Leveraging Fine-grained Error Correction in Korean Speech Recognition for Consultation Services

Yonghyun Jun[a], Jimin Lee[a], Hwan Chang[a], Dongho Shin[b], Seolah Kim and Hwanhee Lee[a,*]

[a] *Chung-Ang University, 84, Heukseok-ro, Dongjak-gu, Seoul, 06974, Republic of Korea*
[b] *120 Dasan Call Foundation, 23, Nangye-ro 28-gil, Dongdaemun-gu, Seoul, Republic of Korea, Seoul, 02586, Republic of Korea*

ARTICLE INFO

Keywords:
Speech-to-text
Data Augmentation
In-context Learning
Supervised Fine-tuning
Pre-trained Language Models

ABSTRACT

Automatic speech recognition (ASR) technology is fundamental to customer service automation and large-scale transcription. However, even advanced ASR models exhibit inevitable errors in complex real-world environments such as call center conversations. When original audio is unavailable due to privacy restrictions or storage limitations, error correction must primarily rely on text-based post-editing. Existing text-only approaches face significant challenges in low-resource languages, mainly due to a critical scarcity of annotated corpora and tailored correction methodologies. For Korean, this resource gap is particularly pronounced.

To address this issue, we introduce the first large-scale Korean benchmark dataset specifically curated for ASR error correction. Derived from genuine call center interactions, it comprises 2k dialogues and 105k utterances, providing a realistic foundation for training and evaluating models. Leveraging this resource, we propose a novel text-only framework for fine-grained ASR error correction. Our system utilizes a two-stage pipeline in which an encoder-based model first performs token-level error detection, followed by a seq2seq corrector trained to rectify errors over fine-grained spans. By employing multi-level granularity and processing context progressively from dialogue to utterance to token or span, our method achieves state-of-the-art performance and significantly surpasses baselines on error detection and correction metrics. Extensive experiments analyzing speaker variation, domain shifts, and error density further illuminate the challenges and demonstrate the effectiveness of our approach. Overall, our contributions establish a vital dataset and a practical baseline that are poised to accelerate future research in Korean ASR post-editing as well as broader text-based error correction techniques for low-resource speech technologies.

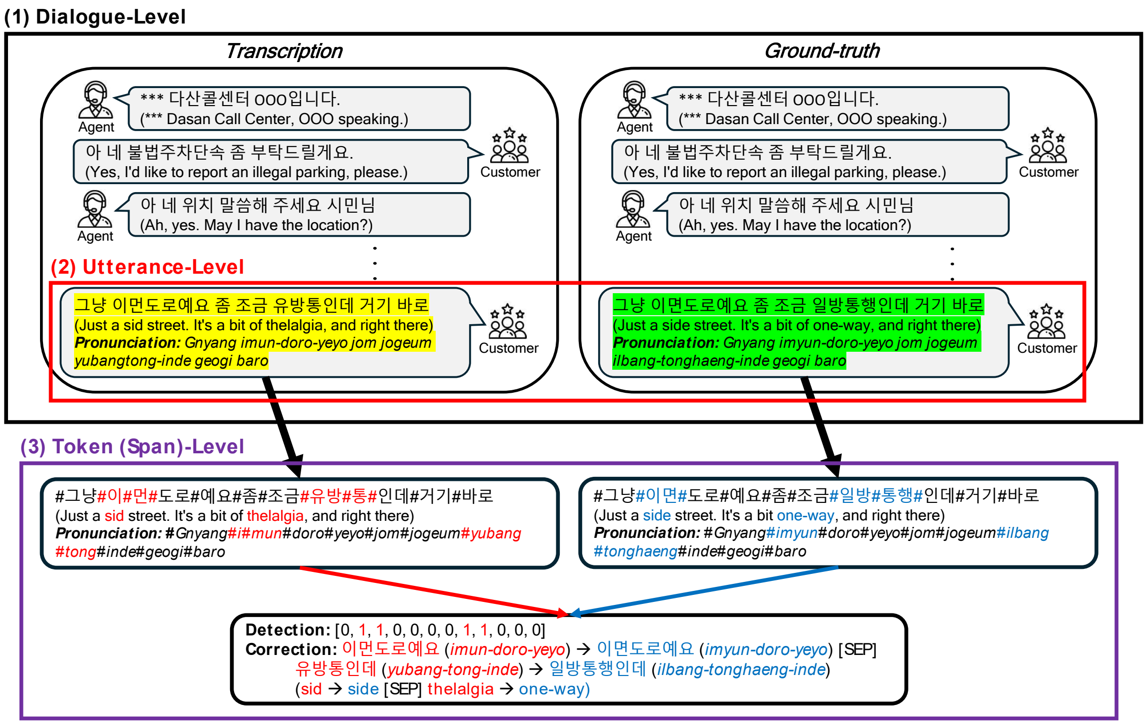
1. Introduction

Speech-to-text (STT) technology is indispensable in domains such as customer support, accessibility services, and automated transcription. Recent advances in deep learning and natural language processing (NLP) have markedly boosted the accuracy of STT systems (Prabhavalkar, Hori, Sainath, Schlüter and Watanabe, 2023; Ahlawat, Aggarwal and Gupta, 2025). However, in complex and dynamic environments, for example, call centers or consultant hotlines, speech recognizers still suffer from high error rates due to pronunciation variance, background noise, overlapping speech, and domain-specific jargon. Such misrecognitions disrupt effective complaint handling and public-service operations.

Research on improving STT performance therefore spans two complementary fronts: (i) enhancing the acoustic and language components of the backbone ASR model (Baevski, Zhou, Mohamed and Auli, 2020; Gulati, Qin, Chiu, Parmar, Zhang, Yu, Han, Wang, Zhang, Wu et al., 2020) and (ii) post-editing transcripts to fix residual errors. In public-service call logs, strict privacy regulations often prevent access to the raw audio, making text-only correction frameworks particularly valuable. Early work framed post-editing as monolingual "translation," fine-tuning sequence-to-sequence models to map noisy transcripts to clean text (Hrinchuk, Popova and Ginsburg, 2020; Dutta, Jain, Maheshwari, Pal, Ramakrishnan and Jyothi, 2022; Ma, Gales, Knill and Qian, 2023a). More recently, large language models (LLMs) have shown the ability to correct errors in context without task-specific tuning (Ma, Qian, Manakul, Gales and Knill, 2023b; Ma, Qian, Gales and Knill, 2024). However, these successes are concentrated in high-resource languages such

*Corresponding author

zgold5670@cau.ac.kr (Yonghyun Jun); ljm1690@cau.ac.kr (Jimin Lee); hwanchang@cau.ac.kr (Hwan Chang); dhs@120dasan.or.kr (Dongho Shin); prcssnowwhite@gmail.com (Seolah Kim); hwanheelee@cau.ac.kr (Hwanhee Lee); hwanheelee@cau.ac.kr (Hwanhee Lee)
ORCID(s):



*** 다산콜센터 ooo입니다.
(*** Dasan Call Center, OOO speaking.)

아 네 불법주차단속 좀 부탁드릴게요.
(Yes, I'd like to report an illegal parking, please.)

아 네 위치 말씀해 주세요 시민님
(Ah, yes. May I have the location?)

그냥 이먼도로예요 좀 조금 유방통인데 거기 바로
(Just a sid street. It's a bit of thelalgia, and right there)
Pronunciation: *Gnyang imun-doro-yeyo jom jogeum yubangtong-inde geogi baro*

그냥 이면도로예요 좀 조금 일방통행인데 거기 바로
(Just a side street. It's a bit of one-way, and right there)
Pronunciation: *Gnyang imyun-doro-yeyo jom jogeum ilbang-tonghaeng-inde geogi baro*

#그냥#이#먼#도로#예요#좀#조금#유방#통#인데#거기#바로
(Just a sid street. It's a bit of thelalgia, and right there)
Pronunciation: #Gnyang#i#mun#doro#yeyo#jom#jogeum#yubang#tong#inde#geogi#baro

#그냥#이면#도로#예요#좀#조금#일방#통행#인데#거기#바로
(Just a side street. It's a bit one-way, and right there)
Pronunciation: #Gnyang#imyun#doro#yeyo#jom#jogeum#ilbang#tonghaeng#inde#geogi#baro

Detection: [0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0]
Correction: 이먼도로예요 (*imun-doro-yeyo*) → 이면도로예요 (*imyun-doro-yeyo*) [SEP]
유방통인데 (*yubang-tong-inde*) → 일방통행인데 (*ilbang-tonghaeng-inde*)
(sid → side [SEP] thelalgia → one-way)

Figure 1: End-to-end illustration of the three-level granularity.

(1) Dialogue-level (black box) shows a dialogue sample from the original dataset. **(2) Utterance-level** (red box) depicts utterances that are split from the dialogue. **(3) Token(Span)-level** (purple box) depicts token-level detection targets and span-level correction targets that are refined from each utterance. The left-hand example is the noisy ASR transcription, while the right-hand example is the human-annotated ground-truth text. Within each example in **(3) Token(Span)-level**, erroneous tokens are marked in red and their correct counterparts in blue.

as English and Chinese; performance drops sharply in low-resource languages, where both error patterns and clean references are scarce (Li, Chen, Kwok, Chu, Chng and Kawai, 2024).

