On the Fragility of Modular Speech Translation: Whisper–T5 Alignment for Jeju Dialect Normalization

Korea University COSE461 Final Project

Douyoung Kwon
Korean Language and Literature
Team 8
2020130024

Yejin Hong
Department of Cyber Defense
Team 8
2021350218

Changyeop Lee
Department of Computer Science
Team 8
2020320060

Jonghyeon Park
Physical Education
Team 8
2019190775

Abstract

This paper investigates a modular speech-to-text translation system that converts Jeju dialect speech into standard Korean using a Whisper encoder, a connector module, and a T5 decoder. Despite the modularity's theoretical appeal, all connector-based configurations—MLP, Q-Former, and STE—perform poorly compared to a fine-tuned Whisper baseline, with BLEU scores near zero. Analysis reveals a mismatch between Whisper's acoustic–phonotactic representations and T5's syntactic–semantic expectations. UMAP visualizations confirm persistent latent-space separation. Interestingly, swapping to an English T5 decoder improves BLEU scores due to stronger pretraining and byte-level tokenization, not language compatibility. These results highlight the importance of decoder priors and representation alignment in low-resource dialect normalization.

1 Introduction

Recent advances in speech recognition technology have brought significant innovation to everyday language processing. However, most of these technologies are designed and trained primarily on standard language, and thus fall short when dealing with diverse linguistic variations such as regional dialects. This study aims to address this limitation by developing a model that takes Jeju dialect speech as input and converts it into standard Korean text. Jeju language is regarded not merely as a regional dialect but as an independent language, and it has been designated by UNESCO as a "critically endangered language," highlighting its cultural and linguistic value. Nevertheless, current automatic speech recognition (ASR) systems perform poorly on Jeju dialect, posing both a technical and cultural challenge.

Large-scale pretrained ASR models such as Whisper demonstrate strong generalization capabilities, but since they are predominantly trained on massive datasets of standard language speech, they fail to adequately handle distinctive speech variants like Jeju dialect. Our experiments confirm that simple fine-tuning is insufficient to achieve satisfactory performance due to the significant lexical, grammatical, and phonological differences between Jeju dialect and standard Korean. Moreover, although larger models like Whisper-large may offer improved performance, they are difficult to fine-tune or extend with additional translation modules in environments with limited GPU and memory resources.

Figure 1: Overview of the proposed Whisper–Connector–T5 model architecture.

Given these constraints, we explore an end-to-end speech recognition and translation architecture that combines a Whisper encoder with a large language model (LLM)-based decoder commonly used in natural language processing. In this architecture, the Whisper encoder extracts audio representations, and the text-based LLM decoder converts them into standard Korean text. This design offers the advantage of integrating recognition and translation into a single pipeline under resource-limited settings. In our study, we experiment with various configurations, such as modifying the connection between the encoder and decoder, and adjusting the freeze/unfreeze strategy of each component, to identify the optimal performance setup.

2 Related Work

Speech-to-Text (ST) translation converts speech in a source language into written text in a target language. Early systems followed a cascade pipeline—first Automatic Speech Recognition (ASR), then Machine Translation (MT)—but this design entails error propagation, higher latency, and separate training costs.[1] Consequently, current research has shifted toward end-to-end (E2E) models.

The large-scale pretrained ASR model Whisper[2] has become a strong baseline after task-specific fine-tuning. Yet Whisper is trained mostly on standard speech, so its generalization to non-standard varieties such as regional dialects remains limited.

Several works try to align pretrained ASR and MT modules inside a single E2E model. Alinejad and Sarkar[3] introduced adversarial regularization to close the ASR–MT representation gap, but their approach requires joint optimization of both encoders, limiting modularity. More recently, Sedláček et al.[4] proposed lightweight connector networks (Q-Former, STE) that map frozen ASR embeddings into the MT space. Although effective, Q-former

3 Approach

In this work, we adopt the alignment-based speech translation framework proposed by Sedláček et al.[4] Our goal is to convert Jeju dialect speech into standard Korean text, leveraging large-scale pretrained models. The system consists of a Whisper encoder, a trainable connector module, and a T5 decoder.

