CoreaSpeech: Korean Speech Corpus via Jamo-based Coreset Selection for Efficient and Robust Korean Speech Generation

Ki-Joong Kwon[1], Jun-Ho So[2], Sang-Hoon Lee[1]*
[1]Department of Artificial Intelligence, Ajou University, South Korea
[2]Department of Mathematics, Ajou University, South Korea
{kijoongkwon, jhso, sanghoonlee}@ajou.ac.kr

Abstract

While substantial advances have been achieved in TTS for languages such as English and Mandarin, Korean remains comparatively underrepresented due to the lack of rigorous preprocessing methods, systematically constructed datasets, a shortage of standardized Korean TTS benchmarks, and explicitly optimized models for Korean. To address these limitations, we propose a Korean-tailored data-refinement and coreset selection pipeline. It refines speech data and performs textual normalization especially for numerals and English terms, followed by a novel coreset selection strategy that leverages Jamo-based linguistic and phonological features unique to Korean. As a result, we release CoreaSpeech, an efficient and robust Korean speech corpus comprising 700 hours across 21,449 speakers. This refined core subset, evenly balanced across utterances ranging from 0 to 30 seconds, is derived from 2,058 hours of widely used Korean datasets. Building on this, we conducted extensive experiments via cross-lingual fine-tuning with our CoreaSpeech dataset. Furthermore, we introduce a new universal Korean TTS benchmark dataset including clean, noisy, and numeric subsets. Additionally, we demonstrate that our Korean-specific text normalization serves as a plug-and-play module, reliably improving performance regardless of the underlying TTS architecture. We publicly release our dataset, pipeline code, and evaluation benchmarks to support reproducible research and further advances in Korean and multilingual speech synthesis: https://coreaspeech.github.io/demo/.

1 Introduction

Recent advancements in Text-to-Speech (TTS) technology have significantly enhanced synthesized speech quality, including improved naturalness, particularly in widely researched languages such as English and Mandarin. Key factors driving these improvements include large-scale high-quality corpora [1, 2], well-established preprocessing methods [2, 3], publicly available and rigorously designed evaluation benchmarks [4–6], and powerful, optimized models [7–9]. However, research in Korean TTS remains comparatively underdeveloped, primarily due to limitations such as the small-scale of available speech corpora, lack of sufficient Korean-specific preprocessing methodologies, a shortage of standardized Korean TTS evaluation benchmarks, and an absence of explicitly optimized models tailored to Korean linguistic and phonological characteristics.

Korean Corpus from large multilingual datasets (e.g., Emilia [10] and Common Voice[11]) while maximizing diversity, often present potential audio quality issues, including severe background noise from online sources and unnatural pronunciations due to non-native speakers. In contrast, the KSS dataset [12], although of high audio quality due to studio recordings, consists exclusively of single-speaker recordings. Moreover, large-scale Korean datasets such as AI-Hub[2]predominantly consist

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Table 1: Publicly Available Speech Corpora Used in TTS Research

Dataset	Hours	#Samples	#Speakers	Languages	License
Non-Korean					
LJSpeech	24	13,100	1	English	Public Domain
LibriTTS	585	–	2,456	English	CC BY 4.0
LibriSpeech	982	292,367	2,484	English	CC BY 4.0
WenetSpeech4TTS	12,800	–	–	Mandarin	CC BY 4.0
Emilia	101,654	–	–	Multilingual	CC BY-NC 4.0
IndicVoices–R [17]	1,704	690,000	10,496	22 Indian languages	CC BY 4.0
Common Voice	33,150	–	–	133 languages	CC0 1.0
Korean-specific					
KSS	12	12,853	1	Korean	CC BY-NC-SA 4.0
Emilia (KO)	213	91,065	27,171	Korean	CC BY-NC 4.0
CoreaSpeech (Ours)	700	168,790	21,449	Korean	CC BY-NC 4.0

of repeated scripts spoken by multiple speakers, resulting in limited linguistic diversity. In addition to their individual drawbacks, these datasets share a common limitation of having predominantly short utterances (3–4 seconds), restricting the models' ability to effectively generate longer, natural-sounding speech segments, as highlighted in previous studies [13, 14].

During preprocessing, Korean speech data presents additional complexity due to two main reasons. First, Korean speech data frequently includes English loanwords, code-switching, and numerals with context-dependent pronunciations. These characteristics necessitate specialized normalization strategies to ensure training consistency and speech quality. Second, Korean's unique syllable block structure composed of Jamo characters requires phonological balance. Recent studies highlight the importance of effective preprocessing for data selection, but previous embedding-based approaches using Large Language Models (LLMs) [15, 16] have primarily focused on coarse aspects, limiting their suitability for Korean's phonological requirements. Thus, there is a clear need for an efficient data selection method that reflects the phonological characteristics of Korean.

Moreover, Korean TTS research lacks standardized evaluation benchmarks covering diverse acoustic and linguistic contexts. A comprehensive benchmark capable of evaluating synthesis quality under optimal acoustic conditions, robustness to noisy environments, and accuracy in handling context dependent numeral pronunciations is therefore essential.

Finally, despite recent advances in multilingual TTS systems, models that faithfully capture Korean's unique phonological and graphemic characteristics remain scarce. To close this gap, there is a need for a dedicated Korean TTS model that can be easily optimized by fine-tuning only a small set of parameters, without relying on extensive computational resources.

To address these limitations, we introduce:

- **CoreaSpeech:** A diverse, and high-fidelity Korean speech corpus (700 hours, 21,449 speakers), refined from raw public audio.
- **Korean Text Preprocessor (N2gk+):** An automatic normalization module handling context-dependent numerals, English loanwords, code-switching, and special characters.
- **Efficient Coreset Selection:** Linguistically aware coreset selection leveraging Korean-specific Jamo pair distributions and acoustically aware selection via dynamic audio quality thresholds (UTMOS)
- **Korean Universal Testset:** Comprehensive Korean TTS benchmark comprising clean, noisy, and numeric subsets to rigorously evaluate synthesis quality across diverse acoustic and linguistic scenarios.
- **Korean Optimized TTS (PEFT-TTS):** A Korean-specialized TTS model optimized via low-resource, parameter-efficient fine-tuning of only 5.81M LoRA parameters, trained on a single GPU.

We publicly release our dataset, preprocessing pipeline, and evaluation benchmarks to facilitate reproducible research, promote further advancements, and accelerate progress in Korean and multilingual TTS research.

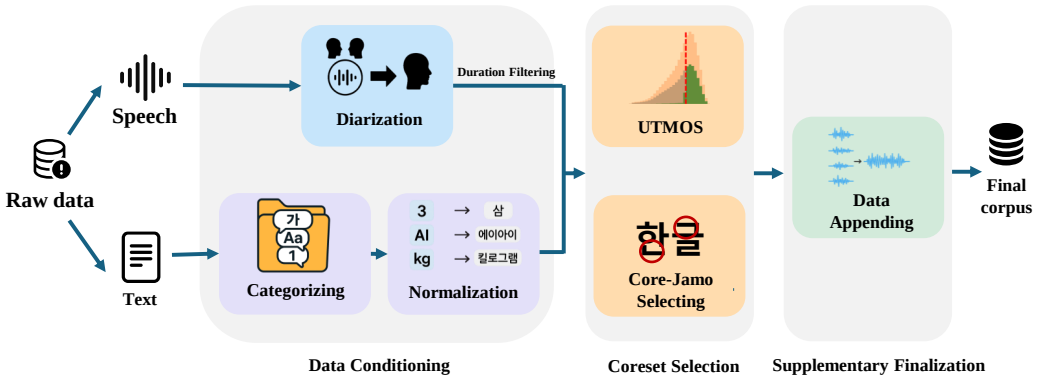


Figure 1: **Overview of the proposed Pipeline.**

2 Data Pipeline

This section introduces a data pipeline designed to construct a high-fidelity Korean speech corpus that not only ensures data quality but also carefully selects core subsets for optimized TTS model training, as illustrated in Figure 1.

2.1 Data Conditioning

2.1.1 Audio

We begin by performing speaker diarization to ensure segments contain speech from a single speaker. Given raw audio from diverse sources, we employ the pyannote/speaker-diarization-3.1 [18] to label speaker turns within each utterance. We only retain segments containing exactly one speaker. Any multi-speaker segments are discarded. This procedure ensures speaker-homogeneity, facilitating subsequent normalization steps. Further evaluation results for diarization are provided in Section 5.4 (see Table 6).

2.1.2 Text Categorizing

Directly discarding all utterances containing non-Korean language tokens may lead to the loss of critical semantic and acoustic information. Thus, to minimize the loss of valuable speech samples, we propose LNCat, a precise text categorization approach that selectively retains utterances based on their convertibility into Korean graphemes (Hangul). Specifically, LNCat identifies English tokens commonly used in everyday speech, such as institutional names and units, which can reliably be converted into Korean pronunciations, while discarding samples containing English words or sentences that may confuse the training process. We implement this selective categorization using a simple yet effective algorithm, with detailed procedures and evaluation provided in Appendix C.

2.1.3 Text Normalization

Normalizing numerals, English tokens, and special characters into consistent Korean pronunciations pose significant challenges due to their inherent complexity and variability in spoken Korean. Thus, we propose N2gk+, an automatic normalization method designed to produce natural Korean pronunciations without relying on manual annotations or resource-intensive speech recognition models. Specifically, N2gk+ robustly handles numerals that potentially have multiple valid pronunciations as well as convertible English tokens and special characters, converting them into Hangul graphemes to ensure consistent and clear textual inputs for subsequent TTS acoustic modeling. Additionally, applying N2gk+ to TTS model input texts significantly improves the quality of synthesized speech, with detailed analyses and evaluation results provided in the Section 5.3. Further methodological details and computational efficiency are presented in Appendix D.

[2] `https://aihub.or.kr/`.

2.2 Coreset Selection

2.2.1 Jamo Bigram (Grapheme level)

Achieving exhaustive phonetic coverage while mitigating phonetic imbalance remains inherently challenging. Further complicating matters, English writing represents phonemes directly with basic graphemes such as individual letters or common digraphs, whereas Korean orthography represents an entire syllable as a single block. Each block is built from up to three sub-graphemes called *jamo*: an initial consonant (IC), a medial vowel (MV), and an optional final consonant (FC). With 19 ICs, 21 MVs, and 28 FCs, these sub-units can generate up to 11,172 distinct syllables. Moreover, Korean pronunciation involves phonetic interactions between adjacent syllables, such as liaison and assimilation. Capturing every possible pairwise interaction would require modelling up to $11,172^2$ combinations, which is practically infeasible.

To address this issue efficiently, we propose a Jamo Bigram coreset selection strategy that explicitly considers pairwise Jamo adjacency. Specifically, we represent each utterance through four distinct adjacent types: (IC, MV), (MV, FC), (FC, *next*-IC), and (MV, *next*-IC), cumulating in a maximum of 1,878 unique Jamo pairs. This method efficiently captures phonetic patterns and interactions between adjacent syllables. Further details are provided in Appendix J.

We then aggregate these Jamo pairs across the dataset and estimate their distributions. If a pair's count is *at or below* a predefined threshold t (e.g., 500), all utterances containing that pair are retained so that comparatively rare phonetic patterns are never lost. For pairs whose counts exceed t, every utterance containing the over-represented pair is retained with probability:

$$p = \exp[-\beta(\textit{global count } - t)] \tag{1}$$

where β is a small constant (e.g., 10^{-4}). This probabilistic filtration curbs the dominance of frequent Jamo patterns, thereby promoting a more even exploration of the language's phonetic space. Further hyperparameter setting details are provided in Appendix H.2. While we specifically utilize Jamo-level pair distributions for efficient and linguistically representative core subset selection, alternative grapheme-level tokenization methods such as Byte-Pair Encoding (BPE) [19] could also potentially be employed.

2.2.2 Dynamic UTMOS threshold (Audio quality level)

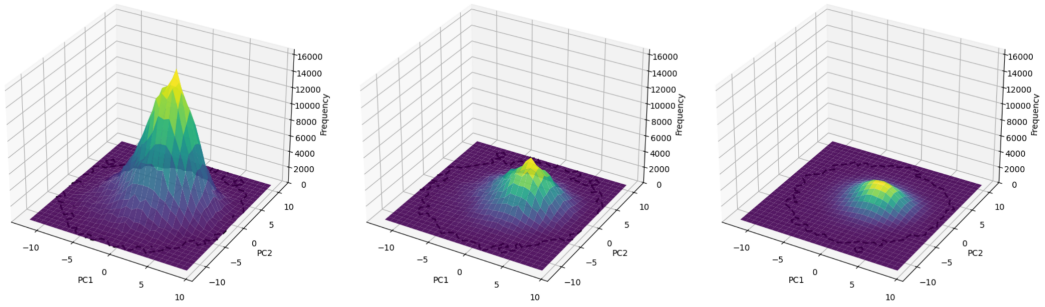
Concurrently, among the utterances retained after stochastic removal step (i.e., those containing sufficiently represented Jamo pairs above threshold t), we further refine sample selection by considering their audio quality as measured by the UTMOS metric [20].

Using a fixed (static) UTMOS threshold to filter low-quality utterances often excluded various audio segments containing slight background noise, adversely affecting the model's ability to generalize and synthesize speech from noisy audio prompts. Moreover, using a fixed threshold sometimes resulted in the unintended exclusion of entire speaker subsets from specific datasets. Since most publicly available Korean speech datasets including those in our study are typically collected from specific platforms or under specific recording conditions, we addressed this limitation by introducing a dynamic, dataset specific UTMOS thresholding method. This approach determines appropriate thresholds tailored to each individual dataset, incorporating a broader and more diverse range of audio qualities and speakers into the training corpus. Detailed experimental results and procedures for determining these dynamic thresholds are provided in the Section 5.4 and Appendix E.