To the best of our knowledge, the Korean ASR correction scenario, especially under the noisy and dynamic conditions of call-center data, remains largely unexplored. We therefore compile a new benchmark comprising real call-center dialogues, automatically transcribed by a production ASR system and manually corrected to create ground-truth references. The final corpus contains roughly 2k dialogue pairs and 105k utterances, with a measured transcription error rate of 9.83

In previous studies, introducing a dedicated detector before the corrector has been shown to improve correction accuracy (Pu, Nguyen and Stüker, 2023; Yeen, Kim and Koo, 2023). Motivated by these **detection-correction pipelines**, we benchmark not only the correction task but also an upstream error-detection task, prepending an individual detector for error detection to our correction framework. Other studies further report that editing models perform best when trained on spans that truly require modification rather than on entire sentences (Malmi, Krause, Rothe, Mirylenka and Severyn, 2019; Stahlberg and Kumar, 2020; Leng, Tan, Liu, Song, Wang, Li, Qin, Lin and Liu, 2023). We therefore **refine sample granularity in three levels**, as illustrated in Fig 1. At the beginning, the dataset comprises full **(1) dialogue-level** samples between call-center agents and customers. Then, each dialogue is split into individual **(2) utterance-level** samples, shortening context and increasing sample count (Sec 3.1.2). We further refine each utterance by computing token-level edit distances between the ASR transcription and ground truth, yielding **(3) token-level** labels for detection (Sec 4.1) and **(3) span-level** targets for correction (Sec 5.1).

The effectiveness of our two-stage architecture and granularity strategy is empirically confirmed in Secs 7.1 and 7.2, where performance increases steadily as each component is introduced. Our final framework outperforms the LLM baseline using dialogue-level granularity by a factor of two to three on average across both detection and correction tasks. In addition, extensive experiments and ablation studies covering class imbalance, speaker variation, out-of-domain inputs, and differing error densities validate the framework's robustness and achieve state-of-the-art results on both detection and correction tasks, as detailed in Sec 7.3. These findings provide concrete guidance for future research on Korean ASR error correction (Sec 7.3). In summary, this paper makes three key contributions:

1. **First Korean ASR benchmark in a real-world setting.** We construct and publicly release the first large-scale Korean STT dataset captured in a dynamic call-center environment and provide manually verified ground-truth corrections, establishing a realistic benchmark for future research.
2. **Widely adoptable Korean-specific framework.** We explore a Korean-specific error correction pipeline and propose an open baseline, combining a fine-grained token-level detector with a fine-grained span-level corrector, that can be readily adopted for this task.
3. **Comprehensive empirical analysis.** Sec 7.3 further elucidates the framework's behavior under diverse configurations, offering actionable insights and directions for future Korean ASR error correction research.

2. Related Work

2.1. Automatic Speech Recognition Error Detection

Automatic speech recognition (ASR) error detection is a long-standing post-processing step that flags recognition mistakes before they propagate to downstream applications. One common approach uses separate acoustic and transcript encoders. Meripo and Konam (2022) jointly encode acoustic and textual inputs, leveraging Wav2Vec 2.0 (Baevski et al., 2020) for speech and BERT (Devlin, Chang, Lee and Toutanova, 2019) for text, and then train an entailment classifier to predict audio–transcript consistency. In many real-world settings, however, raw audio is not always accessible, which motivates text-only detection. Harvill, Khaziev, Li, Cogill, Wang, Chennupati and Thadakamalla (2024) therefore propose a lightweight utterance-level detector that estimates semantic distance to identify critical transcription errors without additional acoustic information. Moving beyond coarse utterance-level labels, Gekhman, Zverinski, Mallinson and Beryozkin (2022) introduce a BERT-based sequence tagger that embeds ASR confidence scores alongside text to label each token as ERROR or NOTERROR, yielding stronger detection performance.

2.2. Automatic Speech Recognition Error Correction

Research on correcting ASR transcriptions has progressed along three intertwined axes: model capacity, pipeline structure, and supervision granularity. These dimensions collectively inform our approach. Early work shows that prompting GPT-style large language models (LLMs) can reduce word-error rate (WER) without fine-tuning (Ma et al., 2023b, 2024), yet these gains are limited to high-resource languages.

When parallel data are scarce, transformer sequence-to-sequence models fine-tuned on 1-best hypotheses provide a stronger baseline (Hrinchuk et al., 2020). Subsequent studies add denoising objectives (Dutta et al., 2022) or N-best conditioning (Ma et al., 2023a), demonstrating that modest corpora still yield competitive improvements when domain knowledge is injected explicitly. We therefore adopt Korean-pretrained seq2seq backbones as a realistic comparison against LLMs.

Accuracy improves further when localization and rewriting are decoupled. Two-stage pipelines first tag erroneous spans and then edit only those regions, outperforming single-stage baselines under data scarcity (Pu et al., 2023; Yeen et al., 2023). Concurrently, fine-grained supervision has proven beneficial: tag-and-realize editors (Malmi et al., 2019), span-level transducers (Stahlberg and Kumar, 2020), and soft masking (Leng et al., 2023) all show that models learn more precise edits when trained only on text that requires modification. Our dataset follows this progression, refining samples from dialogue to utterance and finally to token or span granularity, and applying them to an encoder-based detector and a seq2seq-based corrector.

These advances have rarely been evaluated for Korean, whose agglutinative morphology allows errors to span multiple morphemes. Unsupervised character-level methods succeed in Chinese (Guo, Wang, Qiao, Wei, Shang, Li, Yu, Li, Su, Zhang et al., 2024) but transfer poorly to a word-level language like Korean, and earlier LSTM seq2seq correctors show limited gains (Jin, Lee, Chae, Park, Kang, Lee et al., 2021). By releasing a 2k-dialogue / 105k-utterance benchmark and demonstrating that detector–corrector synergy plus fine-grained supervision yields state-of-the-art results, we close this gap and offer a blueprint for low-resource ASR post-editing.

3. Dataset

3.1. Dataset Construction

3.1.1. Original Dialogue-level Dataset

To construct the dataset for speech recognition error detection and correction, we first collect the original STT-transcribed dialogue data using the speech detection model `HAIV`, which is optimized for cloud environments and operates on on-premises infrastructure utilizing Docker and Kubernetes for AI-driven speech recognition. Based on the dialogues, we manually create the corresponding corrected dialogues with raw audio files to serve as ground truth. Because the data in this study consist of real voice call recordings between citizens and counselors, anonymization is essential to protect privacy. We therefore conduct de-identification of both transcriptions and ground truth by professionals who are well trained for this task. The process follows a structured manual comprising three steps: (1) During the extraction of speech-to-text data, all numerical data are automatically replaced with asterisks (*) using SQL regular expressions to ensure initial de-identification. (2) The data retrieved in step 1 are then manually de-identified by removing personal names and other sensitive information. (3) The de-identified data from step 2 undergo a thorough review to verify the completeness of the anonymization, resulting in the final dataset. In total, 1,974 call instances are processed, yielding 105,133 lines of data each for the raw and corrected transcriptions. Assuming an average call duration of 2 minutes and 30 seconds, the total volume of speech sums to approximately 296,100 seconds (82 hours and 15 minutes).

3.1.2. Dialogue-level to Utterance-level

We initially considered using the entire dialogue as input, hypothesizing that broader conversational context would facilitate more accurate corrections. However, even large-scale language models often struggle to fully leverage lengthy contexts in low-resource languages such as Korean (Aycock, Stap, Wu, Monz and Sima'an, 2024). Drawing on studies indicating that shorter contexts or multiple single-sentence examples can improve performance in low-resource scenarios (Cahyawijaya, Lovenia and Fung, 2024), we refine the sample granularity from **dialogue level** to **utterance level** by splitting each dialogue into individual utterances.

To preserve the information in each original dialogue sample, we create a dataframe with [*speaker, dialogue key, transcribed utterance, ground truth utterance*]. We define a *dialogue key* by concatenating the recording date with a running index for that day, ensuring that all utterances originating from the same conversation share an identical key.

Once we reduce context length by refining the long dialogue to short utterances, large LLMs with substantial parametric knowledge become unnecessary. Instead, the ability to generate correct sentences based solely on the given utterance context takes precedence. Therefore, rather than a general-purpose LLM, we adopt a Korean-pretrained language model to build the post-processing models in Secs 4 and 5.

3.2. Dataset Statistics

This study utilizes a dataset of transcribed Korean civil service conversations, where each conversation contains approximately 57.40 sentences on average. The overall rate of utterance samples containing errors is 9.83%, reflecting 11,355 erroneous sentences out of 115,460 total utterances. These error distributions align with the actual accuracy of the STT model currently in use. To emphasize a real-world configuration that incorporates an advanced STT model's performance in genuine consultation scenarios, no additional preprocessing is conducted.

3.2.1. Error Type Statistics

In terms of error attribution, 61% of the errors are caused by callers (hereafter referred to as "customers"), while the remaining 39% originate from agents. A part-of-speech analysis shows that noun-related errors account for the largest proportion (56.3%), followed by errors involving verbs (21.7%), adjectives (5.9%), prepositions (4.3%), pronouns (4.1%), adverbs (3.8%), conjunctions (2.9%), and interjections (1.0%). For instance, noun errors frequently occur when words such as "jeona" are transcribed instead of "jeonhwa (telephone)" or "charang" instead of "charyang (car)." Verb errors arise when "badumyeon" is recognized in place of "badeumyeon (if receive)" or "andogo" instead of "andoego (it does not work)." Adjective errors appear when "geureomeuro" is transcribed instead of the correct form "geureomuro (hence)" or "gareuchida" is misrecognized as "garikida (indicate)."

From a phonetic perspective, a significant majority of errors (84.5%) involve consonant-level distortions, whereas vowel-related errors account for the remaining 15.5%. For example, the word "bab (rice)" is often misrecognized as "pap," and "seoul (Seoul)" is transcribed as "sseoul." Concerning vowel confusions, "yul" may be interpreted as "ul," and "e" as "ye." Pronunciation-related errors can be further divided into three main categories. The most prevalent type

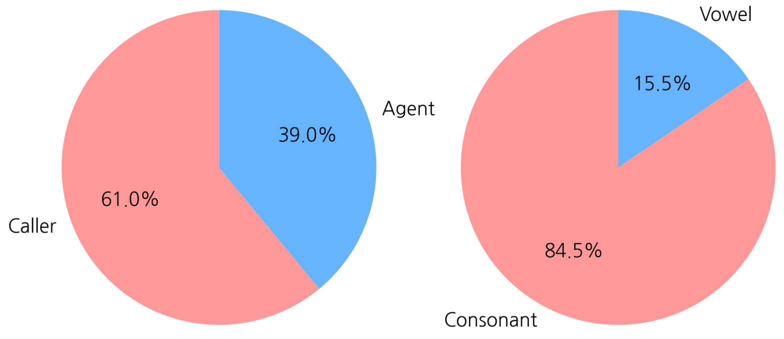


Figure 2: Overall statistics of Seoul Dasan Call Foundation collected data.

