reasoning

In this section, we investigate the multi-step reasoning capabilities in InstructSpeech. Multi-step reasoning includes a step-by-step thought process for arriving at the answer, where solutions typically come before the final answer. Specifically, we use multi-step reasoning for **free-form semantic editing tasks** to perform "add", "remove" and "replace" operations following the user's instruction without a predefined mask region. For example, when being asked "Delete the word sunny", InstructSpeech decomposes the problems into intermediate steps as shown in Algorithm 1.

Algorithm 1 Multi-step reasoning for free-form editing. We use E_{ASR}, E_{Dur}, E_I respectively to denote the task embedding of automatic speech recognition, frame-level duration prediction, and task categories prediction tasks.

1: **Input**: InstructSpeech θ, tuned Flan-T5-XL α, speech before edit x, instruction c, task embedding matrix E
2: Predict phone $y = \theta(E_{\text{ASR}}, x)$.
3: Predict phone frame-level duration alignment $d_{MFA} = \theta(E_{\text{Dur}}, x, y)$.
4: Predict task categories $I = \alpha(c)$.
5: Obtain masked speech x_m given predicted alignment d_{MFA} and derive phones to be edited $\tilde{y}$ from instruction c.
6: Deteriorating to region-based semantic content editing: $\tilde{x} = \theta(E_{\text{I}}, x_m, \tilde{y})$.
7: **RETURN** $\tilde{x}$

As such, InstructSpeech tackles the challenges of accurately locating and manipulating the target context following the user's instruction. We refer the reader to Section 7.2 for a summary of our findings.

4.6. Reconstructing High-Fidelity Waveforms

We train a unit-based neural vocoder from scratch for the acoustic unit to waveform generation. Inspired by BigVGAN (Lee et al., 2022), the synthesizer includes the generator and multi-resolution discriminator (MRD). The generator is built from a set of look-up tables (LUT) that embed the discrete representation and a series of blocks composed of transposed convolution and a residual block with dilated layers. The transposed convolutions upsample the encoded representation to match the input sample rate. Details are included in Appendix D.1.

5. Evaluating Instruction-guided Speech Editing Models

We create the benchmark to evaluate instruction-guided speech editing models. Specifically, for each pair of input speech and editing instructions, we use the following the metrics:

Speech intelligibility. We report word error rate (WER) or phone error rate (PER) to evaluate the intelligibility of speech by transcribing it using a whisper (Radford et al., 2023) ASR system following (Wang et al., 2023a).

Style similarity. SIM assesses the coherence of the generated speech in relation to the speaker's characteristics, and we employ the speaker verification model WavLM-TDNN (Chen et al., 2022) to evaluate the speaker similarity. F0 Frame Error (FFE) measures the prosody similarity of synthesized and reference audio. For emotion and style, we train the classifiers to recognize the categories of output speech with GE2E loss (Wan et al., 2018), which measures if the model can accurately produce the target style or emotion given instruction. For pitch, speaking speed, and volume, we adopt a soft-margin mechanism for accuracy calculation in Appendix G.1.

Constrastive instruction-speech pretraining score evaluation. Most existing contrastive pretraining models are optimized using image (CLIP (Radford et al., 2021)) or audio (CLAP (Elizalde et al., 2023)) data, which are difficult to distinguish human speech's fine-grained prosody information such as speaking speed, volume, style or emotion. As such, we fine-tune the CLAP model to learn a multimodal space of speech and text encoders in our instruction datasets using contrastive loss (Vyas et al., 2023). After training, we evaluate the model in downstream audio-text and text-audio retrieval tasks, where we compute the similarity between the audio and text embeddings. Take audio-text retrieval as

an example, top-N descriptions are computed by picking the descriptions corresponding to the top N values in similarity. More details are included in Appendix C.

	Text → Audio			Audio → Text		
	R@1	R@5	R@10	R@1	R@5	R@10
Base	6.7	26.7	36.7	4.5	17.3	31.2
Ours	56.6	86.6	96.7	90.5	99.3	99.8

Table 2: Retrieval accuracy using a contrastive instruction-speech pretraining model.

As can be seen in Table 2, the constrastive instruction-speech pretraining model (ours) achieves the highest retrieval accuracy with T2A R@10 96.7 and A2T R@10 99.8. It indicates the outperformed capabilities in assessing the coherence of the generated speech in relation to the natural language instruction. After training the CISP model, we use it to evaluate instruction-guided speech editing models by calculating: 1) CISP text-speech direction similarity (CISPtext) – measuring alignment between instruction and edited speech, and 2) CISP speech similarity (CISPspeech) – measuring the change between edited and input speech.

Subjective evaluation. We also conduct a crowd-sourced human evaluation via Amazon Mechanical Turk, which is reported with 95% confidence intervals (CI), and analyze two aspects: style similarity (speaker, emotion, and prosody) and audio quality (clarity, high-frequency), respectively scoring SMOS and MOS. More information on evaluation has been attached in Appendix G.2.

6. Training setup

6.1. Dataset

For speech processing and speech editing tasks, we use Librilight (Kahn et al., 2020), LibriSpeech (Panayotov et al., 2015), LibriTTS (Zen et al., 2019), and VCTK (Veaux et al., 2017) datasets. Besides, the ESD (Zhou et al., 2022) dataset with 5 emotion categories and the synthesized dataset LibriTTS-style with 19 style categories are further included, respectively for emotion and style editing. For region-based content editing, we adopt Montreal Forced Aligner (McAuliffe et al., 2017) to calculate the alignment between phoneme and speech. We tokenize text into the phoneme sequence with an open-source grapheme-to-phoneme conversion tool (Sun et al., 2019) and convert the sampling rate of all data to 16kHz. The detailed data statistics for each task are included in Appendix A.