2.3 Supplementary Finalization

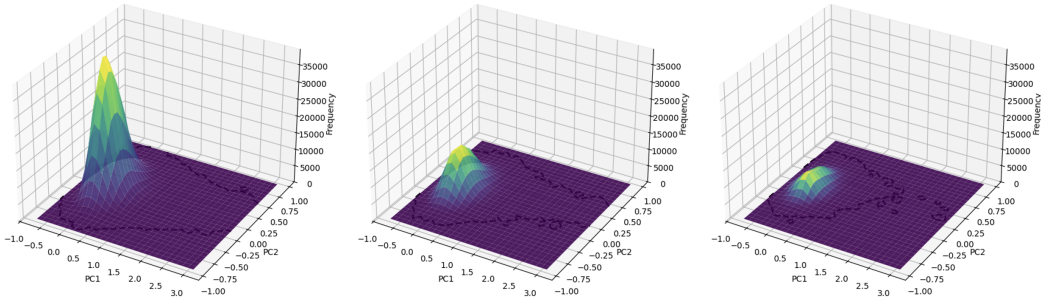
2.3.1 Data Appending for duration balancing

Lastly, recognizing that Korean speech datasets typically contain a large proportion of very short utterances, we explored a data appending method. Specifically, short segments from the same speaker were concatenated using cross-fading to produce longer utterances, thereby adjusting the dataset to include a wider and more balanced range of sample durations up to a maximum of 30 seconds. When selecting utterances, we randomly determined both the number of segments and the specific segments within each speaker, considering both speaker-level and dataset-level duration distributions. Despite potential drawbacks, this approach reduced hallucinations to some extent when synthesizing longer sentences. Further details regarding this method are included in Appendix F.1.



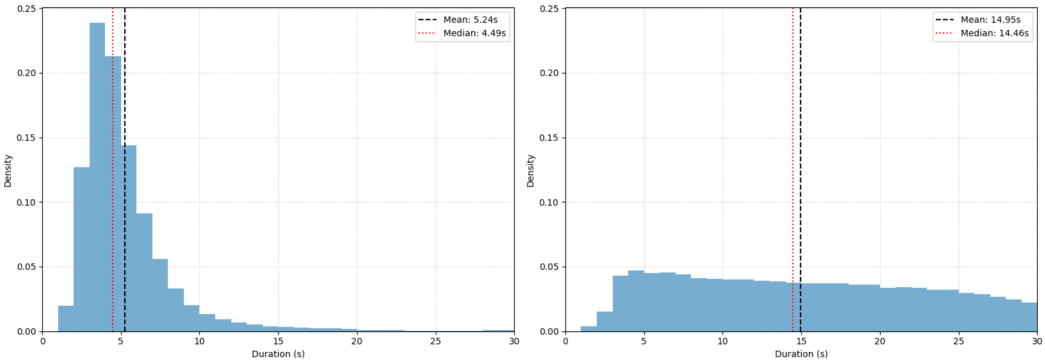
(a) Raw Data (b) Core Data (c) Core-appended Data

Figure 2: Semantic PCA with frequency and projection



(a) Raw data (b) Core data (c) Core-appended data

Figure 3: Acoustic PCA with frequency and projection



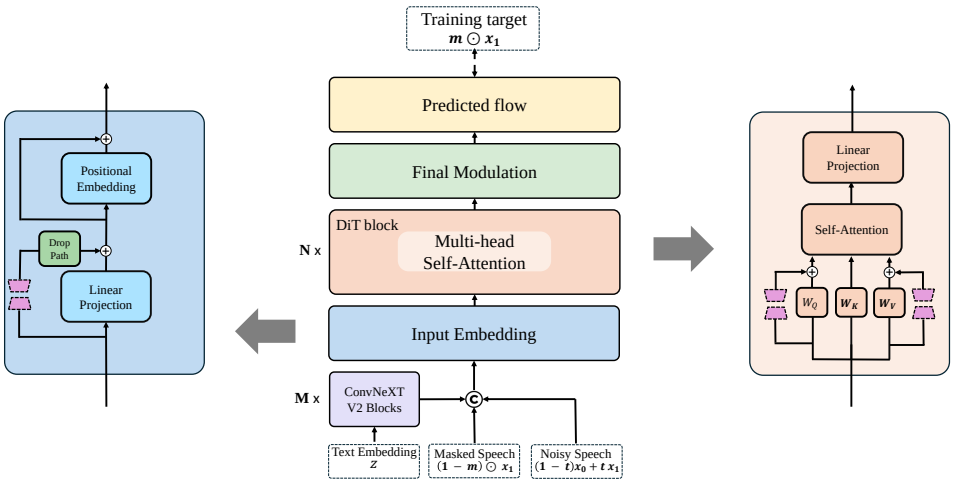
(a) Core data (b) Augmented Core data

Figure 4: Duration Augmentation

3 Dataset Analysis

Semantic & Acoustic features: We assessed the frequency and diversity of the semantic (Figure 2) and acoustic (Figure 3) embedding spaces by fitting PCA on the 768-dimensional ko-sbert-sts [21] and WavLM [22] vectors, then projecting every dataset variant onto the same two principal components axes derived from the raw data. Both analyses reveal that the raw data exhibits highly concentrated regions, which are reduced after coreset selection. Semantic diversity was particularly well preserved, maintaining over 95% of the original diversity, and acoustic diversity held at 82% despite speaker diarization and UTMOS filtering. Additionally, duration appending largely preserves the original (Core data) distributions in both semantic and acoustic features.

Duration distribution: Figure 4 illustrates that duration appending resulting closely to a uniform distribution across 0-30 s range, increasing the mean utterance length from 5.24 s to 14.95 s. Experimental results in 5.4 show that this appending mitigates errors when synthesizing long utterances.



(a) Input Embedding (Prompt Adapter) (b) Training Procedure (c) Multi-head Self-Attention (DiT LoRA Adapter)

Figure 5: *Overview of PEFT-TTS: (a) Prompt Adapter with DropPath, (b) Training procedure, and (c) LoRA adapter for DiT.*

4 Model

This chapter introduces the base model F5-TTS [23] and PEFT-TTS, an application of parameter-efficient fine-tuning, to evaluate the effectiveness and suitability of the CoreaSpeech dataset and to build a model explicitly optimized for Korean speech synthesis.

4.1 F5-TTS

F5-TTS is a non-autoregressive TTS model trained in a multilingual environment (English and Mandarin). It generates mel-spectrograms through a flow-matching approach [24], utilizing a text embedding module based on ConvNeXt V2 [25] and Diffusion Transformer (DiT) blocks [26]. Since F5-TTS directly learns the text-to-speech mapping without forced alignment as in E2-TTS [27], it minimizes the loss in duration diversity that may occur during the forced alignment process.

4.2 PEFT-TTS

Although multilingual TTS models have the advantage of covering a wide range of speech and acoustic characteristics across multiple languages [9], they are not necessarily fully optimized for Korean. To achieve performance specialized for Korean at low computational cost, this study applies a *cross-lingual* parameter-efficient fine-tuning (PEFT) technique to the existing F5-TTS model. In this approach, considering that the pretrained model was not trained on Korean, we applied full fine-tuning to the text embedding and ConvNeXt V2 modules, while the remaining modules were trained only using LoRA adapter [28] parameters specialized for the Korean dataset, CoreaSpeech. As shown in Figure 5, ConvNext V2, Prompt Adapter, and DiT-LoRA Adapter learn Korean speaker characteristics, pronunciation, and prosody in detail during training.

Prompt Adapter Prompt Adapter LoRA (rank=64) is applied to the linear projection layer, which receives concatenated text embeddings and audio features as input. This layer plays a crucial role in balancing the trade-off between pronunciation accuracy and speaker similarity. To fine-tune this adapter, the DropPath technique [29] is used to stochastically drop the adapter's residual path, which alleviates overfitting and improves generalization performance across the entire dataset.

DiT LoRA Adapter LoRA (rank=16) is also applied to the self-attention layer of the DiT block that generates the mel-spectrogram. This design efficiently adjusts the core parameters of the block responsible for capturing speaker characteristics, preserving the existing pre-trained knowledge while flexibly integrating speaker and prosody information from new data. The specific impact of the LoRA rank on synthesis quality is described in detail in Appendix H.1.

Table 2: Comparison of recent text-to-speech models

Models	Korean Text Processor	Training Data	Korean Data(hour)	#Param. (Trainable)
XTTS	BPE + hangul-romanizer [30]	Common Voice (Public)	539	0.4B
CosyVoice 2	raw BPE + Num2Words (En)	Internal	2,200	0.5B
Llasa	Llama BPE	Emilia (Public)	213	1B
Zonos	eSpeak NG	Internal	unknown	1.6B
PEFT-TTS	**Jamo + N2gk+**	**CoreaSpeech (Public)**	**700**	**0.3B (5.81M)**

Table 3: Korean Universal Testset

Eval Set	Model	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
Clean data	GT	3.14	9.46	0.83	3.60
	XTTS	4.22	12.23	0.70	3.16
	CosyVoice 2	3.08	9.05	**0.80**	3.72
	Llasa	8.65	25.37	0.65	**3.90**
	Zonos	5.17	15.83	0.76	3.41
	PEFT-TTS (Ours)	**2.37**	**6.97**	0.80	3.37
Noisy data	GT	9.21	22.64	0.76	1.65
	XTTS	15.68	39.69	0.53	1.68
	CosyVoice 2	22.12	31.27	0.61	2.30
	Llasa	34.10	89.18	0.31	2.20
	Zonos	**8.25**	**20.18**	0.64	**2.58**
	PEFT-TTS (Ours)	8.25	20.61	**0.67**	2.13
Numeric data	GT	6.23	18.93	0.85	3.42
	XTTS	8.21	27.15	0.71	2.81
	CosyVoice 2	17.65	52.60	**0.82**	**3.60**
	Llasa	17.80	58.75	0.66	3.55
	Zonos	21.21	64.08	0.77	3.29
	PEFT-TTS (Ours)	**5.97**	**19.71**	0.81	2.90
Avg.	GT	6.19	17.01	0.81	2.89
	XTTS	9.37	26.35	0.64	2.55
	CosyVoice 2	14.28	30.97	0.74	3.21
	Llasa	20.18	57.76	0.54	**3.22**
	Zonos	11.54	33.37	0.72	3.09
	PEFT-TTS (Ours)	**5.35**	**15.76**	**0.76**	2.80

5 Evaluation

5.1 Experimental Setup

We evaluated the CoreaSpeech dataset by fine-tuning the multilingual F5-TTS, pretrained on English and Mandarin, using the parameter-efficient approach described in Section 4.2. To match the original pretraining conditions, all training samples were downsampled to 24kHz. We maintained the same hyperparameters from the F5-TTS model, specifically setting RMS normalization to 1, classifier-free guidance strength to 2.0, using the Euler ODE solver method, and performing 32 NFE steps. Additional fine-tuning configurations included LoRA with rank of 16 and 64 for the prompt adapter and DiT LoRA, respectively, a DropPath rate of 0.3, the AdamW optimizer, and a learning rate of 1e5 with training conducted on **a single NVIDIA TITAN RTX**. For static UTMOS threshold, we applied 3.5.

Evaluations involved a cross-sentence synthesis task, using objective metrics, Character Error Rate (CER) and Word Error Rate (WER), computed via Whisper-Large-V3 [31], speaker similarity (SIM) via WavLM and synthesis quality via UTMOS. For subjective evaluation, we conducted Naturalness Mean Opinion Score (nMOS) tests involving 20 native Korean listeners, who rated synthesized speech samples on a scale from 1 (poor) to 5 (excellent).

Additionally, to evaluate our proposed Korean text preprocessor (N2gk+), we specifically utilized Korean news data categorized using LNCat for measuring conversion performance.

5.2 Benchmarking: Korean Universal Testset

We introduce the Korean Universal Testset, a novel benchmark designed to rigorously assess the performance and robustness of Korean TTS models. This benchmark consists of three distinct subsets: Clean, Noisy, and Numeric, each comprising 100 utterances selected to represent diverse speakers and linguistic scenarios relevant to practical Korean TTS applications. All subsets are specifically constructed for cross-sentence tasks to evaluate zero-shot synthesis capabilities.

Table 4: Number-to-Korean Comparison: Latency and Accuracy

Method	Latency (ms)	Numeric (%)	English (%)	Numeric & English (%)
g2pk	31.60 ± 0.52	31.03	5.98	1.26
KoNLPy	0.11 ± 0.05	53.13	0.00	0.00
ChatGPT-4.1	API	52.67	34.51	76.69
ChatGPT-o3	API	61.13	45.65	76.92
N2gk+	1.19 ± 0.16	**90.38**	**96.43**	**81.77**

Table 5: Evaluation of N2gk+ as a *Plug-and-Play* Text Normalization Module

Method	N2gk+	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
GT	–	6.23	18.93	0.84	3.42
XTTS	✗	8.21	27.15	0.71	2.81
	✓	6.86	21.29	0.71	2.83
CosyVoice 2	✗	17.65	52.60†	0.82	3.60
	✓	6.90	22.95	0.82	3.43
Llasa	✗	17.80	58.75	0.66	3.55
	✓	11.15	35.45	0.67	3.60
Zonos	✗	21.21	64.08	0.77	3.29
	✓	7.29	24.12	0.78	3.34
Ours	✗	11.11	35.99	0.81	2.86
	✓	5.97	19.71	0.81	2.90

The *Clean set* measures synthesis performance under optimal acoustic conditions. In contrast, the *Noisy set* evaluates robustness in acoustically challenging scenarios typically encountered in everyday life, including urban street noises, vehicle sounds, human conversations, and wind noise. Lastly, given the complexity of numeric pronunciation in Korean, the *Numeric set* includes numerals, floating-point numbers, explicitly pronounced special symbols (e.g., percentages, %), common English abbreviations, and units denoted by English characters.

We benchmarked our model against multilingual TTS models capable of Korean speech synthesis (XTTS[32], CosyVoice 2[33], Llasa[34], and Zonos[35]), as summarized in Table 2. Performance outcomes for each subset are presented in Table3.

Additionally, for supplementary validation using well-known datasets , we conducted Korean Open-source Testset. Detailed results for this set are available in Appendix (see Table 13). The model demonstrates robust overall performance though lower UTMOS scores due to slower pronunciation of numbers in the Numeric Set. Substantial nMOS, sMOS and RTF results are detailed in Appendix A.