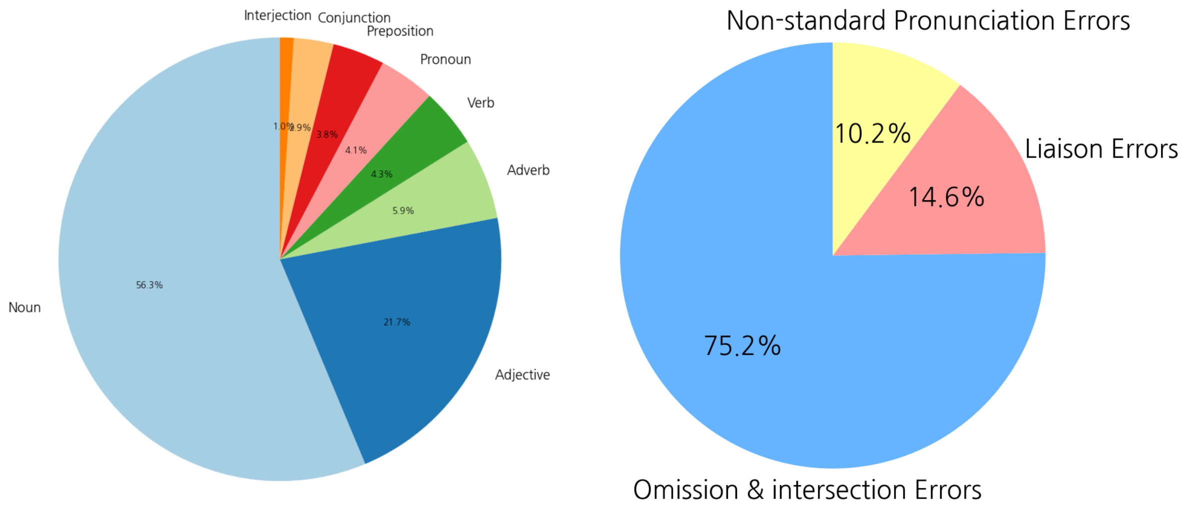


Figure 3: Overall statistics of Seoul Dasan Call Foundation collected data.

is omission and insertion errors (75.2%), in which phonemes are unintentionally dropped or added—for example, the phrase "dongjakgu samuesil e" is transcribed in place of the correct "dongjakgucheong samusil (the Dongjak District Office)." Liaison errors (14.6%) stem from phonological links between adjacent syllables, as when "seo ulsicheong" is interpreted instead of "seoulsicheong (Seoul City Hall)." Finally, non-standard pronunciation errors (10.2%) occur when non-canonical pronunciations are used, such as "unsuhosa" being recognized instead of "unsuhoesa (transport company)."

These statistics highlight the multifaceted nature of speech recognition errors in Korean, spanning lexical, phonological, and pronunciation-level variations. Accordingly, sophisticated error-correction mechanisms tailored to Korean linguistic characteristics are required.

3.2.2. Dataset Split

Call-center data frequently contain repeated phrases such as common inquiries or filler words. To handle this repetition, we first partition the dataset into training, validation, and test sets at a ratio of 0.80 / 0.05 / 0.15. We then rebalance duplicates within each subset to ensure that frequently used expressions remain present across both training and test sets. As a result, the training, validation, and test sets contain 50,896 / 3,065 / 10,261 instances, respectively. This strategy offers a realistic reflection of everyday colloquial usage while preserving a balanced distribution of utterances for model training and evaluation.

4. Automated Speech Recognition Error Detection

In this section, we present an error-detection framework that leverages the dataset described in Sec 3 to automatically locate recognition errors in transcriptions. The approach formulates detection as a sequence-masking task and incorporates a loss function tailored to label imbalance. The resulting detector is later reused within our error-correction pipeline.

4.1. Data Preprocessing: Utterance-level to Token-level

Let U denote an **utterance-level** transcription and G its **utterance-level** ground-truth. Rather than simply determining whether U contains any errors, we refine the task to detect errors in each token of U. Specifically, we utilize a Korean-pretrained tokenizer to convert U and G into **token-level** sequences $U_T = \hat{t}_1, \dots, \hat{t}_n$ and $G_T = t_1, \dots, t_m$, where n and m are their respective lengths. We then compute the edit distance between these tokenized representations using the `SequenceMatcher` from the `difflib` library. Any token in U_T that differs from G_T (because of replacement, deletion, or insertion) is masked with 1, while matched tokens are masked with 0. If an insertion occurs, the token immediately preceding the insertion point is masked; if the insertion is at the start of the utterance, the token immediately following that point is masked. As a result, we obtain a **token-level** mask sequence $M = m_1, \dots, m_n$, where $m_i = 1$ when $\hat{t}_i$ is erroneous and 0 otherwise.

4.2. Training Detector

We formulate the detection task as a sequence-labeling problem (Gekhman et al., 2022), where each token in U_T is assigned a binary mask. To develop a model for this task, we fine-tune the same Korean-pretrained model used in the tokenization phase (Sec 4.1) so that it outputs the mask sequence M given U_T. The loss function is defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{n^{(i)}}\sum_{t=1}^{n^{(i)}}\Big(\lambda\, m_t^{(i)}\log(\hat{m}_t^{(i)}) + (1-m_t^{(i)})\log(1-\hat{m}_t^{(i)})\Big),$$

where N is the number of training samples. For the i-th input $U_T^{(i)}$, $n^{(i)}$ denotes its token length, $m_t^{(i)}$ the binary mask for the t-th token, and $\hat{m}_t^{(i)}$ the predicted probability of assigning an error mask (1) to that token. Because non-erroneous tokens greatly outnumber erroneous ones (Sec 3.2), we introduce a weight factor $\lambda = 8$ to emphasize learning on the sparse error tokens.

5. Automated Speech Recognition Error Correction

In this section, we present an error-correction framework that builds on the dataset described in Sec 3 to automatically correct recognition errors in an utterance-level transcription U. Our primary goal is to guide the model so that the corrected output closely matches the utterance-level ground truth G.

5.1. Data Preprocessing: Utterance-level to Span-level

To correct errors within U, we assume that the model learns more effectively when it focuses not on the entire sentence but specifically on the segments that require repair and their corrected forms. A complete sentence may admit many potential edits; narrowing attention to the precise spans to be modified simplifies the editing objective (Stahlberg and Kumar, 2020; Leng et al., 2023). We therefore refine the **utterance-level** ground truth G into a **span-level** representation that explicitly indicates which segments need modification and how they should be corrected.

Following the notation in Sec 4.1, let $U_T = \hat{t}_1, \hat{t}_2, \dots, \hat{t}_n$ be the tokenized form of U, and let $G_T = t_1, t_2, \dots, t_m$ be the tokenized form of G. Using the same sequence-masking strategy described in Sec 4.2, we measure the edit distance between U_T and G_T to determine which tokens in U_T must be replaced, inserted, or deleted. These tokens constitute the error spans $span^U$, while the corresponding correct tokens constitute $span^G$. In contrast to the detection phase, we treat tokens at the word level and merge consecutive tokens into a single span when warranted—for example, for insertions, deletions, or contiguous erroneous segments.

We then pair each span in $span^U$ with its counterpart in $span^G$ using an arrow token (→) and concatenate multiple pairs with the learnable [SEP] token, creating the final **span-level** label $\mathcal{S}$. If no errors are present in U_T, we simply designate the phrase "No Error" in Korean as the target. Formally:

$$\mathcal{S} = \begin{cases} \text{No Error}, & \text{if } s = 0, \\ \text{span}_1^U \rightarrow \text{span}_1^G \text{ [SEP] } \text{span}_2^U \rightarrow \text{span}_2^G \text{ [SEP] } \cdots, & \text{otherwise}, \end{cases}$$

Here, s is the number of extracted spans, while span_i^U and span_i^G denote the i-th erroneous span in $span^U$ and its corrected form in $span^G$, respectively.

5.2. Training Corrector

We fine-tune a Korean-pretrained seq2seq model to predict the **span-level** representation $\mathcal{S}$ given an utterance-level transcription U. At inference time the predicted $\hat{\mathcal{S}}$ is not a complete sentence, so we apply a post-processing step to construct the corrected utterance $\hat{G}$. Specifically, we split $\hat{\mathcal{S}}$ at each [SEP] token and replace every $\hat{\text{span}}_i^U$ in U with its corresponding $\hat{\text{span}}_i^G$. If the model outputs "No Error", we simply return U unchanged.