3.1 Architecture Overview

Our architecture follows the Encoder-Connector-Decoder (ECD) alignment configuration as described in Sedláček et al.[4]. A high-level overview is shown in Figure 1.

The Whisper encoder f_{ASR} processes the input audio $x \in \mathbb{R}^T$ into a sequence of hidden representations with sequence length n_a and hidden dimension d_a:

$$H_{ASR} = f_{ASR}(x) \in \mathbb{R}^{n_a \times d_a}. \tag{1}$$

A connector module f_{conn} transforms these speech embeddings into a representation space compatible with the T5 decoder(d_t is hidden size of T5 decoder and n_s is sequence length):

$$Z = f_{conn}(H_{ASR}) \in \mathbb{R}^{n_s \times d_t}. \tag{2}$$

The output Z is then passed to the cross-attention block of the T5 decoder, which generates the final standard Korean translation:

$$y = f_{T5}(Z). \tag{3}$$

3.2 Connector Module: Q-Former and STE

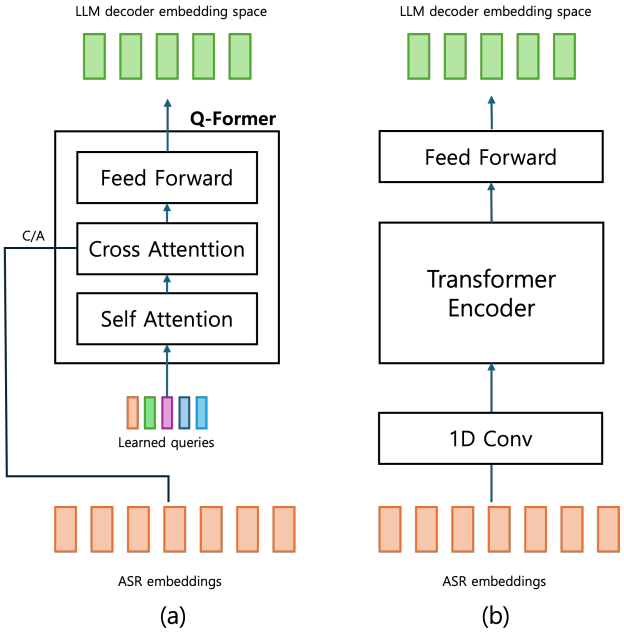

Figure 2: Diagram of the two connector modules. (a) is the Q-Former, (b) is the STE connector.

We use the Q-Former described in Figure3, a lightweight Transformer-based connector. It consists of a fixed set of learned query embeddings $Q \in \mathbb{R}^{n_q \times d_t}$($n_q$ is number of queries), which attend to the ASR output via cross-attention layers:

$$Z = f_{Q-Former}(H_{ASR}) \in \mathbb{R}^{n_q \times d_t} \tag{4}$$

Each query vector aggregates and transforms information from the speech sequence via multiple attention heads and feedforward layers. The output of the Q-Former is directly projected into the T5 decoder's cross-attention layer without passing through the T5 encoder.

Secondly, we use STE(Subsampler-Transformer Encoder) described in Figure3. The subsampling module is 2-layer stack of 1D convolutions, which reduces input ASR embeddings sequence length into $S \in \mathbb{R}^{n_a/4 \times d_c}$. The subsampler is then followed by a stack of transformer encoder blocks, and finally, the output embeddings are projected into the LLM decoder dimension d_t by a feed forward layer.

$$Z = f_{STE}(H_{ASR}) \in \mathbb{R}^{n_a/4 \times d_t} \tag{5}$$

Since neither the Q-Former nor the STE module released their code in previous studies, we implemented them from scratch. For more details, please refer to our GitHub repository.[5]

3.3 Baseline: Fine-tuned Whisper

As a strong baseline, we fine-tune Whisper on the same Jeju dialect–standard Korean dataset. The model is trained end-to-end to produce standard Korean transcriptions. This allows us to compare our modular alignment approach against a fully learned, monolithic system.