5.3 Korean Text Preprocessor: N2gk+

We evaluated our Korean text preprocessor, N2gk+, designed as a plug-and-play module to significantly enhance numeric and English abbreviation handling in Korean TTS. We first compared N2gk+ with existing number-to-Korean conversion methods, including g2pk [36], KoNLPy [37], and ChatGPT [38, 39], focusing on latency and accuracy (Table 5). The KoNLPy library, primarily designed for general Korean natural language processing, relies on simple mapping for numeric conversions, resulting in lower accuracy and lacking English handling capabilities. The g2pk method converts text to phonemes, enabling English handling but offering limited benefits due to its high latency and sensitivity to spacing issues, as detailed in Appendix D.3. ChatGPT exhibited high accuracy on expressions where numeric values co-occured with English unit terms, yet its overall conversion performance was still modest. In contrast, N2gk+ exhibited superior performance across all evaluated categories, while maintaining relatively low latency.

Further, we observed even greater improvements when integrating N2gk+ into multilingual TTS models performing Korean synthesis, likely due to the universal nature of numeric expressions. As demonstrated in Table 6, integrating N2gk+ consistently improved linguistic accuracy across various Korean TTS models compared to their original preprocessing methods.

Table 6: Performance Comparison across Pipeline Components (*Data Cond.*, *Suppl. Final., Spk., Diar., Jamo Bi., Data App.,* and *Dur.* denote Data Conditioning, Supplementary Finalization, Number of Speakers, Diarization, Jamo Bigram, Data Appending, and Total Duration, respectively)

Ablation	Data Cond.		Coreset Selection		Suppl. Final.	Samples	Spk.	CER(↓)	WER(↓)	SIM(↑)	UTMOS(↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Data App.	(Dur.)					
GT	–	–	–	–	–	-	-	6.19	17.01	0.81	2.89
Emilia+KSS	✗	✗	✗	✗	✗	103k (225)	27k	7.17	21.52	0.72	2.81
Base	✗	✗	✗	✗	✗	1,591k (2,058)	28k	9.35	26.83	0.70	2.80
Base-diar.	✓	✗	✗	✗	✗	1,582k (2,031)	26k	9.33	26.82	0.69	2.80
Norm	✓	✓	✗	✗	✗	1,559k (1,993)	25k	7.72	21.84	0.76	2.82
Norm$_{static}$	✓	✓	✗	✓	✗	875k (998)	2k	13.5	38.60	0.76	2.79
Norm-app	✓	✓	✗	✗	✓	448k (1,993)	25k	6.07	17.19	0.74	2.83
Core-app	✓	✓	✓	✗	✓	238k (1,249)	22k	5.98	17.19	0.75	2.79
Core-app$_{static}$	✓	✓	✓	✓	✓	132k (565)	4k	6.03	16.91	0.75	2.79
CoreaSpeech	✓	✓	✓	Dynamic	✓	168k (700)	21k	5.54	15.76	0.76	2.80

5.4 Ablation Study: Pipeline Components

We conducted an ablation study to evaluate the contributions of individual components within the CoreaSpeech pipeline, as summarized in Table 6. Each entry correspond to a dataset variant derived from Base configuration, indicating whether each component is applied or not, while Base denotes the raw unprocessed dataset. Text normalization significantly contributes to model performance, as demonstrated by comparing the Base and Norm datasets. Although speaker diarization is essential, the comparison between Base and Base-diar. reveals that the decrease in the number of samples relative to number of speakers is marginal. Data appending improves WER and CER by alleviating hallucination issues despite certain limitations, as further detailed in Appendix F.2. Coreset selection based on Jamo pairs achieves comparable or superior performance using fewer data points, aiding the model in effectively capturing Korean phonological phenomena. Filtering data based on fixed audio quality threshold severely reduces speaker diversity and negatively impacts performance. Incorporating rare Jamo pairs preserves approximately twice the number of speakers, while applying a dynamic UTMOS threshold retains nearly ten times more speakers, resulting in superior overall performance. Detailed results for each subset across these metrics are provided in Appendix G.1.

6 Potential Broader Impact

Practical Application Despite rapid advancements in multilingual TTS research, the lack of high-quality, language-specific Korean speech-text datasets remains a bottleneck. Our proposed CoreaSpeech (700 h), automated pipeline, plug-and-play text preprocessor (N2gk+), and PEFT-TTS models significantly enhance Korean speech synthesis performance.

Ethical Consideration Improved synthetic speech quality might inadvertently increase misuse risks, such as generating misleading content or deepfake speech, emphasizing the need for careful management and detection strategies. Accordingly, future research will focus on developing detection and monitoring techniques, as well as guidelines for ethical use to mitigate these potential issues.

Limitation The dataset size was limited by available public speech resources, and a larger and more diverse corpus would likely further improve model performance. Additionally, duration appending, though effective for increasing utterance length, may introduce contextual inconsistencies. Lastly, numeric normalization to Korean pronunciations slows the reading speed of synthesized numeric content, potentially affecting naturalness.

7 Conclusion

In this paper, we introduce CoreaSpeech, a large-scale, high-quality Korean speech corpus effectively curated by leveraging the linguistic characteristics of Korean. Our key contributions include the release of extensive, high-quality Korean data, a preprocessing pipeline that effectively addresses phonetic imbalances and contextually variable numerals and English loanwords, and the Korean Universal Testset, designed to robustly evaluate diverse acoustic environments and complex numeral pronunciations. Moreover, we validated the effectiveness of our proposed pipeline through extensive experiments utilizing parameter-efficient fine-tuning on a single GPU. Despite these advancements, achieving optimal naturalness remains challenging. Thus, future research would require developing more sophisticated data appending techniques and methodologies for determining optimal thresholds, aiming to further improve the naturalness and overall quality of synthesized speech.

Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2025-02283048, Developing the Next-Generation General AI with Reliability, Ethics, and Adaptability, IITP-2025-RS-2023-00255968, the Artificial Intelligence Convergence Innovation Human Resources Development, No.RS-2021-II212068, Artificial Intelligence Innovation Hub, RS-2025-02653113, High-Performance Research AI Computing Infrastructure Support at the 2 PFLOPS Scale), National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (RS-2025-16069227) and Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT(MSIT, Korea)&Gwangju Metropolitan City.

This paper used datasets from 'The Open AI Dataset Project (AI-Hub, S. Korea)'. All data information can be accessed through 'AI-Hub (www.aihub.or.kr)'. We acknowledge the licensed use of datasets from MediaZen (미디어젠) (`https://www.aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&dataSetSn=466`) and CommunicationBooks, Inc. (커뮤니케이션북스(주)) (`https://aihub.or.kr/aihubdata/data/view.do?dataSetSn=71349`).

References

[1] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[2] Linhan Ma, Dake Guo, Kun Song, Yuepeng Jiang, Shuai Wang, Liumeng Xue, Weiming Xu, Huan Zhao, Binbin Zhang, and Lei Xie. Wenetspeech4tts: A 12,800-hour mandarin tts corpus for large speech generation model benchmark. In *Interspeech 2024*, pages 1840–1844, 2024.

[3] Jongseok Park, Kyubyong Kim. g2pe. `https://github.com/Kyubyong/g2p`, 2019.

[4] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. In *Interspeech 2019*, pages 1526–1530, 2019.

[5] Georgia Maniati, Alexandra Vioni, Nikolaos Ellinas, Karolos Nikitaras, Konstantinos Klapsas, June Sig Sung, Gunu Jho, Aimilios Chalamandaris, and Pirros Tsiakoulis. Somos: The samsung open mos dataset for the evaluation of neural text-to-speech synthesis. In *Interspeech 2022*, pages 2388–2392, 2022.

[6] Yao Shi, Hui Bu, Xin Xu, Shaoji Zhang, and Ming Li. Aishell-3: A multi-speaker mandarin tts corpus. In *Interspeech 2021*, pages 2756–2760, 2021.

[7] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, Rif A. Saurous, Yannis Agiomvrgiannakis, and Yonghui Wu. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783, 2018.

[8] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021.

[9] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 14005–14034. Curran Associates, Inc., 2023.

[10] Haorui He, Zengqiang Shang, Chaoren Wang, Xuyuan Li, Yicheng Gu, Hua Hua, Liwei Liu, Chen Yang, Jiaqi Li, Peiyang Shi, Yuancheng Wang, Kai Chen, Pengyuan Zhang, and Zhizheng Wu. Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 885–890, 2024.

[11] Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France, May 2020. European Language Resources Association.

[12] Kyubyong Park. Korean single speaker speech dataset, 2018.

[13] Shijie Lai, Minglu He, Zijing Zhao, Kai Wang, Hao Huang, and Jichen Yang. Synthesizing long-form speech merely from sentence-level corpus with content extrapolation and llm contextual enrichment. In *Interspeech 2024*, pages 3430–3434, 2024.

[14] Ramon Sanabria, Wei-Ning Hsu, Alexei Baevski, and Michael Auli. Measuring the impact of individual domain factors in self-supervised pre-training, 2023.

[15] Devleena Das and Vivek Khetan. DEFT-UCS: Data efficient fine-tuning for pre-trained language models via unsupervised core-set selection for text-editing. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20296–20312, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[16] Kentaro Seki, Shinnosuke Takamichi, Takaaki Saeki, and Hiroshi Saruwatari. Diversity-based core-set selection for text-to-speech with linguistic and acoustic features. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12351–12355. IEEE, 2024.

[17] Ashwin Sankar, Srija Anand, Praveen Srinivasa Varadhan, Sherry Thomas, Mehak Singal, Shridhar Kumar, Deovrat Mehendale, Aditi Krishana, Giri Raju, and Mitesh M Khapra. Indicvoices-r: Unlocking a massive multilingual multi-speaker speech corpus for scaling indian TTS. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[18] Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. Pyannote.audio: Neural building blocks for speaker diarization. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7124–7128, 2020.

[19] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[20] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. Utmos: Utokyo-sarulab system for voicemos challenge 2022. *Interspeech 2022*, 2022.

[21] Junyoung (Hugging Face user `jhgan`) Gan. ko-sbert-sts: Sentence-bert model for korean sts. `https://huggingface.co/jhgan/ko-sbert-sts`, 2021.

[22] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, 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.

[23] Yushen Chen, Zhikang Niu, Ziyang Ma, Keqi Deng, Chunhui Wang, Jian Zhao, Kai Yu, and Xie Chen. F5-tts: A fairytaler that fakes fluent and faithful speech with flow matching. *arXiv preprint arXiv:2410.06885*, 2024.

[24] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[25] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16133–16142, 2023.

[26] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[27] Sefik Emre Eskimez, Xiaofei Wang, Manthan Thakker, Canrun Li, Chung-Hsien Tsai, Zhen Xiao, Hemin Yang, Zirun Zhu, Min Tang, Xu Tan, et al. E2 tts: Embarrassingly easy fully non-autoregressive zero-shot tts. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 682–689. IEEE, 2024.

[28] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[29] Gao Huang, Yu Sun, Zhuang Liu, David Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016.

[30] YunWon Jeong. hangul-romanize: Hangul romanization tool. `https://pypi.org/project/hangul-romanize/`, 2020.

[31] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022.

[32] Edresson Casanova, Kelly Davis, Eren Gölge, Görkem Göknar, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, and Julian Weber. Xtts: a massively multilingual zero-shot text-to-speech model. In *Interspeech 2024*, pages 4978–4982, 2024.

[33] Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, Fan Yu, Huadai Liu, Zhengyan Sheng, Yue Gu, Chong Deng, Wen Wang, Shiliang Zhang, Zhijie Yan, and Jingren Zhou. Cosyvoice 2: Scalable streaming speech synthesis with large language models. *CoRR*, abs/2412.10117, 2024.

[34] Zhen Ye, Xinfa Zhu, Chi-Min Chan, Xinsheng Wang, Xu Tan, Jiahe Lei, Yi Peng, Haohe Liu, Yizhu Jin, Zheqi Dai, Hongzhan Lin, Jianyi Chen, Xingjian Du, Liumeng Xue, Yunlin Chen, Zhifei Li, Lei Xie, Qiuqiang Kong, Yike Guo, and Wei Xue. Llasa: Scaling train-time and inference-time compute for llama-based speech synthesis, 2025.

[35] Zyphra. Zonos-v0.1: An open-weight multilingual text-to-speech model. `https://github.com/Zyphra/Zonos`, 2024.

[36] Kyubyong Park. g2pk. `https://github.com/Kyubyong/g2pk`, 2019.

[37] Eunjeong L. Park and Sungzoon Cho. Konlpy: Korean natural language processing in python. In *Proceedings of the 26th Annual Conference on Human Cognitive Language Technology*, Chuncheon, Korea, October 2014.

[38] OpenAI. ChatGPT: GPT-4o. `https://platform.openai.com/docs/models`, 2024.

