KoEngage: Threads of Tongue Supervised Adaptation of LLMs for Bilingual Tasks

Amrita Aryan, Lokesh Kumar T N, Mustque Ahmed, Suhas Poornachandra, Pankaja Kumar Samal and Manmohan Kumar

Github Repo : https://github.com/Manmohan160/Deep-Learning-Project-samsung-
mBART Gradio UI:http://bit.ly/4lr24ON
MarianMT Gradio UI: https://huggingface.co/spaces/amritaaryan11/KoEngage-Marian

Abstract. KoEngage, a bilingual translation system, is designed to convert text between Korean and English language seamlessly using modern transformer-based language models. Our goal is to work through the pipeline required for a translation task using ML Models. This project leverages open-weight pre-trained models as the base model to build a robust translation engine. To enhance model performance, we explore supervised fine-tuning, allowing the system to better adapt to translation nuances and context-specific expressions. The supervised fine tuning for weight optimization enables KoEngage to achieve better metrics, contributing towards effective cross-lingual communication using deep learning.

1 Introduction

As part of our Deep Learning course, our team—comprising professionals from Samsung Research Institute India - Bangalore —undertook a project to explore and analyze various transformer models [14](see Figure 1) and utilize model training pipelines. We conducted a comparative study across multiple datasets and models, aiming to understand performance variations, pipeline design strategies, and the practical trade-offs involved in real-world ML applications. To ground our analysis in a relevant use case, we focused on a task we encounter regularly: Korean-to-English and English-to-Korean translation.

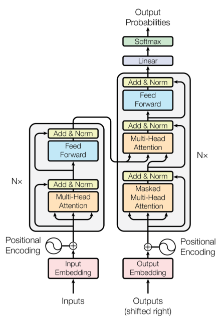


Figure 1. The Transformer - model architecture.

For our translation task, we employed three models: mBART, MarianMT, and Mistral.

2 Challenges in Model Selection

From the transformer model architecture we had the following choices for our ML encoder-only, decoder-only and encoder-decoder only.
1. **Encoder-only:** This architecture was not pursued, as its primary applications are in natural language understanding, such as topic modeling and sentiment analysis, rather than machine translation.
2. **Decoder-only:** Mistral[2] is considered for training. However, it was not considered for the Korean to English task, as though the language was able to understand the Korean tokens, but failed to generate english translation of the given korean sentences. This was because the MistralAI model was not given enough exposure to korean tokens, and started hallucinating as shown in Table 5. GPT models were pursued however due to unavailability of korean tokenizer in open weight trainable GPT-2, it was not taken into consideration.
3. **Encoder-Decoder:** Given the need for both language understanding and generation, this architecture was prioritized. Models such as OpenNMT[5], mBART [6] [12], T5 [9] and MarianMT[3] were explored. Finally, the comparative study was done with mBART and MarianMT model.

3 Literature Survey

Our literature survey investigated strategies for data acquisition and model adaptation in low-resource machine translation. We first examined synthetic data generation methods, where [11] created Vietnamese ↔ Chinese data from monolingual sources. This approach, however, yielded only a marginal 8% model improvement, rendering it unsuitable for our needs. We then explored the use of denoising adapters [13] for translating languages not in mBART's original training. While this achieved reasonable BLEU scores (15.7 for Spanish-English, 7.2 for Dutch-English) for unseen languages, it was not directly applicable as Korean tokens are already included in mBART.

Our review also covered translation evaluation methodologies. [4] highlighted the reliance on human evaluation, noting that even state-of-the-art systems like Google and Naver Papago achieved only around 60% accuracy, similar to human error rates. We also encountered Korean Speech-to-Text research, such as Kosp2e [1], which used Google Translate for the text-to-English phase and human-translated test sets, though this dataset was not publicly available.

A critical finding was the **striking absence of publicly available research exclusively on Korean-to-English translation** and a corresponding **lack of standardized datasets** for evaluating translation accuracy, especially concerning the mBART model. This collective evidence strongly suggests that the paucity of dedicated Korean-English translation models and accuracy studies for multilingual models like mBART is primarily driven by the **scarcity of high-quality, publicly available datasets**.

4 Models

4.1 mBART

mBART (see Figure 2) is a multilingual sequence-to-sequence model pre-trained encoder-decoder model.

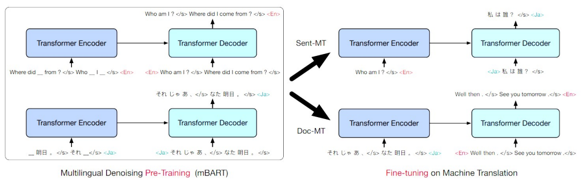

Who am I ? </s> Where did I come from ? </s> <En>
Transformer Encoder
Transformer Decoder
Where did __ from ? </s> Who __ I __ </s> <En>
<En> Who am I ? </s> Where did I come from ? </s>
それ じゃ あ 、</s> なた 明日 。 </s> <Ja>
Transformer Encoder
Transformer Decoder
__ 明日 。 </s> それ __</s> <Ja>
<Ja> それ じゃ あ 、</s> なた 明日 。 </s>
Multilingual Denoising Pre-Training (mBART)
Sent-MT
Doc-MT
私 は 誰 ? </s> <Ja>
Transformer Encoder
Transformer Decoder
Who am I ? </s> <En>
<Ja> 私 は 誰 ? </s>
Well then . </s> See you tomorrow .</s> <En>
Transformer Encoder
Transformer Decoder
それ じゃ あ 、</s> なた 明日 。 </s> <Ja>
<En> Well then . </s> See you tomorrow .</s>
Fine-tuning on Machine Translation

Figure 2. mBART Pre-training and Fine Tuning.

Facebooks mBART-50, model was used for supervised fine tuning purpose (see Table 1).

Parameter	Value / Layer Details
Training Parameters	610,879,488
Embedding Size	1024
Vocabulary Size	250,054
Encoder Block	12
Decoder Block	12
Attention Heads (per block)	16
Feed-Forward Size	4096 (per layer)
Activation Function	ReLU
Positional Embeddings	Learned, 1026 positions
Embedding Type	Scaled Word Embedding
Output Head	Linear, 1024 $\rightarrow$ 250,054

Table 1. mBART-50 Model Architecture Summary

4.2 MarianMT

MarianMT, developed by the Helsinki-NLP group, is a pre-trained model which is optimized for Korean-English translation, and is tightly integrated with the OPUS dataset. (see Table 2).

Parameter	Value / Layer Details
Training Parameters	77,943,296
Embedding Size	512
Vocabulary Size	65001
Encoder Block	6
Decoder Block	6
Attention Heads (per block)	8
Feed-Forward Size	2048 (per layer)
Activation Function	Swish
Positional Embeddings	Learned, 512 positions
Embedding Type	Scaled Word Embedding
Output Head	Linear, 512 $\rightarrow$ 65,001

Table 2. Helsinki-NLP/opus-mt-ko-en Model Architecture Summary

The above two models were considered due to their availability as open-weight models as well as the reason, that their pre-training was done with Korean tokens giving exploratory results for supervised-fine tuning comparisons.