6.2. Model Configurations

For acoustic representation, we train the SoundStream model with 12 quantization levels, each with a codebook of size 1024 and the same downsampling rate of 320. We train three sets of InstructSpeech, with 160M (base), 520M (medium), and 1.2B (large) parameters. As for the unit-based vocoder, we use the modified V1 version of BigVGAN. A comprehensive table of hyperparameters is available in Appendix D. Except explicitly stated, we use our 520M (medium) model for downstream evaluation.

During training, we train InstructSpeech for 100K steps using 8 V100 GPUs with a batch size of 6000 tokens for each GPU on the publicly-available *fairseq* framework (Ott et al., 2019). Adam optimizer is used with $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-9}$. BigVGAN is optimized with a segment size of 8192 and a learning rate of 1×10^{-4} until 500K steps using 4 V100 GPUs. For sampling, we employ top-p (Holtzman et al., 2019) sampling with $p = 0.25$.

6.3. Baseline models

We also compare the InstructSpeech with other systems, including 1) GT, the ground-truth audio; 2) GT (voc.), where we first convert the ground-truth audio into tokens and then convert them back to audio using BigVGAN; 3) EditSpeech (Tan et al., 2021) and A3T (Bai et al., 2022) perform region-based content editing with deletion, insertion and replacement of words in a given speech utterance; 4) YourTTS (Casanova et al., 2022), as we are investigating a new task with no previous work to compare with, we train YourTTS with speech prompt for emotion or style guidance.

7. Results

7.1. Quantitative Results

The objective and subjective evaluation is presented in Tables, and extensive experiments are included in Appendix F.

Semantic content editing. We first test the intelligibility of speech in semantic content editing: 1) For **region-based semantic editing**, InstructSpeech has achieved a WER of 5.6 averaged across different operations, indicating that InstructSpeech could generate accessible speech of good quality as previous non-autoregressive speech editing families. 2) For **free-form semantic editing**, InstructSpeech has demonstrated an overall PER of 7.3 in phone recognition, as well as a mean absolute error (MAE) of 0.61 in frame-level phone alignment prediction averaged across different operations. InstructSpeech is aware of the speech segmentation and their corresponding phones, which to the best of our knowledge is first model available for free-form editing following human's instruction, attributing to the multi-step reasoning.

Acoustic attribute editing. We also evaluate our model to manipulate acoustic attributes (e.g., gender, emotion, and style) and keep the semantic content unchanged: 1) Regarding similarity, InstructSpeech scores the highest accu-

	Speed	Volume	Emotion		Style		CISP score		Subjective evaluation	
	Acc	Acc	Acc	FFE	Acc	FFE	CISPtext	CISPspeech	MOS	SMOS
GT	86.4	91.6	86.9	/	96.1	/	/	/	4.32±0.08	/
GT (voc.)	86.2	90.9	82.8	0.04	93.1	0.03	0.62	0.71	4.25±0.07	4.21±0.06
YourTTS	/	/	38.7	0.39	30.3	0.29	/	/	3.91±0.07	3.86±0.06
Base	81.2	88.1	46.3	0.42	88.3	0.46	0.58	0.66	3.98±0.07	3.92±0.08
Medium	84.0	90.6	55.2	0.41	90.6	0.52	0.59	0.68	4.01±0.06	3.95±0.07
Large	84.9	91.3	57.3	0.39	92.3	0.38	0.61	0.69	4.05±0.07	4.04±0.06

Table 3: Acoustic editing results. We modify YourTTS to include speech prompt for emotion or style guidance.

	Add	Remove	Replace	SIM
GT	5.8	5.7	7.5	0.98
EditSpeech	9.2	5.9	6.5	0.97
A3T	7.5	7.8	6.9	0.96
Base	6.3	4.2	5.9	0.97
Medium	6.4	5.5	5.8	0.98
Large	5.1	4.9	5.6	0.98

Table 4: Region-based content editing. We report WER of adding, removing, and replacing operation, as well as the overall SIM in LibriSpeech-test set.

Multi-step reasoning	Results		
Step 1: Phone recognition	PER: 7.3		
Step 2: Alignment Prediction	Acc: 62.8; MAE: 0.61		
Step 3: Overall WER	Add	Remove	Replace
Base	13.3	12.5	15.0
Medium	13.1	12.0	14.5
Large	12.6	11.9	14.1

Table 5: Multi-step reasoning for free-form content editing. We evaluate our models in LibriSpeech-test set.

racy of 55.2 and 90.6 respectively in emotion and style editing, showing that InstructSpeech excels at transferring the prosody of custom voices following instruction. Informally, InstructSpeech is optimized in a large amount of self-supervised data, which contains many speakers with various accents and diverse demographics to improve robustness and generalization. 2) For speed and volume, InstructSpeech also effectively alters its speaking style guided by human instruction. 3) Regarding CISP direction similarity (CISPtext and CISPSpeech), InstructSpeech presents that strong coherence of 0.59 between text instruction and edited speech, as well as a high similarity of 0.68 between the speech before and after edit. It suggests the precise speech editing following user's instruction while keeping the other attributes in consistent with the speech before edit.

Subjective Human Evaluation The evaluation of the instruction editing models is very challenging due to its subjective nature in perceptual quality, and thus we include a human evaluation: InstructSpeech achieves the high perceptual quality with MOS of and SMOS of 4.01 and 3.95. It indicates that raters prefer our model synthesis against baselines in terms of audio naturalness and faithfulness.