3.4 Implementation Details

We use the official Hugging Face implementations of Whisper and T5. The Q-Former connector is implemented following the code and design principles outlined in the BLIP-2 framework [6] and adapted for speech input as described by Sedláček et al. [4]. All training code for the connector is implemented by us and will be released publicly.

Since our implementation reuses the model structure and methodology of Sedláček et al. [4] as-is, we do not claim novelty in the model design. Our contribution lies in empirically validating this framework in a new language and domain: dialect-to-standard normalization in Korean. Additionally, we conducted experiments to evaluate the effectiveness of Q-Former and STE modules as alternatives to the MLP connector. We also constructed a paired dataset of Jeju dialect and English translations, and performed a task of translating Jeju dialect into English. Through this, we investigated how the prior knowledge of LLMs and domain mismatch affect end-to-end speech-to-text (ST) translation.

4 Experiments

4.1 Data

In this study, we utilized the Korean Dialect Speech (Jeju) dataset[1] provided by AI Hub. This dataset consists of speech recordings from Jeju region speakers paired with corresponding standard Korean text. It is a large-scale resource totaling approximately 600GB, constructed for the preservation and technological utilization of endangered languages such as Jeju dialect.

Due to limited computational resources (GPU and memory constraints), our team selectively used a portion of the dataset: approximately 70GB for training, 13GB for validation, and 5GB for testing.

Although the dataset provides raw audio files along with their corresponding standard Korean transcripts, many of the audio clips are too long to be directly used for model training. To address this, we segmented all audio files at the sentence level by using the start and end time annotations provided in the dataset. As part of preprocessing, we removed sentences containing stopwords or ambiguous text, which accounted for 3.9

4.2 Evaluation method

In this study, we evaluate the performance of an end-to-end model that converts Jeju dialect speech into standard Korean text using three metrics: BLEU (Bilingual Evaluation Understudy), WER (Word Error Rate), and CER (Character Error Rate). These metrics are widely used in the fields of machine translation and automatic speech recognition (ASR), making them suitable for the characteristics of our task.

BLEU is a standard metric in machine translation that computes a score based on the n-gram overlap between the generated sentence and the reference sentence. It is useful for quantifying how naturally and contextually appropriate the model generates standard Korean sentences, and is thus well-suited for evaluating the fluency and adequacy of the translation.

WER measures the quality of ASR outputs by calculating the differences between the predicted and reference sentences based on insertion, deletion, and substitution errors. It allows for a quantitative assessment of how accurately Jeju dialect speech is converted into standard Korean.

CER is a finer-grained version of WER that measures errors at the character level rather than the word level. Given that dialectal speech often includes many phonetic variations, character-level evaluation is important for capturing subtle differences and can reflect recognition accuracy at the phoneme level.

4.3 Experimental details

Whisper Baseline. We fine-tune the Whisper encoder and decoder as a standalone model on dialect speech input. The training is performed using a batch size of 2 with gradient accumulation steps set to 2, effectively resulting in an update batch size of 4. We use a learning rate of 1×10^{-4} and train for 5 epochs using the Adam optimizer.

Whisper + Connector + T5. We adopt a two-stage encoder-decoder architecture consisting of the Whisper encoder and a T5 decoder. For the Korean model, we use `paust/pko-t5-base`, and for the English variant, we use `google/t5-base`. The models are trained for 5 epochs with a base batch

[1] `https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=121`

Table 1: BLEU, WER, and CER scores for the Whisper–Connector–T5 model under various training configurations. **FF** denotes the freeze–freeze setting, while **UF** and **FU** represent unfreeze–freeze and freeze–unfreeze settings, respectively. **En** indicates that the T5 English model is used for the Jeju dialect speech-to-English translation task.