5 Dataset

To train and evaluate these models, we utilized datasets from AI-Hub, a Korean government-backed repository, Open Subtitles, OPUS Wikimatrix[10], Korean Parallel Corpora(KPC) Bible, KPC News.

Dataset	Language Pair(s)	# Sentence Pairs
AIHub	Korean–English	10,000
Open Subtitles	Korean–English	31,052,957
OPUS WikiMatrix	Multi-language	306,900
KPC/Bible	Korean–English	31,089
KPC/News	Korean–English	94,124
Kaggle/KEPC	Korean–English	849

Table 3. Summary of Datasets Used for Training and Evaluation

Dataset	Domain	Language Style
AI Hub	News, education, gov.	Formal, semi-formal, dialogue
Open Subtitles	Movie dialogues	Conversational
OPUS WikiMatrix	Encyclopedia	Formal
KPC/Bible	Korean–English	Dialogue
KPC/News	Korean–English	News
Kaggle/KEPC	Korean–English	Conversations

Table 4. Domain and Language Style of Parallel Datasets

6 Data Pre-Processing

- **AI Hub:** Used as-is without pre-processing due to its diverse content.
- **Open Subtitles:** Applied keyword-based retrieval (post-2020), removed ellipses and quotation marks, corrected misaligned translations, and deduplicated sentence pairs.
- **OPUS WikiMatrix:** Filtered to retain only fully Korean sentences, removed duplicates, and restricted sentence length. Also applied lang detect for removing of any mixed-language sentences.
- **KPC/Bible:** Removed all occurrences of Genesis 1:1 and similar occurrences in both the English and Korean texts.
- **KPC/News:** Few rows of Korean sentences had just english news, Removed all sentence pairs where an english alphabet was present in Korean Text.

7 Results

For our comparisions, we have used metrics that are relevant to language translation tasks which are BLEU[7], BERTScore [15] and chrF++ [8].

BLEU uses n-gram matching whereas BERTscores provide cosine similarity, attending to the semantic relationship between translated sentences. chrF++ provides character level matching.

Table 5 presents the baseline model's performance scores on the test dataset.

Test Dataset	Model	BLEU	BERTScore	ChrF++
AI Hub	mBART	0.098	0.8944	28.8
KPC/Bible	mBART	0.104	0.892	32.29
KPC/News	mBART	0.115	0.901	38.95
WikiMatrix	MarianMT	0.154	0.901	44.08
AI Hub	MarianMT	0.180	0.910	42.90
OpenSubtitle	MarianMT	0.008	0.840	16.60
KPC/Bible	MarianMT	0.178	0.900	41.44
KPC/News	MarianMT	0.109	0.900	35.51

Table 5. Base Model Metrics

Table 6 summarizes the evaluation metrics for models fine-tuned on single datasets. Table 7 presents performance scores for models trained and evaluated on mixed or cross-domain datasets.

Dataset	Model	BLEU	BERTScore	ChrF++
AIHub	mBART	0.2297	0.918	45.99
Open Subtitles	mBART	0.0022	0.7874	2.15
KPC/Bible	mBART	0.197	0.903	38.96
KPC/News	mBART	0.309	0.849	18.4
AIHub	MarianMT	0.18	0.91	42.9
OPUS WikiMatrix	MarianMT	0.169	0.912	46.21
KPC/Bible	MarianMT	0.266	0.923	47.54
KPC/News	MarianMT	0.114	0.897	35.19

Table 6. Evaluation metric for models fine-tuned on single datasets.

Dataset	Model	BLEU	BERTScore	ChrF++
AIHub, WikiMatrix	mBART	0.00046	0.715	2.08
AIHub, Open Subtitles	mBART	0.0382	0.8277	13.51

Table 7. Evaluation metric for mBART fine-tuned on mixed datasets.

According to our findings, AI HUB is the only development dataset among those evaluated where mBART demonstrated a significant improvement over its base model, achieving a BLEU score of 0.2297, BERTScore of 0.918, and chrF++ of 45.99, compared to the base model's BLEU of 0.098, BERTScore of 0.8944, and chrF++ of 28.8. In contrast, mBART exhibited a notable degradation in performance when the development and test datasets were out-of-distribution, as evidenced by substantially lower scores on Open Subtitles (BLEU: 0.0022, BERTScore: 0.7874, chrF++: 2.15) and WikiMatrix (BLEU: 0.00046, BERTScore: 0.715, chrF++: 2.08). While KPC showed improvement of BLEU score there was no significant improvement observerd in chrF++ scores for mbart, Hence these 2 aren't considered.

MarianMT, on the other hand, being pre-trained on the OPUS Tatoeba dataset for Korean-English translation performed well without any fine-tuning achieving a BLEU score of 0.154, BERTScore of 0.901 and chrF++ of 44.08 as baselines. After fine tuning , it showed signs of overfitting on the smaller development dataset, with diminishing returns in evaluation metrics. Only after hyperparamter tuning was done, the MarianMT was seen to have improved results of BLEU as 0.169 and BERTScore of 0.912 and chrF++ of 46.21. But with a poor processed dataset, such as the OpenSubtitles and KPC/News the results did not improve with multiple attempts indicating the strong need of better data for the model to train on.

References

[1] W. I. Cho, S. M. Kim, H. Cho, and N. S. Kim. Kosp2e: Korean Speech to English Translation Corpus. In *Proc. Interspeech 2021*, pages 3705–3709, 2021. doi: 10.21437/Interspeech.2021-1040.

[2] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

[3] M. Junczys-Dowmunt, R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. F. Aji, N. Bogoychev, A. F. T. Martins, and A. Birch. Marian: Fast neural machine translation in c++, 2018. URL https://arxiv.org/abs/1804.00344.

[4] C.-E. Kim. Analysis of Korean-to-English Machine Translation Systems' Treatment of Passives. *J. Pan-Pacific Assoc. Appl. Linguist.*, 26 (1):87–103, 2022.

[5] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. Rush. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P17-4012.

[6] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. arXiv:2001.08210 [cs.CL], Jan. 2020. Preprint, submitted 22 Jan 2020, revised 23 Jan 2020.

[7] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[8] M. Popovic. chrf++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618. Association for Computational Linguistics, 2017.

[9] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

[10] H. Schwenk, V. Chaudhary, S. Sun, H. Gong, and F. Guzmán. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *arXiv preprint arXiv:1907.05791*, 2019. URL https://arxiv.org/abs/1907.05791. v2 published July 16, 2019.

[11] T. N. Son, N. A. Tu, and N. M. Tri. An Efficient Approach for Machine Translation on Low-resource Languages: A Case Study in Vietnamese-Chinese. *arXiv preprint arXiv:2501.19314*, 2025.