[39] OpenAI. ChatGPT: GPT-o3. `https://platform.openai.com/docs/models`, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the paper's contributions, including CoreaSpeech, the processing pipeline, coreset selection, a new benchmark, and the PEFT-TTS model, all of which are detailed in the subsequent sections.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are explicitly discussed in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper contains no formal theorems, so no mathematical assumptions or proofs are required.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All data, code, hyper-parameters, and checkpoints are publicly released and fully documented in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The CoreaSpeech Dataset, Korean Universal Testset, and model checkpoints are released under the CC-BY-NC 4.0 license, and the preprocessing, training, and evaluation code are released under the MIT license at the project URL.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The correspondingpp details are described in Sections 4, 5.1 and Appendix H.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Although Section 5.2 Appendix A reports error bars for experimental results, other sections lack explicit information on statistical significance, variability factors, underlying assumptions (e.g., normality), and the methods used to calculate error bars or confidence intervals.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 5.1 Experimental Setup states that fine-tuning was carried out on a single NVIDIA TITAN RTX GPU.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We comply with the NeurIPS Code of Ethics and discuss all relevant issues in Section 6.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed all relevant issues in Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We explicitly restrict the use of our data and models to non-commercial purposes through licensing.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites numerous existing datasets and tools used in the research. Specific licensing information for each item can be found in an appendix.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Detailed documentation of the assets introduced in this paper is provided on the demo page specified in the abstract.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We conducted Naturalness Mean Opinion Score (nMOS) tests in which Korean native listeners evaluated the samples. Further details are provided in Appendix A.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The nMOS listening tests involved anonymous adult participants and did not collect any personally identifiable information. Thus, explicit IRB approval was not obtained.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We do not use LLM.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Table 7: Subjective Evaluation & Real-Time Factor

Eval Set	Model	nMOS	sMOS	RTF
Clean data	GT	3.91±0.13	4.09±0.12	-
	XTTS	2.22±0.13	2.32±0.14	0.265
	CosyVoice 2	**3.57±0.13**	**3.91±0.11**	0.725
	Llasa	2.36±0.13	2.65±0.14	0.798
	Zonos	2.57±0.13	2.62±0.14	5.230
	PEFT-TTS (Ours)	3.51±0.13	3.71±0.12	**0.242**
Noisy data	GT	4.57±0.10	4.65±0.08	-
	XTTS	1.97±0.12	2.00±0.13	0.255
	CosyVoice 2	**3.47±0.14**	**3.51±0.15**	0.626
	Llasa	1.75±0.12	1.78±0.13	0.807
	Zonos	2.85±0.13	2.75±0.14	5.220
	PEFT-TTS (Ours)	3.01±0.13	3.02±0.14	**0.281**
Numeric data	GT	4.76±0.07	4.40±0.12	-
	XTTS	2.11±0.11	2.15±0.13	0.257
	CosyVoice 2	2.61±0.14	3.33±0.16	0.542
	Llasa	1.78±0.11	2.31±0.14	0.793
	Zonos	1.65±0.10	2.17±0.14	5.387
	PEFT-TTS (Ours)	**3.48±0.12**	**3.94±0.11**	**0.192**
Avg.	GT	4.41±0.06	4.38±0.06	-
	XTTS	2.10±0.07	2.15±0.08	0.259
	CosyVoice 2	3.22±0.08	**3.58±0.08**	0.631
	Llasa	1.96±0.07	2.24±0.08	0.799
	Zonos	2.36±0.08	2.51±0.08	5.279
	PEFT-TTS (Ours)	**3.33±0.07**	3.55±0.08	**0.238**

A Subjective Evaluation Results (nMOS & sMOS)

We conducted naturalness (nMOS) and similarity (sMOS) evaluations across three distinct testsets (clean, noisy, and numeric), comparing our PEFT-TTS model against several baseline models (see Table 7), with all reported scores accompanied by 95% confidence intervals (CIs). On the clean set, our model achieved comparable naturalness to CosyVoice 2. In contrast, results on the noisy set showed differences compared to CosyVoice 2, likely because our model simultaneously generates persistent background noise throughout synthesized speech, while CosyVoice 2 effectively mitigates this noise via inherent speech enhancement capabilities, leading to differences between objective neural evaluations and subjective scores. Additionally, the noisy set revealed discrepancies between objective ground-truth (GT) scores and subjective neural model evaluations, likely because everyday background noises do not typically cause discomfort for human listeners but are perceived as unnatural by neural models. In the numeric set, our model outperformed the baselines, primarily reflecting the effectiveness of our proposed text preprocessor in accurately handling numeric pronunciations. Moreover, our model achieved the fastest inference speed measured by Real-Time Factor (RTF), emphasizing its computational efficiency, especially given its small parameter size and capability to run effectively on a single GPU.

All scores were obtained through a crowdsourced online listening test. To evaluate the nMOS and sMOS of Korean TTS models, we recruited native Korean participants and provided them with detailed instructions specifying the use of headphones, maintaining a noise-free environment, and conducting absolute evaluations, as well as guidance for navigating a Gradio-based evaluation webpage. Participants listened to randomly ordered speech samples from six different models across three distinct testsets (clean, noisy, numeric) and rated each sample using a five-point. Each participant received monetary compensation of KRW 50,000 upon completion of all evaluations, an amount exceeding Korea's minimum hourly wage at the time of evaluation. Screenshots of the nMOS and sMOS evaluation interfaces are provided in Figure 6.

(a) nMOS (b) sMOS

Figure 6: Screenshots of the Gradio interface used for nMOS and sMOS evaluations.

B Related Works

B.1 Acoustic Normalization

The VCTK dataset (44 hours, 109 English speakers), composed of studio-quality recordings, further exemplified controlled acoustic conditions. It underwent minimal acoustic processing, mainly normalization and spectral filtering, given the inherently low background noise and reverberation from professional recording setups. Recent large-scale speech datasets, however, have primarily relied on diverse internet sources, resulting in considerable acoustic variability. The Emilia dataset (101,654 hours, multilingual) addressed this by introducing "Emilia-Pipe," a scalable open-source pipeline involving standardization, vocal extraction (using Ultimate Vocal Remover), speaker diarization, VAD-based segmentation, and deep-learning-driven speech enhancement. Final filtering was applied via quality metrics (e.g., DNSMOS), ensuring high acoustic quality from inherently noisy sources.

Similarly, WenetSpeech4TTS (12,800 hours, Mandarin) transformed in-the-wild data through speaker diarization and deep neural networks for noise reduction and reverberation control. The processed segments were subsequently filtered based on acoustic clarity and speech intelligibility metrics. IndicVoices-R (1,704 hours, 10,496 speakers, 22 Indian languages) followed a comprehensive cleaning pipeline incorporating state-of-the-art neural models, HTDemucs for noise removal, VoiceFixer for dereverberation, and DeepFilterNet3 for artifact suppression producing high-quality speech data from diverse real-world environments. These works collectively demonstrate the evolution of acoustic data cleaning techniques, incorporating sophisticated deep learning models alongside traditional signal processing to produce extensive, multilingual datasets optimized for advanced speech generation tasks.

B.2 Text Normalization

Text normalization is a crucial preprocessing step for TTS datasets, commonly involving lowercasing, numeral expansion, abbreviation standardization, and consistent punctuation rules. Traditional English datasets like LibriTTS and LibriSpeech typically provide transcripts processed through standard normalization procedures. These include conversion of numbers into written forms, abbreviation expansions, removal or normalization of punctuation, and consistent casing. Such transformations ensure homogeneity in textual input, facilitating more effective TTS model training.

Recently developed large-scale datasets like Emilia and WenetSpeech4TTS also employ similar methods, but with additional multilingual considerations. Emilia uses a multilingual ASR model (Whisper-Medium) combined with efficient segmentation techniques, resulting in consistently annotated multilingual transcripts. WenetSpeech4TTS similarly leverages ASR-based transcription followed by language-specific normalization rules to maintain uniformity across its large Mandarin corpus. IndicVoices-R, on the other hand, carefully filters and validates speech-text pairs using multiple levels of standardized transcription, explicitly differentiating between verbatim and normalized forms, thus accommodating diverse linguistic phenomena across Indian languages.

However, text normalization for Korean poses unique challenges due to the language's agglutinative characteristics, nuanced pronunciation rules, and complex numeral systems. Widely used Korean NLP tools like KoNLPy offer morphological analysis and tokenization, yet suffer from significant limitations in numeral handling. KoNLPy's numeric processing relies primarily on simple direct mappings, rendering it insufficient for representing numerals accurately beyond very basic use cases. Another commonly used library, g2pK, converts graphemes to phonemes to handle Korean pronunciation complexities, explicitly addressing context-sensitive pronunciation rules. However, g2pK struggles with English and numerals embedded within Korean sentences, as it forcibly maps them to Hangul-based pronunciations. Moreover, g2pK's performance is highly sensitive to spacing errors, a frequent issue given Korean's optional and inconsistent spacing rules. Consequently, this sensitivity often leads to erroneous pronunciations, diminishing the reliability of the processed transcripts.

Given these limitations, there is a clear need for a robust and efficient normalization pipeline specifically tailored to handle Korean text's unique attributes, especially the frequent occurrence of English words, abbreviations, and numerals within the linguistic context.

B.3 Korean Text Preprocessing in other models

XTTS (Coqui XTTS-v2) XTTS[3] uses a custom multilingual Byte-Pair Encoding (BPE) tokenizer with a vocabulary of 6681 tokens. For Korean (as well as Japanese and Chinese), XTTS applies a special preprocessing step. All Hangul text is romanized (transliterated to Latin script) before tokenization. This is done using the open-source hangul-romanize library, ensuring Korean characters are converted to a Latin alphabet representation (following the standard Korean romanization scheme) prior to encoding. Consequently, XTTS does not feed native Hangul graphemes into the model, nor does it perform a separate G2P conversion to Korean phonemes, instead, the romanized Korean text is tokenized like any other Latin-script input. XTTS's pipeline does not document any unique handling of numerals in Korean. Presumably, numbers are left as Arabic digits in the text and would be processed through the same BPE tokenizer. Embedded English words in a Korean utterance are left unchanged which means code-switched English terms can be tokenized alongside the romanized Korean in a unified token space.

CosyVoice 2 CosyVoice 2[4] is a multilingual streaming TTS model utilizing a large language model (Qwen-2.5-0.5B LLM) as its text encoder. It inherits the LLM's multilingual subword tokenizer, which covers multiple languages. CosyVoice 2 does not perform any Korean-specific grapheme-to-phoneme conversion or transliteration, directly processing Korean Hangul text through the tokenizer. Unlike Chinese, which receives dedicated preprocessing to prevent incorrect subword merges, Korean (alongside English and Japanese) is directly processed by the tokenizer without specific grapheme-to-phoneme conversions or transliterations. Notably, CosyVoice 2 explicitly normalizes numeric tokens in Korean input texts by converting them into corresponding English words (e.g., "44" $\rightarrow$ "forty-four") before tokenization. Embedded English words remain unchanged, directly tokenized into English subwords by the multilingual tokenizer. Consequently, CosyVoice 2 processes Korean text at a combined Hangul grapheme and English-word level, explicit conversion of numeric tokens into appropriate Korean words is essential to maintain naturalness in synthesized speech.

Llasa TTS Llasa TTS is a LLaMA-based speech synthesis model (1B/3B/8B), initially trained primarily on a large bilingual Chinese–English corpus. It leverages the LLaMA text tokenizer for input, inheriting the same SentencePiece BPE vocabulary as the base LLaMA language model. Korean text is also processed through this general tokenization system without language-specific rules. This method tokenizes input without specialized handling such as transliteration or phonemization. Numerals and embedded English tokens are directly tokenized without specific normalization, processed simply as digit sequences or Latin script tokens. A variant of Llasa trained explicitly on Korean datam among other languages has been introduced (llasa-1B-multi-speakers-genshin-zh-en-ja-ko)[5]. Despite the inclusion of Korean speech data in training, the lack of specialized Korean text normalization may limit the model's performance in accurately and naturally synthesizing Korean speech containing numerals and mixed-language tokens.

Zonos TTS Zonos TTS[6]utilizes eSpeak NG, an external grapheme-to-phoneme (G2P) tool, to enable multilingual text processing. This approach allows Zonos to quickly support multiple languages without dedicated language-specific preprocessing or tokenization modules. Specifically, for Korean, eSpeak NG converts Hangul text directly into phonetic sequences represented typically by IPA or X-SAMPA, which are then consumed by the TTS model. However, eSpeak NG demonstrates several significant limitations when handling Korean text, particularly with numerals, units, and mixed-language scenarios.

eSpeak NG struggles with Korean text, particularly in handling units, numerals, and mixed-language expressions. It misreads abbreviations like "mm" as letter-by-letter English, mispronounces numerals after English words (e.g., "GPT-3" as "GPT sam"), and fails to apply context-aware rules for time expressions (e.g., "7 o'clock" as "chil si" instead of "ilgop si").

Due to these limitations, employing eSpeak NG in Korean TTS results in unnatural pronunciation, particularly in sentences combining numerals, units, and English tokens. Hence, achieving high-quality, natural-sounding Korean TTS would necessitate a dedicated, linguistically aware preprocessing strategy that properly addresses these language-specific challenges.

[3] `https://huggingface.co/coqui/XTTS-v2`.

[4] `https://huggingface.co/FunAudioLLM/CosyVoice2-0.5B`

[5] `https://huggingface.co/HKUSTAudio/Llasa-1B-multi-speakers-genshin-zh-en-ja-ko`

[6] `https://github.com/Zyphra/Zonos`

B.4 Coreset Selection

Coreset selection, an approach first highlighted in machine learning by Sener and Savarese for active learning in convolutional neural networks, has recently gained attention in the context of TTS. The underlying goal of coreset selection is to identify a small but representative subset of data, preserving the original dataset's diversity and effectiveness for model training.

In the large-scale corpus WenetSpeech4TTS, data was strategically divided into subsets based on linguistic and acoustic criteria, focusing on diversity in speaker identity, speech styles, and textual content. This allowed the efficient construction of representative subsets, significantly reducing computational resources without compromising model performance.

A recent study introduced a method specifically tailored for TTS datasets. This approach extracts linguistic embeddings using BERT and acoustic embeddings using wav2vec 2.0, alongside speaker embeddings (x-vectors). These embeddings are concatenated into joint feature vectors to measure diversity. The method employs a greedy algorithm to incrementally build a coreset by maximizing the diversity metric, defined by squared Euclidean distances among data points. Experimental evaluations across multiple languages (Japanese, Chinese, English) demonstrated that this diversity-based approach outperformed traditional phoneme-balanced selection in terms of naturalness and intelligibility, even when using significantly reduced dataset sizes.

Further emphasizing the importance of coreset selection, DEFT-UCS leveraged unsupervised clustering methods to efficiently select data subsets without reliance on annotations. By embedding textual data using models like Sentence-T5, and performing K-means clustering, this method sampled "easy" and "hard" examples (based on centroid distances) to construct representative coresets. DEFT-UCS demonstrated that using only 32.5 of the original dataset could achieve comparable or superior performance in fine-tuning language models for various text-generation tasks.

These studies collectively underscore the practicality and effectiveness of coreset selection methods, highlighting their potential to significantly reduce data requirements and computational costs while maintaining or even enhancing model quality. However, these methods, whether relying on simple metric-based clipping or deep learning-based embedding techniques, cannot guarantee the perfect preservation of all linguistic diversity within the selected text subsets.

B.5 Handling Duration

In previous studies, discrepancies between training and inference durations have led to critical issues in text-to-speech (TTS) synthesis. Specifically, acoustic models trained primarily on sentence-level datasets often struggle with length generalization, causing significant deterioration in speech quality when synthesizing long-form content. Such mismatches typically result in pronunciation errors, unnatural prosody, erratic durations, and irregular pitch patterns.

To address these challenges, several methodologies have been proposed. Common approaches include integrating context encoders to enrich acoustic features, employing models like Tacotron2 or FastSpeech2 variants for enhanced length generalization, and introducing pause prediction modules to improve naturalness during inference. However, these solutions have their limitations, such as autoregressive models exhibiting slow inference speeds and context encoders encountering generalization issues, failing to reliably extrapolate durations beyond their training scope.

