Thesis for the Degree of M.S. in Engineering

Prompt-based Learning for Natural Language Processing

Department of Engineering, Major in Electronics and Electrical Engineering
The Graduate School

Eunchan Lee

December, 2022

The Graduate School
Kyungpook National University

Prompt-based Learning for Natural Language Processing

Eunchan Lee

School of Electronic and Electrical Engineering

The Graduate School

Supervised by professor Sangtae Ahn

Approved as a qualified thesis of Eunchan Lee

for the degree of M.S. by the Evaluation Committee

December, 2022

Chairman: Prof. Gil-jin Jang

Prof. Seokjin Lee

Prof. Sangtae Ahn

The Graduate School

Kyungpook National University

Contents

List of Tables

List of Figures

ABSTRACT

Natural language processing technology can be divided into natural language understanding and natural language generation, and in both directions, bidirectional encoder representations from transformers (BERT) in the field of natural language understanding and generative pre-trained transformer (GPT) model in the field of natural language generation have changed the paradigm of existing natural language processing models and methods.

They have something in common: the Transformer-based model and the Pre-trained Language Model pre-trained with large corpus, BERT is divided into transformer encoder and GPT is divided into decoder only.

However, in-context learning, a few-shot learning technique from the prompt-based paper, proved that it can achieve a very high level of performance by generating labels for natural language comprehension tasks either with natural language generation or with decoders. This trend has brought a new paradigm called prompt-based learning, which is one step further from the existing BERT and GPT. However, the studies of Prompt-based learning have not had enough research in the early stages, and only a few Korean-based research has been conducted.

In this paper, to reflect this research field's trend, I apply it directly to two Korean natural language understanding tasks using a model, a type of Korean GPT. In addition, various comparative experiments were applied to prove important points through experimental results. As a result, several factors that cause many performance differences in the experiment were found and presented

in Section 4.

Specifically, I use KoGPT (6.16B) as a neural network model (language model), and apply prompt-based few-shot learning to NAVER sentiment movie corpus (NSMC) and Korean question and answering dataset (KorQuAD), benchmark datasets for Korean Sentiment Analysis, Korean Question Answering tasks, respectively. And finally, we design and apply a new arbitrary text style transfer task with same method and same language modeling condition to see its various potential applicable field.

1 INTRODUCTION

Artificial intelligence technology has been advanced from various perspectives such as models and computing resources based on neural network-based algorithms, and has successfully learned data from tasks such as computer vision (CV), natural language processing (NLP), and speech recognition. Among them, natural language processing technology developed rapidly in 1990, based on statistics in a way that computers aim to understand human language, and by the 2010s, machine learning showed high performance in rather complex natural language data learning as models advanced based on deep learning. In deep learning-based NLP, the language model, a deep neural network trained by text dataset for understanding human language and performing a lot of natural language-based tasks, is the key mechanism that allows machines to understand human's language patterns.

Currently, techniques specializing in NLP have developed, allowing techniques such as sequence-to-sequence [1], attention mechanism [2], and transformer [3] to understand the data patterns of more complex language-related NLP tasks.

Sequence-to-sequence (seq2seq) [1] is a method in which neural network models such as recurrent neural network (RNN) [4] and transformer [3] divide the roles of models into two types, encoder and decoder, and combine them to model them for better language understanding and processing.

In this case, the encoder is optimized to understand the input text and to send it to the hidden vector, and the decoder is optimized and learned to generate

text from the hidden vector of the neural network. In the case of seq2seq-based language modeling, encoders can be used to perform natural language understanding (NLU) tasks (from 'text' to 'predicted label'), decoder alone to perform natural language generation (NLG) (from 'no input' to 'text'), and encoder-decoder together to perform text-to-text tasks such as machine translation and document summarization (from 'source text' to 'target text').

In this modeling stage, RNNs are mainly used for modeling, and its improving methods such as long-short term memory architecture (LSTM), sequence-to-sequence (seq2seq), and attention mechanisms are proposed, but RNN's learning performance has fundamental limitations such as the vanishing gradient problem. Transformer architecture [3], an encoder-decoder model based on only an attention mechanism without RNNs, contributed significantly to performance improvement in the neural machine translation (NMT) task.

Specifically, it received parallel at a time and further advanced correlation through the output data and attention mechanism [2] to learn a huge transformer pattern. Due to such influence, the performance of most natural language processing tasks has been significantly improved by constructing these transformer-based encoder decoder layers [3].

The paradigm for fine-tuning pre-trained models based on Transfer Learning can be seen as the third breakthrough of NLP, which allows language models to learn word embeddings in advance and perform subsequent downstream tasks more efficiently[5]–[7].

ULMFiT [5], ELMo [6] are the first studies to successfully bring this method to language modeling, where LSTM-based model layers are used. BERT

[7] and GPT [8] apply these transfer learning-based paradigms to Transformer modeling, and in particular improve the quantity and quality of the pre-training corpus and independently pre-train and deploy the encoder (BERT) and decoder (GPT) models of the Transformer model, a type of sequence-to-sequence, to achieve high task performance specifically.