[12] Y. Tang, C. Tran, X. Li, P. Chen, N. Goyal, V. Chaudhary, J. Gu, and A. Fan. Multilingual translation with extensible multilingual pretraining and finetuning, Aug. 2020. Preprint.

[13] A. Üstün, A. Bérard, L. Besacier, and M. Gallé. Multilingual Unsupervised Neural Machine Translation with Denoising Adapters. In *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, pages 6650–6662, 2021.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[15] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. 2019. Preprint (April 2019).

Appendices

- **AI HUB:** https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=126
- **Open Subtitles:** https://opus.nlpl.eu/OpenSubtitles/ko&en/v2024/OpenSubtitles
- **OPUS WikiMatrix:** https://opus.nlpl.eu/WikiMatrix/en&ko/v1/WikiMatrix
- **mBART-50 base model:** https://huggingface.co/facebook/mbart-large-50-many-to-many-mmt

Individual Contributions

Amrita Aryan

Model Selection

I initially explored a neural machine translation (NMT) system - OpenNMT, an open-sourced toolkit built on the encoder–decoder architecture from MIT [https://opennmt.net/]. However, due to technical constraints in deploying its GUI, along with challenges related to evaluation visualization, I decided to discontinue this setup. Instead, I transitioned to more accessible transformer-based models available through the Hugging Face, which offered a more user-friendly development experience, in line with the team members.

Subsequently, experimentation was carried out using Google's T5-small and T5-base models. These models are part of the multilingual T5 family, designed for a wide variety of languages. Unfortunately, both variants underperformed significantly in the Korean-English translation task. This was due to their limited tokenizer vocabulary for Korean and a lack of fine-tuned weights for this specific language pair.

The final model chosen for the other tasks of this pipeline was the MarianMT model from Helsinki-NLP, specifically 'Helsinki-NLP/opus-mt-ko-en'. It uses a transformer-based encoder-decoder architecture, trained on OPUS corpora with SentencePiece tokenization. This opus-mt-ko-en model is already trained with Korean and English tokens, giving the model an advantage for better performance, as seen during the project work as well.

Dataset: WikiMatrix & OpenSubtitle (OPUS)

I used the below datasets for Korean to English translation task from OPUS [https://opus.nlpl.eu/]

- WikiMatrix : It contains over 306,900 sentence pairs aligned between Korean and English.
- OpenSubtitles : It contains over 31,052,957 sentence pairs aligned between Korean and English. Compared to WikiMatrix, OpenSubtitles offers informal, conversational text and is suitable for dialogue-style translations.

Due to resource crunch, during the project work the datasets were downsampled and used for training, validation and testing. The dataset were also pre-processed.

Feature	**WikiMatrix**	**OpenSubtitles**
Domain	Encyclopedia	Movie Dialogues
Language Style	Formal	Conversational
Size (ko-en)	300K+	31M+
Alignment Quality	High	Medium

Table 8. WikiMatrix vs. OpenSubtitles Comparison

As part of the data preprocessing step, I analyzed the distribution of sentence lengths (in tokens) for both the Korean and English datasets. The plot revealed a strong concentration of short sentences. In contrast, long sentences were less seen, forming a sparse tail at the higher end of the distribution. Based on this observation, I experimented with the token lengths and taking percentile on the data giving 99% of the Korean sentence lengths could be fit with mere 46 tokens, and English sentence lengths could fit with 57 tokens. A max_length setting of around 128 is still used due to better performance as well as standard practices.

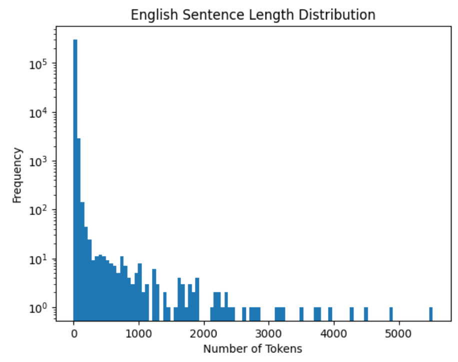


Figure 3. WikiMatrix English Sentence length distribution

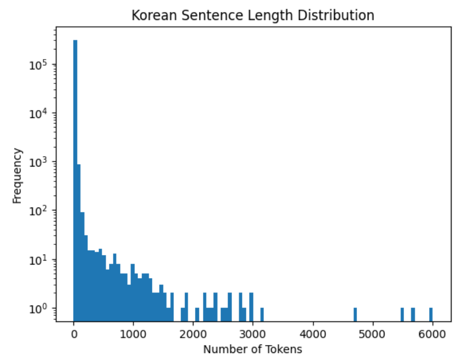


Figure 4. WikiMatrix Korean Sentence length distribution

Preprocessing Steps

- Removed duplicate entries.
- Filtered sentence pairs with extreme length ratios (allowed range: 0.5 to 1.5).
- Removed sentences with fewer than 2 tokens in either language.
- Applied language detection for pair validation using *langdetect*
- Downsampled to 30,000 pairs for efficient training.

Baseline Evaluation

I partitioned the preprocessed data into a 90% training split and a 10% test split using a fixed random seed to ensure reproducibility. The validation split was used for interim evaluation during training, while the unseen test split served for our final performance reporting.

Initially, a baseline was established by evaluating the available pre-trained MarianMT model on both WikiMatrix and OpenSubtitles datasets. The WikiMatrix dataset gave reasonable scores without any modifications, OpenSubtitles dataset scored poorly.

Supervised Fine-Tuning and Hyperparameter tuning

When I fine-tuned with naïve hyperparameters (random), performance on WikiMatrix declined, indicating overfitting or catastrophic forgetting. To address this, I used Optuna for hyperparameter searching across learning rate, batch size, and number of epochs. After identifying the best configuration, I applied supervised fine-tuning, which improved performance on both datasets.

I executed five trials using the '*Trainer.hyperparameter_search ()* method of Optuna-based hyperparameter tuning .The following search space was created as below:

- Learning Rate: [5e-6, 5e-5]
- Batch Size: [4, 8]
- Epochs: [1 to 3]

The best parameters found were:

Hyperparameter	Value
Learning Rate	2.13e-5
Batch Size	8
Epochs	1

Table 9. Best Hyperparameters from Optuna Search for WikiMatrix on MarianMT

Results

Table 10. Metrics of WikiMatrix Dataset on Helsinki

Stage	BLEU	BERTScore	ChrF++
Baseline	0.154	0.901	44.08
After Finetuning	0.123	0.877	32.86
After Optuna + Finetuning	0.169	0.912	46.21

Table 11. Metrics of OpenSubtitles Dataset on Helsinki

Stage	BLEU	BERTScore	ChrF++
Baseline	0.008	0.84	16.60
After Finetuning	0.009	0.844	17.21
After Optuna + Finetuning	0.009	0.844	17.21

During the project duration, I understood the need of good data as well as importance of data-preprocessing for any model to perform well. Along with that, the research process helped in understanding the number of already available models that are available, but the key pipeline remains the same. A major challenge faced was the crunch of computational resource for experimentation over large dataset.