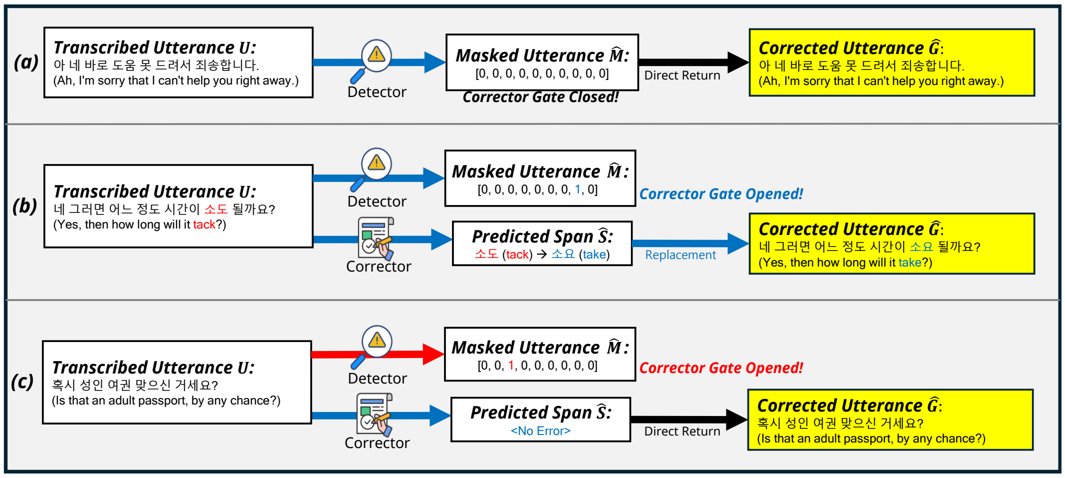

(a)
Transcribed Utterance U:
아 네 바로 도움 못 드려서 죄송합니다.
(Ah, I'm sorry that I can't help you right away.)
Detector
Masked Utterance M̂:
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Corrector Gate Closed!
Direct Return
Corrected Utterance Ĝ:
아 네 바로 도움 못 드려서 죄송합니다.
(Ah, I'm sorry that I can't help you right away.)
(b)
Transcribed Utterance U:
네 그러면 어느 정도 시간이 소도 될까요?
(Yes, then how long will it tack?)
Detector
Masked Utterance M̂:
[0, 0, 0, 0, 0, 0, 0, 1, 0]
Corrector Gate Opened!
Corrector
Predicted Span Ŝ:
소도 (tack) → 소요 (take)
Replacement
Corrected Utterance Ĝ:
네 그러면 어느 정도 시간이 소요 될까요?
(Yes, then how long will it take?)
(c)
Transcribed Utterance U:
혹시 성인 여권 맞으신 거세요?
(Is that an adult passport, by any chance?)
Detector
Masked Utterance M̂:
[0, 0, 1, 0, 0, 0, 0, 0, 0]
Corrector Gate Opened!
Corrector
Predicted Span Ŝ:
<No Error>
Direct Return
Corrected Utterance Ĝ:
혹시 성인 여권 맞으신 거세요?
(Is that an adult passport, by any chance?)

Figure 4: Illustration of our detector–corrector inference pipeline. Blue arrows denote correct actions; red arrows indicate mistakes. (a) Detector sees no error, gate stays closed, and U is returned. (b) Detector opens the gate and the corrector repairs the flagged span. (c) Detector raises a false positive, but the corrector detects no error and echoes U.

5.3. Incorporating Detector

Inspired by prior work demonstrating that a detector-corrector pipeline can improve ASR error correction (Pu et al., 2023; Yeen et al., 2023), we integrate the detector from Sec 4 into our seq2seq approach.

First, for each utterance-level training sample U_{train}, we obtain a predicted mask sequence $\hat{M}$train from the detector: if $\hat{M}$train contains at least one 1, U_{train} is flagged as erroneous; otherwise, it is flagged as error-free. We then assemble a new training set for the seq2seq corrector by collecting

1. all samples whose M_{train} actually contain at least one erroneous token (1), and
2. all samples that the detector incorrectly flags as erroneous even though U_{train} is error-free.

The Korean-pretrained seq2seq model is further fine-tuned on this curated set following the fine-grained scheme in Sec 5.2. This refinement enables the model to focus on genuine correction behavior while preventing spurious detections from propagating downstream.

At inference time, we first apply the detector to an utterance-level transcription U. As illustrated in Fig 4(a), when the detector deems U error-free, the gate to the corrector remains closed and U is returned unchanged. Otherwise, the detector flags an error and opens the gate to the corrector, forwarding U to be refined into $\hat{G}$. Fig 4(b) shows a typical success case in which the detector identifies the error span and the corrector produces an accurate revision. Even when the detector mistakenly opens the gate for an error-free sentence, as in Fig 4(c), the corrector can recognize that no change is required and simply echoes U, thereby compensating for the detector's false positive.

6. Experimental Setup

6.1. Baselines

6.1.1. Baselines for Error Detection

Utterance-level Detection To verify the effectiveness of our **token-level** detection after granularity refinement (Sec 4.1), we reframe the detection task as a binary classification problem at the **utterance level**. The model predicts whether each utterance contains any error. If the transcribed utterance U perfectly aligns with its ground-truth counterpart G, the label y is 0; otherwise, the label is 1. We fine-tune a Korean-pretrained encoder model for sequence classification, employing the following loss function:

$$\mathcal{L} = -\sum_{i=1}^{N} \Big(\lambda\, y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \Big),$$

where N is the number of training samples. For the i-th utterance sample U, $y^{(i)}$ is the binary label and $\hat{y}^{(i)}$ is the probability of assigning the erroneous label (1). We set $\lambda = 2$ as a weight factor to address label imbalance.

Token-level Detection with Seq2Seq We also adopt Korean-pretrained seq2seq models to generate **span-level** error representations directly. Specifically, we align the masked utterance M obtained in Sec 4.1 with the original utterance-level transcription U to locate error spans. We then concatenate each span with the [SEP] token and fine-tune the seq2seq model to output the concatenated spans. If no errors are present, the model produces "No Error", as in Sec 5.1.

6.1.2. Baselines for Error Correction

Starting from **dialogue-level** correction using multilingual large language models (LLMs), we progressively refine the granularity and the backbone architecture to construct a series of baselines.

Dialogue-level Correction with LLM We fine-tune each LLM on the **dialogue-level** transcription as a single context, pairing it with the corresponding ground-truth dialogue as the target. This setup enables the model to correct the entire conversation in one pass. We also evaluate in-context learning with a more advanced LLM that, instead of fine-tuning, is given two **dialogue-level** transcriptions along with their ground-truth dialogues as examples. For evaluation, the corrected dialogue is split into individual utterances.

Dialogue-level to Utterance-level Following the approach in Sec 3.1.2, we segment dialogue samples into utterances, providing each **utterance-level** transcription as a single context and its corresponding ground truth as the target for LLM fine-tuning. For in-context learning, we present a more advanced model with ten **utterance-level** transcriptions (five containing errors and five error-free) and their ground-truth utterances.

Large Model to Small Model As noted in Sec 3.1.2, we hypothesize that large LLMs may be excessive for the simplified error-correction task. Smaller, linguistically targeted models are both more efficient and better suited. We therefore fine-tune Korean-pretrained seq2seq models on **utterance-level** transcriptions to produce the corresponding ground-truth utterances.

Utterance-level to Span-level Lastly, following Sec 5.1, we refine the **utterance-level** targets into **span-level** representations and fine-tune a seq2seq corrector accordingly. This baseline uses only the seq2seq corrector, without a preceding detector.

6.2. Backbone Models

We use `monologg/koelectra-base-v3-discriminator`(Park, 2020) as the backbone encoder for our detector in Sec 4, leveraging its strong performance on token-classification tasks. For the utterance-level baseline detector, we employ `klue/roberta-base` (Park, Moon, Kim, Cho, Han, Park, Song, Kim, Song, Oh, Lee, Oh, Lyu, Jeong, Lee, Seo, Lee, Kim, Lee, Jang, Do, Kim, Lim, Lee, Park, Shin, Kim, Park, Oh, Ha and Cho, 2021), known for its robust performance across the KLUE benchmark.

For the correctors and seq2seq-based baseline detectors, we use `google/mt5-base`(Xue, Constant, Roberts, Kale, Al-Rfou, Siddhant, Barua and Raffel, 2020), `paust/pko-t5-base`(Park, 2022), and `hyunwoongko/kobart` as backbones. For the LLM-based baseline correctors, we fine-tune `Llama-3.1-8b-Instruct`(Dubey, Jauhri, Pandey, Kadian, Al-Dahle, Letman, Mathur, Schelten, Yang, Fan et al., 2024) and `Qwen-2.5-7b-Instruct`(Yang, Yang, Zhang, Hui, Zheng, Yu, Li, Liu, Huang, Wei et al., 2024), while in-context learning relies on `GPT-4o` (Hurst, Lerer, Goucher, Perelman, Ramesh, Clark, Ostrow, Welihinda, Hayes, Radford et al., 2024).

6.3. Evaluation Metrics

6.3.1. Metrics for Error Detection

We assess error detection using **coarse-grained metrics**, which judge whether an utterance-level transcription U contains any errors, and **fine-grained metrics**, which measure how precisely the model localizes error spans.

Coarse-grained Metrics Since our detector outputs token-level predictions, we map an utterance to 0 (error-free) if all tokens are predicted correct and to 1 (erroneous) otherwise. The ground-truth status is 0 when U exactly matches its ground-truth G and 1 otherwise. Comparing these two labels yields **F1** and **Accuracy (Acc.)** on the test set. Given the pronounced class imbalance (that is, fewer erroneous utterances than correct ones) in Sec 3.2, we also report **Balanced Accuracy (Bal Acc.)** for a more equitable measure.

Fine-grained Metrics Beyond coarse-grained detection, we assess whether the model can precisely localize errors. Rather than inspecting only error-containing segments, we compare the predicted token-level mask sequence $\hat{M}$ to the corresponding ground-truth M (from Sec 4.1). We compute:

- **Exact Match Ratio (EM)**: The fraction of samples for which $\hat{M}$ exactly matches M.
- **Balanced Exact Match Ratio (Bal EM)**: **EM** is computed separately for genuinely error-free and erroneous samples, then averaged.
- **Balanced Token-Accuracy (Bal T-Acc.)**: Given the sparsity of erroneous tokens, we measure **balanced accuracy** between $\hat{M}$ and M for each sample and then average over all samples.