7.2. Qualitative Findings

Firstly, we explore the region-based content editing and compare the results with baseline models (i.e., A3T or EditSpeech). As shown in Figure 4, InstructSpeech presents a smooth transition (i.e., pitch contours) between the edited and origin region and demonstrates good intelligibility. We attach more mel-spectrograms of edited samples and corresponding pitch tracks in Appendix E.

We present the free-form semantic editing examples in Table 9: InstructSpeech step-by-step recognizes the phone sequence and frame-level duration alignment, in the following InstructSpeech leverages the region-based speech editing with user-drawn mask to manipulate the sequence. As such, the multi-step reasoning process tackles the challenges of executing user's instructions, enpowering InstructSpeech to accurately locate and manipulate the target.

To visualize whether different style are identified in generated samples, we randomly sample 19 styles; Each is converted into a 256-dimensional embedding and reduced to 2-dimensional with Uniform Manifold Approximation and Projection (UMAP). As can be seen in Figure 3, InstructSpeech presents style-aware acoustic editing with significant inter-class distance. We also include the UMAP of emotional samples in Figure 6 in Appendix E.

7.3. Analysis and Ablation Studies

To verify the capabilities of InstructSpeech, we conduct ablation studies and discuss the key findings in this section.

Combining speech tasks. To demonstrate the effectiveness of optimizing instruction editing models on an extensive and diverse set of tasks, we train two additional models on all tasks except: (i) frame-level TTS task, and (ii) VC task. As we show in Table 6, adding the frame-level TTS improves the model performance in region-based content editing, where it assists to encode aligned phone sequences and generate intelligible speech. Similarly, VC assists the

Task	Content editing		Emotion editing	
	WER	SIM	Acc	FFE
InstructSpeech	5.9	0.98	55.2	0.41
w/o multitask	7.4	0.97	51.3	0.43

(a) **Multitask learning**

Task embedding	Acc
GT embed	100%
Predicted embed	100%
w/o embed	73.2%

(b) **Task embedding conditions**

	Params	WER	SIM
All	438M	0.32	0.86
Lora	8.97M	0.41	0.84
HA	3.03M	0.38	0.85

(c) **Finetuning in emotion task**

Table 6: Ablation studies. In Figure (a), we train two models on all tasks except frame-level TTS and VC task, and test them respectively in region-based content editing and emotion editing. In Figure (c), we use HA to denote the hierarchical adapter.

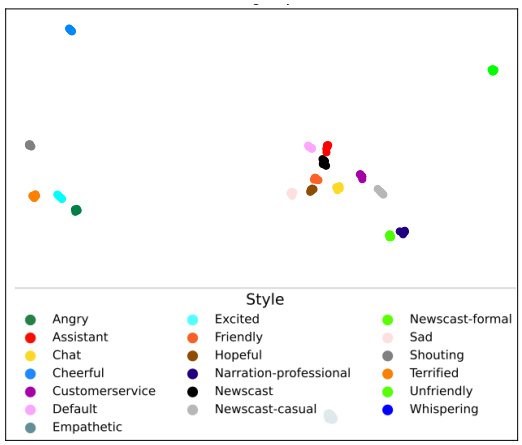


Figure 3: UMAP of style manipulation results.

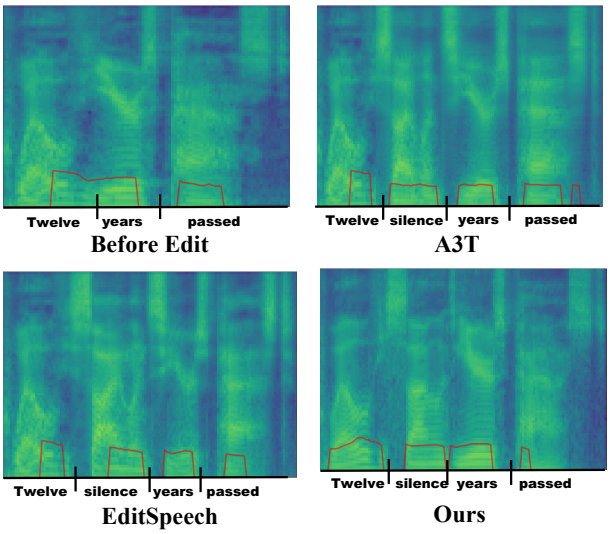


Figure 4: Region-based speech editing results. We add the word "silence" to the speech "Twelve years passed".

model in understanding semantic representation, thereby improving model robustness and generalization.

Scalability to improve performance. We also report results for different model sizes, namely 160M (base), 520M (medium), and 1.2B (large) parameter models. As expected, scaling the model size results in better scores. For example, increasing the model size from 520M to 1.2B leads to a 0.75% reduction in WER averaged across region-based speech editing tasks and 3.7% accuracy improvement in speed manipulation tasks.

Task embedding. Inspired by (Sheynin et al., 2023), we compare to condition InstructSpeech on the ground-truth task embedding or the task embedding predicted by the task predictor. Additionally, a model without a task embedding (still having T5 instruction condition) is also investigated. To conclude, the T5 task embedding predictor demonstrates high accuracy in recognizing task categories from human instruction, and thus, whether the task embedding is predicted or not makes no difference. In contrast, we observe that without conditioning on the task type, the model may perform the wrong editing operation.