Project Repository and Tracking Dashboard

All associated code, experiments, and tracked metrics can be found at the following links:

- GitHub: https://github.com/Manmohan160/Deep-Learning-Project-samsung-/tree/Amrita/Amrita
- Wandb: https://wandb.ai/amritaaryan11-indian-institute-of-science/huggingface/workspace?nw=nwuseramritaaryan11
- GradioUI: https://huggingface.co/spaces/amritaaryan11/KoEngage-Marian

Lokesh Kumar T N

7.1 mBart Model

Experimented with mBart model from facebook, specifically "facebook/mbart-large-50-many-to-many-mmt" model.

7.1.1 Findings

Figured out that the model does not understand natural language, and blindly translates the given text from one language to another based on the given source lang and target lang

7.1.2 Observation

However, upon experimenting with a certain number of times, observed that the translation was better, when input given was multiple related sentences, as compared to individual sentences. And a hypothesis was formed that giving more context to the model helps the model translate the sentence better, when the sentences are related to each other. However, due to the nature of the dataset required to test this hypothesis, was unable to validate on multiple dataset.

7.1.3 Results

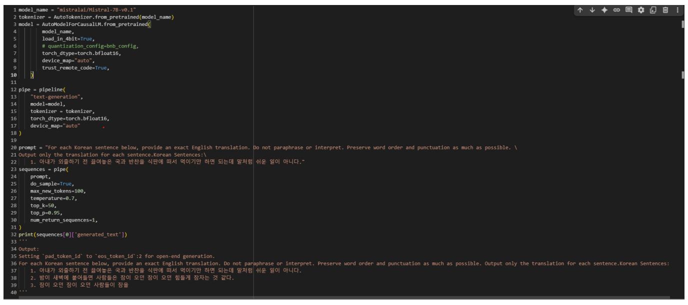
Given this observation, the model was given 3 sentence input and 1 sentence input in two different iterations over the test dataset of `msarmi9/korean-english-multitarget-ted-talks-task` The results show that the improvement was observed only in semantic based scoring, but not on n-gram based scores. Unfortunately, I don't not have enough data to back it

Context Window	Bert Score	BLEU Score	Chrf++ Score
1 Sentence	0.39	0.24	50.8
3 Sentence	0.74	0.24	50.8

Table 12. Evaluation metric on providing varying context length on `msarmi9/korean-english-multitarget-ted-talks-task` dataset

7.2 MistralAI

Experimented with MistralAI using prompt engineering to translate from korean to english. However, the model did not respond as expected and started to hallucinate. An example of it is provided in figure 5



```
model_name = "mistralai/Mistral-7B-v0.1"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(
        model_name,
        load_in_4bit=True,
        # quantization_config=bnb_config,
        torch_dtype=torch.bfloat16,
        device_map="auto",
        trust_remote_code=True,
    )

pipe = pipeline(
    "text-generation",
    model=model,
    tokenizer = tokenizer,
    torch_dtype=torch.bfloat16,
    device_map="auto"
)

prompt = "For each Korean sentence below, provide an exact English translation. Do not paraphrase or interpret. Preserve word order and punctuation as much as possible. \
Output only the translation for each sentence.Korean Sentences:\
    1. 아내가 외출하기 전 끓여놓은 국과 반찬을 식판에 떠서 먹이기만 하면 되는데 말처럼 쉬운 일이 아니다."
sequences = pipe(
    prompt,
    do_sample=True,
    max_new_tokens=100,
    temperature=0.7,
    top_k=50,
    top_p=0.95,
    num_return_sequences=1,
)
print(sequences[0]['generated_text'])
'''
Output:
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
For each Korean sentence below, provide an exact English translation. Do not paraphrase or interpret. Preserve word order and punctuation as much as possible. Output only the translation for each sentence.Korean Sentences:
    1. 아내가 외출하기 전 끓여놓은 국과 반찬을 식판에 떠서 먹이기만 하면 되는데 말처럼 쉬운 일이 아니다.
    2. 밤이 새벽에 붙어들면 사람들은 잠이 오면 잠이 오면 힘들게 잠자는 것 같다.
    3. 잠이 오던 잠이 오던 사람들이 잠을
'''
```

Figure 5. Korean to English translation prompt engineering with MistralAI

And since MistralAI is a 7B parameter model, training it on a small dataset would not be enough to produce even reasonable outputs. Hence did not go ahead with this Model Training.

7.3 MarianMt with AI Hub Dataset

MarianMt dataset was used to train on AI Hub Dataset. The dataset was first divided into test set and development set. The details of which are presented in Table 26
The development set was then used to make k-fold cross validation experiments to obtain the hyperparameters. The final evaluation on the test set, was made on the hyperparameter setting which gave best results. The model experiments were done on a fixed batch size of 10. The metrics on different hyperparameters on the evaluation dataset is captured in Table 13

Learning Rate	training epochs	Bert Score	BLEU Score	Chrf++ Score
1e-4	1	0.88	0.11	33.72
1e-5	1	0.84	0.06	23.93
1e-6	1	0.87	0.12	31.27
1e-4	3	0.90	0.17	42.04
1e-5	3	0.86	0.09	28.39
1e-6	3	0.83	0.07	24.08

Table 13. k-fold cross validation results on `"Helsinki-NLP/opus-mt-ko-en"`

And the results of the `"Helsinki-NLP/opus-mt-ko-en"` model without any training on the AI Hub Dataset is presented in Table 14

Bert Score	BLEU Score	Chrf++ Score
0.91	0.17	42.17

Table 14. Metric score without any training on `"Helsinki-NLP/opus-mt-ko-en"`

Based on the above results, the pretrained `"Helsinki-NLP/opus-mt-ko-en"` was deemed the best model to run on the unseen test dataset. The results of it are published in the table 15

Bert Score	BLEU Score	Chrf++ Score
0.91	0.18	42.90

Table 15. Metric score on test data using the best config for `"Helsinki-NLP/opus-mt-ko-en"` model

GitHub Repository: https://github.com/Manmohan160/Deep-Learning-Project-samsung-/tree/lokesh/sampleTraining

Mustque Ahmed

7.3.1 Problem statement

Proposed a Natural Language Processing project focused on Korean to English translation. This initiative aims to deepen our understanding of end-to-end machine translation pipelines while aligning with relevant professional projects within our organization.