The recent study introduced techniques like Moving Average Equipped Gated Attention (MEGA), Global-information-enhanced Classification Pause Insertion (GCPI), and context generation through Large Language Models (LLMs) as solutions aimed at mitigating these duration-related challenges.

Nevertheless, despite advancements achieved through these methodologies, the distribution of durations in the training dataset itself remains fundamentally important. Balanced and sufficiently diverse duration coverage within training corpora significantly enhances a model's capability to generalize effectively to varying lengths during inference, complementing existing techniques. Therefore, attention to duration diversity is essential for robust and natural speech synthesis performance in TTS applications.

C Text Categorization and Filtering: LNCat

To effectively handle the frequent occurrence of English tokens, numerals, and foreign language elements within Korean speech datasets, we introduce a dedicated categorization and filtering strategy termed LNCat. This method serves as a critical preprocessing step, distinguishing samples that can reliably be converted into standardized Korean pronunciations from those that cannot. In Korean speech corpora, it is common for utterances to contain English terms, numerals, or mixed-language tokens. Completely discarding all samples with English or numerals can result in a significant loss of valuable semantic and acoustic information, negatively impacting TTS model training. Therefore, distinguishing samples based on their convertibility into standardized Korean pronunciation is essential. The LNCat algorithm achieves this goal by identifying which samples should be retained and which should be discarded.

C.1 LNCat Categorizing Methodology

The LNCat method assigns language-category tags to each utterance based on the linguistic composition of the input text:

- **ko_only** Only Korean Hangul characters
- **ko_en** Korean and English tokens
- **ko_num** Korean and numerals
- **ko_en_num** Korean, English tokens, and numerals
- **ko_jp/ko_zh/ko_other** Korean mixed with Japanese, Chinese, or other languages (discarded)
- **en_only/jp_only/zh_only/other_other/other** Utterances without Korean content (discarded)

Only utterances classified as **ko_only, ko_en, ko_num** and **ko_en_num** are retained. All other categories are discarded from further processing.

C.2 Determining English Convertibility (en_convertable)

Samples tagged as ko_en or ko_en_num undergo an additional evaluation to determine if their embedded English tokens can reliably be converted into Korean pronunciation. We define this convertibility (en_convertable) based on the following conditions applied to each English token in the text:

1. **Units** The token, converted entirely to lowercase, matches a predefined list of standard measurement units (e.g., "kg", "mm", "cm", "ml").
2. **Upper-case abbreviations** Tokens composed entirely of uppercase letters and limited to 4 characters or fewer. This rule accommodates commonly pronounced abbreviations and acronyms such as "CCTV", "BMW", "SKT".
3. **Single-letter tokens** Tokens consisting of a single alphabet character (upper or lower case), commonly used in Korean speech to represent anonymous entities or placeholders (e.g., "A 회사" – "company A").

If any English token in an utterance fails to meet at least one of these three criteria, the sample is flagged as non-convertible (en_convertable = False) and subsequently discarded to prevent confusion or degradation during TTS model training. For example, the utterance "a 지점에서 오늘 3kg 짜리 TV를 샀다." contains tokens that all satisfy the defined convertibility conditions: "a" (single-letter token), "3kg" (recognized measurement unit), and "TV" (uppercase abbreviation). Therefore, this utterance is classified as convertible. Conversely, the utterance "a 지점에서 오늘 3kg 짜리 TV를 샀는데, 너무 awesome했어!" includes the token "awesome," which does not meet any of the established criteria for convertibility. As a result, this second example is categorized as non-convertible and subsequently discarded.

This precise categorization and filtering process ensures we retain as many valuable speech samples as possible without negatively affecting model training quality. Consequently, LNCat significantly improves dataset efficiency and linguistic coherence in the preprocessing pipeline.

D Korean Text Preprocessor: N2gk, N2gk+

This section describes the detailed functionalities implemented in our proposed Korean-specific text preprocessing modules, N2gk (Num2Grapheme Korean) and N2gk+, highlighting their comprehensive handling of numerals, English tokens, and special symbols in Korean TTS datasets.

D.1 Numeral Conversion (N2gk)

The N2gk module robustly addresses the inherent complexity of Korean numeral pronunciation, which arises primarily from two different numeral systems: Sino-Korean (Hanja-based) and native Korean (pure Korean). In Korean speech, a single digit may have multiple valid pronunciations depending on context, unit suffix, and grammatical structures. For instance, the numeral '1' can be pronounced as il (일), hana (하나), or han (한), each preferred in different contexts or when paired with certain units or suffixes. Since using an inappropriate pronunciation can lead to misunderstandings, we prioritize the most common and contextually appropriate pronunciation. To systematically handle these nuances, N2gk categorizes numerals based on linguistic context and unit-specific usage, explicitly addressing the following numeral distinctions and special pronunciation cases:

- **Sino-Korean vs. Native Korean Numerals** Sino-Korean numerals are derived from Hanja (Chinese characters), such as 일 (il, one), 이 (i, two), 삼 (sam, three), and used predominantly in formal contexts, units of measure, and counting larger numbers. Conversely, native Korean numerals (하나 (hana, one), 둘 (dul, two), 셋 (set, three)) are frequently used with certain everyday units, counting objects, and for age expressions. N2gk explicitly handles these two numeral systems by contextually mapping numerals based on the following unit categories:
  1. **Native Korean units** e.g., 명 (myeong, persons), 사람 (saram, people), 마리 (mari, animals), 번째 (beonjjae, ordinal numbers), 시 (si, hours on the clock), 개 (gae, items), 가지 (gaji, kinds or types), 잔 (jan, cups or glasses), 번 (beon, times or occasions), 장 (jang, sheets or pages), 병 (byeong, bottles), 살 (sal, years of age), 연세 (yeonse, age, honorific), etc.
  2. **Sino-Korean units** e.g., 초 (cho, seconds), 분 (bun, minutes), 시간 (sigan, hours, duration), 일 (il, days), 주 (ju, weeks), 월 (wol, months), 년 (nyeon, years), 원 (won, Korean Won), 달러 (dalleo, dollars), kg (kilogram), mm (millimeter), cm (centimeter), °C (degree Celsius), % (percent), 포인트 (pointeu, points), etc.
- **Numeral Reading Mechanism** Korean numerals employ positional reading, similar to English (tens, hundreds, thousands), with units such as 십 (sip, tens), 백 (baek, hundreds), 천 (cheon, thousands), and extend up to larger units like 경 (gyeong). Notably, the numeral '1' in thousands is typically silent (e.g., 1000 is pronounced 천 (cheon), not 일천 (il-cheon)), explicitly accommodated by N2gk.
- **Special Numeral Cases** Certain numerals have irregular pronunciations similar to English numerals (e.g., twelve, fifteen). For example, the number 20 can be pronounced 이십 (i-sip), but more commonly 스물 (seumul) or 스무 (seumu). Months also follow special pronunciations, such as June (6월) as 유월 (yu-wol) rather than 육월 (yuk-wol), and October (10월) as 시월 (si-wol) instead of 십월 (sip-wol). N2gk explicitly handles these irregular cases.
- **Commas and Decimal Points** Numbers with commas (e.g., 1,000,000) are processed by ignoring commas during conversion. Decimal numbers follow English-style pronunciation, using 점 (jeom, point), with each digit pronounced individually (e.g., 3.14 as 삼점일사 (sam-jeom-il-sa)).
- **English Numbers within Korean Text** Numerals following English terms are typically pronounced using English pronunciations. For example, the numeral '3' in 'GPT3' is pronounced 쓰리 (sseuri) rather than the Korean 삼 (sam). N2gk explicitly encodes common patterns.
- **Phone Number Handling** Korean phone numbers (e.g., 010-1234-5678) are explicitly converted into digit-by-digit Korean pronunciation (gong-il-gong il-i-sam-sa o-yuk-chil-pal).

D.2 Extended Functionalities (N2gk+)

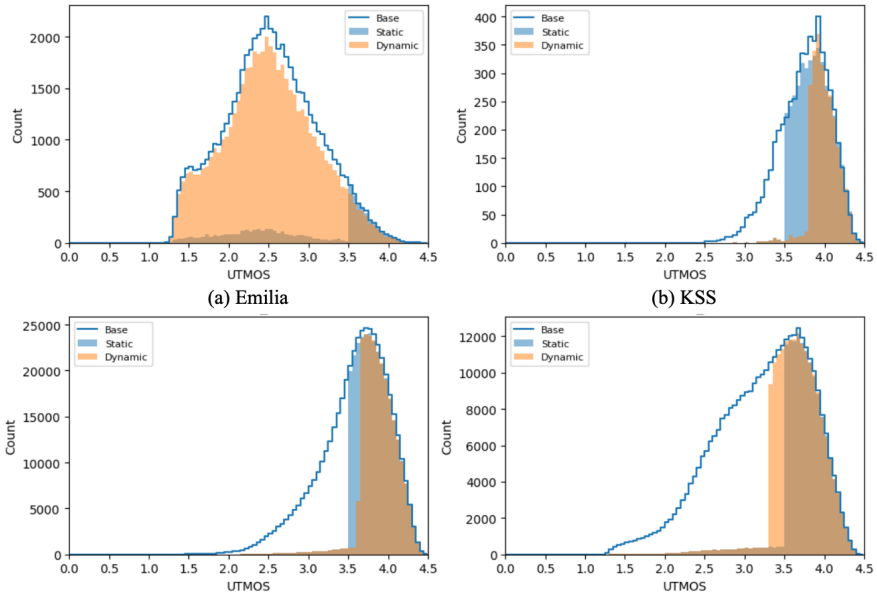
The extended N2gk+ module inherits all functionalities of N2gk and adds robust handling for English tokens, special characters, abbreviations, historical dates, and numeric ranges. Specifically, it incorporates the following capabilities.

- **English to Korean Grapheme Conversion** All English letters are mapped to their Korean phonetic representations, e.g., A (에이, ei), B (비, bi), C (씨, ssi). Common abbreviations, acronyms, and proper nouns (e.g., WHO (더블유에이치오), SKT (에스케이티), BMW (비엠더블유), CCTV (씨씨티비), TV (티비)) are explicitly converted to their standard Korean pronunciations. Established exceptions such as FIFA (피파, pipa) and NASA (나사, nasa) are individually handled due to their common usage in Korean.
- **Special Symbol Handling** Special symbols are systematically converted into their Korean pronunciation equivalents, e.g., % (퍼센트, peosenteu), (앤, aen), $ (달러, dalleo), °C (도씨, dossi), + (플러스, peulleoseu), - (마이너스, mainaseu), (샵, syap), etc.
- **Historical Dates Handling** Historical dates with special pronunciation conventions are explicitly managed. For example, the Korean historical date 3.1절 (March 1st) is correctly converted to 삼일절 (samiljeol) rather than a literal decimal pronunciation 삼점일절 (samjeomiljeol). N2gk+ explicitly recognizes and converts numerals in these historical contexts appropriately.
- **Ranges Handling** The tilde symbol, frequently used to indicate ranges (e.g., 34), is naturally read as "에서" (eseo) in Korean. Moreover, when numerals within a range have a following unit or suffix, both numerals are consistently pronounced using either Sino-Korean or native Korean numerals, depending on context.
- **Removal of Unnecessary Special Characters and Symbols** To improve clarity and consistency, N2gk+ removes extraneous special symbols and handles parentheses by deleting their contents if necessary, ensuring clean textual input for TTS.
- **Single Korean Character Mapping** Individual consonants such as ㄱ, ㄴ, ㄷ, ㅂ, ㅅ, and ㅇ are mapped to their complete Korean pronunciations: ㄱ (기역, giyeok), ㄴ (니은, nieun), ㄷ (디귿, digeut), ㅂ (비읍, bieup), ㅅ (시옷, siot), ㅇ (이응, ieung).
- **Prioritization and Complexity Management** The functionalities mentioned above are deeply intertwined, often overlapping within text inputs. To handle these complexities, we implemented carefully prioritized rules ensuring a coherent and conflict-free text preprocessing pipeline. The prioritization ensures numerals, English tokens, special symbols, and historical events are consistently and accurately transformed according to contextual requirements, maximizing naturalness in synthesized Korean speech.

D.3 Preliminary Experiments for Text Preprocessor Evaluation

Prior to evaluating the effectiveness of our proposed text preprocessor, N2gk+, we conducted preliminary experiments to establish baseline performance of comparative models. Specifically, to evaluate the performance of the Korean grapheme-to-phoneme converter, g2pK, we converted both input texts (containing numerals and English) and corresponding ground-truth (GT) texts (pure Hangul without numerals or English) using g2pK and measured accuracy. Since phoneme outputs from g2pK are highly sensitive to text spacing, we further investigated performance by removing all spaces entirely. This extreme scenario resulted in approximately a two-fold increase in accuracy for numeral-containing texts, highlighting g2pK's instability due to common spacing inconsistencies in real-world Korean texts. However, our reported results are based on original texts with spacing preserved, as completely removing spacing is impractical in realistic scenarios.

Additionally, we conducted experiments with ChatGPT-based prompts, evaluating two distinct types under fully reset memory conditions, one minimal and one detailed. The generic prompt requested ChatGPT to convert numerals and embedded English into their most natural Korean textual forms, providing relevant conversion examples. In contrast, the detailed prompt explicitly included the key mappings and conversion rules used by our proposed N2gk+ algorithm, instructing ChatGPT to strictly follow these specified guidelines. Counterintuitively, the detailed prompt yielded lower performance, we utilized results obtained with the generic prompt for subsequent performance comparisons.



(a) Emilia (b) KSS

(c) CoreaSpeech (Diverse Style Subset) (d) CoreaSpeech (Emotional Subset)

Figure 7: UTMOS threshold comparison (Static vs. Dynamic)

E UTMOS threshold

E.1 Static vs. Dynamic

Figure 7 compares static and dynamic UTMOS thresholding across four subsets of CoreaSpeech. Note that UTMOS thresholding is applied during the selection of Jamo bigrams to preserve rare pairs, which is crucial for phonetic coverage. When a static threshold is used, the model is primarily trained on high-quality samples above a fixed score, which limits its robustness to noisy or low-quality conditions. To address this, we allow the threshold to adapt dynamically to the score distribution, enabling the inclusion of a proportion of lower-quality samples when appropriate.