The success of BERT and GPT has led to a tremendous paradigm shift in NLP, with performance improvement to a high level in most natural language understanding and generation tasks compared to traditional state-of-the-art models.

Due to the successful and significant impact of the two language models, their various advanced follow-up studies such as RoBERTa, ALBERT, DeBERTa, ELECTRA (BERT-Family), GPT-2, GPT-3, DialoGPT (GPT-Family) have been actively considered to be the main topics of language models[9]–[15]. In addition, models distributed after fine-tuning suitable for various tasks such as finance-specific and bio-specific text understanding and generation, and programming-code-specific understanding and generation include FinBERT [16], BioBERT [17], and Codex [18].

With this paradigm established, the rule that encoder-based modeling is applied to natural language understanding tasks, and decoder-based modeling is mainly used to generate natural language is typically established. However, in the papers of [GPT understand too.] [19] and [Language Models are few-shot learners] [14], successfully introduced a method called prompt-based few-shot learning, they show that NLU can be performed at a high level even with a decoder-based model, GPT.

It was also used based on a futuristic learning-based method that allows learning in an environment with less data, which is an efficient alternative in that it takes high labor-costs to label or process natural language data. To apply the decoder model GPT-3 to natural language understanding (NLU) tasks, we present a learning design method for generating outputs of natural language comprehension tasks with 'generation' (abstractive) rather than output 'prediction' (extractive) used by existing encoder models, which the researchers named successfully presented techniques 'in-context learning'.

In-context learning notes that pre-trained language models (PLMs) with model parameters in Billion units, such as GPT-3, have already been completely understood through large-scale corpora. In other words, in order to more efficiently leverage these already learned model parameters, it is a method of inducing the model to learn tasks without backpropagating them in a learning method through few-shot learning (and meta-learning).

P. Liu et al [20] summarized the prompt paradigm through a survey of the very simple method called 'Pretrain, prompt, and predict'. They also stated that PLMs with model parameters of more than 1 billion (1B) can perform well on prompt-based futuristic learning performance of typical tasks such as NLU and NLG, and can perform as well as fine-tune based state-of-the-art models if the model scales even larger.

That is, from a model point of view, it can be seen that if a pre-trained language model with a model parameter of 1B or more is secured, the task can be performed by applying prompt-based few-shot learning without fine-tuning the model's parameters. They also suggested that one of the key factors that

determines a model's prompt-based few-shot learning performance when it is fixed is prompt engineering, which appropriately handles the format of the prompt, merges the data with the prompt, and gives it as a description of few-shot learning.

Prompt engineering, which properly converts data and tasks into input descriptions of models after understanding them, is largely divided into human-set manual prompt template engineering [14], [21] and automated template learning, which automatically generates prompts through neural network layers (smaller than language models).

Brown et al [14] used the 'manual prompt' template to train the GPT3-175B, a model parameter 175 Billions pre-trained model, to outperform the fine-tuning-based state-of-the-art model. Sanh et al [21] presented the formats of 'manual prompt' for each task in an appendix to more than 100 page papers through a study of the T0 model that pre-trained to enable multi-task zero-shot learning (Huggingface, github). The example of the manual prompt format they presented is based on an intuitive understanding of the task, and in IMDB, a typical emotional classification task, they devised 11 prompts themselves, one of which, like "text : Did the reviewer find this movie "good or bad"?" (prompt format of text description), is very intuitive.

Compared to Automated prompt, manual prompt engineering requires some tasks that humans have to perform directly, but it has a significant advantage of being able to perform tasks with less constraints and less fully learned gradient free techniques compared to automated prompt template learning, which requires layer learning and dataset.

Sebastian Ruder,one of the influential people in the deep learning field, also cited ‘Prompting’ as one of the 15 highlights of [ML and NLP Research Highlights of 2021] [22] in his technical report. In addition, keywords such as ”Beyond The Transformer”, ”Meta-learning”, ”Universal Models”, and ”Massive Multi-task Learning” that he selected together can also be considered to be significantly related to NLP task application of large-scale pre-trained language models through ‘prompt-based few-shot learning’. He noted that after GPT-3, the combination with the few-shot setting is very important, and that there are still countless points to be studied about the prompter, and that it will become more elaborated in the future.

To summarize these long background and NLP paradigm changes mentioned above, ”Prompt-based few-shot learning for NLP” is an increasingly recognized field of potential post-transformer paradigm, with two key elements (i) Pre-trained Language Models which have more than 1 Billion model parameters. (ii) Prompt engineering with the dataset well. Large-scale PLMs are key to pre-training processes that require very large amounts of GPU computing sources and very large collected natural language corpus, so research on models pre-trained by corporate research teams is recognized for their achievements and is very active through open sources such as Github and Hugging Face.

As a representative example of the models released, GPT-3 [14] released model versions of 1.3B, 6.7B, and 175B and made them available to individual and educational researchers with permission. Google released the T5 (Text to Text Transfer Transformer) [23] model as a version with five different model parameters: small (60M), base (220M), large (770M), XL (3B), and XXL (11B), creating an environment where researchers can study based on large-scale PLM.