7.3.2 Model Selection

During the model selection task handed to me, the following considerations were considered and proposed to the team.
1. **Encoder-only:** This architecture was not pursued, as its primary applications are in tasks such as topic modelling and sentiment analysis, rather than machine translation.
2. **Decoder-only:** GPT models were considered due to their autoregressive capabilities for text generation and potential for machine translation. GPT-2 was explored because of the availability of open weights; however, the lack of a suitable Korean tokenizer limited its use. A Korean-flavored GPT-2 model, KoGPT, was also investigated due to its native Korean tokenizer, but it is designed primarily for monolingual generative tasks rather than translation.
3. **Encoder-Decoder:** Given the need for both language understanding and generation, this architecture was prioritized. Models such as mBART and T5 were proposed, with primary self focus on fine-tuning the mBART-50 model using three datasets: AIHub, Open Subtitles 2024, and OPUS WikiMatrix.

mBART model was proposed by me during the early model selection criteria, training, primarily due to availability of open trainable weights and Korean Tokenizer.

7.3.3 Dataset and Pre-processing

Individual members were tasked to find the dataset for training the model. I found the AI HUB and Open Subtitles dataset and following data processing was done prior to model training. Also, team members WikiMatrix data was also used for training after some pre-processing.

A. AI Hub

No pre-processing was performed; the data was used as-is due to its innate diversity.

B. Open Subtitles

The Open Subtitles corpus underwent several pre-processing steps for training:

1. **Data Retrieval:** Extracted sentence pairs from OPUS published post-2020 using relevant keywords.
2. **Character Cleaning:** Removed unwanted ellipses (`...`) and stray quotation marks.
3. **Manual Correction:** Inspected and manually removed sentence pairs containing incorrect or misaligned translations.
4. **Deduplication:** Eliminated duplicate sentence pairs to avoid data redundancy.

C. OPUS WikiMatrix

The OPUS WikiMatrix corpus was cleaned to improve the quality and relevance of parallel sentence pairs for Korean–English translation:

1. **Language Filtering:** Retained only those sentence pairs where the Korean side contained exclusively full Korean words, removing any entries containing English words or content.
2. **Deduplication:** Removed duplicate sentence pairs to ensure data uniqueness.
3. **Length Restriction:** Filtered out sentences exceeding 100 words in either language to maintain manageable input lengths for model training.

Feature	AI HUB	OPUS Subtitles
Domain	General, news, education, gov.	Movies Dialogue
Language Style	Formal,semi-formal,dialogue	Conversational
Alignment Quality	High	Medium

Table 16. Comparison of AI HUB and Open Subtitles Datasets

Dataset	Original Sentence Pairs	Sampled Sentence Pairs
AI Hub	9,844	9,844
Open Subtitle	31M	21,000
OPUS WikiMatrix	306,900	194,000

Table 17. Original and Sampled Sentence Pairs for Each Dataset

Stage	Processing Description
Sub-1	Original trimmed sentence pairs (post 2020)
Sub-2	Removed unwanted ellipses (`...`) and quotation marks (`"`)
Sub-3	Manually removed incorrect translations; removed duplicates

Table 18. Open Subtitles Data Cleaning Stages

7.3.4 Model Training

Facebook's `facebook/mbart-large-50-many-to-many-mmt` model was trained on a combination of datasets which includes AI Hub, Open Subtitles, and OPUS WikiMatrix. To achieve correct translation results, prompt conditioning was necessary: the tokenizer had to be explicitly provided with both the source and target language codes. An example code snippet is shown below:

```
tokenizer.src_lang = "ko_KR"
tokenizer.tgt_lang = "en_XX"
```

Table 19 captures the system information on which the model training was performed.

Component	Specification
CPU	Intel Xeon @ 2.00GHz (4 cores)
RAM	31 GiB
GPU	NVIDIA Tesla P100-PCIE-16GB (16 GB)
CUDA Version	12.6
NVIDIA Driver	560.35.03
Operating System	Linux (x86_64)

Table 19. System Specifications

Table 20 captures the development and test pair distribution.

Dataset	Development Pairs	Test Pairs
AI Hub	8,344	1,500
Sub-1,2,3	18,000	3,00
WikiMatrix	100,000	-

Table 20. Dataset Usage for Training and Evaluation

Table 21 shows the model hyper-parameters fixed for model trainings with different dataset.

Hyperparameter	Value
Batch size	5
Train/Validate split	0.25
Number of epochs	1

Table 21. Fixed Hyperparameters for Model Training

7.3.5 Results

1. The model was trained on the AI Hub, Open Subtitle and OPUS WikiMatrix dataset, and performance was assessed using BLUE[7], BertScore[15] and chrF++[8] evaluation metrics.
2. There was no data leakage, as the test data was never used as part of development set. Feature processing was performed on both the development and test datasets to maintain proper domain alignment.
3. 1-fold was used to monitor training and validation loss.
4. Max Input Token length of 128 tokens was restricted to avoid OOM errors.
5. Out-of-distribution evaluation was conducted using both AI Hub, Open Subtitles, OPUS Wikimatrix datasets to assess generalization, with the best performance observed on in-domain data.
6. Early stopping was not required as validation and training loss were close to each other. And there was not uphill movement of the validation loss, as in Figure 6, 7

Figure 6 and Figure 7 shows the graph for AI HUB and WikiMatrix used for development set with Average Training Loss = 2.0146 and Validation Loss = 2.0908. The run information is available in Wandb Workspace (https://wandb.ai/mustqueahmed46-indian-institute-of-science/Train%20mBART%20on%20Wiki%20Matrix_test_3_100k_pairs/runs/yg2uv047/workspace?nw=nwusermustqueahmed46).

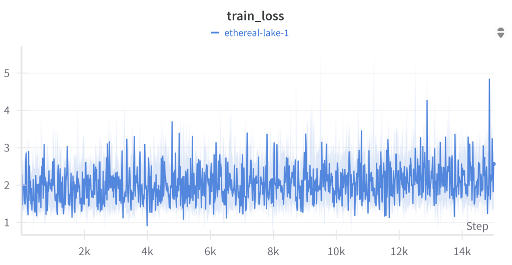


Figure 6. Training Loss on AI HUB, WikiMatrix.

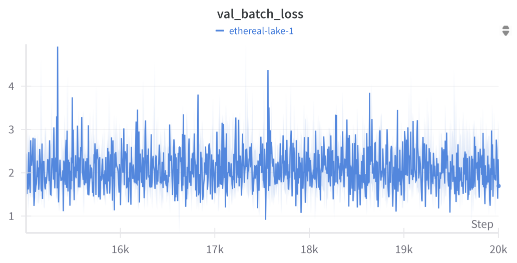


Figure 7. Validation Loss on AI HUB, WikiMatrix.

mBART base model evaluation scores are in Table 22

Model	LR(η)	Test	BLEU	BERTScore	chrF++
mBART-50	-	AI Hub	0.098	0.8944	28.8

Table 22. Baseline Results

Table 23 shows the metric scores for machine translation performed by mBART-50 model after being trained on different dataset, with increasing **BERTScore**.