Few-shot learning with hierarchical adapter. To enable few-shot learning of new tasks without losing the general abilities, we fine-tune InstructSpeech in only 1-hour unseen ESD data, and compare the results among different adaptation methods. Illustrated in Table 6(c), as a lightweight plug-and-play module, the proposed hierarchical adapter enjoys superior training efficiency with only 1% parameters in contrast to lora or full finetuning and demonstrates the outperformed WER and SIM score. This enables us to fine-tune instruction-editing LLMs on cheap devices.

8. Conclusion

In this work, we presented InstructSpeech with the first attempt to edit the acoustic and semantic attributes of speech given a user's instruction input. To steer the generation process toward the correct generative task, we included learned task embeddings and fine-tuned a Flan-T5-XL to identify the task given the input instruction. InstructSpeech included multitask learning on an extensive and diverse set of speech editing and speech processing tasks, enhancing its capabilities to manipulate a speech's semantic and acoustic attributes. We investigated the multi-step reasoning to alleviate the difficulties in following human instruction, and thus, InstructSpeech was the only model available for free-form semantic editing. To generalize to unseen tasks, we proposed a hierarchical adapter to update only 1% of parameters efficiently. The comprehensive metrics with contrastive instruction-speech pretraining (CISP) demonstrated that InstructSpeech achieved state-of-the-art results in 11 editing tasks with superior speech quality and instruction-speech alignment faithfulness. We envisage that our work serves as a basis for future speech editing studies.

9. Potential Negative Societal Impacts

This paper aims to advance open-domain instruction-guided speech editing, which will ease the effort of speech and digital art creation. The multitask learning on an extensive and diverse set of speech editing and speech processing tasks, enhancing its capabilities to manipulate a speech's semantic and acoustic attributes. A negative impact is the risk of misinformation. To alleviate it, we can train an additional classifier to discriminate the fakes. We believe the benefits outweigh the downsides.

InstructSpeech lowers the requirements for high-quality instruction-guided speech editing, which may cause unemployment for people with related occupations, such as speech engineers and radio hosts. In addition, there is the potential for harm from non-consensual voice cloning or the generation of fake media, and the voices in the recordings might be overused than they expect.

References

Bai, H., Zheng, R., Chen, J., Ma, M., Li, X., and Huang, L. A3t: Alignment-aware acoustic and text pretraining for speech synthesis and editing. In *International Conference on Machine Learning*, pp. 1399–1411. PMLR, 2022.

Borsos, Z., Sharifi, M., and Tagliasacchi, M. Speechpainter: Text-conditioned speech inpainting. *arXiv preprint arXiv:2202.07273*, 2022.

Casanova, E., Weber, J., Shulby, C. D., Junior, A. C., Gölge, E., and Ponti, M. A. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*, pp. 2709–2720. PMLR, 2022.

Chan, C. H., Qian, K., Zhang, Y., and Hasegawa-Johnson, M. Speechsplit2. 0: Unsupervised speech disentanglement for voice conversion without tuning autoencoder bottlenecks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6332–6336. IEEE, 2022.

Chen, M., Tan, X., Li, B., Liu, Y., Qin, T., Zhao, S., and Liu, T.-Y. Adaspeech: Adaptive text to speech for custom voice. *arXiv preprint arXiv:2103.00993*, 2021.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

Défossez, A., Copet, J., Synnaeve, G., and Adi, Y. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

Elizalde, B., Deshmukh, S., Al Ismail, M., and Wang, H. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Guo, Z., Leng, Y., Wu, Y., Zhao, S., and Tan, X. Prompttts: Controllable text-to-speech with text descriptions. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Han, B., Dai, J., Song, X., Hao, W., He, X., Guo, D., Chen, J., Wang, Y., and Qian, Y. Instructme: An instruction guided music edit and remix framework with latent diffusion models. *arXiv preprint arXiv:2308.14360*, 2023.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Kahn, J., Riviere, M., Zheng, W., Kharitonov, E., Xu, Q., Mazaré, P.-E., Karadayi, J., Liptchinsky, V., Collobert, R., Fuegen, C., et al. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7669–7673. IEEE, 2020.

Kong, J., Kim, J., and Bae, J. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Proc. of NeurIPS*, 2020.

Le, M., Vyas, A., Shi, B., Karrer, B., Sari, L., Moritz, R., Williamson, M., Manohar, V., Adi, Y., Mahadeokar, J., et al. Voicebox: Text-guided multilingual universal speech generation at scale. *Advances in neural information processing systems*, 36, 2024.

Lee, S.-g., Ping, W., Ginsburg, B., Catanzaro, B., and Yoon, S. Bigvgan: A universal neural vocoder with large-scale training. *arXiv preprint arXiv:2206.04658*, 2022.

Leng, Y., Guo, Z., Shen, K., Tan, X., Ju, Z., Liu, Y., Liu, Y., Yang, D., Zhang, L., Song, K., et al. Prompttts 2: Describing and generating voices with text prompt. *arXiv preprint arXiv:2309.02285*, 2023.

Liu, A. H., Le, M., Vyas, A., Shi, B., Tjandra, A., and Hsu, W.-N. Generative pre-training for speech with flow matching. *arXiv preprint arXiv:2310.16338*, 2023.

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., and Sonderegger, M. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pp. 498–502, 2017.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

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

Popov, V., Vovk, I., Gogoryan, V., Sadekova, T., and Kudinov, M. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pp. 8599–8608. PMLR, 2021.

Qian, K., Zhang, Y., Chang, S., Yang, X., and Hasegawa-Johnson, M. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*, pp. 5210–5219. PMLR, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518. PMLR, 2023.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Sheynin, S., Polyak, A., Singer, U., Kirstain, Y., Zohar, A., Ashual, O., Parikh, D., and Taigman, Y. Emu edit: Precise image editing via recognition and generation tasks. *arXiv preprint arXiv:2311.10089*, 2023.

