

UnitCorrect: Unit-Based Mispronunciation Correcting System With a DTW-Based Detection

Hyun-Woo Bae , Hyung-Seok Oh , Seung-Bin Kim , and Seong-Whan Lee , *Fellow, IEEE*

Abstract—With recent globalization trends, the importance of language education for nonnative speakers has significantly increased. Research on pronunciation correction and speech editing continues to be actively explored to reduce the pronunciation gap between native and nonnative speakers. Traditional speech editing models frequently suffer from unnaturalness at the boundary of the corrected parts and demand substantial input information to effectively synthesize corrected speech. Furthermore, the lack of well-curated annotations for mispronounced speech presents a challenge for effective training. This paper introduces UnitCorrect, a mispronunciation correction system that leverages self-supervised unit representations for synthesizing natural-sounding speech and utilizing more phoneme information effectively. Additionally, we propose a mispronunciation detection approach based on dynamic time warping, which efficiently identifies mispronounced segments by aligning input audio with the target text, helping to mitigate the data scarcity issue. In addition, frame-level text is incorporated into the decoder’s input to supplement the text information, and a conditional flow-matching decoder enables high-quality speech synthesis. The experimental results demonstrate that UnitCorrect outperforms existing models in terms of mispronunciation corrections and naturalness of speech.

Index Terms—Conditional flow matching, dynamic time warping, mispronunciation detection, pronunciation correction, speech editing.

I. INTRODUCTION

RECENT advances in deep learning have significantly improved the field of speech synthesis. Research efforts have been directed toward generating human-like, natural [1], [2], [3], [4], [5], and expressive speech [6], [7], [8], [9] in text-to-speech (TTS) applications.

With globalization, nonnative speakers have more opportunities to use foreign languages. However, the pronounced linguistic disparity between native and nonnative speakers is evident, as the pronunciation of a second language is influenced by the

phonetic characteristics of the first language and the educational environment experienced by the speaker. This disparity presents difficulties in effective communication. Recognizing and correcting these linguistic differences is particularly challenging in language education. Several studies [10], [11], [12] have been conducted on pronunciation to highlight its important role in effective communication and language acquisition.

Speech editing, in particular, has drawn significant attention because of its extensive potential in video/audio editing and educational settings. In speech editing tasks, the traditional cut-copy-paste method [13], [14], [15] simply cuts the sections of audio or spectrograms that are to be edited, copies the desired parts, and pastes them into the cut parts. However, this method exhibits the drawback of unnatural prosody at the transition points. Mask prediction approaches [16], [17] resolve the above problem by masking and predicting contextual information, thereby aiming to synthesize smoother and more natural transitions in the edited speech. However, issues still exist such as segments that are not sufficiently natural, a sense of awkwardness in newly generated segments, and the need to specify the locations that require editing still remaining manually. Furthermore, the lack of well-curated well-pronounced and mispronounced data pairs with the corresponding text poses significant challenges for training traditional models in speech editing and pronunciation correction tasks. This issue naturally leads us to explore frameworks that obviate the necessity of curated data pairs and use self-supervised learning representations [18], [19], [20], [21], [22] or neural codecs [23], [24], [25]. In our study, we define “well-pronounced” as standard American English, whereas “mispronounced” refers to phonological differences from standard American English found in non-native speech.

To address these issues, we propose UnitCorrect, a unit-based mispronunciation correction system with a dynamic time warping (DTW)-based detection. First, we design an alignment-guided system to detect mispronunciations for alleviating the need for human supervision (i.e., annotation of edited parts). Our detection method aims to identify mispronounced parts from input speech and target text without requiring any additional input. To mitigate the data scarcity issue caused by the lack of mispronounced speech datasets, we leverage self-supervised unit representations, which effectively capture a text information while reducing reliance on factors such as emotion, speaker identity, or gender characteristics [26]. The abundance of text content renders unit representations highly useful in scenarios with limited textual data availability, facilitating more natural and adequate modifications of text representation. We apply

Received 25 July 2024; revised 20 February 2025; accepted 2 April 2025. Date of publication 9 April 2025; date of current version 25 April 2025. This work was supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korea government (MSIT) under Grant RS-2019-II190079, Grant RS-2021-II212068, and Grant RS-2024-00336673, in part by Artificial Intelligence Graduate School Program (Korea University) under Grant RS-2021-II212068, in part by Artificial Intelligence Innovation Hub under Grant RS-2024-00336673, and in part by the AI Technology for Interactive Communication of Language Impaired Individuals. The associate editor coordinating the review of this article and approving it for publication was Dr. Suma Bhat. (*Corresponding author: Seong-Whan Lee.*)

The authors are with the Department of Artificial Intelligence, Korea University, Seoul 02841, Republic of Korea (e-mail: hw_bae@korea.ac.kr; hs_oh@korea.ac.kr; sb-kim@korea.ac.kr; sw.lee@korea.ac.kr).

Digital Object Identifier 10.1109/TASLPRO.2025.3559344

connectionist temporal classification (CTC) [27] loss at the unit representation to better capture relevant text information while maintaining robustness. For the parts to be preserved, we aim for a representation that effectively captures adequate text information, whereas for the parts that require corrections, we suppress the inadequate text information and retain the other necessary information. To improve the naturalness of the corrected speech, we adjust the duration to better align with the intended pronunciation. To correct the mispronounced speech and obtain well-pronounced speech, we use duration-corrected frame-level text conditioning, which incorporates target text information, and apply it as input to the decoder. Consequently, our model can synthesize well-pronounced speech from mispronounced speech using the target text. Experimental results show that our approach outperforms existing speech editing models in pronunciation correction and achieves superior performance in terms of naturalness and speaker similarity. The audio samples are available on our demo page¹.

II. RELATED WORK

A. Speech Editing

Speech editing models have been developed to improve the quality and naturalness of modified speech. Traditional approaches often rely on cut-copy-paste operations, which are only applied to waveforms or spectrograms. However, these methods typically encounter two problems. The first is the unnatural connection at the boundaries between the edited and original parts. Another limitation is the difficulty of synthesizing new words. Researchers have explored text-based editing methods to address these issues. For instance, EditSpeech [28] utilizes an autoregressive TTS model to synthesize better modified speech. CampNet [17] and A³T [16] employ mask prediction-based methods to predict contextual information, thereby enhancing the naturalness of the edited speech. EdiTTS [29] uses a score-based generative modeling method for pitch control. In addition to these models, FluentSpeech [30] uses a context-aware spectrogram denoiser to focus on stutter removal and production of more natural speech. Recent research on speech editing includes Voicebox [31], which uses large-scale data for region-based content editing, and InstructSpeech [32], which leverages large language model instructions as conditions for editing. However, these models require large-scale data during the pretraining process or substantial amounts of information as input.