This approach improves generalization, especially in acoustically diverse environments. Importantly, our current implementation uses only simple summary statistics (mean and median) to determine dataset-specific thresholds. Despite this simplicity, we observe consistent improvements. Eventually, each dataset,Emilia, KSS, CoreaSpeech (Diverse Style subset), and CoreaSpeech (Emotional subset), received a dynamically computed UTMOS threshold of 1.040, 3.807, 3.538, and 3.305, respectively, based on their individual score distributions. In future work, we plan to investigate more sophisticated thresholding strategies by incorporating additional distributional features to better tailor the threshold to each dataset.

E.2 Implementation

To determine a threshold τ for filtering based on UTMOS scores, we considered multiple strategies. While the standard approach $\tau = \mu - x\sigma$ is simple, it is sensitive to outliers and inappropriate for skewed distributions. Since many speech datasets, including ours, are recorded under consistent conditions or extracted from homogeneous sources, their UTMOS distributions tend to be skewed. Therefore, we adopted a robust alternative using the median absolute deviation (MAD):

$$\tau = \text{median} - k \cdot \text{MAD} \tag{2}$$

To further adapt the strictness of filtering based on the overall dataset quality, we dynamically adjust the coefficient k depending on the deviation of the mean μ from a reference mean μ_{ref}:

$$k = \max(k_{\min}, k_{\max} \cdot \frac{\mu}{\mu_{\text{ref}}}) \tag{3}$$

F Data Appending

F.1 Implementation

This section describes the detailed implementation of our proposed data appending strategy employed to address the prevalent short-duration problem in speech datasets. Notably, our method is designed under realistic conditions where the number of speakers, the number of available samples per speaker, and individual utterance durations are completely unknown in advance. We further split speakers by annotated emotion and style before concatenation

Still, the key idea is to strategically concatenate shorter utterances from the same speaker, thereby achieving balanced coverage across a wider range of utterance durations (0-30 seconds), as illustrated at Figure 4.

Given a set of speech utterances $U = \{u_i\}_{i=1}^{N}$, each utterance u_i has an associated duration $d(u_i)$, speaker identity $s(u_i)$, and corresponding text transcript. The goal is to generate a new set of augmented utterances U', such that each augmented utterance $u \in U'$ satisfies:

$$0 < d\left(u'\right) \leq 30 \text{ seconds, and } s\left(u'\right) \text{ remains unchanged.} \tag{4}$$

We first organize utterances by speaker identity to maintain speaker consistency within augmented samples. For each speaker s_j:

$$U_{s_j} = \{u \in U \mid s(u) = s_j\} \tag{5}$$

Only utterances whose duration is less than or equal to the maximum allowed duration (30 seconds) are selected for further processing:

$$U_{s_j}^{\text{filtered}} = \left\{u \in U_{s_j} \mid d(u) \leq 30\right\} \tag{6}$$

To achieve balanced coverage of durations between 1 to 30 seconds, we introduce a probabilistic selection mechanism. Specifically, we define buckets B_i representing integer durations from 0 to 29 seconds. We calculate bucket frequencies based on two distributions:

- Global bucket frequency $n_{bucket}(i)$, representing the total count of samples across all speakers.
- Speaker-specific bucket frequency $m_{bucket}(i, s_j)$, representing the count of samples within a specific speaker s_j.

The weight for selecting k-utterances (k in seconds) for concatenation from a particular speaker s_j is computed as follows:

$$w_{\text{bucket}}\ (i, s_j) = \frac{(N_{\text{total}}\ - n_{\text{bucket}}\ (i))}{N_{\text{total}}} \cdot (1 - \frac{1}{S_{\text{total}}}) + \frac{(M_{\text{total}}\ (s_j) - m_{\text{bucket}}\ (i, s_j))}{M_{\text{total}}\ (s_j)} \cdot \frac{1}{S_{\text{total}}} \tag{7}$$

where $N_{total} = \sum_{i=0}^{29} n_{bucket}(i)$ is the total number of utterances across all speakers, $M_{total}(s_j) = \sum_{i=0}^{29} m_{bucket}(i, s_j)$ is the total number of utterances for speaker s_j, and S_{total} is the total number of remaining speakers yet to be processed.

Notably, the term S_{total} (the number of remaining speakers) serves as a dynamic balancing factor. Initially, when many speakers remain, the weight emphasizes speaker-specific diversity, ensuring each speaker's samples cover a wide range of durations. However, as the process continues and fewer speakers remain, the influence of global distribution increases, prompting a globally uniform duration distribution across the dataset. This mechanism addresses two practical challenges:

- Relying solely on global distribution might lead individual speaker's samples to cluster in specific buckets, limiting intra-speaker variety
- Conversely, relying only on speaker specific distributions might result in global imbalances.

To ensure no weight becomes zero, we enforce a minimum weight threshold ϵ:

$$w_{\text{bucket}}\ (i, s_j) = \max\left\{w_{\text{bucket}}\ (i, s_j), \epsilon\right\}, \quad \epsilon \text{ is a small constant } \left(\text{ e.g., } 10^{-6}\right) \tag{8}$$

We further define selection weights according to an expected number of utterances γ:

$$\gamma = \frac{D_{\max}}{d_{\text{avg}}\ (s_j)}, \quad \text{where } d_{\text{avg}}\ (s_j) = \frac{\sum_{u \in U_{s_j}^{\text{filtered}}} d(u)}{\left| U_{s_j}^{\text{filtered}} \right|} \tag{9}$$

Selection weights for sampling $k \leq \lfloor \gamma \rfloor$ utterances are then :

$$W\left(k, s_j\right) = w_{bucket}\left(\min\left(k \cdot \frac{30}{\lfloor \gamma \rfloor}, 29\right), s_j\right) \tag{10}$$

We randomly sample utterances according to these computed weights and concatenate them into a single audio segment per augmented utterance. Concatenation employs a cross-fade window of 0.5 seconds to avoid abrupt spectral discontinuities:

Given audio segments $x_1, x_2, ..., x_k$:

$$u' = x_1 \oplus_{\text{fade}}\ x_2 \oplus_{\text{fade}}\ \cdots \oplus_{\text{fade}}\ x_k \tag{11}$$

where the cross-fade concatenation $(\oplus_{fade})$ operation between two segments x_a, x_b is:

$$x_a \oplus_{\text{fade}}\ x_b = \text{FadeOut}\left(x_a, 0.5s\right) + \text{FadeIn}\left(x_b, 0.5s\right) \tag{12}$$

This ensures natural acoustic continuity and spectral smoothness within concatenated utterances.

To prevent any duration bucket from becoming excessively dominant, we impose a dynamic threshold $m_{threshold}(s_j)$:

$$n_{\text{threshold}}\ (s_j) = \left\lfloor \frac{\left| U_{s_j}^{\text{filtered}} \right|}{\sum_1^{30} t} \right\rfloor + 1 \tag{13}$$

If the frequency of a specific duration bucket exceeds $m_{threshold}(s_j)$, we temporarily avoid adding further utterances into that bucket for the current speaker, ensuring balanced and even duration distribution across the dataset.

F.2 Limitation

Our data appending method, while effective, has several limitations. Utterances were concatenated only when style and emotion annotations matched within the same speaker. However, since the concatenation process was random, the resulting sequences did not always maintain semantic coherence, and punctuation such as sentence-ending periods occasionally caused unnatural transitions.

When style or emotion annotations are unavailable, it becomes more difficult to preserve acoustic consistency. This highlights the need for future work that can ensure semantic and acoustic alignment without relying on explicit annotations. Although the method was introduced to address hallucinations caused by discrepancies between training and inference durations, it did not fully resolve the issues, indicating that more advanced solutions are necessary for stable long-form generation.

Table 8: Performance comparison across pipeline components on Clean Data

Method	Data Cond.		Coreset Selection		Suppl. Final.	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Duration App.				
GT	–	–	–	–	–	3.140	9.463	0.834	3.604
Base	✗	✗	✗	✗	✗	3.824	11.264	0.764	3.396
Base-diar.	✓	✗	✗	✗	✗	3.804	11.240	0.754	3.408
Norm	✓	✓	✗	✗	✗	3.290	9.740	0.807	3.394
Norm$_{static}$	✓	✓	✗	✓	✗	4.168	12.231	0.803	3.407
Norm-app	✓	✓	✗	✗	✓	2.905	8.645	0.800	3.343
Core-app	✓	✓	✓	✗	✓	2.806	8.260	0.801	3.302
Core-app$_{static}$	✓	✓	✓	✓	✓	2.779	8.022	0.801	3.380
CoreaSpeech	✓	✓	✓	Dynamic	✓	2.373	6.970	0.803	3.378

Table 9: Performance comparison across pipeline components on Noisy Data

Method	Data Cond.		Coreset Selection		Suppl. Final.	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Data App.				
GT	–	–	–	–	–	9.213	22.640	0.761	1.657
Base	✗	✗	✗	✗	✗	12.459	30.830	0.661	2.056
Base-diar.	✓	✗	✗	✗	✗	12.629	31.400	0.652	2.091
Norm	✓	✓	✗	✗	✗	12.347	30.869	0.657	2.070
Norm$_{static}$	✓	✓	✗	✓	✗	23.080	59.417	0.671	2.041
Norm-app	✓	✓	✗	✗	✓	9.521	23.470	0.609	2.157
Core-app	✓	✓	✓	✗	✓	8.916	22.874	0.645	2.131
Core-app$_{static}$	✓	✓	✓	✓	✓	9.460	23.295	0.652	2.020
CoreaSpeech	✓	✓	✓	Dynamic	✓	8.259	20.618	0.673	2.131

Table 10: Performance comparison across pipeline components on Numeric Data

Method	Data Cond.		Coreset Selection		Suppl. Final	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Data App.				
GT	–	–	–	–	–	6.236	18.936	0.854	3.427
Base	✗	✗	✗	✗	✗	11.790	38.407	0.682	2.954
Base-diar.	✓	✗	✗	✗	✗	11.560	37.818	0.677	2.918
Norm	✓	✓	✗	✗	✗	7.526	24.940	0.824	2.966
Norm$_{static}$	✓	✓	✗	✓	✗	13.450	44.161	0.820	2.998
Norm-app	✓	✓	✗	✗	✓	5.812	19.469	0.817	2.923
Core-app	✓	✓	✓	✗	✓	6.239	20.697	0.818	2.954
Core-app$_{static}$	✓	✓	✓	✓	✓	5.856	19.442	0.816	2.972
CoreaSpeech	✓	✓	✓	Dynamic	✓	5.973	19.716	0.818	2.903

G Korean Testset

G.1 Detailed evaluation for Korean Universal subsets

Tables 8, 9, and 10 present detailed evaluation metrics for subsets of the Korean Universal Testset, specifically categorized into Clean, Noisy, and Numeric testsets.

In the Clean and Noisy sets (Tables 8 and 9), one notable metric is the performance of Norm$_{static}$, which uses a fixed UTMOS threshold (set to 3.5) for data filtering. While this static threshold approach resulted in a moderate increase in WER and CER on the Clean set compared to Norm, it caused a nearly two-fold increase in WER on the Noisy set. This substantial performance drop suggests that the Norm$_{static}$ method significantly reduced speaker diversity and text coverage, indicating limited robustness and adaptability to noisy prompts. On the Numeric set (Table 10), the introduction of our proposed N2gk+ preprocessor resulted in substantial improvements in WER starting from the Norm subset. However, consistent with findings from the Noisy set, the application of a static UTMOS threshold (Norm$_{static}$) again notably degraded WER performance. Overall, the inclusion of data appending methods showed minor but consistent WER improvements, indicating their positive impact on longer utterances, which constitute a relatively small portion of the dataset. Furthermore, the difference in performance between Norm$_{static}$ and Core-app$_{static}$ highlights the effectiveness of preserving Jamo-pair diversity.

Table 11: Performance comparison across pipeline components on Emilia Data

Method	Data Cond		Coreset Selection		Suppl. Final.	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Data App..				
GT	–	–	–	–	–	10.306	26.509	0.774	2.442
Base	✗	✗	✗	✗	✗	9.220	23.505	0.751	2.422
Base-diar.	✓	✗	✗	✗	✗	9.015	23.481	0.756	2.436
Norm	✓	✓	✗	✗	✗	8.887	23.546	0.757	2.436
Norm$_{static}$	✓	✓	✗	✓	✗	19.822	54.399	0.736	2.413
Norm-app	✓	✓	✗	✗	✓	9.187	24.908	0.729	2.413
Core-app	✓	✓	✓	✗	✓	7.871	20.318	0.752	2.404
Core-app$_{static}$	✓	✓	✓	✓	✓	7.980	20.496	0.737	2.422
CoreaSpeech	✓	✓	✓	Dynamic	✓	7.156	18.805	0.757	2.449

Table 12: Performance comparison across pipeline components on KSS Data

Method	Data Cond.		Coreset Selection		Suppl. Final.	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
	Diar.	N2gk+	Jamo Bi.	UTMOS	Data App.				
GT	–	–	–	–	–	2.558	6.219	0.645	3.777
Base	✗	✗	✗	✗	✗	4.319	10.051	0.652	3.253
Base-diar.	✓	✗	✗	✗	✗	3.778	10.372	0.652	3.176
Norm	✓	✓	✗	✗	✗	3.380	8.094	0.652	3.321
Norm$_{static}$	✓	✓	✗	✓	✗	2.533	6.177	0.649	3.399
Norm-app	✓	✓	✗	✗	✓	2.555	6.266	0.638	3.234
Core-app	✓	✓	✓	✗	✓	2.128	5.041	0.646	3.383
Core-app$_{static}$	✓	✓	✓	✓	✓	1.573	3.906	0.640	3.336
CoreaSpeech	✓	✓	✓	Dynamic	✓	2.039	4.958	0.647	3.365

Table 13: Korean open source testset: Note that the Emilia dataset was already included in the training of the LLaSA model, whereas speakers in this testset were excluded from our model's training. For the KSS dataset, the corresponding speaker was included in our model's training.