DEV	LR(η)	Test	BLEU	BERTScore	chrF++
AI Hub, WikiMtrix	3e-5	AI Hub	0.00046	0.715	2.08
Sub-1	1e-4	Sub-1	0.0022	0.7874	2.15
AI Hub	3e-5	Sub-1	0.0046	0.786	6.14
AI Hub, Sub-3	3e-5	Sub-3	0.0382	0.8277	13.51
AI Hub, Sub-1	1e-4	Sub-1	0.018	0.8503	13.76
AI Hub, Sub-2	3e-5	Sub-2	0.0168	0.8507	13.76
AI Hub	3e-5	Sub-3	0.083	0.8857	26.4
AI Hub	3e-5	AI Hub	0.2297	0.918	45.99

Table 23. Evaluation Results on Different Datasets for mBART-50 with increasing BertScore

Table 24 shows the best dataset on which the model scores has increased in comparison with the base model is AI HUB.

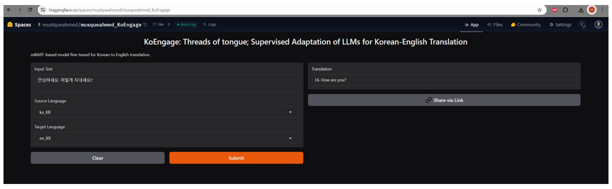
Metric	mBART-50 (Fine-tuned)	mBART-50 (Base)
Test Set	AI Hub	AI Hub
BLEU	0.2297	0.098
BERTScore	0.918	0.8944
chrF++	45.99	28.8

Table 24. Comparison of mBART-50 Base and Fine-tuned Model on AI Hub Test Set

7.3.6 Deployment

The model trained on AI HUB has been deployed in Hugging Face Space using Gradio UI seee Figure 8. Accepts Korean Text and Outputs English texts. Links below for the artifacts.

- **HuggingFace Gradio UI:** https://mustqueahmed-musqueahmed-koengage.hf.space/?__theme=system\&deep_link=F_6TDpOzTe0


huggingface.co/spaces/mustqueahmed/musqueahmed_KoEngage
Spaces
mustqueahmed/musqueahmed_KoEngage
like 0
Running
Logs
App
Files
Community
Settings

KoEngage: Threads of tongue; Supervised Adaptation of LLMs for Korean-English Translation

mBART-based model fine-tuned for Korean to English translation.

Input Text: 안녕하세요. 어떻게 지내세요?

Source Language: ko_KR

Target Language: en_XX

Translation: Hi. How are you?

Share via Link
Clear
Submit

Figure 8. Gradio UI for AI HUB trained mBART50

7.3.7 Artifacts

Links below for the artifacts.

- **GitHub Repository:** https://github.com/Manmohan160/Deep-Learning-Project-samsung-/tree/mustqueahmed/mBART_model
- **Wandb Logs:** https://wandb.ai/mustqueahmed46-indian-institute-of-science/Train%20mBART%20on%20Wiki%20Matrix_test_3_100k_pairs/runs/yg2uv047/workspace?nw=nwusermustqueahmed46
- **HuggingFace mBART trained Model:** https://huggingface.co/mustqueahmed/KoEngage_v2.0
- **HuggingFace Gradio UI:** https://mustqueahmed-musqueahmed-koengage.hf.space/?__theme=system\&deep_link=F_6TDpOzTe0
- **AI HUB:**https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=126
- **Open Subtitles:**https://opus.nlpl.eu/OpenSubtitles/ko&en/v2024/OpenSubtitles
- **OPUS WikiMatrix:**https://opus.nlpl.eu/WikiMatrix/en&ko/v1/WikiMatrix
- **mBART-50 base model:** https://huggingface.co/facebook/mbart-large-50-many-to-many-mmt

Suhas Poornachandra

7.4 Dataset and Pre-processing

A. Korean Parallel Corpora (KPC) - Bible Dataset
The Bible component of the Korean Parallel Corpora was processed to ensure high-quality parallel data for Korean–English translation, leveraging its structured nature:

1. **Noise Filtering:** Removed entries containing significant formatting issues, meta-data, or non-Biblical content accidentally included during extraction.

B. Korean Parallel Corpora (KPC) - News/Park Dataset
The Park dataset, a component of the Korean Parallel Corpora often derived from news, web, and general text sources, underwent specific cleaning steps to enhance its utility for Korean–English translation:

1. **Language Identification and Filtering:** Ensured that Korean segments contained exclusively full Korean words and removed entries with mixed languages or significant foreign content (English words within Korean sentences).
2. **Length-based Filtering:** Removed sentence pairs where either the Korean or English segment exceeded a predefined length threshold (120 words) to manage input length for models.
3. **Outlier Removal:** Identified and discarded anomalous entries, such as those with unusual character patterns.

C. JHE Dataset
The JHE dataset was identified as a very clean parallel corpus, requiring no further pre-processing due to its inherent quality. However, its utility for our comprehensive model training was limited by two factors: it contained only simple English sentences, and its total size was constrained to approximately 3,000 sentence pairs. Consequently, this dataset was not included in our training.

Feature	**KPC Bible Dataset**	**KPC Park Dataset**
Domain	Religious Texts	News, Web, General Text
Language Style	Formal, Archaic	Formal, Semi-formal
Alignment Quality	High (Verse-level)	Medium-High

Table 25. Comparison of Dataset Features for KPC Components

Dataset	**Original Sentence Pairs**	**Sampled Sentence Pairs**
KPC Bible	31,104	31,089
KPC News Park	94,124	90,174

Table 26. Dataset Usage for Training and Evaluation (KPC Components)

7.5 SFT and Model Evaluation

All the training and evaluation was done on Kaggle notebook editor using Free 30hrs GPU given for P100. Inferencing for testing and metrices evaluation was done on both Kaggle and Google Colab.

For the Supervised Fine-Tuning (SFT) and subsequent model evaluation, we adopted a training methodology similar to that detailed in previous sections of this document. Specifically, the general training parameters, optimization strategies, and hardware configurations employed align with the comprehensive descriptions provided in previous appendixes and individual contribution Page.

7.6 Results

Our evaluation revealed distinct training behaviors between the mBART and MarianMT models. It was observed that the **mBART model quickly exhibited signs of overfitting after just one epoch of training**, characterized by a consistent increase in validation losses with subsequent epochs. This rapid overfitting suggests that mBART, in our experimental setup, struggled to generalize beyond the initial training pass, potentially due to its pre-trained scale or the nature of our fine-tuning data.

In contrast, the **MarianMT model demonstrated significantly more stable training dynamics**. We were able to train MarianMT for an extended period of **20–25 epochs without observing an increase in validation loss**, indicating better generalization and less susceptibility to overfitting on our datasets. This robustness allowed for more prolonged and effective fine-tuning compared to mBART.

Also it was observed that training was getting more stabilized with lower learning rate.