B. Unit Representation

Discrete speech unit representations have emerged as a useful approach in speech synthesis [33], [34], [35], [36] and speech-to-speech translation [37], [38], [39], [40]. Unit representations are clustered from self-supervised learning speech representations to capture essential phonetic information. These unit representations concentrate more on phoneme information than the information related to speaker or gender [26], making them particularly effective for tasks where textual data are scarce.

By focusing primarily on the text content, unit representations provide a robust and flexible foundation for generating natural and high-quality speech. In this study, we utilized unit representations to address the data scarcity of annotated text and leverage phoneme information for pronunciation correction.

C. Conditional Flow Matching

Conditional flow matching (CFM) [41] is an approach in generative modeling that addresses the limitations of traditional methods such as continuous normalizing flows (CNFs) [42]. Unlike traditional CNFs that require simulation-based training, CFM introduces a simulation-free training paradigm. The core idea of the CFM is to regress vector fields that define conditional probability paths, enabling the efficient and scalable training of CNFs. This method constructs paths based on Gaussian distributions, enabling straightforward and unbiased sampling. CFM has the advantage of simplifying the calculation of the objective function by conditioning based on auxiliary variables. This conditioning mechanism allows the model to generate data that follow more accurately through a desired distribution. A key application of CFM is in the field of generative modeling, where it can be used to produce high-quality samples conditioned on specific inputs. This technique is robust and stable, making it a preferable alternative to the existing score-matching methods. Furthermore, CFM can be applied to TTS tasks [31], [43], [44]. CFM facilitates faster and more efficient training, resulting in high-quality audio outputs with fewer synthesis steps than those of traditional methods. The flexibility and efficiency of CFM make it a powerful tool for modern deep learning tasks, enhancing both the quality and control of generative models.

III. METHOD

In this section, we introduce UnitCorrect, a mispronunciation correction system that incorporates DTW-based detection. Our goal is to detect mispronounced segments based on the input speech and target text and then correct these segments to improve pronunciation. To address the scarcity of annotated data for mispronounced speech, which poses challenges for detecting mispronounced parts, we utilize unit representations and employ a detection method based on dynamic time warping (DTW) algorithm. Moreover, the mispronounced parts that are identified by the mispronunciation detection method are processed using a correction module. The text information of the intended pronunciation is then fed into the CFM decoder using frame-level text conditioning. Frame-level text conditioning help to synthesize speech that aligns more closely with natural pronunciation. The overall structure and process of UnitCorrect are shown in Fig. 1. Further details are presented in the following subsections.

A. Unit Encoder

In our approach, we aim to address the scarcity of annotated text data for mispronounced speech while effectively incorporating phonetic information for both detection and correction. To achieve this, we utilize frame-level unit representations, which are obtained by applying K-Means clustering to the

¹<https://prml-lab-speech-team.github.io/demo/UnitCorrect/>

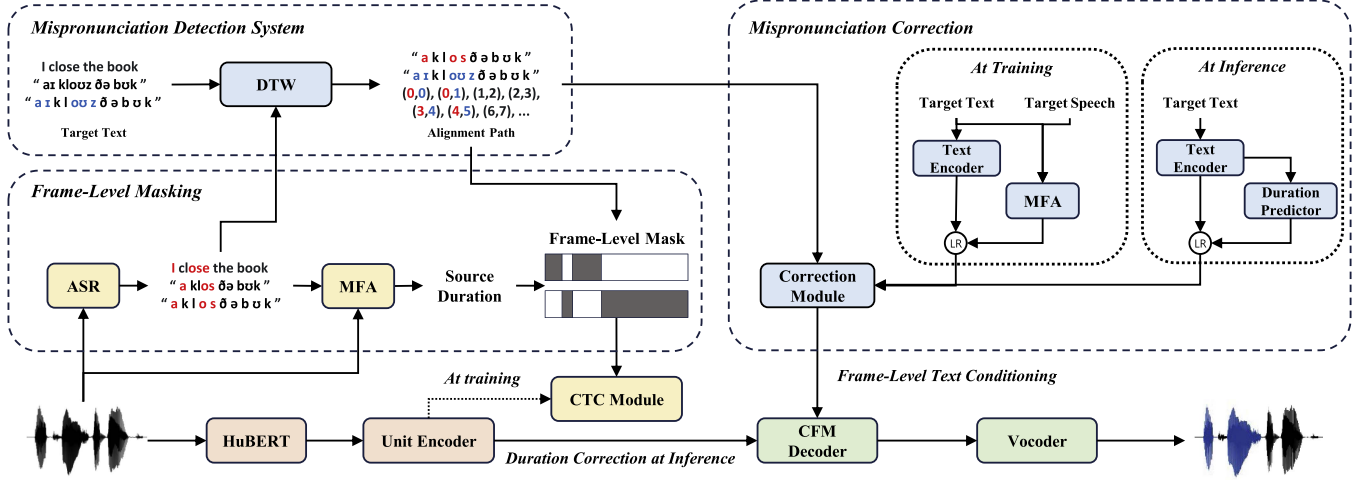


Fig. 1. Overall architecture of UnitCorrect. Based on the predicted text obtained from the input speech, the output of the unit encoder is corrected using a mispronunciation detection method and CTC module. This corrected output is then decoded with frame-level text conditioning to generate corrected speech.

features obtained using HuBERT [20]. These frame-level unit representations are derived to have the same length as the mel-spectrogram, ensuring alignment between the two representations. The unit encoder encodes speech content by leveraging these frame-level unit representations. We train the unit encoder using the prior loss \mathcal{L}_{enc} as described in [35]. Unlike in [35], where an alignment process is necessary, the output of the unit encoder in our study is already aligned at the frame level. A robust representation from the unit encoder serves as the prior of the flow-matching decoder.

B. CTC Module

During training, the CTC module effectively incorporates desired phonetic information into representations, enhancing the ability of the model to understand and generate coherent outputs that better align with the given text. To learn a robust representation and enhance the effectiveness of the target text conditioning, the output of the unit encoder is divided into well-pronounced and mispronounced parts using a frame-level mask. The frame-level mask is generated from the duration of the source speech, which is obtained using the Montreal forced aligner (MFA) [45] and the alignment path from the mispronunciation detection method. The source text obtained from the automatic speech recognition (ASR) model is separated into text that requires correction and text that should be preserved using text masking. For text that needs to be preserved, we use connectionist temporal classification (CTC) loss, \mathcal{L}_{ctc} , to train the well-pronounced parts of the CTC module, helping them better capture and represent the text information. For text requiring correction, we additionally apply a gradient reversal layer (GRL) along with the CTC loss, \mathcal{L}_{grl_ctc} , to the output of the encoder. As these parts contain mispronounced information, the gradient reversal layer suppresses the gradients during backpropagation to prevent the model from learning these mispronunciations, thereby effectively guiding the encoder to avoid learning the