6.3.2. Metrics for Error Correction

We evaluate how effectively the corrector transforms each transcription U into its revised form $\hat{G}$ with respect to the ground truth G. First, we employ **coarse-grained metrics** to determine whether the model correctly identifies whether U needs correction. To gauge the model's proficiency in fixing errors, we also track:

- **Exact Match Ratio (EM)**: The proportion of samples for which $\hat{G}$ exactly matches G.
- **Balanced Exact Match Ratio (Bal EM)**: **EM** is computed separately for genuinely error-free and erroneous utterances, and the two results are averaged.
- **Balanced Word-Error-Rate (Bal WER)**: We compute the Levenshtein distance (Heeringa, 2004) between $\hat{G}$ and G, normalizing by the number of words in G, yielding the usual **Word-Error-Rate (WER)**. Because most tokens in transcriptions are already correct, averaging WER across all samples yields a near-zero value, making comparisons among baselines difficult. Consequently, we compute **WER** separately for error-free and erroneous subsets and average these values to obtain **Bal WER**. This metric offers a softer, more nuanced assessment than strict **EM** evaluation.

6.4. Implementation Details

When training encoder models, we set the learning rate to `2e-5`, and for seq2seq models, we use `1e-4`. Both categories adopt a batch size of 24 with the AdamW optimizer (Loshchilov and Hutter, 2017). Training spans 15 epochs by default, and we select the checkpoint with the highest validation performance—token-level accuracy for detectors and balanced WER for correctors. For seq2seq models, we apply a weight decay of `1e-3` and use a beam size of 4 for decoding. For LLM baselines, we utilize LoRA (Hu, Shen, Wallis, Allen-Zhu, Li, Wang, Wang, Chen et al., 2022) for fine-tuning and perform greedy (zero-temperature) decoding. All experiments are conducted on NVIDIA RTX 4090 GPUs.

Model	Utterance-Level			Token-Level		
	F1 ↑	Acc. ↑	Bal Acc. ↑	EM ↑	Bal EM ↑	Bal T-Acc. ↑
KoRoberta$_U$	66.64	93.50	80.55	–	–	–
			Utterance → Tokens			
KoElectra$_T$	70.72(+4.08)	93.96(+0.46)	84.35(+3.80)	96.03	59.00	**92.53**
mT5$_T$	52.64(-18.08)	91.14(-2.82)	72.35(-12.00)	95.90(-0.13)	58.58(-0.42)	89.45(-3.08)
pkoT5$_T$	62.38(-8.34)	93.58(-0.38)	75.33(-9.02)	98.19(+2.16)	65.03(+6.03)	90.63(-1.90)
KoBart$_T$	43.28(-27.44)	90.24(-3.72)	66.56(-17.79)	96.39(+0.36)	56.15(-2.85)	88.15(-4.38)

Table 1
Combined detection results from coarse-grained and fine-grained metrics. Metrics with (↑) indicate higher is better, while those with (↓) indicate lower is better. Best results are in bold. To highlight the influence of successive design choices (except for backbone model changes), we report the change relative to the immediately preceding configuration in parentheses: red indicates an improvement, blue a degradation. The suffix *U* means the model is trained with utterance-level samples, while *T* indicates token-level samples.

7. Results & Analysis

7.1. Detection Results

We compare our proposed detector with the baselines described in Sec 6.1.1 to demonstrate the robustness of our detection framework, with a particular focus on the impact of transitioning from an utterance-level to a token-level approach.

7.1.1. *Utterance-level vs. Token-level*

As summarized in Tab 1, refining the sample granularity from **utterance level** to **token level** using KoElectra yields improvements even in coarse-grained metrics. Moreover, the model achieves strong results on the more challenging fine-grained metrics. Seq2seq backbones likewise exhibit high fine-grained metric scores, underscoring the benefits of our proposed granularity refinement and training strategy.

7.1.2. *Encoder vs. Seq2Seq*

We also examine performance differences between encoder backbones (e.g., KoRoberta, KoElectra) and seq2seq backbones (e.g., mT5, pkoT5, KoBART). Although seq2seq models generally attain high exact-match metrics (EM and balanced EM), KoElectra achieves the highest balanced token accuracy (Bal T-Acc). Encoder models also surpass seq2seq models on coarse-grained metrics. Given that encoder models such as KoElectra are relatively lightweight, these findings suggest that encoder backbones may be more suitable for ASR error detection.

7.2. Correction Results

We evaluate our error-correction framework and various baselines using the metrics in Sec 6.3.2, progressively adding each proposed component to illustrate its contribution. Tab 2 presents an overview of the findings.

7.2.1. *Dialogue-level vs. Utterance-level*

Applying dialogue-level samples directly to the error-correction task severely impairs the multilingual LLM baselines, which fail not only to generate properly corrected Korean text but also to produce outputs in the intended format. In contrast, splitting **dialogue-level** samples into **utterance-level** samples (Sec 3.1.2) yields a marked improvement in error detection and enables the models to address correction tasks effectively. These results highlight that, in low-resource language environments, providing multiple exemplar shots at the utterance level can be more beneficial than relying on a lengthy dialogue context.

Notably, GPT-4o in an utterance-level in-context learning setup achieves higher **Balanced EM** and **Balanced WER** than many other variants, indicating that advanced LLMs can capitalize strongly on our granularity strategy.

7.2.2. *Large Language Models vs. Korean-pretrained Language Models*

Following Sec 3.1.2, we adopt a Korean-pretrained seq2seq model as our default corrector backbone and compare its performance with that of multilingual LLM baselines.

Granularity	Model	Detection			Correction		
		F1 (↑)	Acc. (↑)	Bal Acc. (↑)	EM (↑)	Bal EM (↑)	Bal WER (↓)
Dialogue	*LLama-3.1-8b*$_{D\text{-}IT}$	24.56	16.21	40.62	–	–	–
	Qwen-2.5-7b$_{D\text{-}IT}$	14.21	75.18	50.56	–	–	–
	GPT-4o$_{D\text{-}ICL}$	35.22	29.41	51.65	–	–	–
			Dialogue → Utterances				
Utterance	*LLama-3.1-8b*$_{U\text{-}IT}$	28.13(+3.57)	53.79(+37.58)	63.01(+22.39)	48.83	38.45	32.51
	Qwen-2.5-7b$_{U\text{-}IT}$	48.18(+33.97)	83.71(+8.53)	69.88(+19.32)	78.65	44.80	25.74
	GPT-4o$_{U\text{-}ICL}$	37.42(+2.20)	78.77(+49.36)	69.13(+17.48)	76.12	**56.82**	**12.44**
			LLM → Seq2Seq				
Utterance	*mT5*$_U$	56.44(+18.53)	82.98(+10.89)	67.53(+0.19)	83.91(+16.04)	44.36(-2.33)	22.31(-1.25)
	pkoT5$_U$	57.35(+19.44)	83.22(+11.13)	71.15(+3.81)	83.29(+15.42)	44.56(-2.13)	22.14(-1.42)
	KoBart$_U$	58.14(+20.23)	87.53(+15.44)	65.63(-1.71)	91.12(+23.25)	47.91(+1.22)	17.56(-6.00)
			Utterance → Error Spans				
Span	*mT5*$_S$	63.92(+7.48)	92.06(+9.08)	71.63(+4.10)	94.96(+11.05)	52.00(+7.64)	15.04(-7.27)
	pkoT5$_S$	68.28(+10.93)	93.56(+10.34)	74.46(+3.31)	96.82(+13.53)	52.12(+7.56)	14.68(-7.46)
	KoBart$_S$	60.55(+2.41)	89.91(+2.38)	65.12(-0.51)	91.89(+0.77)	50.12(+2.21)	14.67(-2.89)
			Incorporating Detector				
Token+Span	*KoElectra*$_T$ + *mT5*$_S$	69.88(+5.96)	93.12(+1.06)	83.95(+12.32)	91.33(-3.63)	50.12(-1.88)	14.91(-0.13)
	KoElectra$_T$ + pkoT5$_S$	**72.43**(+4.15)	**95.18**(+1.62)	**85.31**(+10.85)	**97.12**(+0.30)	52.67(+0.55)	14.67(-0.01)
	KoElectra$_T$ + *KoBart*$_S$	68.35(+7.80)	92.74(+2.83)	83.78(+18.66)	95.60(+3.71)	51.75(+1.63)	14.90(+0.23)

Table 2
Combined utterance-level detection and correction results for various baselines. Metrics with (↑) indicate higher is better, while those with (↓) indicate lower is better. Best results are in bold. To highlight the effect of successive design choices, we list the change relative to the immediately preceding configuration (within the same backbone) in parentheses: red for gains, blue for drops. For the "LLM →Seq2Seq" block, we average each metric over the three LLMs and compare the mean to each corresponding seq2seq model. Suffixes are defined as follows: *D-IT*=instruction-tuned at dialogue-level; *D-ICL*=in-context learning with dialogue-level examples; *U-IT* and *U-ICL*=instruction-tuned or in-context at utterance-level; *U*=utterance-level training; *S*=span-level training.

Korean-pretrained seq2seq models generally yield results on par with or surpassing those of the LLM baselines, despite being smaller and more cost-effective. The most striking differences arise in detection metrics: LLM baselines often misjudge repetitive interjections and colloquialisms prevalent in Korean, whereas the seq2seq models handle them accurately (see Sec 7.3.4). Moreover, LLMs frequently exhibit over-editing, altering utterances beyond the original meaning and diverging from the ground truth. Certain LLM baselines (e.g., GPT-4o) nonetheless deliver impressive scores, indicating room for further enhancement, but overall the targeted, less resource-intensive seq2seq models remain an appealing choice.