Sun, H., Tan, X., Gan, J.-W., Liu, H., Zhao, S., Qin, T., and Liu, T.-Y. Token-level ensemble distillation for grapheme-to-phoneme conversion. *arXiv preprint arXiv:1904.03446*, 2019.

Tae, J., Kim, H., and Kim, T. Editts: Score-based editing for controllable text-to-speech. *arXiv preprint arXiv:2110.02584*, 2021.

Tan, D., Deng, L., Yeung, Y. T., Jiang, X., Chen, X., and Lee, T. Editspeech: A text based speech editing system using partial inference and bidirectional fusion. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 626–633. IEEE, 2021.

Veaux, C., Yamagishi, J., MacDonald, K., et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 6:15, 2017.

Vyas, A., Shi, B., Le, M., Tjandra, A., Wu, Y.-C., Guo, B., Zhang, J., Zhang, X., Adkins, R., Ngan, W., et al. Audiobox: Unified audio generation with natural language prompts. *arXiv preprint arXiv:2312.15821*, 2023.

Wan, L., Wang, Q., Papir, A., and Moreno, I. L. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879–4883. IEEE, 2018.

Wang, C., Chen, S., Wu, Y., Zhang, Z., Zhou, L., Liu, S., Chen, Z., Liu, Y., Wang, H., Li, J., et al. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*, 2023a.

Wang, Y., Ju, Z., Tan, X., He, L., Wu, Z., Bian, J., and Zhao, S. Audit: Audio editing by following instructions with latent diffusion models. *arXiv preprint arXiv:2304.00830*, 2023b.

Yang, D., Liu, S., Huang, R., Lei, G., Weng, C., Meng, H., and Yu, D. Instructtts: Modelling expressive tts in discrete latent space with natural language style prompt. *arXiv preprint arXiv:2301.13662*, 2023a.

Yang, D., Tian, J., Tan, X., Huang, R., Liu, S., Chang, X., Shi, J., Zhao, S., Bian, J., Wu, X., et al. Uniaudio: An audio foundation model toward universal audio generation. *arXiv preprint arXiv:2310.00704*, 2023b.

Yu, L., Simig, D., Flaherty, C., Aghajanyan, A., Zettlemoyer, L., and Lewis, M. Megabyte: Predicting million-byte sequences with multiscale transformers. *arXiv preprint arXiv:2305.07185*, 2023.

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J., Jia, Y., Chen, Z., and Wu, Y. Libritts: A corpus derived from librispeech for text-to-speech. *arXiv preprint arXiv:1904.02882*, 2019.

Zhang, R., Han, J., Zhou, A., Hu, X., Yan, S., Lu, P., Li, H., Gao, P., and Qiao, Y. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023a.

Zhang, Z., Zhou, L., Wang, C., Chen, S., Wu, Y., Liu, S., Chen, Z., Liu, Y., Wang, H., Li, J., et al. Speak foreign languages with your own voice: Cross-lingual neural codec language modeling. *arXiv preprint arXiv:2303.03926*, 2023b.

Zhou, K., Sisman, B., Liu, R., and Li, H. Emotional voice conversion: Theory, databases and esd. *Speech Communication*, 137:1–18, 2022.

Appendices

InstructSpeech: Following Speech Editing Instructions via Large Language Models

A. Data

Semantic Editing
Add/Remove/Replace: LibriTTS, VCTK
Acoustic Editing
Style: LibriTTS-style
Emotion: ESD
Speed/Volume/Gender: LibriTTS
Background sound: Audioset
Speech tasks
TTS: VCTK, LibriTTS
VC: Librilight
ASR: LibriSpeech

Table 7: Original data to construct instruction speech editing dataset.

B. Instruction speech editing dataset construction

To construct instruction speech editing datasets, we train speech LLMs to synthesize speech samples (after editing) that align well with human instruction, where we use the same training objective and architecture as InstructSpeech. Specifically, we 1) tokenize speech samples and construct sequence for in-context learning. 2) build the architecture upon end-to-end differentiable multiscale transformers to predict long sequences as described in Section 4.3; and 3) train the model with a language modeling objective (i.e., next-token prediction task) with the same configurations in Section 6.2.

As illustrated in Figure (a), given a training sample (phone a and speech b pair) in the ESD dataset, the overall token sequence includes 1) phones a or Hubert tokens [5] of b, where we use k-means to discretize 12th-layer embeddings into Hubert tokens with a codebook of size 1000; 2) emotion prompt (acoustic tokens of a randomly chosen sample with the same emotion as b's), 3) speaker prompt (acoustic tokens of a randomly chosen sample with the same speaker as b's), and 4) acoustic tokens of b.

During inference, the model can be prompted with phones/Hubert tokens and emotion, speaker prompts to generate the target with acoustic attributes to be coherent with the prompt. For example, given a sample and instruction with "convert it into happy emotion", we take its phone or Hubert tokens as input, and utilize a randomly chosen "happy" sample as the emotion prompt, and a randomly chosen sample with the same timbre as the speaker prompt, to synthesize speech samples (after editing) that align well with human instruction.

Librilight is included to promote the data scale. As illustrated in Figure 2 (a), given a training speech, the overall token sequence becomes 1) Hubert tokens [5], 2) emotion prompt (acoustic tokens of a randomly chosen Librilight sample to denote the Neutral emotion), 3) speaker prompt (acoustic tokens of a randomly chosen sample with the same speaker), and 4) acoustic target. Inspired by [4], we combine both unsupervised and supervised datasets in speech generative models for better generalization to unseen speaking style prompts, in a zero-shot fashion.