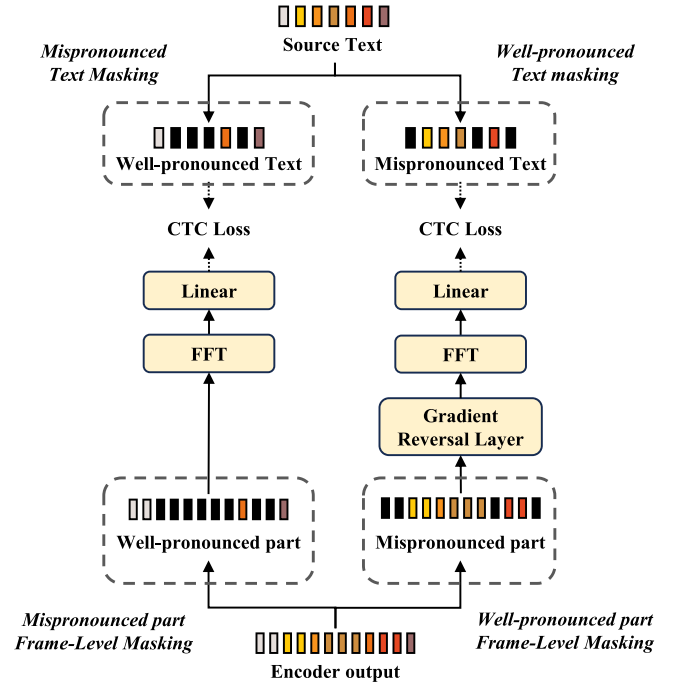


Fig. 2. Details of CTC module. For well-pronounced parts, text and frame-level masks are added to the source text and output of the encoder, respectively. For mispronounced parts, a gradient reversal layer is added to the mispronounced parts of the encoder output to mitigate mispronunciation. The representation is trained by calculating the Connectionist Temporal Classification loss from each well-pronounced and mispronounced parts.

errors. The structure of the CTC module and the details of the CTC loss are illustrated in Fig. 2.

C. Mispronunciation Detection Method

We utilize a dynamic time warping (DTW) algorithm-based approach to create the text and frame-level masks used in the CTC module. Dynamic time warping [46] is an algorithm used

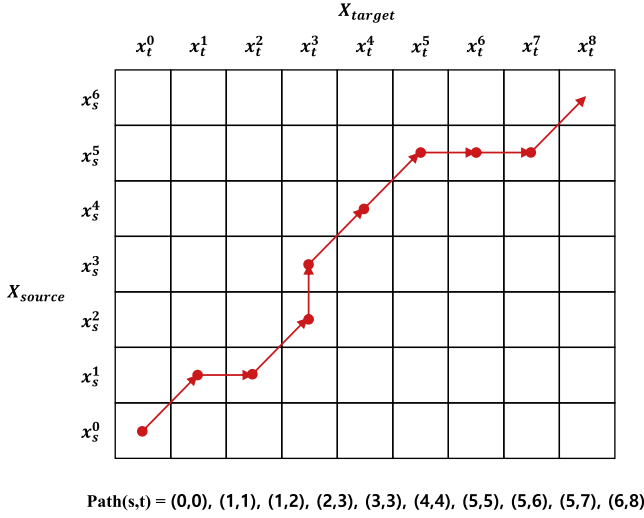


Fig. 3. Example of an alignment path calculated using the DTW algorithm. The DTW algorithm determines the optimal alignment path between two sequences by sequentially comparing each element from start to end.

to measure similarity between two temporal sequences that may vary in length. In this study, we employ the DTW algorithm to facilitate the alignment and comparison of sequences with different lengths more effectively. The method by which the DTW algorithm calculates the optimal path is illustrated in Fig. 3. Using the mispronunciation detection with DTW algorithm, we create text and frame-level masks. To generate a text mask, the target and source text sequences need to be compared. The source text sequence is obtained from the output of the ASR model. By comparing these two sequences using a detection method, we obtain an alignment path of the sequences, particularly, the segments that may require correction. Using this alignment information, we create a text mask that divide the text into parts that require correction and those that do not. Duration information required for the frame-level mask is obtained using the montreal forced aligner (MFA) [45], which provides alignment information between text and speech. Using the duration information, we convert the text mask into a frame-level mask. The details of the mispronunciation detection method are shown in Fig. 4.

D. Mispronunciation Correction

The desired target speech duration is derived from the source duration. For duration correction, we predict the target speech duration using the target text, source speech, and source text. First, the source duration is divided into well-pronounced and mispronounced parts using the alignment path obtained from the mispronunciation detection method. For the well-pronounced parts, the duration is aligned with the source speech length based on the duration obtained from the MFA. For the mispronounced parts, the duration from the target text is used to align the speech length with the corrected pronunciation. During training, the duration of the target text is obtained using MFA. During inference, a duration predictor is used to estimate durations. It takes phoneme-level representations as input and predicts how many

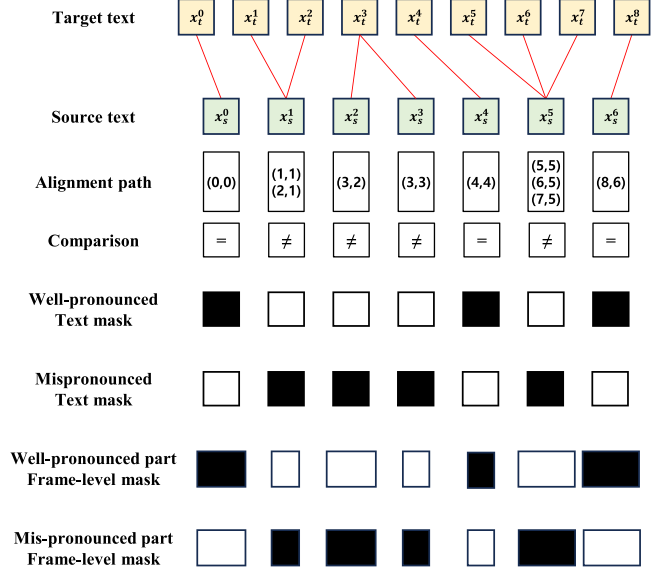


Fig. 4. Details of the mispronunciation detection method and mask creating. To create the text mask used in the CTC module, the alignment path obtained from the DTW algorithm is used to distinguish between well-pronounced and mispronounced parts. A frame-level mask can be generated using the duration information obtained from the Montreal Forced Aligner.

mel frames correspond to each phoneme, representing durations in the logarithmic domain for improved learning stability. Using this corrected duration, we construct a frame-level text of the target duration. The details for creating the frame-level text are as follows: Let, $\mathbf{X}_t = (x_t^0, x_t^1, \dots, x_t^N)$, $\mathbf{Y}_t = (y_t^0, y_t^1, \dots, y_t^M)$, and $\mathbf{D}_t = (d_t^0, d_t^1, \dots, d_t^N)$ where \mathbf{X}_t represents the target text, N represents the number of target phonemes, \mathbf{Y}_t represents the frame-level representation of the target speech, M denotes the number of frames and \mathbf{D}_t represents the per-phoneme duration of the target text obtained from MFA. The frame-level text is generated by repeating each phoneme in the target text according to its duration value, ensuring that the total duration of the phonemes matches the duration of the corresponding frame-level speech representation. In (2), len denotes the time length of component. The details of duration correction are shown in Fig. 5 and procedure for duration correction is presented in Algorithm 1.