Connector	**BLEU** ↑	**WER** ↓	**CER** ↓
MLP (ff)	0.03	1.35	1.03
MLP (fu)	0.03	1.35	1.03
MLP (uf)	0.04	1.14	0.85
Q-Former (ff)	**0.03**	**1.41**	**1.09**
STE (ff)	0.01	1.01	0.93
Whisper Finetuning (baseline)	10.51	1.44	1.26
Q-Former (En-ff)	0.30	1.14	0.81
STE (En-ff)	0.47	1.10	0.80

size of 16. When either the Whisper encoder or T5 decoder is unfrozen, we reduce the batch size to 8 due to GPU memory constraints. Optimization is performed using the Adam optimizer with a learning rate of 5×10^{-5}, a warmup ratio of 0.1, and a weight decay of 0.01.

In addition to Korean generation, we also perform a dialect-to-English translation task by pairing Whisper with the `google/t5-base` decoder. To construct training data, we take the standard Korean references from the existing AI Hub Jeju corpus and translate them into English using GPT-4.1 nano API. This allows us to evaluate the impact of decoder language and pretraining scale on alignment performance.

Connector Architectures. We experiment with two connector modules to bridge the Whisper encoder and the T5 decoder:

- **Q-Former:** A transformer-based connector with 6 layers, 100 learnable queries, and a hidden dimension of 256.
- **STE (Simple Transformer Encoder):** A 6-layer transformer encoder with a hidden dimension of 256.

4.4 Results

Table1 shows the performance of models that apply various connection methods such as MLP, Q-Former, and STE along with the combination of the encoder and decoder (freeze/unfreeze) based on the Whisper encoder and T5 decoder structure. We expected that these models would perform better than finetuning the Whisper sole model. Because Whisper is specialized in standard language and has low performance on Jeju language, we thought that better translation results would be obtained using large language models such as T5 and connectors. However, the actual experimental results turned out to be significantly poorer than we anticipated.

The BLEU scores were extremely low in most model evaluations, ranging from 0.03 to 0.04. This appears to be due to the decoder frequently generating sentences that were completely unrelated to the input speech, or repeatedly producing meaningless tokens. In addition, WER and CER were generally worse than those of the baseline. In some experiments, the WER and CER values exceeded 1.0, suggesting that the predicted sentences were excessively longer than the reference sentences or suffered from frequent insertion errors.

As a result, while the Whisper-T5 architecture is resource-efficient and theoretically promising, it underperformed in our current experiments. These results have led us to investigate potential causes of performance degradation, including data quality issues, the choice of pretrained models, and design flaws in the model architecture.

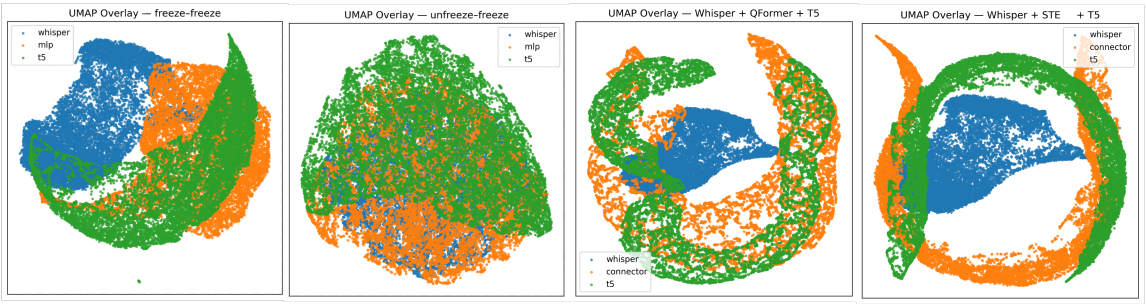

Figure 3: Diagram of UMAP. (a) and (b) use an MLP as the connector, while (c) and (d) use Q-Former and STE as the connector, respectively. In (a), the Whisper encoder is frozen and the T5 decoder is also frozen. In (b), only the T5 decoder is frozen. In (c) and (d), both the Whisper encoder and T5 decoder are frozen.