C. Contrastive instruction-speech pretraining (CISP)

The CISP model jointly trains audio and text encoder to learn a common multimodal space using contrastive learning. Let the training data be $D = \{(X_i^a, X_i^t)\}_{i=1}^{i=N}$. Let f_{audio} be the audio encoder and f_{text} be the text encoder which are learnable

embedding functions. The audio encoder converts the raw audio into a log Mel spectrogram followed by a learnable embedding function.

The model is trained with the contrastive learning paradigm between the audio E_i^a and text embeddings E_i^t in pair:

$$\begin{aligned} E_i^a &= MLP_{\text{audio}}\left(f_{\text{audio}}\left(X_i^a\right)\right) \\ E_i^t &= MLP_{\text{text}}\left(f_{\text{text}}\left(X_i^t\right)\right) \end{aligned} \tag{1}$$

$$L = \frac{1}{2N}\sum_{i=1}^{N}\left(\log\frac{\exp\left(E_i^a \cdot E_i^t/\tau\right)}{\sum_{j=1}^{N}\exp\left(E_i^a \cdot E_j^t/\tau\right)} + \log\frac{\exp\left(E_i^t \cdot E_i^a/\tau\right)}{\sum_{j=1}^{N}\exp\left(E_i^t \cdot E_j^a/\tau\right)}\right) \tag{2}$$

Where τ is a learnable temperature parameter for scaling the loss, and N is the number of data. Following (Radford et al., 2021; Elizalde et al., 2023), two logarithmic terms consider either audio-to-text logits or text-to-audio logits.

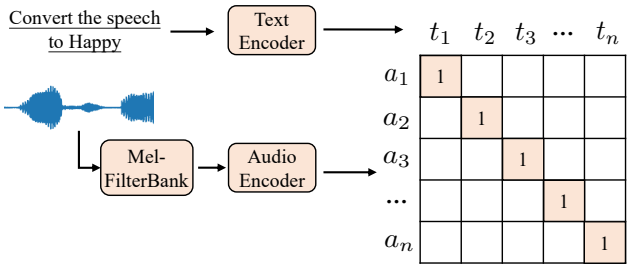


Figure 5: The contrastive instruction-speech pretraining process.

D. Model Architectures

In this section, we list the model hyper-parameters of InstructSpeech in Table 8.

Hyperparameter		MVoice
Global Base	Transformer Layer Transformer Embed Dim Transformer Attention Headers Number of Parameters	16 768 12 114 M
Global Medium	Transformer Layer Transformer Embed Dim Transformer Attention Headers Number of Parameters	20 1152 16 320 M
Global Large	Transformer Layer Transformer Embed Dim Transformer Attention Headers Number of Parameters	24 1536 32 930 M
Local	Transformer Layer Transformer Embed Dim Transformer Attention Headers Number of Parameters	6 Same as global 8 46/101/303 M
BigVGAN Vocoder	Upsample Rates Hop Size Upsample Kernel Sizes Number of Parameters	[5, 4, 2, 2, 2, 2] 320 [9, 8, 4, 4, 4, 4] 121.6M

Table 8: Hyperparameters of InstructSpeech.

D.1. Unit-based Vocoder

The generator of the unit-based vocoder is built from a set of look-up tables (LUT) that embed the discrete representation, and a series of blocks composed of transposed convolution and a residual block with dilated layers. We train the enhanced vocoder with the weighted sum of the least-square adversarial loss, the feature matching loss, and the spectral regression loss on mel-spectrogram, where the training objective formulation and hyperparameters follow (Kong et al., 2020; Lee et al., 2022).

E. Case study

In this section, we visualize pairs of speech samples (i.e., before and after the edit) via semantic or acoustic manipulation using InstructSpeech.

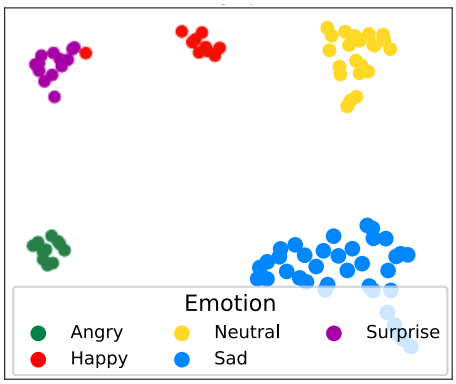


Figure 6: UMAP of emotion manipulation results.

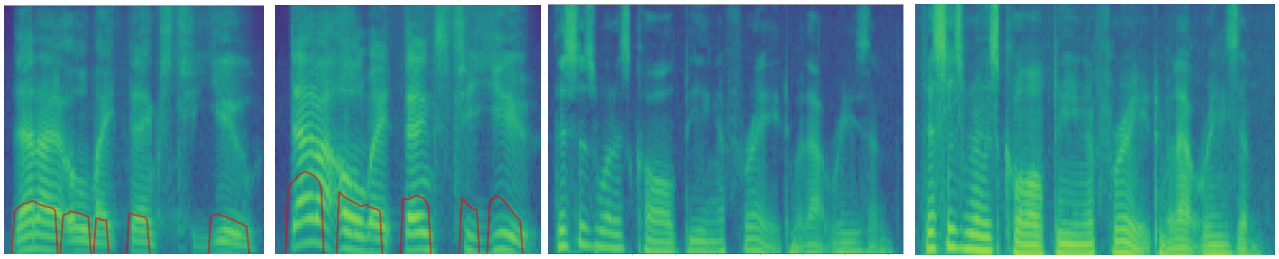


Figure 7: Acoustic editing results. Left: “manipulate the emotion to angry”. Right: “changes its volume to high”