7.2.3. *Utterance-level vs. Span-level*

Moving beyond **utterance-level** correction, we adopt the **span-level** target refinement introduced in Sec 5.1 and observe gains in both detection and correction performance. By explicitly instructing the model on which segments need to be edited and how, we strengthen its comprehension of the task, reinforcing the value of finer-grained targets. In these experiments, T5-based models (mT5, pkoT5) outperform BART-style models such as KoBART. This disparity aligns with their pre-training objectives: BART learns to reconstruct the entire corrupted sentence, whereas T5 learns to generate only the corrupted spans. Because our task likewise tunes the model to generate span-level edits, the T5 pre-training scheme is naturally better aligned and thus delivers stronger results.

7.2.4. *Effect of Incorporating Detector*

Finally, we integrate the token-level KoElectra detector ahead of the corrector framework, forming a two-stage pipeline. This approach outperforms a standalone fine-tuned corrector. In particular, combining our detector with pkoT5 not only boosts detection performance beyond the level of the detector alone but also proves more robust to misclassifications, as it can filter out utterances that do not need correction. Consequently, this two-stage design achieves top marks across most metrics, particularly for **detection metrics** and **EM**, affirming the feasibility of our system in real-world scenarios.

7.3. Analysis

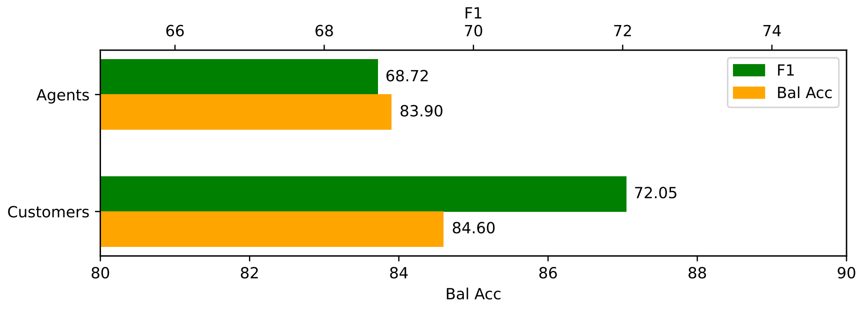
After training our detector and corrector within the proposed framework, we conduct additional analyses to explore their characteristics and performance trends. Unless otherwise noted, all experiments in this section employ $KoElectra_T$ as the detector and $KoElectra_T$ + $pkoT5_S$ as the corrector.

7.3.1. *Performance Gap between Speaker Types*

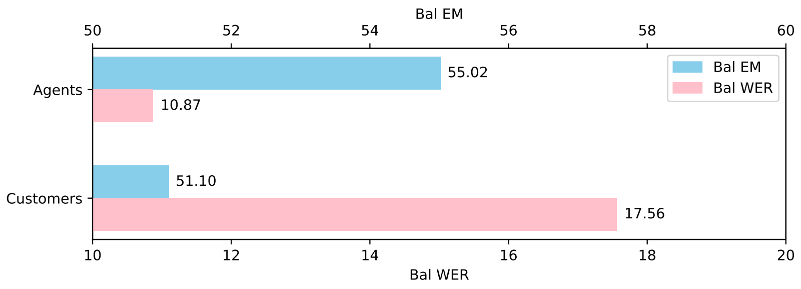
We investigate how our corrector framework performs on utterances spoken by call-center agents versus customers. We report **F1** and **Bal Acc** for detection, and **Bal EM** and **Bal WER** for correction.

As depicted in Fig 5a, the detector achieves better performance on customer utterances than on agent utterances for both detection metrics. This trend likely stems from pronunciation differences: customers often produce more misrecognized tokens, which reduces the likelihood that a token naturally fits the surrounding context. Consequently, these tokens are more easily identified as errors.

By contrast, Fig 5b shows that the corrector scores higher on agent utterances. Although agent utterances may still be misrecognized, these errors often involve tokens whose pronunciations closely resemble the ground truth, creating a predictable error pattern. Once detected, such errors are easier to correct. Customer utterances, on the other hand, may include a wider range of random tokens, complicating the corrector's inference of the original intent.



(a) Our corrector tends to be more accurate for customer's utterances in terms of detection.



(b) Our corrector tends to be more accurate for agent's utterances in terms of correction.

Figure 5: Speaker-wise performance analysis. The top panel shows detection metrics (F1 and Bal-Acc) for utterances spoken by agents versus customers. The bottom panel displays correction metrics (Bal-EM and Bal-WER)

7.3.2. *Out-of-Domain Analysis*

When constructing our train, validation, and test splits, we ensured that frequently occurring utterances were evenly distributed among the splits and removed duplicates after splitting to reflect a realistic speech-recognition scenario (Sec 3.2.2); we refer to this setting as *partial OOD*. Nevertheless, because of the large variety of potential utterances and the variability of ASR performance, assessing generalization on genuinely unseen data is essential.

To this end, we create a new partition by first removing any utterance shared across the three sets, guaranteeing no overlap. We retrain the corrector on this partition and evaluate it on the resulting out-of-domain (OOD) test set, denoted *utterance OOD* in Tab 3. As shown in Tab 3, our corrector maintains performance similar to that in the original setup, with only about a one percent performance decline.

OOD Type	Detection		Correction	
	F1 ↑	Bal Acc. ↑	Bal EM ↑	Bal WER ↓
Partial OOD	70.72	84.35	52.67	14.67
Utterance OOD	69.62(-1.10)	85.46(+1.11)	51.69(-0.98)	15.11(+0.44)
Dialogue OOD	68.92(-1.80)	81.88(-2.47)	50.99(-1.68)	15.32(+0.65)

Table 3
Performance under out-of-domain (OOD) conditions. We denote our default setting as *Partial OOD*, and use its result as the baseline. Although detection and correction metrics dip slightly, and more so for *dialogue OOD*, our corrector remains robust relative to the *Partial OOD* baseline.

To probe an even more divergent setting, we further prevent utterances from the same dialogue from appearing in different splits. This approach enables us to gauge how effectively our framework generalizes to dialogue topics unseen during training, yielding a *dialogue OOD* scenario. As indicated in Tab 3, performance remains stable, showing a one to two percent drop relative to both the *partial* and *utterance OOD* configurations. These findings affirm our framework's resilience in out-of-domain contexts, suggesting it can manage extensive varieties of potential errors in real-world applications.

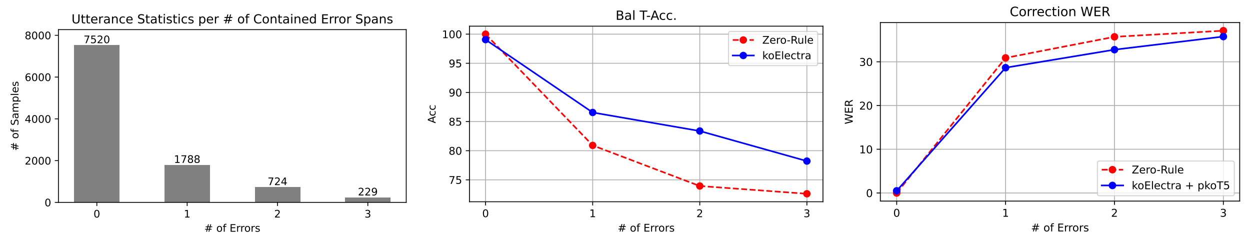
7.3.3. *Performance Trend by Number of Error Spans*

We also evaluate our detector's and corrector's abilities as a function of the number of error spans in each utterance. Because utterances with four or more error spans are extremely rare (about four samples), we merge these with the subset containing three error spans. Fig 6a shows the utterance distribution by error span. We adopt **Bal T-Acc** for detection and **WER** for correction.

To establish a baseline, we compare our system with a zero-rule approach that simply retains the transcribed utterance as is—performing no modifications—and assumes that all tokens are correct for detection. Because most utterances (about 90%) do not require correction, even this naive method produces deceptively high performance figures. Therefore, surpassing the zero-rule baseline confirms that our system can detect and correct errors effectively despite the highly imbalanced nature of real-world data.

As shown in Figs 6b and 6c, for utterances containing errors, our detector consistently attains higher **Bal T-Acc** and our corrector yields lower **WER** than the zero-rule baseline, regardless of the number of errors. For error-free utterances, we record performance comparable to the baseline. These results imply that as the number of errors in an utterance increases, our framework's finer granularity and the complementary interplay between the detector and corrector bolster resilience across diverse error patterns, thereby mitigating performance degradation more effectively than the baseline.

Although the overall **WER** gains over the zero-rule baseline might appear modest, they remain notable achievements, given the relative sparsity of errors even in erroneous utterances and the resulting difficulty of inflating the baseline's **WER**. This robustness underscores our framework's practicality for real-world deployment.



(a) Error-span Count Statistics (b) Detection Performance (Bal T-Acc) (c) Correction Performance (WER)

Figure 6: Impact of error-span count. The left histogram shows the distribution of utterances by number of error spans. The middle plot charts token-level detection accuracy, and the right plot charts correction WER, comparing our detector and corrector (blue) against a zero-rule baseline (red). Although accuracy declines as errors accumulate, our framework consistently outperforms the baseline across all error counts.

Korean proper nouns

U = 예 시민님 망으로 *** 주소 확인했는데 어떤 거 때문에 그러십니까 (Yes, citizen, I've checked the address at Mhang-uh street ***; what seems to be the issue?)

G = 예 시민님 망우로 *** 주소 확인했는데 어떤 거 때문에 그러십니까? (Yes, citizen, I've checked the address at Mhang-oo street ***; what seems to be the issue?)

$\hat{G}$ = 예 시민님 망우로 *** 주소 확인했는데 어떤 거 때문에 그러십니까? (Yes, citizen, I've checked the address at Mhang-oo street ***; what seems to be the issue?)