$$\text{Frame-level text} = \left\{ \overbrace{x_t^0, \dots, x_t^0}^{d_t^0}, \dots, \overbrace{x_t^N, \dots, x_t^N}^{d_t^N} \right\}, \quad (1)$$

$$\sum_{i=1}^N \text{len}(x_t^i) * d_t^i = \sum_{j=1}^M \text{len}(y_t^j). \quad (2)$$

E. Conditional Flow Matching Decoder

To improve pronunciation and supplement insufficient text information, we concatenate the output of the encoder with the frame-level text obtained through the correction module and use this as the input to the decoder. In this process, because the duration of the frame-level target text may differ from that of the encoder output, both lengths need to be matched before

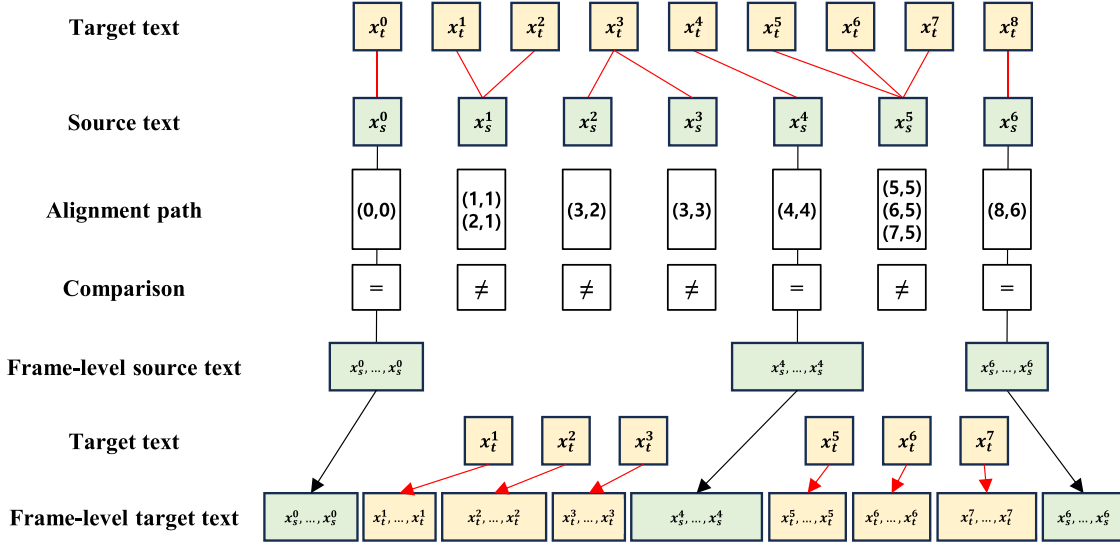


Fig. 5. Details of correction module. The alignment path obtained from the mispronunciation detection method is used. For the well-pronounced parts, the duration is maintained as provided by the MFA. For the mispronounced parts, the duration obtained from the MFA during training and from the duration predictor during inference is used to obtain the corrected target speech duration.

Algorithm 1: Duration Correction.

Input:

Target text \mathbf{X}_t , source text \mathbf{X}_s , frame-level representation of source speech \mathbf{Y}_s , source duration \mathbf{D}_s

Output:

Target speech \mathbf{Y}_t \hrulefill

Step 1:

Align target and source text for searching alignment path with detection method.

Step 2:

Detect mispronounced parts of source and target text from comparison.

Step 3:

$$\mathbf{D}_s = \{\mathbf{D}_s^{well}, \mathbf{D}_s^{mis}\}$$

Divide source duration from MFA into well-pronounced and mispronounced parts.

Step 4:

$$\mathbf{D}_t^{pred} = \text{DP}(\mathbf{X}_t)$$

Predict duration of mispronounced parts from target text and duration predictor DP .

Step 5:

$$\mathbf{D}_t = \{\mathbf{D}_s^{well}, \mathbf{D}_t^{pred}\}$$

Align duration from remained duration of well-pronounced parts and predicted duration of mispronounced parts.

concatenation. For concatenation, we correct the mispronounced segments of the encoder output via interpolation according to the corrected duration obtained from the correction module, as explained in Section III-D. We utilize the entire representation as the input to the decoder, unlike the CTC module, where frame-level masking was applied. Frame-level masking is not applied to the decoder because the representation has been trained to mitigate potential mispronunciations using the gradient reversal

layer and CTC loss at the CTC module, and we incorporate the target text information through text conditioning. Moreover, this approach transforms text information for pronunciation correction while preserving other speech characteristics. To achieve high-quality speech synthesis, we adopt a generative modeling approach based on flow matching [41]. Specifically, we model the continuous transformation of an input sample x over time using an ordinary differential equation (ODE) parameterized by a time-dependent vector field $\mathbf{v}_t(x; \theta)$. Here, θ denotes learnable parameters of vector field. This transformation is described as:

$$\frac{d}{dt}\phi_t(x) = \mathbf{v}_t(\phi_t(x); \theta), \quad \phi_0(x) = x. \quad (3)$$

Here, $\phi_t(x)$ defines a flow function that transports the initial sample x along a trajectory dictated by $\mathbf{v}_t(x; \theta)$. This process induces a probability path $p_t(x|x_1)$, which gradually evolves over time, transitioning from a simple prior distribution toward a more structured representation. To align the probability flow with a structured trajectory, we employ conditional flow matching (CFM) [41], which trains the learnable vector field $\mathbf{v}_t(x; \theta)$ to follow a reference conditional vector field $\mathbf{u}_t(x|x_1)$. The CFM loss function is formulated as:

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t, q(x_1), p_t(x|x_1)} \|\mathbf{u}_t(x|x_1) - \mathbf{v}_t(x; \theta)\|^2, \quad (4)$$