We present several examples sampled from the free-form editing in Table 9,

Instruction:	Add the word “virtue” between the word “I” and “have”
Source:	I have played the flute to the hurricane.
Target:	I virtue have played the flute to the hurricane.
Target frame-level duration alignment	13
Predicted frame-level duration alignment	12
Instruction:	Add the word “preceded” between the word “were” and “of”
Source:	The walls were of mud, the roof was of straw, and there was more thatch than wall
Target:	The walls were preceded of mud, the roof was of straw, and there was more thatch than wall
Target frame-level duration alignment	37
Predicted frame-level duration alignment	37

Table 9: Two examples comparing free-form editing produced by InstructSpeech.

F. Extensive experimental results

In this section, we include the detailed extensive experiments to further demonstrate the results.

F.1. Region-based content editing

For region-based content editing, We include a subjective evaluation by respectively scoring MOS and SMOS, rated from 1 to 5 and reported with 95% confidence intervals (CI). For easy comparison, the results are compiled and presented in the following table.

	Add	Remove	Replace	SIM	MOS	SMOS
GT	5.8	5.7	7.5	0.98	4.26±0.07	/
EditSpeech	9.2	5.9	6.5	0.97	3.89±0.07	4.02±0.07
A3T	7.5	7.8	6.9	0.96	3.93±0.08	3.98±0.07
Base	6.3	4.2	5.9	0.97	3.95±0.06	4.01±0.06
Medium	6.4	5.5	5.8	0.98	3.96±0.07	4.01±0.07
Large	5.1	4.9	5.6	0.98	3.99±0.09	4.03±0.05

InstructSpeech (medium) achieves high perceptual quality with MOS and SMOS of 3.96 and 4.01. It indicates that raters prefer our model synthesis against baselines in terms of speech intelligibility and style similarity, which is consistent with the objective evaluation results.

F.2. Free-form content editing

Model	Add	Remove	Replace
Cascaded	14.5	11.0	17.3
Base	13.3	12.5	15.0
Medium	13.1	12.0	14.5
Large	12.6	11.9	14.1

We include the optimized way for speech editing, which can be decomposed into multiple sub-tasks using the following external cascaded models:

- The speech sample is transcribed by the whisper ASR system.
- The transcription is tokenized into phones using grapheme-to-phoneme tools.
- We mask the original speech we want to update based on phone-frame duration alignment, which is generated using the MFA tool trained on LibriTTS dataset.
- Given phone and masked speech, the edited speech is generated by a region-based speech editing model (i.e., InstructSpeech).

InstructSpeech prompts step-by-step, significantly tackling the challenges of accurately locating and manipulating the target context following the user's instruction. InstructSpeech presents its advantages in multitask prediction since it is trained on an extensive and diverse set of tasks to enhance capabilities, while these cascaded models are usually optimized in varying ways and datasets, where the domain gap can lead to cascaded error and quality degradation.

F.3. Acoustic editing

We include the comparison with several baselines (i.e. VALL-E (Wang et al., 2023a) and Spear-TTS (Zhang et al., 2023b)) on the benchmark zero-shot TTS tasks in speaker transferring. Specifically, we report the WER and SIM to respectively assess the audio quality and style similarity, using a small-scale test set with the examples provided on the demo page. We also score MOS and SMOS for subjective evaluation, rated from 1 to 5 and reported with 95% confidence intervals (CI). For easy comparison, the results are compiled and presented in the following table.

Model	MOS (↑)	SMOS (↑)	WER (↓)	SIM (↑)
VALL-E	3.92±0.12	3.81±0.07	6.5	0.79
Spear-TTS	3.97±0.06	3.89±0.04	5.7	0.83
InstructSpeech	4.04±0.08	3.94±0.06	5.0	0.85

InstructSpeech presents a 1.5 lower score WER and 0.06 higher SIM than VALL-E, also achieving superior results compared to Spear-TTS. To conclude, InstructSpeech avoids cascaded errors (VALL-E's cascaded NAR and AR models, and Spear-TTS's cascaded semantic and acoustic tokens), and trains the LLM on an extensive and diverse set of tasks at the scale of around 100K hours including both speech editing and speech processing tasks to enhance its capabilities.

G. Evaluation

G.1. Objective Evaluation

For emotion and style controlling accuracy, we train an open-source GE2E with our speech data. We train the emotion and style classifiers respectively on ESD dataset and the constructed LibriTTS-style dataset with GE2E loss, which achieve the high accuracy of 99.8 and 97.6 averaged across emotions and styles, serving as metrics to evaluate the similarity of generated samples.

For controlling accuracies on volume, pitch, and speaking speed, considering that the values of generated singing may slightly deviate from the boundaries used for categorization, we adopt a soft-margin mechanism for accuracy calculation. Specifically, we take the accuracy of data falling within the correct range as 100, and calculate the accuracy with $100 * \exp(-k\epsilon)$ for data outside the correct range, where ϵ is the error between the data value and the boundary, and k is a hyper-parameter controlling the decay rate of accuracy at the margins, with larger k corresponding to faster decay. We take accuracy curves of high vocal-range of female, low speed, and medium volume as examples and illustrate them in Figure 8.