Eval Set	**Model**	**CER** (↓)	**WER** (↓)	**SIM** (↑)	**UTMOS** (↑)
Emilia (KO)	GT	10.306	26.059	0.774	2.442
	XTTS	32.326	62.795	0.629	2.250
	CosyVoice 2	**6.083**	**16.657**	0.749	**3.012**
	Llasa	24.985	70.169	0.497	2.757
	Zonos	7.737	20.922	0.718	2.919
	PEFT-TTS (Ours)	7.156	18.805	**0.757**	2.449
KSS	GT	8.272	2.558	0.645	3.777
	XTTS	109.281	274.108	0.578	2.413
	CosyVoice 2	3.443	9.241	0.610	3.506
	Llasa	5.525	14.562	0.496	**3.635**
	Zonos	5.759	14.194	0.596	3.557
	PEFT-TTS (Ours)	**2.039**	**4.958**	**0.647**	3.365

G.2 Korean Open-source Testset

Tables 11 and 12 report metrics for the Emilia and KSS data, respectively. The Emilia and KSS datasets were intentionally excluded from the Korean Universal Testset due to the high likelihood that existing TTS models have trained on these datasets. For fair evaluation, our model specifically excluded all samples from both Emilia and KSS datasets during training. Consequently, the Emilia testset represents unseen speaker validation, whereas the KSS testset, being a single-speaker dataset, represents seen speaker validation. Table 13 provides a benchmarking comparison of our model against other models using these two testsets, with the important note that each model differs in terms of its training exposure to the Emilia and KSS datasets.

Table 14: Ablation: LoRA rank

DiT rank	#Trainable Params.	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
GT	-	7.220	15.180	0.786	2.087
16	5.81M	4.957	12.838	0.720	2.831
64	10.10M	4.844	13.405	0.736	2.783

Table 15: Ablation: Jamo extraction coefficient (β) with threshold (t) as 500 - Korean Universal Testset : †denotes that Gini coefficient computed for Jamo bigram counts over 1000, while unmarked Gini coefficient is computed over threshold (t).

Jamo Sel. (β)	Total Duration	Gini coefficient	Gini coefficient†	CER (↓)	WER (↓)	SIM (↑)	UTMOS (↑)
GT	-	-	-	7.023	15.931	0.773	2.981
random	100	0.7701	0.7539	7.173	21.384	0.763	2.784
0.01	68.58	0.5965	0.5347	6.515	18.491	0.766	2.781
0.001	214.53	0.7258	0.6348	6.146	17.394	0.759	2.809
0.0001	1249.86	0.7240	0.6947	5.631	15.945	0.762	2.795
0	1871.21	0.7724	0.7528	6.562	18.645	0.752	2.813

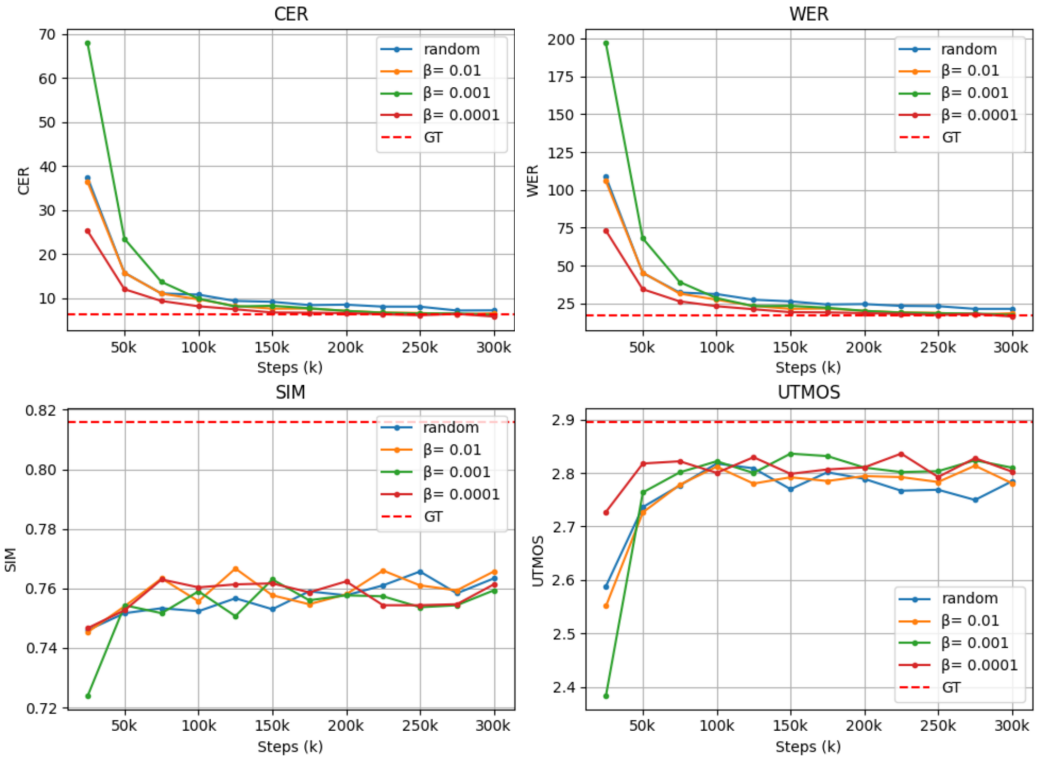


Figure 8: Evaluation metrics (CER, WER, SIM, and UTMOS) on the Korean Universal Testset up to 300k steps, according to different values of the compression factor (β).

H Ablation

H.1 LoRA rank effectiveness depends on training data size

Table 14 shows that increasing the LoRA rank of the DiT block from 16 to 64 yields virtually no difference across all objective metrics. Although the rank of 64 model tends to capture phonological characteristics slightly faster than rank of 16 in fewer epochs, after sufficient training, both models ultimately exhibit nearly identical performance. In other words, the improvement achieved by increasing adapter size is marginal and falls within the typical variation observed across runs, despite quadrupling the number of trainable parameters. Therefore, we selected the rank of 16 configuration for the final model due to its higher parameter efficiency.

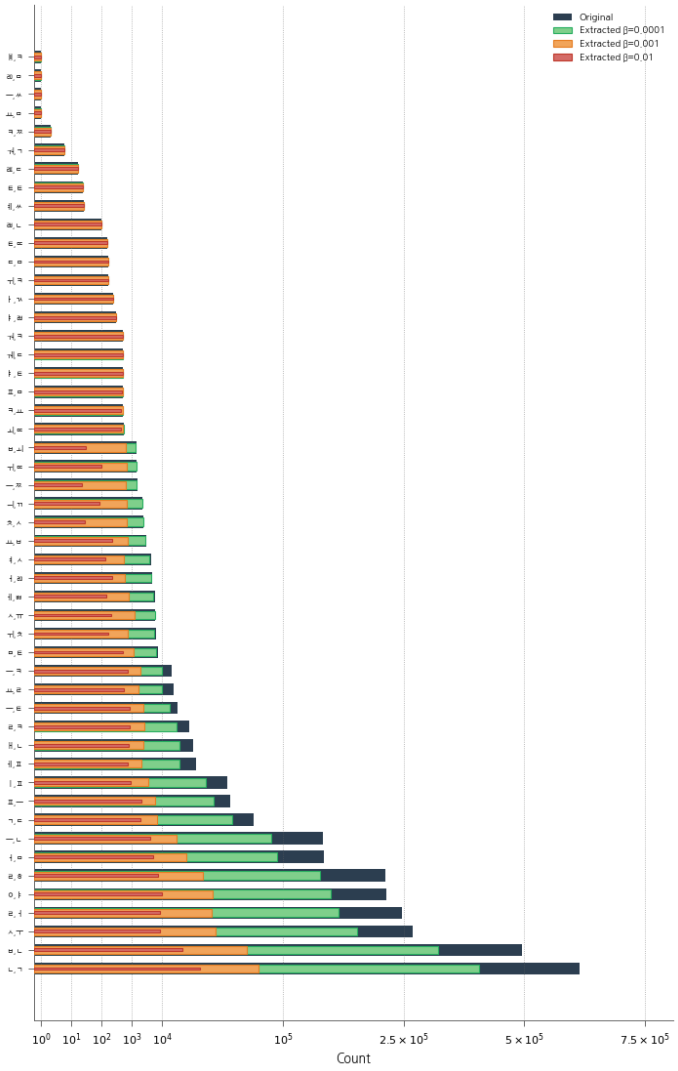


Figure 9: Distribution of surviving Jamo–bigram counts after probabilistic pruning with different coefficients β.

H.2 Jamo extraction coefficient

Table 15 and Figure 8 present the effects of varying the Jamo extraction coefficient (β) compared to a random data selection baseline on the Korean Universal Testset. We experimented with three β values (0.01, 0.001, 0.0001) to identify the optimal setting, while t was driven by the lowest-frequency 25% of types of Jamo pairs, collectively representing only 0.05% of the total frequency.

The distributional imbalance was quantified using the Gini coefficient, calculated at two thresholds (500, used during extraction, and 1000). Random selection resulted in Gini coefficients nearly identical to the unfiltered set, showing limited capability to address imbalance, reflected by higher WER and CER metrics as shown in Figure 8).

Higher β values removed more data, lowering the Gini coefficient (particularly notable for bigram counts above 1000) thus reducing imbalance while preserving linguistic (Jamo-pair) coverage. However, despite similar Gini coefficients, $\beta = 0.001$ and $\beta = 0.0001$ exhibited noticeable performance differences, indicating the significant influence of total data duration. Ultimately, the best performance was achieved with $\beta = 0.0001$, effectively balancing linguistic coverage and dataset size.

Additionally, Figure 9 illustrates the actual distribution of randomly selected 30 Jamo pairs for each β value, visually confirming the impact of different extraction coefficients on pair distributions.

I License

Table 16: Public speech corpora mentioned in this study and their licenses.

Dataset	Languages	License
LJSpeech	English	Public Domain
LibriTTS	English	CC BY 4.0
LibriSpeech	English	CC BY 4.0
WenetSpeech4TTS	Mandarin	CC BY 4.0
Emilia	Multilingual	CC BY-NC 4.0
Common Voice	133 languages	CC0 1.0
IndicVoices–R	22 Indian languages	CC BY 4.0
KSS (Korean SS)	Korean	CC BY-NC-SA 4.0

Table 17: External pretrained models and tools employed in this work.

Model / Tool	License
F5-TTS (base model)	MIT (code) / CC BY-NC 4.0 (models)
Coqui XTTS v2	Mozilla Public License 2.0 (code) /Coqui Public Model License 1.0.0 (models)
CosyVoice 2	Apache-2.0
Llasa	CC BY-NC 4.0
Zonos	Apache-2.0
Whisper-Large v3	MIT
WavLM (Base / Large)	MIT
pyannote-diarization 3.1	MIT
ko-sbert-s	Apache-2.0
g2pk	Apache-2.0
KoNLPy	GPL-3.0-or-later

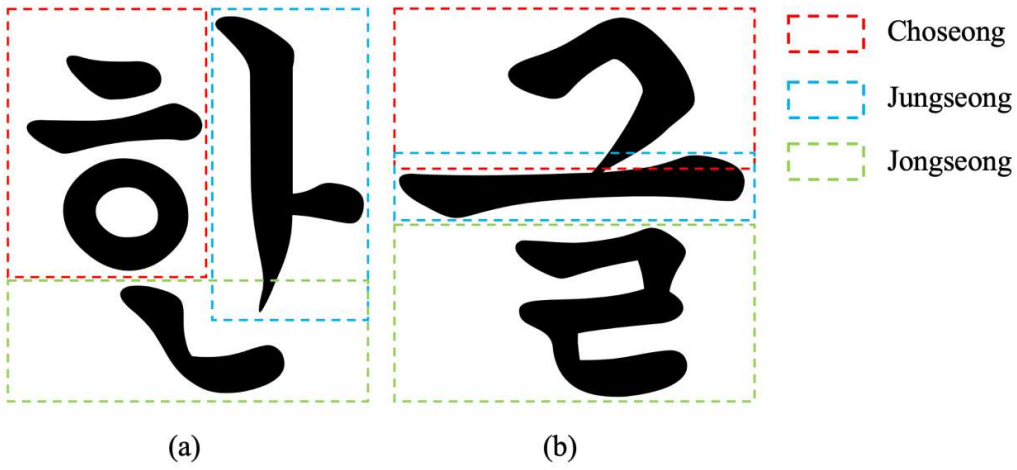


Figure 10: (a) Decomposition of the first syllable *han* (한), (b) decomposition of the second syllable *geul* (글)

J Jamo

Korean Hangul is an alphabetic writing system where letters called Jamo are grouped into blocks representing syllables. Each Hangul syllable block is composed of at least two Jamo, an initial consonant (also called *choseong*, and a medial vowel (*jungseong*), and optionally a final consonant (*jonseong*) at the end. For example, as illustrated in Figure 10 (a), the syllable "한" is written as a single block but consists of the Jamo 'ㅎ' (initial consonant *h*), 'ㅏ' (medial vowel *a*), and 'ㄴ' (final consonant *n*). In total, Korean has 19 possible initial consonants, 21 vowels, and 28 possible final consonant positions. This combinatorial design yields $19 \times 21 \times 28 = 11,172$ theoretically possible distinct syllable blocks in Hangul.

Initial consonants include distinctive tense or fortis forms (ㄲ, ㄸ, ㅃ, ㅆ, ㅉ), which are articulated with increased muscular tension and no aspiration, clearly distinguishing them from aspirated counterparts (e.g., tense ㄸ [t] versus aspirated ㅌ [t]). Notably, the consonant ㄹ varies contextually between a flap [r] and a lateral [l], and ㅇ uniquely remains silent at syllable onset but is pronounced [ŋ] in the final position. Medial vowels encompass 21 distinct forms, consisting of 10 monophthongs and 11 diphthongs or vowel combinations, including glide prefixes such as y- (ㅑ, ㅕ,ㅛ,ㅠ, ㅒ, ㅖ) and w- (ㅘ,ㅙ,ㅝ,ㅞ,ㅟ), along with the distinctive vowel ㅡ and the compound vowel ㅢ. Every syllable block contains exactly one medial vowel. The final consonant (batchim) can appear in 27 forms, including consonant clusters, or 28 if counting syllables without a final consonant. However, actual pronunciation often differs from the written form; in practice, only seven consonant sounds ([ㄱ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅇ]) are realized in final positions. Specifically, final consonants ㅂ and ㅍ are pronounced as [ㅂ], ㄷ, ㅌ, ㅅ, ㅆ, ㅈ, ㅊ, and ㅎ are all pronounced as [ㄷ], and ㄱ, ㄲ, ㅋ as [ㄱ], while ㅁ, ㄴ, ㅇ, and ㄹ maintain their respective pronunciations.