To improve the transport efficiency, we adopt optimal transport conditional flow matching (OT-CFM) [41], [43], which optimizes the probability flow using an interpolation method inspired by optimal transport. The OT-CFM loss is defined as:

$$\mathcal{L}_{OT_CFM}(\theta) = \mathbb{E}_{t, q(x_1), p_0(x_0)} \|\mathbf{u}_t^{\text{OT}}(\phi_t^{\text{OT}}(x)|x_1) - \mathbf{v}_t(\phi_t^{\text{OT}}(x)|x_{enc}, x_{txt}; \theta)\|^2, \quad (5)$$

where $\mathbf{u}_t^{\text{OT}}(x|x_1)$ represents the optimal transport vector field, and $\phi_t^{\text{OT}}(x)$ defines the corresponding transport mapping. By structuring the probability path in this manner, OT-CFM can help

regularize the probability flow, potentially improving stability in the generative process. To enhance the robustness of the model to unseen data, we apply random masking strategies to perturb the inputs during training. The encoder outputs x_{enc} and frame-level text representations x_{txt} are concatenated and provided as inputs to the CFM decoder. The decoder is then trained with the CFM loss, leveraging random masking strategies, which can enhance robustness. The final loss function integrates multiple components, including the encoder loss \mathcal{L}_{enc} , CTC loss \mathcal{L}_{ctc} , gradient reversal CTC loss \mathcal{L}_{grl_ctc} , and the optimal transport-based flow matching loss:

$$\mathcal{L}_{total} = \mathcal{L}_{enc} + \mathcal{L}_{ctc} + \mathcal{L}_{grl_ctc} + \mathcal{L}_{OT_CFM}. \quad (6)$$

At inference, before generating the waveform with the vocoder, we replace the masked parts of the original representation with newly generated parts.

IV. EXPERIMENTS

A. Datasets

We utilize three datasets: VCTK [47], LibriTTS-R [48], and L2-ARCTIC [49]. The VCTK dataset consists of 44 k recording samples of 109 speakers with different accents spanning a period of 44 hours. From the LibriTTS-R dataset, which is an enhanced version of LibriTTS [50], we utilize the `test-clean-360` and `train-clean-100` sets, comprising 149k samples spoken by 1,151 speakers spanning over a period of 262 hours. The L2-ARCTIC dataset is a nonnative English multi-speaker dataset with 24 speakers, 26 k samples, and 27 hours of recordings. For evaluation, test samples were selected from the L2-ARCTIC. The VCTK and LibriTTS-R datasets are used to train the model. The L2-Arctic dataset was used exclusively for evaluation.

B. Experimental Setup

All data are downsampled to 16 kHz for consistency in comparison. We transform the audio into 80 bins of mel-spectrograms using short-time Fourier transform with a hop size of 320, fft size of 1280 and window size of 1280. The BASE HuBERT model comprising 12 transformer layers is used to generate unit representations by extracting the output from the sixth layer and applying K-Means clustering. Then, the length of the unit representations are matched with the length of the mel-spectrogram. If a resolution mismatch problem occurs, interpolation is used to adjust the resolution of the unit representation to match that of the mel-spectrogram, ensuring that both have the same frame length. Owing to the lack of paired datasets of well-pronounced and mispronounced speeches, we utilize the speech from existing datasets and the scripts generated by the ASR model from this speech as text-speech pairs. We employed Whisper [21] to extract scripts. The original texts from these datasets are used as the target texts. The source text from the ASR scripts and target text from the original dataset are converted to phonemes using an open-source module grapheme-to-phoneme module². Moreover, the test samples

were selected from L2-ARCTIC based on specific criteria³ to ensure that the mispronounced data was explicitly defined, allowing for a noticeable performance difference in pronunciation correction evaluation. Specifically, we choose test samples from the original dataset that had both a word error rate (WER) and phoneme error rate (PER) of over 30%, and where the length of the mispronounced segments exceeded 10% of the total speech length. We conducted evaluations with two types of samples. One type consisted of whole synthesized samples comprised of original and corrected parts. The other type consisted of segments of the corrected parts. Based on this selection of test samples, meaningful results could be obtained for evaluating the performance of the correction model.

C. Implementation Details

The overall architecture of UnitCorrect is shown in Fig. 1. We extract unit representations using a pre-trained HuBERT model during the preprocessing stage. The unit encoder is structured with six layers of feed-forward transformer blocks using a hidden size of 256 and filter size of 1280. The extracted unit representations are used as inputs and equipped with trainable embeddings of 320 dimensions. For the ctc module, the CTC encoder employs a hidden size of 80, which matches the number of mel-spectrogram bins. The gradient reversal layer is implemented based on a previous study [51]. The duration predictor and length regulator are based on the FastSpeech 2 [2]. For the speaker condition, we use speaker embeddings extracted from Resemblyzer⁴, an open-source model trained with generalized end-to-end (GE2E) loss [52], setting the dimensions to 64. The flow matching decoder is based on the implementation in [43]. For the vocoder, we train a BigVGAN [53] at 16 kHz by utilizing the implementation, which is known for its robustness to unseen speech. We use the same speech synthesis model, speaker embedding extractor, and vocoder for all baselines.

D. Evaluation Metrics

1) *Mean Opinion Score*: We utilized the mean opinion score (MOS) test as a subjective evaluation metric to assess the quality of the generated speech. The evaluation was conducted using the Amazon Mechanical Turk (mTurk)⁵ crowd-sourcing platform, where 20 listeners were recruited to rate a randomly selected subset of 50 test samples from our dataset. Each sample was evaluated on a five-point scale (ranging from 1 to 5) based on specific perceptual criteria. To ensure statistical reliability, the evaluation was performed with a 95% confidence level.

2) *UTokyo-SaruLab Mean Opinion Score Prediction System*: We use the open-source UTokyo-SaruLab mean opinion score prediction system (UTMOS)⁶ [54] as a metric to measure the naturalness of speech. UTMOS is a powerful automatic speech quality assessment tool that evaluates speech naturalness.

³We referred to the test samples that meet this specific criteria as the “Input” in the tables.

⁴<https://github.com/resemble-ai/Resemblyzer>

⁵<https://www.mturk.com/>

⁶<https://github.com/sarulab-speech/UTMOS22>

²<https://github.com/Kyubyong/g2p>

TABLE I
OBJECTIVE EVALUATION FOR COMPARISON OF THE ASR MODELS

Model	WER (↓)	PER (↓)	UTMOS (↑)	SECS (↑)
Input	24.04	27.17	3.58	-
w HuBERT-large	25.67	27.36	3.19	0.965
w Wav2Vec2.0-large	23.26	26.78	3.20	0.968
w Whisper-large	21.04	24.73	3.38	0.977

Although it is not an absolute measure, it is as a sufficient metric for comparative purposes.

3) *Word and Phoneme Error Rate*: We use the WER as a metric to measure the accuracy and intelligibility of speech and PER for assessing pronunciation accuracy and intelligibility at the phoneme level. To transcribe the generated audio, we use the open-source pre-trained wav2vec2.0 model for determining the WER⁷ [19] and PER⁸ [55] of the ASR system.

4) *Speaker Embedding Cosine Similarity*: To compare speaker similarity between the corrected speech and input speech, we utilize speaker embedding cosine similarity (SECS). For this purpose, we employ a pre-trained WavLM-TDNN⁹ [22] speaker verification model.