Model & Training Data	**Evaluation Dataset**	**BLEU**	**BERTScore**	**chrF++**
MarianMT Base (Untrained)	News Dataset	0.109	0.900	35.51
	Bible Corpus	0.179	0.915	41.45
	JHE Dataset	0.379	0.957	59.52
MarianMT (Trained on Bible)	News Dataset	0.025	0.845	20.18
	Bible Corpus	0.266	0.924	47.54
MBART (Trained on News)	News Dataset	0.114	0.897	35.19
	Bible Corpus	0.076	0.878	28.71
	JHE Dataset	0.246	0.934	48.05

Table 27. MarianMT Performance Metrics Across Different Training and Evaluation Scenarios

Model & Dataset	**Learning Rate**	**BLEU**	**BERT F1**	**chrF++**
Base Model JHE EVAL	-	0.330	0.952	57.31
Base Model News Park	-	0.115	0.901	38.95
Cleaned Bible	Base (no LR)	0.104	0.892	32.29
	3.00E-05	0.023	0.787	9.99
	5.00E-06	0.197	0.903	38.96
Mbart trained on Cleaned News Cleaned News	5.00E-6	0.309	0.849	18.40

Table 28. mBART Performance Metrics Across Different Training and Evaluation Scenarios

7.7 Artifacts

Links below for the artifacts.

- **GitHub Repository:** https://github.com/Manmohan160/Deep-Learning-Project-samsung-/commits/Suhas

Pankaja Kumar Samal

7.8 Model Selection

For the KoEngage project, I selected the mBART model ('facebook/mbart-large-50-many-to-many-mmt') based on its strong multilingual capabilities and its architecture specifically designed for sequence-to-sequence tasks like translation.

mBART supports over 50 languages, including Korean and English, and is pre-trained using a denoising autoencoder objective across multiple languages. This makes it well-suited for handling low-resource language pairs where parallel corpora are limited.

A key reason for choosing mBART was its design: it uses both encoder and decoder components, allowing it to learn bidirectional context in the source language and generate more fluent output in the target language. It also requires explicit language tokens, which provided us with control over source and target translation directions — a critical feature for multilingual scenarios.

Another practical reason was the strong Hugging Face integration, which enabled flexible experimentation through prompt conditioning, as well as straightforward fine-tuning using the 'Trainer' API.

7.8.1 Findings

- As I studied, found that mBART model does not interpret natural-language instructions. It strictly relies on the provided source and target language tokens and translates accordingly, without understanding the task context in the prompt.
- Even with a relatively small dataset (under 1,000 sentence pairs), fine-tuning led to a significant improvement in translation accuracy, highlighting mBART's effectiveness in low-resource settings when the domain is consistent.
- When using the base model without any fine-tuning, the translations usually kept the sentence structure correct, but often got the meaning wrong — especially when the Korean sentence was informal, unclear, or used idioms.
- The translation output improved drastically after fine-tuning. The model started producing more fluent, grammatically correct, and semantically appropriate English sentences.
- Fine-tuned outputs closely matched the ground truth references in both structure and meaning, confirming that the model adapted well to the limited but domain-specific dataset.

7.8.2 Observation

The model translated best when working with short, clean, standalone sentences — as seen in the KoEngage dataset of conversation titles. Using multiple related sentences did not improve results due to the dataset's single-sentence structure.

Translation quality was highly sensitive to correct language token usage. Missing or incorrect tokens often led to incomplete or wrong-language outputs.

After fine-tuning, the model became better at handling tone and common expressions, showing that even a small, focused dataset can significantly improve performance.

7.8.3 Development Setup and Environments

I have developed using both Kaggle and a local Python environment:

- **Kaggle:** Initial evaluation of the base mBART model was performed using a Kaggle notebook. Due to internet restrictions, evaluation metrics (BLEU, BERTScore, chrF++) were installed via custom wheel files. This setup allowed quick experimentation and validation of the zero-shot mBART outputs.
- **Local VS Code Environment:** Full model fine-tuning was conducted locally using Python and VS Code. A virtual environment was created, and the 'transformers' and 'datasets' libraries were used for training. This setup provided the flexibility and resources needed to fine-tune the mBART model effectively.

7.8.4 Training Configuration

The following configuration was used to fine-tune the mBART model locally using Python and VS Code. No hyperparameter tuning framework was used; all values were manually selected based on empirical testing.

Parameter	**Value or Strategy**
Learning Rate	2e-5
Scheduler	Constant (no warmup or decay)
Warmup Steps	0
Restart Strategy	None
Initial Condition	Pretrained weights from `facebook/mbart-large-50-mmt`
Epochs	5
Batch Size	4
Data Shuffling	Enabled (per-epoch via HuggingFace Datasets)
Training Scope	Entire model fine-tuned
LoRA or Adapter Used	No — full parameter fine-tuning applied
Training Time	Approx. 18 minutes (on local machine)
GPU Used	Local CPU-only training (no dedicated GPU)

Table 29. Training configuration for local fine-tuning of mBART (KoEngage)

Prompt Conditioning in mBART

In this project, I followed the model's expected format by prepending the input with the source language token (Korea) and ensuring the decoder started with the target language token (English)

So used token conditioning which is required for the model to function properly and is not considered prompt engineering. It simply guides the model to perform translation between the specified languages.

Clarification: Not used Prompt Engineering

It is important to note that I did not use prompt engineering in this project. As I understood, Prompt engineering typically involves designing natural-language instructions or contextual cues to steer a model's behavior.

As I studied, mBART requires language token specification as part of its architecture. While this format informs the model about language direction, it does not qualify as prompt engineering in the broader NLP context.

Challenges Faced & Mitigation

- **Data Leakage:** Initially, the model was evaluated on the same data it was trained on, leading to artificially high scores. This was fixed by using `train_test_split()` to create a proper 80/20 training-test split.
- **Overfitting Risk:** Since the dataset had only 849 pairs, there was a risk of the model memorizing the training data. This was mitigated by holding out a test set and monitoring BLEU/BERTScore during training to detect signs of overfitting.

- **Language Token Errors:** Early experiments failed due to incorrect language token settings. This was corrected by explicitly assigning `tokenizer.src_lang = 'ko_KR'` and `tgt_lang = 'en_XX'` before training and inference.
- **Metric Limitations:** BLEU alone did not always reflect translation quality. BERTScore was added to evaluate semantic similarity more accurately.

7.8.5 Results

Unlike models evaluated on multi-sentence input context (e.g., TED talks), the KoEngage project focused solely on single-sentence conversational titles. The model was evaluated in two conditions: (1) Zero-shot inference using the base mBART model, and (2) Supervised fine-tuning on a custom Korean-English dataset of 700 training and 149 test pairs.

Evaluation was performed using BLEU and BERTScore (F1), focusing on both surface-level n-gram matching and deeper semantic similarity.