5 Analysis

5.1 Quantitative Results

Table 1 shows BLEU scores between 0.01 and 0.04, and WER/CER values exceeding 1.0 across the board, indicating that every configuration underperforms even a vanilla Whisper ASR baseline. Notably, altering the freeze policy has *negligible* impact: MLP(*ff*) and MLP(*fu*) yield identical scores, while MLP(*uf*) gains a marginal CER improvement but still remains unusable. The best relative outcome—STE(*ff*) with WER 1.01—suggests that connector architecture exerts more influence than parameter count, yet the gain is far from practical. Collectively, the data point to a fundamental incompatibility rather than a capacity bottleneck.

5.2 Representation–Space Analysis (UMAP)

Figures 3a–d plot Whisper (blue), connector (orange), and T5 (green) embeddings after dimensionality reduction. In (a), (c), and (d), the three modalities are clearly separated, with (a)—which uses an MLP connector—showing almost completely disjoint clusters. Even in (b), where the Whisper encoder is unfrozen, the separation between the embeddings remains, and alignment is not significantly improved. In (c) and (d), which use Q-Former and STE respectively, the connector embeddings (orange) lie along a curved manifold between the Whisper and T5 clusters, forming a "two-ring" structure. This suggests that the connector is not fully aligning the two representation spaces, but rather linking them through intermediate interpolation. Since UMAP preserves local neighborhoods but distorts global geometry, superficial overlap does not necessarily indicate lexical alignment. Nevertheless, the consistent separation across embeddings aligns well with the observed low BLEU/WER scores.

5.3 Failure Hypotheses

Latent-space mismatch Whisper focuses on acoustic and phonotactic features, while T5 expects syntactic and semantic token representations. Because these two types of information are so different, it's naturally difficult to connect them using a simple connector. Recent models like SpeechT5 [7] address this issue by learning speech and text representations together, which supports the idea that this mismatch is a real problem.

Objective misalignment The model is trained using only decoder cross-entropy, which means it doesn't directly encourage the speech and text features to match well. In other speech translation models [8][9][10], special alignment losses—like contrastive or distillation losses—are added to help the speech and text representations line up better, and these methods have shown better results.

Decoder prior & tokenizer granularity Table 1 shows that BLEU scores are higher when using the English T5 decoder. This highlights the importance of decoder priors in cross-modal generation tasks. The performance gain in the English setting is not due to a better language match, but rather because the English decoder has stronger priors and uses a byte-level tokenizer that happens to align

well with Whisper's syllable rhythm. In contrast, the public PKO-T5-BASE model is trained solely on Korean-specific datasets such as Namuwiki, Korean Wikipedia, and the Modu Corpus, with the exact data size unspecified. Meanwhile, GOOGLE/T5-BASE has been pre-trained on approximately 750GB of the C4 corpus and further fine-tuned on over 20 supervised downstream tasks. As a result, when both Whisper and T5 are frozen, the Korean decoder provides little guidance and collapses into short or repetitive outputs. The English decoder—despite lacking knowledge of Hangul—can still rely on its rich sequence-to-sequence priors to generate plausible outputs from byte-level fallback tokens. Furthermore, byte-level subwords help align Whisper frames to target tokens on a nearly one-to-one basis, drastically simplifying the alignment problem. Similar advantages of byte-level tokenization have been reported in ablation studies on ByT5 [11].

Take-away. The performance jump after swapping the decoder is *not* evidence that Whisper encodes English better; it exposes how brittle a low-capacity connector becomes when paired with a decoder that (i) was under-trained and (ii) uses coarse morpheme-level tokens. Strengthening the Korean decoder (extra supervised pre-training) or switching to a byte-level Korean variant is therefore a more principled fix than abandoning Korean altogether.

6 Conclusion

In this project, we explored an end-to-end speech translation architecture combining Whisper and T5, aimed at converting Jeju dialect speech into standard Korean text. Despite the intuitive appeal of modularizing the encoder and decoder via a connector, our experiments showed that such configurations—particularly under freeze–freeze training constraints—failed to match the performance of a simple fine-tuned Whisper baseline.