V. RESULTS

A. Automatic Speech Recognition Model Selection

As mentioned previously, owing to the limitations of the dataset, we used speech from existing datasets and the text generated by applying an ASR model for training. Therefore, selecting an appropriate ASR model is crucial because it can significantly influence the experimental results. We compared HuBERT-Large [20], Wav2Vec2.0-Large [19], and Whisper-Large [21] models as ASR candidates. Each model was used to generate scripts, and the models were each trained using these scripts for 800 k steps. For the test samples to be evaluated, we chose the same samples for each ASR model for a fair comparison. For a fair comparison, we selected test samples from the test set that satisfied the existing criteria (WER of over 30%, PER of over 30% and a mispronounced portion of over 10%) by considering $K = 200$ clusters across all three ASR models. As shown in Table I, Whisper was chosen as the ASR model due to its better performance.

B. Number of Clusters in K-Means Clustering

To investigate the effect of an appropriate number of clusters, K , on the experimental results, we conducted evaluations for $K = 50, 100$, and 200 . For a fair comparison, we also selected test samples from the test sample set that met the existing criteria (WER of over 30%, PER of over 30%, a mispronounced portion of over 10%) with the Whisper-Large ASR model. Moreover, we evaluated the performance of complete sentences of the test samples and the corrected segments. We analyzed the performance in terms of pronunciation accuracy and naturalness to

provide a comprehensive assessment of the effect of variations in K on the outcomes. The number of clusters appeared to have little impact on the speaker verification results; however, some differences were observed in the accuracy and intelligibility test performances. Similar results were observed for complete sentences and segments. An increase in the value of K tended to yield more accurate pronunciation scores. We also evaluated the naturalness and speaker similarity of the segments. For pronunciation accuracy, we determined that evaluating the complete sentence provides a more accurate assessment, owing to variations in audio length and the proportion of mispronounced parts. Therefore, we did not evaluate the segmented parts. As shown in the results listed in Table II, the most suitable performance of the proposed pronunciation correction model is achieved when $K = 200$.

C. Correction Performance

Based on the above results, we evaluated UnitCorrect using $K = 200$ and the Whisper model as the ASR. To visualize and explain the pronunciation correction process and its before-and-after comparison, we utilized mel-spectrograms, as shown in Fig. 6. First, the mispronounced parts are detected using the detection system. Subsequently, a masking strategy is applied to the mispronounced parts of the original speech. The correction module predicts and adjusts the duration of the mispronounced segments to target speech duration. Finally, the corrected speech is synthesized from the corrected duration of the representation. We compared our baseline model with other speech editing models such as A³T [16], CampNet [17], and FluentSpeech [30] in terms of pronunciation accuracy, naturalness, and speaker similarity of the corrected speech. We implemented and trained each baseline model using the configurations and hyperparameters presented in the studies related to these models. We evaluated the models with the best performance for our test samples. However, in this experiment, all test samples were from an unseen speaker dataset for both the proposed model and baseline models. As mentioned above, our test samples focused more on mispronunciations than on the clean samples evaluated by the baseline models, resulting in lower evaluation scores than those reported in the original papers. As shown in Table III, UnitCorrect exhibits superior performance in all metrics for both complete sentences and segments of sentences. The evaluation results demonstrate that UnitCorrect is robust in correcting mispronounced speech.

D. Ablation Study

In this section, we describe the ablation studies that were conducted to demonstrate the performance of each component of UnitCorrect as summarized in Table IV. We trained our model without considering CTC loss. In the “w/oGRL_CTC” scenario, we removed mispronounced part of the CTC module and only the CTC loss from these components was eliminated during training. In the “w/oGRL_CTC&CTC” scenario, we removed both types of CTC losses entirely from the training process. In both scenarios, the performance in terms of pronunciation accuracy and speech naturalness was

⁷<https://huggingface.co/facebook/wav2vec2-large-960h-lv60-self>

⁸<https://huggingface.co/facebook/wav2vec2-lv-60-espeak-cv-ft>

⁹<https://huggingface.co/microsoft/wavlm-base-sv>

TABLE II
OBJECTIVE EVALUATION FOR COMPARISON OF THE NUMBER OF UNIT CLUSTERS

Model	Whole				Segment	
	WER (\downarrow)	PER (\downarrow)	UTMOS (\uparrow)	SECS (\uparrow)	UTMOS (\uparrow)	SECS (\uparrow)
Input	24.04	27.17	3.58	-	2.84	-
Ours ($K=50$)	24.18	27.24	3.33	0.977	2.63	0.862
Ours ($K=100$)	23.04	26.70	3.36	0.977	2.69	0.864
Ours ($K=200$)	21.04	24.73	3.38	0.977	2.81	0.864

Target: The warden with a quart of champagne.
ASR Result: The warden with a swatch of champagne.

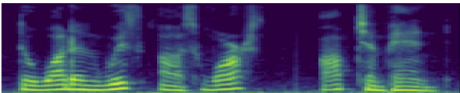
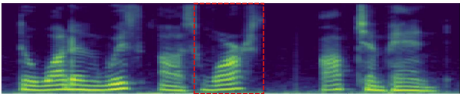
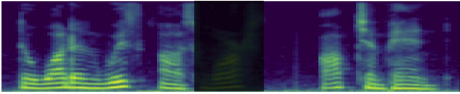
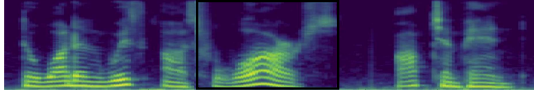
Task	Character & Phoneme Sequence	Mel Spectrogram
Step 1: ASR Prediction	The warden with a swatch of champagne.	
	DH AH0 W A01 R D AH0 N W IH1 DH AH0 S W A A I C H AH1 V SH AE0 M P EY1 N	
Step 2: Mispronunciation Detection	The warden with a swatch of champagne.	
	DH AH0 W A01 R D AH0 N W IH1 DH AH0 S W A A I C H AH1 V SH AE0 M P EY1 N	
Step 3: Masking Mispronounced Part	The warden with a [Mask] of champagne.	
	DH AH0 W A01 R D AH0 N W IH1 DH AH0 [Mask] AH1 V SH AE0 M P EY1 N	
Step 4: Mispronunciation Correction	The warden with a quart of champagne.	
	DH AH0 W A01 R D AH0 N W IH1 DH AH0 K W A01 R T AH1 V SH AE0 M P EY1 N	

Fig. 6. Visualization of the pronunciation correction procedure. We aim to correct mispronounced speech to the speech of the target text. In step 1, the source text is predicted from the result of the automatic speech recognition model. In step 2, the mispronounced parts are detected using a detection system. In step 3, the mispronounced parts are masked, and the duration is corrected in step 4. Finally, the corrected speech is synthesized with a corrected duration.