G.2. Subjective Evaluation

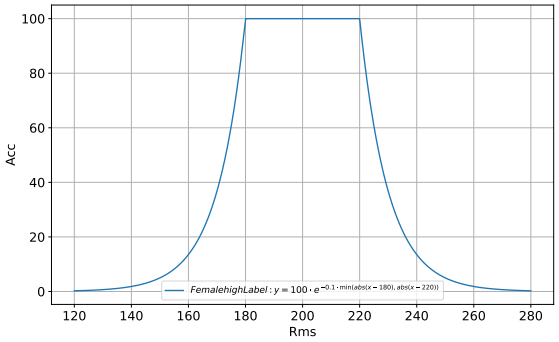
All our Mean Opinion Score (MOS) tests are crowd-sourced and conducted by native speakers. The scoring criteria have been included in Table 10 for completeness. The samples are presented and rated one at a time by the testers, each tester is asked to evaluate the subjective naturalness of a sentence on a 1-5 Likert scale. The screenshots of instructions for testers are shown in Figure 9. We paid $8 to participants hourly and totally spent about $400 on participant compensation.

Table 10: Ratings that have been used in the evaluation of speech naturalness of synthetic and ground truth samples.

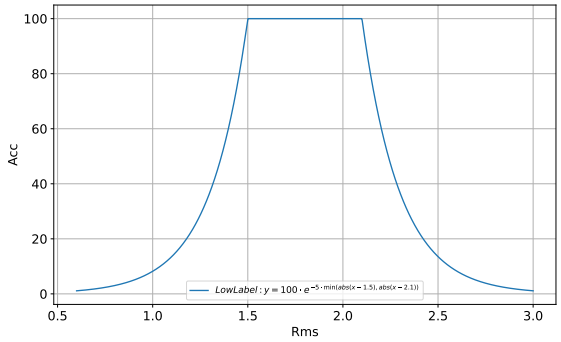
Rating	Naturalness	Definition
1	Bad	Very annoying and objectionable dist.
2	Poor	Annoying but not objectionable dist.
3	Fair	Perceptible and slightly annoying dist
4	Good	Just perceptible but not annoying dist.
5	Excellent	Imperceptible distortions

H. Limitation and Potential Risks

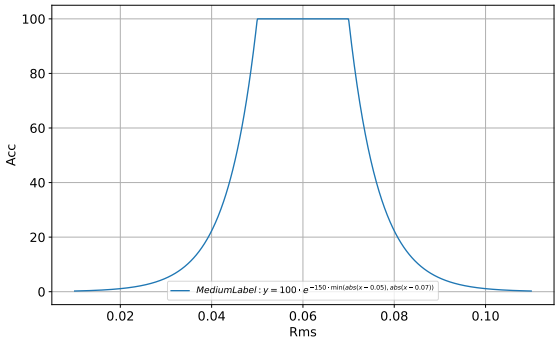
We control synthesized speech tempo by multiplying durations by a factor λ, and thus InstructSpeech supports global speed editing. To enable fine-grained control, the model may be prompted with phoneme-level durations, which can be left for



(a) Pitch.

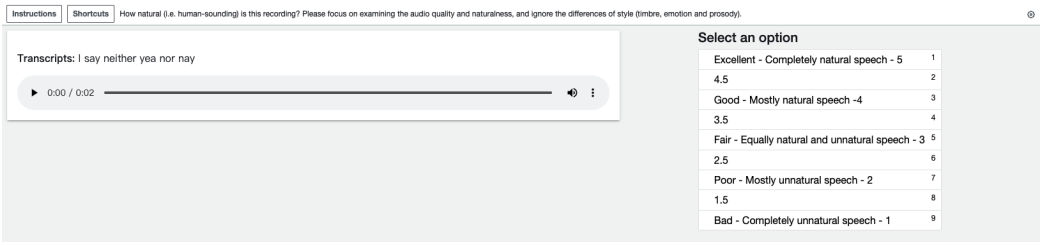


(b) Speed.

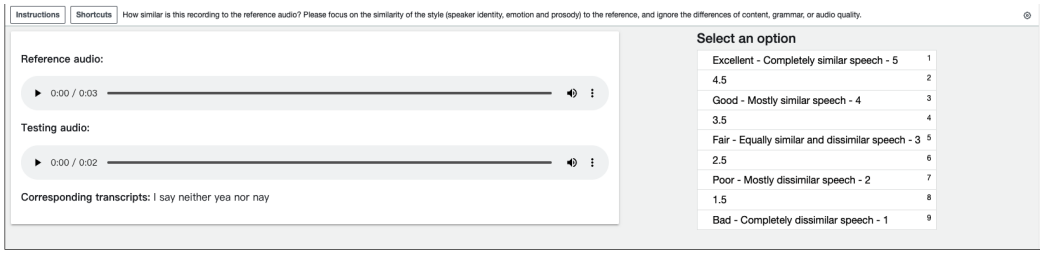


(c) Volume.

Figure 8: Soft-margin accuracy curve.



(a) Screenshot of MOS testing.



(b) Screenshot of SMOS testing.

Figure 9: Screenshots of subjective evaluations.

future work.

Although InstructSpeech as a voice LLM is successfully applied to zero-shot voice signals at scale, it still suffers from some limitations: 1) InstructSpeech introduces a strong dependency on the quality of the audio tokenizer. 2) We test in-context learning ability of our model on manipulation testing set, and there are still challenges in open-domain instruction editing, and 3) a longer sequence length typically requires more computational resources, and degradation could be witnessed with decreased training data.

I. Reproducibility Statement

We will release our code in the future. The InstructSpeech model that we build upon is publicly available through the fairseq code repository (Ott et al., 2019). To aid reproducibility, we have included a schematic overview of the algorithm in Algorithm 1, and hyperparameters in Table 8.