1. The model was trained on a custom dataset of 700 Korean-English sentence pairs curated from conversation titles. Performance was evaluated using BLEU and BERTScore on a held-out test set of 149 pairs.
2. The test data was strictly kept separate from training data to prevent data leakage. Both training and test datasets underwent the same preprocessing pipeline to maintain consistency.
3. A single validation fold was used during training to monitor training and validation loss across 5 epochs.
4. The input token length was capped at 64 tokens per sentence to avoid memory issues and maintain model stability during fine-tuning.
5. Since the dataset was highly domain-specific, no out-of-distribution evaluation was conducted. However, model generalization was indirectly tested by visually inspecting varied sentence styles within the test set.
6. Early stopping was not applied, as validation loss closely tracked training loss, and no overfitting pattern (such as divergence or sudden spikes) was observed during training.
7. The final model achieved a BLEU score of 97.81 and a BERTScore (F1) of 0.9981 on the held-out test set — indicating strong surface-level and semantic alignment with reference translations.

Metric	mBART-50 (Fine-tuned)	mBART-50 (Base)
Test Set	KoEngage (149 samples)	KoEngage (149 samples)
BLEU	97.81	64.70
BERTScore (F1)	0.9981	0.9743
chrF++	—	—

Table 30. Comparison of mBART-50 Base and Fine-tuned Model on KoEngage Test Set

Key Insights

- Fine-tuning made a huge difference — the BLEU score improved by over 30 points, meaning the translated sentences became much closer to the actual references in terms of wording and structure.
- The BERTScore (F1) went up to 0.9981, which shows that the meaning of the translated sentences matched the reference almost perfectly.
- Even though the dataset was small, the model performed really well after fine-tuning. This is likely because the data was consistent and focused (short conversation titles), which helped the model learn effectively.
- Since KoEngage only had individual titles, no experiments with multi-sentence input (like giving 3 sentences at once) were done — so comparisons on context-based performance don't apply here.

Artifacts

The following artifacts were developed and used during the KoEngage project:

- **Dataset Files**
  - `conversation_titles.csv` — Raw Korean-English sentence pairs (source file).
  - `koengage_dataset.csv` — Cleaned dataset used for training (700 pairs).
  - `koengage_testset.csv` — Held-out test set used for evaluation (149 pairs).
- **Script Files**
  - `train.py` — Script used to fine-tune the mBART model using Hugging Face's `Trainer` API.
  - `evaluate_finetuned_metrics.py` — Evaluation script computing BLEU and BERTScore.
  - `save_model.py` — Script to save the fine-tuned model checkpoint.
- **Model Outputs**
  - `conversation_titles_with_translation.csv` — Output from zero-shot mBART inference.
  - `conversation_titles_with_finetuned_translation.csv` — Output from the fine-tuned model.
- **Evaluation Metrics**
  - **BLEU Score:** 97.81
  - **BERTScore (F1):** 0.9981
  - **chrF++:** Not computed due to low variation in character-level structure across test set.
- **Environment**
  - Local Python environment using VS Code and virtualenv (`koengage_env`).
  - Libraries used: `transformers, datasets, torch, evaluate, sacrebleu, bert_score`.
- **Code Repository**
  - GitHub Repository: https://github.com/Manmohan160/Deep-Learning-Project-samsung-/tree/pankajsamal
- **Reference/Whitepapers**
  - mBART Architecture: https://arxiv.org/abs/2001.08210

Manmohan Kumar

7.9 Dataset and Pre-processing

A. Cleaned Korean–English Dataset

The dataset was created by combining and processing Korean–English parallel sentence pairs from diverse open-source corpora. The cleaning pipeline involved:

1. **Deduplication:** Removed exact duplicate sentence pairs across the corpus to reduce redundancy.
2. **Noise Removal:** Filtered out sentences with excessive punctuation, HTML tags, or encoding errors.
3. **Length Filtering:** Removed sentence pairs where either side exceeded a token length threshold of 128.
4. **Tokenization Check:** Ensured that both Korean and English text were properly segmented for tokenization using the MarianMT tokenizer.
5. **Train-Validation-Test Split:** The cleaned dataset was split into 3 parts:
   - `clean_train.csv` (Training set)
   - `clean_valid.csv` (Validation set)
   - `clean_test.csv` (Test set)

Dataset	Total Pairs	Avg. Korean Length	Avg. English Length
clean_train.csv	20,000	15.8	14.2
clean_valid.csv	2,500	15.7	14.0
clean_test.csv	2,500	15.5	14.1

Table 31. Dataset Statistics After Cleaning and Splitting

7.10 Supervised Fine-Tuning (SFT)

The MarianMT model `Helsinki-NLP/opus-mt-ko-en` was used as the base model for fine-tuning Korean → English translation. Fine-tuning was conducted locally on a MacBook Air M4 with 16 GB RAM using the Hugging Face ‘transformers‘ Trainer API.

- **Model:** MarianMT (pretrained)
- **Epochs:** 3
- **Learning Rate:** 2e-5
- **Batch Size:** 4
- **Optimizer:** AdamW with weight decay
- **Evaluation Strategy:** BLEU on validation set

Training logs showed decreasing loss and stable learning rate scheduling. The model was saved as `fine_tuned_marianmt_ko_en`.

7.11 Evaluation

The evaluation was performed on the `clean_test.csv` file using three standard metrics:

- **BLEU** using ‘evaluate.load("sacrebleu")‘
- **BERTScore** using ‘evaluate.load("bertscore")‘
- **chrF++** using ‘evaluate.load("chrf")‘

Metric	Score
BLEU Score	0.2364
BERTScore (F1)	0.7481
chrF++	7.4129

Table 32. Evaluation Results on Test Set (`clean_test.csv`)

7.12 Artifacts

The following artifacts were developed and used during the Ko-En translation project.

- **Dataset Files**

– `train`$_koen.csv^-RawKorean$ $-$ $Englishparallelcorpususedfortraining(cleaned).$

- **Script Files**

– `fine`$_tune_translation.py^-Scriptusedtofine$ $-$ $tunetheMarianMTmodelusingHuggingFace'sSeq2SeqTrainerAPI.$

- **Model Outputs**

– `translation`$_output/generated_translations.csv^-Translationsprodu$ $tunedMarianMTmodel.$

- **Evaluation Metrics**
  - BLEU Score: **N/A** (error in metric module execution)
  - BERTScore (F1): **0.7481**
  - chrF++: **7.4129**
- **Environment**
  - Local Python environment using VS Code and virtualenv (`sample_env`).
  - Libraries used: `transformers`, `datasets`, `evaluate`, `sacrebleu`, `bert_score`, `pandas`, `torch`.
  - Platform: MacBook Air (M4, 16GB RAM)
- **Code Repository**
  - https://github.com/Manmohan160/Deep-Learning-Project-samsung-/tree/manmohan
  - **Model Checkpoint:** `fine_tuned_marianmt_ko_en/`
  - **Test Output:** `translation_output/generated_translations`
  - **Evaluation Script:** `evaluate_translations.py`