TABLE III
OBJECTIVE COMPARISON WITH BASELINE MODELS

Model	Whole					Segment	
	WER (\downarrow)	PER (\downarrow)	MOS (\uparrow)	UTMOS (\uparrow)	SECS (\uparrow)	UTMOS (\uparrow)	SECS (\uparrow)
Input	24.04	27.17	4.10	3.58	-	2.84	-
A ³ T [16]	48.04	42.08	2.28	2.26	0.951	1.62	0.817
CampNet [17]	51.95	46.70	2.13	2.20	0.951	1.40	0.813
FluentSpeech [30]	44.44	42.80	3.30	2.72	0.941	2.34	0.827
UnitCorrect	21.04	24.73	4.03	3.38	0.977	2.81	0.864

TABLE IV
OBJECTIVE EVALUATION OF ABLATION STUDIES

Model	Whole					Segment	
	WER (\downarrow)	PER (\downarrow)	MOS (\uparrow)	UTMOS (\uparrow)	SECS (\uparrow)	UTMOS (\uparrow)	SECS (\uparrow)
Input	24.04	27.17	4.10	3.58	-	2.84	-
UnitCorrect	21.04	24.73	4.03	3.38	0.977	2.81	0.864
w/o GRL_CTC	22.26	26.22	3.92	3.34	0.979	2.66	0.865
w/o GRL_CTC & CTC	23.10	26.96	3.90	3.34	0.981	2.70	0.858
w/o Random masking	23.93	26.39	3.96	3.33	0.979	2.69	0.860
w Frame-level masking	23.43	26.11	3.93	3.36	0.968	2.70	0.842

reduced. This indicates that CTC loss is essential for enabling the model to synthesize accurate and natural-sounding speech. In the “w/o Random masking” scenario, we trained our model without applying the random masking method to the decoder input. Consequently, the performance metrics across all metrics were lower. This suggests that the random masking method contributes significantly to building a robust model, as anticipated. In the “w Frame-level masking” scenario, we applied the frame-level masking method to the encoder output to remove mispronounced parts. Only the frame-level target text from the correction module was fed into the decoder as input for mispronounced segments. However, this approach carelessly removed non-linguistic information as well. Particularly, we observed significantly lower results in terms of speaker similarity as compared to the other metrics. Consequently, we confirmed that not applying the frame-level masking method to the decoder input improves the overall performance.

VI. DISCUSSION

A. Limitation

Our model has the advantage of generating the desired corrected speech using only the input speech and target text. However, because we used an ASR model for mispronunciation detection, the performance is also influenced by the mispronunciation detection performance of the ASR model. Selecting an ASR model that performs well in the detection of a specific language can enhance the overall performance. Moreover, although our model show improved performance in terms of WER and PER as compared to the ground truth, the results were not yet satisfactory, indicating room for improvement in pronunciation accuracy.

B. Future Work

Future research will focus on refining both mispronunciation detection and correction to further enhance pronunciation accuracy. This includes the development of more advanced ASR models tailored to specific languages to enhance the detection accuracy. We also plan to explore multi-lingual datasets to train UnitCorrect for broader applications, thereby enabling effective pronunciation correction across various languages. Furthermore, we will work on improving pronunciation accuracy by fine-tuning correction systems to achieve performance levels that surpass current benchmarks and user satisfaction levels. By focusing on these areas, we hope to make significant strides in the field of speech correction and editing.

VII. CONCLUSION

In this paper, we propose UnitCorrect, a unit-based mispronunciation correction model with a detection method. By leveraging self-supervised unit representations, our approach synthesizes natural-sounding speech even in data-scarce scenarios, particularly at transition boundaries. Moreover, utilizing an alignment-based mispronunciation detection method, we effectively detect the mispronounced parts of speech, effectively overcoming the issue of limited annotated text for identifying mispronounced speech. Furthermore, by appropriately

correcting the duration based on the detected parts and alignment information, well-pronounced speech is successfully synthesized. The experimental results demonstrate that our model outperform existing comparative models in terms of naturalness, pronunciation accuracy, and speaker similarity. We believe that by extending our model with training in multiple languages, multilingual pronunciation corrections can be performed.

REFERENCES

- [1] Y. Ren et al., “Fastspeech: Fast, robust and controllable text to speech,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 3171–3180.
- [2] Y. Ren et al., “FastSpeech 2: Fast and high-quality end-to-end text to speech,” in *Proc. 9th Int. Conf. Learn. Representations*, 2021.
- [3] J. Kim, J. Kong, and J. Son, “Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech,” in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 5530–5540.
- [4] K. Shen et al., “Natalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers,” in *Proc. 12th Int. Conf. Learn. Representations*, 2024.
- [5] S.-H. Lee, H.-Y. Choi, S.-B. Kim, and S.-W. Lee, “HierSpeech : Bridging the gap between semantic and acoustic representation of speech by hierarchical variational inference for zero-shot speech synthesis,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, vol. 35, pp. 16624–16636.
- [6] R. Skerry-Ryan et al., “Towards end-to-end prosody transfer for expressive speech synthesis with tacotron,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4693–4702.
- [7] D. Min, D. Lee, E. Yang, and S. Hwang, “Meta-StyleSpeech: Multi-speaker adaptive text-to-speech generation,” in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 7748–7759.
- [8] Y. Wang et al., “Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5180–5189.
- [9] H.-S. Oh, S.-H. Lee, and S.-W. Lee, “DiffProsody: Diffusion-based latent prosody generation for expressive speech synthesis with prosody conditional adversarial training,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 2654–2666, 2024.
- [10] I. McGraw, I. Badr, and J. R. Glass, “Learning lexicons from speech using a pronunciation mixture model,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 2, pp. 357–366, Feb. 2013.
- [11] X. Wang, Y. Liu, J. Li, V. Miljanic, S. Zhao, and H. Khalil, “Towards contextual spelling correction for customization of end-to-end speech recognition systems,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 3089–3097, 2022.
- [12] J.-F. Yeh and C.-H. Wu, “Edit disfluency detection and correction using a cleanup language model and an alignment model,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 5, pp. 1574–1583, Sep. 2006.
- [13] Z. Jin, G. J. Mysore, S. Diverdi, J. Lu, and A. Finkelstein, “VoCo: Text-based insertion and replacement in audio narration,” *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017, Art. no. 96.
- [14] Z. Jin et al., *Speech Synthesis for Text-Based Editing of Audio Narration*. Princeton, NJ, USA: Princeton Univ., 2018.
- [15] M. Morrison, L. Rencker, Z. Jin, N. J. Bryan, J. -P. Caceres, and B. Pardo, “Context-aware prosody correction for text-based speech editing,” in *Proc. 2021 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 7038–7042.
- [16] H. Bai, R. Zheng, J. Chen, M. Ma, X. Li, and L. Huang, “A³T: Alignment-aware acoustic and text pretraining for speech synthesis and editing,” in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 1399–1411.
- [17] T. Wang, J. Yi, R. Fu, J. Tao, and Z. Wen, “CampNet: Context-aware mask prediction for end-to-end text-based speech editing,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 2241–2254, 2022.
- [18] S. Schneider, A. Baevski, R. Collobert, and M. Auli, “wav2vec: Unsupervised pre-training for speech recognition,” in *Proc. Interspeech*, 2019, pp. 3465–3469.
- [19] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 12449–12460.
- [20] W. -N. Hsu et al., “Hubert: Self-supervised speech representation learning by masked prediction of hidden units,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3451–3460, 2021.