For a complete list of Jamo for each initial (Choseong), medial (Jungseong), and final (Jongseong) element, their International Phonetic Alphabet (IPA) transcriptions, and other details, see Tables 18, 19, and 20.

Table 18: Initial consonants (Choseong)

Hangul	Name	IPA	Notes
ㄱ	giyeok	[g] or [k]	-
ㄴ	nieun	[n]	-
ㄷ	digeut	[t] or [d]	-
ㄹ	rieul	[l] or [r]	-
ㅁ	mieum	[m]	-
ㅂ	bieup	[p] or [b]	-
ㅅ	siot	[s]; [ɕ] before [i]	(soft s; before [i] it sounds like [ɕ])
ㅇ	ieung	silent	silent when used as an initial
ㅈ	jieut	[j] or [tɕ]	-
ㅊ	chieut	[ch] or [tɕ^h]	Aspirated
ㅋ	kieuk	[k] or [k^h]	Aspirated
ㅌ	tieut	[t] or [t^h]	Aspirated
ㅍ	pieup	[p] or [p^h]	Aspirated (typically [p^h])
ㅎ	hieut	[h]	-
Tense (Fortis) Consonants: higher tension, no aspiration			
ㄲ	ssang-giyeok	[k’] (tense [k])	Stiff ’k’ sound
ㄸ	ssang-digeut	[t’] (tense [t])	Stiff ’t’ sound
ㅃ	ssang-bieup	[p’] (tense [p])	Stiff, unaspirated ’p’
ㅆ	ssang-siot	[s’] (tense [s])	Forced ’s’ sound
ㅉ	ssang-jieut	[tɕ’] (tense [tɕ])	Stiff ’jj’ sound

Table 19: Medial vowels (Jungseong)

Hangul	Name	IPA	Notes	Hangul	Name	IPA	Notes
ㅏ	a	[a]	-	ㅜ	u	[u]	-
ㅐ	ae	[ɛ]	-	ㅠ	yu	[ju]	-
ㅑ	ya	[ja]	-	ㅡ	eu	[ɯ]	Unrounded (’ugh’-like)
ㅒ	yae	[jɛ]	-	ㅣ	i	[i]	-
ㅓ	eo	[ʌ] or [ɔ]	-	ㅘ	wa	[wa]	-
ㅔ	e	[e]	-	ㅙ	wae	[wɛ]	-
ㅕ	yeo	[jʌ]	-	ㅚ	oe	[we] or [ø]	-
ㅖ	ye	[je]	-	ㅝ	wo	[wʌ]	-
ㅗ	o	[o]	-	ㅞ	we	[we]	-
ㅛ	yo	[jo]	-	ㅟ	wi	[wi]	-
				ㅢ	ui	[ɯi] or [i]	Context-dependent

Table 20: Final consonants (Jongseong)

Hangul	Name	IPA	Hangul	Name	IPA	Hangul	Name	IPA
ㄷ	digeut	[t̚]	ㄱ	giyeok	[k̚]	ㄹ	rieul	[l]
ㅅ	siot	[t̚]	ㅋ	kieuk	[k̚]	ㄾ	rieul-bieup	[l]
ㅈ	jieut	[t̚]	ㄳ	giyeok-siot	[k̚]	ㄽ	rieul-siot	[l]
ㅊ	chieut	[t̚]	ㄺ	rieul-giyeok	[k̚]	ㄼ	rieul-tieut	[l]
ㅌ	tieut	[t̚]	ㄲ	ssang-giyeok	[k̚]	ㅀ	rieul-hieut	[l]
ㅎ	hieut	[t̚]	ㅂ	bieup	[p̚]	ㄴ	nieun	[n]
ㅆ	ssang-siot	[t̚]	ㅍ	pieup	[p̚]	ㄵ	nieun-jieut	[n]
ㅁ	mieum	[m]	ㄿ	rieul-pieup	[p̚]	ㄶ	nieun-hieut	[n]
ㄻ	rieul-mieum	[m]	ㅄ	bieup-siot	[p̚]	ㅇ	ieung	[ŋ]

J.1 Decomposition of Hangul and representations

Every Korean syllable is structured as initial consonant + Vowel + optional final consonant in that order. Decomposing Hangul into Jamo thus yields the basic phonemic letters that make up the syllables. For example, word '한글' (Figure 10) can be decomposed in Jamo sequence as ㅎ, ㅏ, ㄴ, ㄱ, ㅡ, ㄹ representing h a n g u l. This shows how any Hangul text can be broken down into a linear sequence of consonant and vowel symbols.

In Unicode, there are two ways to represent Hangul text, either as precomposed syllable codepoints (one per block), or as a sequence of Jamo characters. Jamo have their own Unicode ranges (U+1100-U+11FF for initial medial letters, and U+11XX for final variants, or the Hangul Compatibility Jamo U+3130-U+318F). In the Hangul Jamo block, the initial versus final forms of the same letter are encoded separately (e.g., an initial ㄱ and a final ㄱ are different code points). This allows explicit distinction of position in text encoding.

For training a TTS model, we conceptually treat each Jamo as a distinct token in sequence which model doesn't necessarily need to distinguish by code point if we ensure the sequence ordering. The key is that any Korean syllable can be represented by at most three Jamo tokens, rather than as a single indivisible unit.

J.2 Rationale for Jamo level Representation in TTS

Using Jamo as the basic input units for a Korean TTS model offers several linguistic and practical advantages over using whole syllable blocks or word-level units:

Phonetic Granularity Each Jamo roughly corresponds to a phonetic sound. By decomposing text into Jamo, we supply the TTS model with a sequence of units that are very close to the actual pronunciation. Essentially, Jamo are an alphabetic phonemic script. This means a TTS system can learn the mapping from letters to audio without an intermediate phonemic transcription, because Hangul "sounds as it is written" in most cases. In contrast, treating each syllable block as an atomic unit would obscure the internal phonetic structure. For instance, the model would treat "가" and "구" as completely unrelated symbols, even though they share the consonant ㄱ and only differ in the vowel. Jamo-level input makes such relationships explicit – the model sees that "가" (ㄱ+ ㅏ) and "구" (ㄱ+ㅜ) share the letter ㄱ, so it can more easily generalize the pronunciation of ㄱ across different vowels.

Open Vocabulary & Generalization Building on the above point, Jamo decomposition guarantees an open vocabulary system. Any Korean word, even if OOV (out-of-vocabulary) for the training corpus, can be synthesized as long as its letters have been seen. This is crucial because Korean is an agglutinative language with a very large lexicon. Fore perspective, an authoritative Korean dictionary contains over 1.1 million unique words (counting distinct headword entries). It is impractical to include or explicitly model all words in a TTS system. Instead, by training at the letter level, the model can construct new words from known pieces. Jamo sequences cover every possible syllable that can be formed in Korean, so the TTS can potentially read any word or name, even unfamiliar ones, by combining the learned pronunciations of the constituent Jamo. This compositional generalization is one the core motivations for using Jamo units. In contrast, a syllable level model could fail on a syllable it never saw during training. A word level model would be even more brittle. It would only know the specific words seen in training and would treat an unknown word as entirely out-of-vocabulary. Jamo level input elegantly avoids this issue.

Morphological Transparency Korean orthography often preserves morphological structure even when pronunciation changes. By using Jamo directly, the model can implicitly leverage morphology. For example, the verb ending -습니다 is composed of s + eu + p + ni + da (스+ㅂ니다) but pronounced "씀니다" [s^{h}ʌmnida] due to assimilation. A Jamo based model sees the underlying letters ㅅ, ㅡ, ㅂ, ㄴ, ㅣ, ㄷ, ㅏ and can learn the pattern that ㅂㄴ of 스+ㅂㄴ yields a [mn] sound in that context. Meanwhile, it still recognizes this sequence as the polite verb ending (common morpheme) and might generalize timing or intonation for it. If we had converted everything to surface phonemes, some of that orthographic morphological clue would be lost. Essentially, Jamo input can carry latent morphological information, which might help the model produce more consistent pronunciation for inflections or particle attachments and handle systemic sound changes as they occur across morpheme boundaries.

Simpler Text Processing Pipeline Using Jamo directly means we rely on the model to learn Korean pronunciation rules, rather than hand-coding them. Korean has a well-understood set of pronunciation adjustment rules (liaison, nasalization, tensification, etc., discussed below). A common alternative approach is to perform a grapheme-to-phoneme (G2P) conversion with a tool like G2pK before feeding to the TTS. G2p will output a phonetic transcription (often still Hangul in IPA) that reflects how the word should be pronounced in standard Korean. While effective, this adds complexity: it requires maintaining a dictionary or rule engine and handling exceptions. A Jamo based end-to-end model, on the other hand, can learn these context-dependent pronunciation from data. It has been observed that modern neural TTS models are capable of learning the pronunciation rules of phonetic writing systems without an explicit G2P step, especially if the orthography is mostly regular. Hangul is largely phonemic, so many TTS systems skip an explicit phoneme conversion. Eliminating the G2P step reduces potential error propagation and avoids sensitivity to spacing variations, which can notably influence pronunciation due to phonological phenomena in Korean.

Memory Efficient With Jamo, because the total number of symbols is small, the model's embedding layer is much more compact. For example, 68 symbols versus 11,172 means nearly 165 times fewer embedding vectors. This not only saves memory but can also speed up training and inference due to fewer distinct classes. A smaller vocabulary might also require less data to adequately learn each symbol's usage. The trade-off is that the sequence length is longer (each syllable becomes 2-3 tokens instead of 1). However, modern sequence-to-sequence models (like Transformers) can handle the increased length easily since Korean words are not extremely long (and the increase is linear). The benefit of a compact, fully covered vocabulary often outweighs the cost of slightly longer sequences.

J.3 Phonological Phenomena in Korean

Phonological phenomena in Korean can be broadly categorized into the following types:

- **Liaison (Cases: Common)** Final consonant moves to the next syllable if it begins with a vowel. E.g., 맛있어요 (mat-it-seo-yo) is pronounced [마시써요] (ma.ɕisʌ.jo).
- **Final Consonant Neutralization (Cases: 7 sounds)** Final consonant in Korean are simplified into one of seven consonant sounds: [ㄱ (kˀ), ㄴ (n), ㄷ (tˀ), ㄹ (l) , ㅁ (m), ㅂ (pˀ), ㅇ]. E.g., 옷 (ot) is pronounced [온] (otˀ), and 밥 (bab) is pronounced [밥] (papˀ).
- **Nasalization (Cases: 3 types)** When followed by nasal consonants, final obstruents (ㄱ , ㄷ , ㅂ) become nasalized (ㅇ , ㄴ , ㅁ respectively). E.g., 앞문 (ap-mun) is pronounced [암문] (am.mun)
- **Lateralization (Cases: 2 types)** When ㄴ and ㄹ meet across syllable boundaries, they merge into a double [ㄹ ㄹ] ([ll]) sound. E.g., 신라 (sin-la) is pronounced [실라] (sil.la).
- **/n/-Insertion (Cases: Common in compounds)** An [ㄴ] sound is inserted between consonant-final syllables and vowel-initial syllables starting with /i/ or glide /j/. E.g., 꽃잎 (kkot-ip) is pronounced [꼰닙] (kon.nip).
- **Aspiration and ㅎ Assimilation (Cases: 2 main types)** Final consonant ㅎ aspirates following consonants ㄱ , ㄷ , ㅂ , ㅈ , turning them into their aspirated counterparts (ㅋ , ㅌ , ㅍ , ㅊ). Additionally, ㅎ is frequently deleted between vowels or sonorants. E.g., 좋다 (joh-ta) becomes [조타] (tɕo.t^{h} a)
- **Tensification/Fortition (Cases: Common after obstruents)** Plain consonants become tense after syllables ending in obstruents. E.g., 학교 (hak-gyo) is pronounced [학꾜] (hakˀ.kjo).
- **Palatalization (Cases: 2 main types)** Alveolar consonants ㄷ and ㅌ become palatalized to ㅈ and ㅊ respectively before the vowels 이 [i] or 히 [hi]. Additionally, ㅅ is pronounced as [ɕ] before /i/. E.g., 같이 (gat-i) is pronounced [가치] (ka.tɕ^{h} i).

These phonological phenomena represent very specific cases, most commonly arising from interactions between the final consonant of one syllable and the initial consonant of the next. Although such specific phonological cases constitute a relatively small proportion of all possible syllable combinations in Korean, accurately modeling these phenomena is crucial for natural and fluent Korean speech synthesis. In other words, the pronunciation of a given syllable can vary significantly depending on its surrounding context, an effective method is required for a TTS model to properly learn these context-dependent pronunciation variations.

J.4 Pair-wise Jamo Representation

Given the linguistic advantages and phonological characteristics described above, we propose a pair-wise representation based on Jamo to effectively leverage these strengths. By explicitly modeling pairs of adjacent Jamo, we effectively address both phonetic granularity within syllables and context-dependent phonological variations across syllable boundaries, particularly interactions involving the final consonant of one syllable and the initial consonant of the next. Specifically, we represent each utterance through four distinct pair types to comprehensively cover all possible Jamo pairs: (initial consonant, medial vowel), (medial vowel, final consonant), (final consonant, next initial consonant), and (medial vowel, next initial consonant) since the final consonant is optional. These pairs are extracted using a sliding window of length two with stride one over each decomposed sequence, ensuring comprehensive coverage without added complexity, resulting in a maximum of 1,878 unique Jamo pairs. This significantly reduces complexity compared to modeling complete syllable pairs, while still capturing critical pronunciation phenomena such as liaison, nasalization, and tensification.

Using this pair-wise representation offers several important advantages. Firstly, it simplifies modeling complexity and enhances computational feasibility. Secondly, by focusing explicitly on inter-syllabic relationships, the model more effectively generalizes pronunciation patterns rather than memorizing isolated syllables, improving the naturalness and accuracy of synthesized speech, especially in cases of unseen or rare combinations. Finally, applying a pair-wise Jamo representation when selecting core datasets ensures balanced phonetic coverage. By identifying and retaining utterances containing infrequent but phonologically critical Jamo pairs, and probabilistically filtering overly frequent patterns, this method achieves a more representative and linguistically robust dataset.