The main findings are as follows:

1. Connector modules (MLP, Q-Former, STE) failed to bridge the representational gap between Whisper's acoustic–phonotactic embeddings and T5's syntactic–semantic expectations.

2. Quantitative metrics (BLEU, WER, CER) indicate that all connector models severely underperform, often generating semantically irrelevant or repetitive outputs.

3. UMAP analyses confirmed a persistent separation in representation space, supporting the hypothesis of latent-space mismatch.

4. Using the English T5 decoder significantly improved BLEU scores, highlighting the importance of decoder prior strength and tokenizer granularity. This improvement was not due to better linguistic match but rather to the English decoder's richer pretraining (750GB+ of supervised data and over 20 tasks) and its use of byte-level tokenization, which aligns well with Whisper's frame structure. By contrast, the Korean PKO-T5-BASE decoder, trained only on a small Korean-specific dataset with limited supervision, offered little guidance when frozen.

Through this investigation, we learned that cross-modal alignment is not a trivial plug-and-play task, especially when pretrained components operate in mismatched representational domains. A strong decoder prior, fine-grained tokenization, and supervised pretraining are critical for bridging such gaps.

6.1 Future Directions

Joint CTC+Seq2Seq Training. Attach a CTC head to the Whisper encoder during fine-tuning to anchor phonetic cues while the connector learns higher-level abstractions.

Contrastive Alignment. Borrow InfoNCE or discrete alignment losses from SLAM/Mu^{2}SLAM to explicitly minimise speech–text distance.

Two-Stage Bridging. First distil Whisper into a text-encoder (e.g. Marian) to obtain a truly textual latent; then fine-tune a T5 decoder on that space, decoupling acoustic noise from generation.

Connector Capacity Sweep. Expand STE into a Mixture-of-Experts or depth-wise Q-Former to test whether richer cross-modal attention improves recognition without unfreezing Whisper/T5.

References

[1] Nivedita Sethiya and Chandresh Kumar Maurya. End-to-end speech-to-text translation: A survey, 2024.

[2] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[3] Ashkan Alinejad and Anoop Sarkar. Effectively pretraining a speech translation decoder with machine translation data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8014–8020, 2020.

[4] Šimon Sedláček, Santosh Kesiraju, Alexander Polok, and Jan Černockỳ. Aligning pre-trained models for spoken language translation. *arXiv preprint arXiv:2411.18294*, 2024.

[5] Douyoung Kwon. Jeju dialect speech-to-text. `https://github.com/douyoung89/Jeju-dialect-speech-to-text`, 2025. Accessed: 2025-05-30.

[6] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023.

[7] Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, Zhihua Wei, Yao Qian, Jinyu Li, and Furu Wei. Speecht5: Unified-modal encoder-decoder pre-training for spoken language processing, 2022.

[8] Rong Ye, Mingxuan Wang, and Lei Li. Cross-modal contrastive learning for speech translation. *arXiv preprint arXiv:2205.02444*, 2022.

[9] Ziqiang Zhang, Long Zhou, Junyi Ao, Shujie Liu, Lirong Dai, Jinyu Li, and Furu Wei. Speechut: Bridging speech and text with hidden-unit for encoder-decoder based speech-text pre-training. *arXiv preprint arXiv:2210.03730*, 2022.

[10] Hao Zhang, Nianwen Si, Yaqi Chen, Wenlin Zhang, Xukui Yang, Dan Qu, and Wei-Qiang Zhang. Improving speech translation by cross-modal multi-grained contrastive learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1075–1086, 2023.

[11] Yi Tay Xu, Noam Shazeer, and Donald Metzler. Byt5: Towards a token-free future with pre-trained byte-to-byte models. In *Proc. ACL*, 2021.

A Appendix: Team contributions

Douyoung Kwon : Implementation of training code and Writing the paper. Changyeop Lee : Implementation and experimentation of Q-Former and STE Yejin Hong : Implementation and experimentation of baseline and UMAP Jonghyeon Park : Writing the paper and translating the dataset into English using OpenAI API