- [21] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 28492–28518.
- [22] S. Chen et al., "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 6, pp. 1505–1518, Oct. 2022.
- [23] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "SoundStream: An end-to-end neural audio codec," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 495–507, 2022.
- [24] X. Wang et al., "SpeechX: Neural codec language model as a versatile speech transformer," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 3355–3364, 2024.
- [25] C. Wang et al., "Neural codec language models are zero-shot text to speech synthesizers," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 33, pp. 705–718, 2023.
- [26] A. Sicherman and Y. Adi, "Analysing discrete self supervised speech representation for spoken language modeling," in *Proc. 2023 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.
- [27] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 369–376.
- [28] D. Tan, L. Deng, Y. T. Yeung, X. Jiang, X. Chen, and T. Lee, "EditSpeech: A text based speech editing system using partial inference and bidirectional fusion," in *Proc. 2021 IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 626–633.
- [29] J. Tae, H. Kim, and T. Kim, "EdiTTS: Score-based editing for controllable text-to-speech," in *Proc. Interspeech*, 2022, pp. 421–425.
- [30] Z. Jiang et al., "FluentSpeech: Stutter-oriented automatic speech editing with context-aware diffusion models," in *Proc. Findings Assoc. Comput. Linguistics*, 2023, pp. 11655–11671.
- [31] M. Le et al., "Voicebox: Text-guided multilingual universal speech generation at scale," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, vol. 36, pp. 14005–14034.
- [32] R. Huang et al., "InstructSpeech: Following speech editing instructions via large language models," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 19886–19903.
- [33] A. Polyak et al., "Speech resynthesis from discrete disentangled self-supervised representations," in *Proc. Interspeech*, 2021, pp. 3615–3619.
- [34] E. Kharitonov et al., "textless-lib: A library for textless spoken language processing," in *Proc. 2022 Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., System Demonstrations*, 2022, pp. 1–9.
- [35] H. Kim, S. Kim, J. Yeom, and S. Yoon, "UnitSpeech: Speaker-adaptive speech synthesis with untranscribed data," in *Proc. Interspeech*, 2023, pp. 3038–3042.
- [36] H.-S. Oh, S.-H. Lee, D.-H. Cho, and S.-W. Lee, "DurFlex-EVC: Duration-flexible emotional voice conversion with parallel generation," 2024, in *IEEE Trans. Affect. Comput.*, 2025, pp. 1–15.
- [37] A. Lee et al., "Direct speech-to-speech translation with discrete units," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, 2022, pp. 3327–3339.
- [38] A. Lee et al., "Textless speech-to-speech translation on real data," in *Proc. NAACL-HLT*, 2022, pp. 860–872.
- [39] H. Inaguma et al., "UnitY: Two-pass direct speech-to-speech translation with discrete units," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 15655–15680.
- [40] S.-B. Kim, S.-H. Lee, and S.-W. Lee, "TranSentence: Speech-to-speech translation via language-agnostic sentence-level speech encoding without language-parallel data," in *Proc. 2024 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2024, pp. 12722–12726.
- [41] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.
- [42] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, vol. 31, pp. 6572–6583.
- [43] S. Mehta, R. Tu, J. Beskow, É. Székely, and G. E. Henter, "Matcha-TTS: A fast TTS architecture with conditional flow matching," in *Proc. 2024 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2024, pp. 11341–11345.
- [44] N. Kanda et al., "Making flow-matching-based zero-shot text-to-speech laugh as you like," 2024, *arXiv:2402.07383*.
- [45] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kald," in *Proc. Interspeech*, 2017, pp. 498–502.
- [46] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 26, no. 1, pp. 43–49, Jan. 1978.
- [47] J. Yamagishi, C. Veaux, and K. MacDonald, "CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92)," Centre Speech Technol. Res., Univ. Edinburgh, 2019.
- [48] Y. Koizumi et al., "LibriTTS-R: A restored multi-speaker text-to-speech corpus," in *Proc. Interspeech*, 2023, pp. 5496–5500.
- [49] G. Zhao et al., "L2-ARTIC: A non-native english speech corpus," in *Proc. Interspeech*, 2018, pp. 2783–2787.
- [50] H. Zen et al., "Libritts: A corpus derived from librispeech for text-to-speech," in *Proc. Interspeech*, 2019, pp. 1526–1530.
- [51] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by back-propagation," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.
- [52] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4879–4883.
- [53] S.-G. Lee, W. Ping, B. Ginsburg, B. Catanzaro, and S. Yoon, "BigVGAN: A universal neural vocoder with large-scale training," in *Proc. 12th Int. Conf. Learn. Representations*, 2023.
- [54] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "UTMOS: UTokyo-SaruLab system for VoiceMOS challenge 2022," in *Proc. Interspeech*, 2022, pp. 4521–4525.
- [55] Q. Xu, A. Baevski, and M. Auli, "Simple and effective zero-shot cross-lingual phoneme recognition," in *Proc. Interspeech*, 2022, pp. 2113–2117.



Hyun-Woo Bae received the B.S. degree in electrical and electronic engineering from SungKyunKwan University, Seoul, South Korea, in 2019. He is currently working toward the integrated master's and Ph.D. degree with the Department of Artificial Intelligence, Korea University, Seoul, South Korea. His research interests include artificial intelligence and audio signal processing.



Hyung-Seok Oh received the B.S. degree in computer science and engineering from Konkuk University, Seoul, South Korea, in 2021. He is currently working toward the integrated master's and Ph.D. degree with the Department of Artificial Intelligence, Korea University, Seoul, South Korea. He was an Intern with Voice&Avatar, Naver Cloud, and Seongnam, Korea, in 2023. His research interests include artificial intelligence and audio signal processing.



Seung-Bin Kim received the B.S. degree in physics from the University of Seoul, Seoul, South Korea, in 2021. He is currently working toward the integrated master's and Ph.D. degree with the Department of Artificial Intelligence, Korea University, Seoul, South Korea. His research interests include artificial intelligence and audio signal processing.



Seong-Whan Lee (Fellow, IEEE) received the B.S. degree in computer science and statistics from Seoul National University, Seoul, South Korea, in 1984, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology, Seoul, South Korea, in 1986 and 1989, respectively. He is currently the Head with the Department of Artificial Intelligence, Korea University, Seoul. His research interests include artificial intelligence, pattern recognition, and brain engineering. He is a Fellow of the International Association of Pattern Recognition, Korea Academy of Science and Technology, and National Academy of Engineering of Korea.