Utterance with extensive filler words

U = 그렇죠 예 예 종료가 돼야 돼 그런 그런 분은 하는데 좀 예 예 예 아니 규양 저 교양교육을 좀 시키셔야 되는 거 아니에요 그게 (Right, yeah, yeah, it has to end. Those kinds of people do it, but, uh, yeah, ah, yes, yes, yes. No, livoral, you need to give them some liberaleducation, don't you? That's)

G = 그렇죠 예 예 종료가 돼야 돼 그런 그런 분은 하는데 좀 예 예 예 아니 교양 저 교양 교육을 좀 시키셔야 되는 거 아니에요 그게 (Right, yeah, yeah, it has to end. Those kinds of people do it, but, uh, yeah, ah, yes, yes, yes. No, liberal, you need to give them some liberal education, don't you? That's)

$\hat{G}$ = 그렇죠 예 예 종료가 돼야 돼 그런 그런 분은 하는데 좀 예 예 예 아니 교양 저 교양 교육을 좀 시키셔야 되는 거 아니에요 그게 (Right, yeah, yeah, it has to end. Those kinds of people do it, but, uh, yeah, ah, yes, yes, yes. No, liberal, you need to give them some liberal education, don't you? That's)

Undetected errors

U = 예 욕실장 매출 신고라는 말씀이세요? (Yes, you mean bathhouse sales report?)

G = 예 욕실장 배출 신고라는 말씀이세요? (Yes, you mean bathhouse discharge report?)

$\hat{G}$ = 예 욕실장 매출 신고라는 말씀이세요? (Yes, you mean bathhouse sales report?)

Incorrect corrections

U = 아 그러니까 경찰서에 먼저 점하고 얘한테 기다리라고 법률 판을 불러줬어요. (Ah, so I first cold the police station and, to him to wait, called out a legal plate.)

G = 아 그러니까 경찰서에 먼저 전화하고 얘한테 기다리라고 하고 번호판을 불러줬어요. (Ah, so I first called the police station, told him to wait, and then called out the license plate.)

$\hat{G}$ = 아 그러니까 경찰서에 먼저 전화하고 얘한테 기다리라고 하고 법무부를 불러줬어요. (Ah, so I first called the police station, told him to wait, and called out the Ministry of Justice.)

Over-correction

U = 네 맞아요 아파트 또 나이 (yes, that's right apartment and age)

G = 네 맞아요 아파트 또 나이 (yes, that's right apartment and age)

$\hat{G}$ = 네 맞아요 아파트 단지 (yes, that's right apartment complex)

Figure 7: Case-study examples. The first two blocks (light blue background) illustrate scenarios where our corrector succeeds: (i) mistranscribed Korean proper nouns and (ii) utterances laden with filler words. The remaining three blocks (light red) highlight failure modes: (iii) undetected errors, (iv) incorrect corrections, and (v) over-corrections. Within each row, erroneous tokens are marked in red and their correct counterparts in blue.

7.3.4. Case Study

We conduct a qualitative analysis of the outputs generated by our corrector. As summarized in Fig 7, we identify two representative *success* scenarios and three representative *failure* scenarios.

Success Cases. **(1) Korean proper nouns.** Our corrector robustly fixes mistranscribed Korean-specific proper nouns, such as addresses, institutions, and event names. In the first example of Fig 7, the erroneous "Mahng-uh ro" is accurately corrected to "Mahng-oo ro (Mahng-oo Street)." Such corrections are difficult unless the backbone model possesses specialized knowledge of Korean toponyms, underscoring the advantage of adopting a Korean-pretrained model over a general-purpose LLM.
(2) Utterances with extensive filler words. Our corrector remains stable even when the input contains heavy verbal padding. The second example in Fig 7 shows an utterance cluttered with fillers, yet our framework successfully isolates

and amends the true error spans. This resilience stems from the finer granularity of both inputs and targets, which allows the model to ignore meaningless tokens and focus on genuinely erroneous segments.

Failure Cases. **(1) Undetected errors.** When the prepended detector misclassifies an erroneous utterance as error-free, the seq2seq corrector is never invoked. As illustrated in the third example, the detector overlooks the misrecognized word "maechul (sales)" because the utterance appears locally fluent; identifying the error would require broader dialogue context.
(2) Incorrect corrections. In certain customer utterances, the seq2seq corrector fails to propose an appropriate fix even after the detector flags an error. These cases typically involve severely fragmented or context-dependent speech, making it difficult to infer the intended form without additional audio or dialogue cues. The fourth example highlights such a scenario.
(3) Over-correction of fluent utterances. Occasionally, the two-stage pipeline mistakenly presumes that a fluent utterance is erroneous and generates an alternative form. The fifth example shows an utterance whose original version is arguably acceptable, yet our framework produces a more coherent variant. Although this technically constitutes an error, the resulting sentence may be more fluent than the ground truth.

Overall, these observations reaffirm the trade-off introduced by splitting dialogues into utterances: while the approach increases robustness to **token (span)-level** errors, it can falter when corrections require **dialogue-level** information. Nevertheless, given the poor performance of dialogue-level LLM baselines in our low-resource setting (Sec 3.1.2 and Tab 2), the benefits of granularity adjustment outweigh its limitations.

8. Discussion

8.1. Potential Limitations and Future Directions

Our study focuses on post-editing transcripts produced by a Korean ASR system, yet the proposed methodology could be extended to other languages and tasks. Evaluating whether the same granularity and detector-corrector design benefits other low-resource languages remains an important avenue for future work. Furthermore, while our experiments assume a text-only setting that reflects scenarios where raw audio is inaccessible, richer multimodal approaches could be explored when audio is available.

A key limitation of the current pipeline is its coarse interface between detector and corrector: although the detector outputs token-level masks, we decide whether to invoke the corrector solely at the utterance level. Passing token-level error locations directly to the corrector could yield more targeted edits, but it also risks propagating detector errors and increasing sensitivity to span design. We leave this finer integration for future research. Moreover, as discussed in Sec 7.3.4, finer granularity reduces the contextual information available to the model, potentially limiting the correction of errors that rely on broader dialogue context. Future studies might address this by incorporating compressed dialogue-context embeddings or utilizing context windows around target utterances.

Finally, although the dataset captures speech from a diverse user base, it lacks meta-information about speakers. Even minimal persona descriptors (e.g., regional background) could enable personalized correction strategies, such as dialect-aware editing. Addressing privacy concerns appropriately, future datasets could integrate such metadata to enhance correction accuracy.

8.2. Ethical and Copyright Considerations

Ethical considerations were carefully managed throughout this study, which involved analyzing real-world call-center data provided by the 120 Dasan Call Foundation, an official government organization in Korea. The creation of ground-truth annotations and the de-identification process were carried out securely by two trained Call Foundation employees with direct audio access (details in Sec 3.1.1). Only anonymized datasets were shared with Chung-Ang University strictly for research purposes.

Regarding copyright, the dataset derived from authentic call transcripts complies with legal standards for academic research, provided it is non-commercial. Our study adheres to these regulations, explicitly limiting the dataset's usage to scholarly research aimed at advancing language model development, without intent for distribution or commercial exploitation.

9. Conclusion

We introduced the first large-scale benchmark for Korean ASR error correction, comprising 105k call-center utterances with manually verified references, and used it to develop a two-stage detector-corrector pipeline that progressively refines granularity from dialogues to utterances and finally to token- and span-level edits. An encoder-based KoElectra detector and a span-focused pkoT5 corrector jointly deliver state-of-the-art detection and correction accuracy, outperforming dialogue-level and single-stage baselines. Extensive experiments spanning speaker roles, error density, and out-of-domain scenarios highlight the distinctive hurdles in correcting Korean ASR output. Collectively, our work provides a strong baseline and a shareable dataset that can drive future progress in Korean post-editing and other low-resource speech applications.

CRediT authorship contribution statement

Yonghyun Jun: Writing – original draft, Methodology, Formal analysis, Conceptualization, Validation, Investigation, Visualization. **Jimin Lee:** Writing – original draft, Methodology, Conceptualization, Visualization. **Hwan Chang:** Writing – review & editing, Writing – original draft, Methodology, Validation. **Dongho Shin:** Data Curation. **Seolah Kim:** Data Curation. **Hwanhee Lee:** Writing – review & editing, Supervision, Conceptualization, Project administration, Funding acquisition.

Declaration of competing interest

The authors confirm that they have no identified financial interests or personal relationships that could have potentially impacted the work presented in this paper.

Acknowledgements

This research was supported by the Chung-Ang University Graduate Research Scholarship in 2025. This research was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [RS-2021-II211341, Artificial Intelligent Graduate School Program (Chung-Ang University)].

A. Dataset Examples

This section presents concrete examples of the original **dialogue-level** dataset alongside the **utterance-level** version released for public use. The side-by-side tables illustrate the dataset's structure, typical content, and the outcome of our de-identification process. Fig 8 shows a raw dialogue-level sample, whereas Fig 9 displays the corresponding data after granularity refinement to utterances.

2024.05.09 124th Dialogue-level Transcription	**2024.05.09 124th Dialogue-level Ground-truth**
Agent: *** 다산콜센터 OOO입니다. 시민님 무엇을 도와드릴까요? (*** Dasan Call Center, OOO speaking. How may I help you, citizen?)	**Agent:** *** 다산콜센터 OOO입니다. 시민님 무엇을 도와드릴까요? (*** Dasan Call Center, OOO speaking. How may I help you, citizen?)
Customer: 네 안녕하세요 (Yes, hello.)	**Customer:** 네 안녕하세요 (Yes, hello.)
Agent: 네 감사합니다 무엇을 도와드릴까요? (Yes, thank you. What can I help you with?)	**Agent:** 네 감사합니다 무엇을 도와드릴까요? (Yes, thank you. What can I help you with?)
Customer: 인도에 탑차를 세워놔서 (They've parked a box truck on the sidewalk. . .)	**Customer:** 인도에 탑차를 세워놔서 (They've parked a box truck on the sidewalk. . .)
Agent: 예 타 (Yes, I see.)	**Agent:** 예 타 (Yes, I see.)
Customer: 보행하기가 힘들어 가지고요 (It's making it difficult to walk.)	**Customer:** 보행하기가 힘들어 가지고요 (It's making it difficult to walk.)
Agent: 예 탑차가 불법주차가 되어 있으니까 제가 접수하겠습니다. 신고하시는 장소가 어떻게 되십니까? (Yes, since that box truck is illegally parked, I will file a report. May I ask for the location you are reporting?)	**Agent:** 예 탑차가 불법주차가 되어 있으니까 제가 접수하겠습니다. 신고하시는 장소가 어떻게 되십니까? (Yes, since that box truck is illegally parked, I will file a report. May I ask for the location you are reporting?)
Customer: 서울마당로 ***이요. (It's on Seoul Madang-street, number ***)	**Customer:** 어울마당로 ***이요. (It's on Eoul Madang-street, number ***)
Agent: 어울 마당로 ***번 (Eoul Madang-street, number ***)	**Agent:** 어울 마당로 ***번 (Eoul Madang-street, number ***)
Customer: 네 인도에다가 차를 세워놨어요 (Yes, they've parked the vehicle on the sidewalk.)	**Customer:** 네 인도에다가 차를 세워놨어요 (Yes, they've parked the vehicle on the sidewalk.)

Figure 8: Dialogue-level Dataset: 2024.05.09 124th Call

Speaker	Key	Utterance-level Transcription	Utterance-level Ground-truth
Agent	240509124	*** 다산콜센터 OOO입니다. 시민님 무엇을 도와드릴까요? (*** Dasan Call Center, OOO speaking. How may I help you, citizen?)	*** 다산콜센터 OOO입니다. 시민님 무엇을 도와드릴까요? (*** Dasan Call Center, OOO speaking. How may I help you, citizen?)
Customer	240509124	네 안녕하세요 (Yes, hello.)	네 안녕하세요 (Yes, hello.)
Agent	240509124	네 감사합니다 무엇을 도와드릴까요? (Yes, thank you. What can I help you with?)	네 감사합니다 무엇을 도와드릴까요? (Yes, thank you. What can I help you with?)
Customer	240509124	인도에 탑차를 세워놔서 (They've parked a box truck on the sidewalk. . .)	인도에 탑차를 세워놔서 (They've parked a box truck on the sidewalk. . .)
Agent	240509124	예 타 (Yes, box. . .)	예 타 (Yes, box. . .)
Customer	240509124	보행하기가 힘들어 가지고요 (It's making it difficult to walk.)	보행하기가 힘들어 가지고요 (It's making it difficult to walk.)
Agent	240509124	예 탑차가 불법주차가 되어 있으니까 제가 접수하겠습니다. 신고하시는 장소가 어떻게 되십니까? (Yes, since that box truck is illegally parked, I will file a report. May I ask for the location you are reporting?)	예 탑차가 불법주차가 되어 있으니까 제가 접수하겠습니다. 신고하시는 장소가 어떻게 되십니까? (Yes, since that box truck is illegally parked, I will file a report. May I ask for the location you are reporting?)
Customer	240509124	서울마당로 ***이고요. (It's on Seoul Madang-street, number ***)	어울마당로 ***이고요. (It's on Eoul Madang-street, number ***)
Agent	240509124	어울 마당로 ***번 (Eoul Madang-street, number ***)	어울 마당로 ***번 (Eoul Madang-street, number ***)
Customer	240509124	네 인도에다가 차를 세워놨어요 (Yes, they've parked the vehicle on the side-walk.)	네 인도에다가 차를 세워놨어요 (Yes, they've parked the vehicle on the side-walk.)

Figure 9: Utterance-level Dataset: 2024.05.09 124th Call

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used ChatGPT in order to improve readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

Ahlawat, H., Aggarwal, N., Gupta, D., 2025. Automatic speech recognition: A survey of deep learning techniques and approaches. International Journal of Cognitive Computing in Engineering .

Aycock, S., Stap, D., Wu, D., Monz, C., Sima'an, K., 2024. Can llms really learn to translate a low-resource language from one grammar book? arXiv preprint arXiv:2409.19151 .

Baevski, A., Zhou, Y., Mohamed, A., Auli, M., 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. Advances in neural information processing systems 33, 12449–12460.

Cahyawijaya, S., Lovenia, H., Fung, P., 2024. Llms are few-shot in-context low-resource language learners. arXiv preprint arXiv:2403.16512 .

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), pp. 4171–4186.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al., 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783 .

Dutta, S., Jain, S., Maheshwari, A., Pal, S., Ramakrishnan, G., Jyothi, P., 2022. Error correction in asr using sequence-to-sequence models. arXiv preprint arXiv:2202.01157 .

Gekhman, Z., Zverinski, D., Mallinson, J., Beryozkin, G., 2022. Red-ace: Robust error detection for asr using confidence embeddings. URL: https://arxiv.org/abs/2203.07172, arXiv:2203.07172.

Gulati, A., Qin, J., Chiu, C.C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., et al., 2020. Conformer: Convolution-augmented transformer for speech recognition. arXiv preprint arXiv:2005.08100 .

Guo, J., Wang, M., Qiao, X., Wei, D., Shang, H., Li, Z., Yu, Z., Li, Y., Su, C., Zhang, M., et al., 2024. Ucorrect: An unsupervised framework for automatic speech recognition error correction. arXiv preprint arXiv:2401.05689 .

Harvill, J., Khaziev, R., Li, S., Cogill, R., Wang, L., Chennupati, G., Thadakamalla, H., 2024. Significant asr error detection for conversational voice assistants, in: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 11606–11610.

Heeringa, W.J., 2004. Measuring dialect pronunciation differences using levenshtein distance .

Hrinchuk, O., Popova, M., Ginsburg, B., 2020. Correction of automatic speech recognition with transformer sequence-to-sequence model, in: Icassp 2020-2020 ieee international conference on acoustics, speech and signal processing (icassp), IEEE. pp. 7074–7078.

Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al., 2022. Lora: Low-rank adaptation of large language models. ICLR 1, 3.

Hurst, A., Lerer, A., Goucher, A.P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al., 2024. Gpt-4o system card. arXiv preprint arXiv:2410.21276 .

Jin, H.w., Lee, A., Chae, Y.J., Park, S.H., Kang, Y.J., Lee, S., et al., 2021. Error correction for korean speech recognition using a lstm-based sequence-to-sequence model. Journal of the Korea Society of Computer and Information 26, 1–7.

Leng, Y., Tan, X., Liu, W., Song, K., Wang, R., Li, X.Y., Qin, T., Lin, E., Liu, T.Y., 2023. Softcorrect: Error correction with soft detection for automatic speech recognition, in: proceedings of the AAAI conference on artificial intelligence, pp. 13034–13042.

Li, S., Chen, C., Kwok, C.Y., Chu, C., Chng, E.S., Kawai, H., 2024. Investigating asr error correction with large language model and multilingual 1-best hypotheses, in: Proc. Interspeech, pp. 1315–1319.

Loshchilov, I., Hutter, F., 2017. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 .

Ma, R., Gales, M.J., Knill, K.M., Qian, M., 2023a. N-best t5: Robust asr error correction using multiple input hypotheses and constrained decoding space. arXiv preprint arXiv:2303.00456 .

Ma, R., Qian, M., Gales, M., Knill, K., 2024. Asr error correction using large language models. arXiv preprint arXiv:2409.09554 .

Ma, R., Qian, M., Manakul, P., Gales, M., Knill, K., 2023b. Can generative large language models perform asr error correction? arXiv preprint arXiv:2307.04172 .

Malmi, E., Krause, S., Rothe, S., Mirylenka, D., Severyn, A., 2019. Encode, tag, realize: High-precision text editing. arXiv preprint arXiv:1909.01187 .

Meripo, N.V., Konam, S., 2022. Asr error detection via audio-transcript entailment. URL: https://arxiv.org/abs/2207.10849, arXiv:2207.10849.

Park, D., 2022. pko-t5: Paust korean t5 for text-to-text unified framework. URL: https://github.com/paust-team/pko-t5.

Park, J., 2020. Koelectra: Pretrained electra model for korean. https://github.com/monologg/KoELECTRA.

Park, S., Moon, J., Kim, S., Cho, W.I., Han, J., Park, J., Song, C., Kim, J., Song, Y., Oh, T., Lee, J., Oh, J., Lyu, S., Jeong, Y., Lee, I., Seo, S., Lee, D., Kim, H., Lee, M., Jang, S., Do, S., Kim, S., Lim, K., Lee, J., Park, K., Shin, J., Kim, S., Park, L., Oh, A., Ha, J., Cho, K., 2021. Klue: Korean language understanding evaluation. arXiv:2105.09680.

Prabhavalkar, R., Hori, T., Sainath, T.N., Schlüter, R., Watanabe, S., 2023. End-to-end speech recognition: A survey. IEEE/ACM Transactions on Audio, Speech, and Language Processing 32, 325–351.

Pu, J., Nguyen, T.S., Stüker, S., 2023. Multi-stage large language model correction for speech recognition. arXiv preprint arXiv:2310.11532 .

Stahlberg, F., Kumar, S., 2020. Seq2edits: Sequence transduction using span-level edit operations. arXiv preprint arXiv:2009.11136 .

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., Raffel, C., 2020. mt5: A massively multilingual pre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934 .

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., et al., 2024. Qwen2. 5 technical report. arXiv preprint arXiv:2412.15115 .

Yeen, H.Y., Kim, M.J., Koo, M.W., 2023. I learned error, i can fix it!: A detector-corrector structure for asr error calibration, in: Proc. INTERSPEECH, pp. 2693–2697.