However, since most of these PLMs have pre-training datasets based on English, NLP Tasks in other languages have shown poor performance with few-shot learning. Finding a PLM with a model parameter of a billion (1B) or more in the same language as Korean has become quite necessary, and based on demand, researchers at NAVER CLOVA unveiled a large-scale Korean-based PLM in June 2021.

HyperCLOVA released two versions of 6.9B and 82B, showing the possibility of becoming a Korean version of GPT-3 with 175B and 6.7B, but the disadvantage is that individual researchers must obtain meticulous permission for use and are not open-source available. [24].

However, in November 2021, when kakaobrain unveiled KoGPT, a 6.7B model-sized Korean large-scale PLM, as an open source (on Github, Huggingface), a suitable model for prompt-based few-shot learning for Korean NLP appeared to individual researchers. [25] From the point of view of prompt engineering, English-based research is the same as from the model point of view, but since the format of the prompt can be applied in Korean, it has the advantage of less language restrictions. In other words, the core findings of the English prompt engineering study apply equally to Korean-based prompt engineering.

A typical study of this prompt engineering is T0 [21]. The paper discloses prompt formats for most natural language processing tasks. In addition, since the manual prompt template has not yet been formalized, different prompt engineering is applied according to the tasks and datasets used in related studies, so there are many points that can be referred to by comparing performance when applying various methods.

As such, there are various comparative studies dealing with how it is appropriate to give prompt-based description that affects the performance of prompt-based few-shot learning through prompt engineering. Min et al [26] suggested that accurate labels may not be important by comparing performance when learning gold and random labels of few-shot data, respectively, Zhao et al [27] compared the selection of various prompt format and selection of few-shot data, showing that the selection of manual prompt and few-shot data can greatly influence the performance of the same model parameter.

However, unlike the formatting of the prompt, which is less subject to language constraints, it is necessary to check and verify whether the factors that can affect the performance actually affect or change the Korean large-scale PLM, Korean Task, and Dataset. In fact, since performance varies widely from study to study, it can be seen that there is still a high demand in the process of studying important factors to control the variance of performance by applying a direct Korean-based PLM to a Korean-based task.

In this thesis, I directly study the performance of prompt-based few-shot learning based on the Korean-based large-scale pre-trained language model. The target task is benchmark datasets of several Korean NLU tasks, and based on theories and international academic research about prompt-based few-shot learning, I aim to present how to stabilize the performance of Korean prompt-based few-shot learning through various comparison methods for each task. Our main contributions are as follows.

● Using KoGPT (Kakaobrain) with 6.7B as model parameter, a Korean-based large-scale pre-trained language model, I verify the performance of prompt-

based few-shot learning in Korean natural language processing tasks.

● As a Korean NLP task, Sentiment Analysis and Question Answering tasks belonging to NLU's typical tasks are applied. NSMC and KorQuAD v1 are used as benchmark datasets, respectively.

● I intend to suggest critical points by applying various comparative studies, not just looking at the task learning performance of the model.

● In addition, to take advantage of the free-to-data constraint due to the nature of few-shot learning, I apply KoGPT and prompt-based few-shot learning directly to implement a customized new task, the Arbitrary Text Style Transfer task (Sec 4.4)

Later in this paper, Chapter 2 contains Related Work to help understand this research concept, and Chapter 3 presents data and theories on specific methods. In addition, Chapter 4 presents the experimental results, and the experimental setup and summary for this, and the description for each task. Finally, I conclude in chapter 5.

2 Background

2.1 Prompting Paradigm

In terms of the meaning of the word, ”prompt” is often used to mean ”immediate” and ”punctual”, but ”prompt” in fields such as deep learning and natural language processing is used to mean ”guide by giving a hint to a question” (e.g. a natural language instruction). From the perspective of the NLP model, prompt can be viewed in a large range by giving a text guide as shown in Figure 1 so that the model can learn data better or output interference better.

BERT [7] pre-trained the model with large corpus data through a technique called Masked Language Modeling (MLM), where MLM replaces some (15 percent in BERT paper) of textual data to be learned with [MASK] token, so that the model sees the surrounding words and predicts mask token. The pre-training method of these language models can also be seen as a representative example of a text guide-based prompting method. [7], [20]

In addition, GPT also appears to utilize text guides as pre-training strategies, when pre-training the sentence ’I like you’, GPT learns large-scale text using a conditional probability-based left-to-right LM method that allows it to see the current word and predict the next word. For this reason, it shows very good performance in natural language generation tasks, and this method can also be seen as pre-training based on the prompting method./[8], [13], [14]

In addition, Brown et al [14] demonstrated that for very large pre-trained models, they efficiently learn and perform the task when they give the text

description-based prompting method-based few-shot data as a description even in k few-shot settings. [14]

This prompting-based few-shot learning method is called in-context learning. In-context learning differs from fine-tuning in that it freezes the model's parameters and allows the model to learn tasks as input instruction text with prompts.

The main reason why this is possible is that the model has learned a large-capacity text corpus in advance through pre-training. For example, Shin et al [28] demonstrates that when other pre-train corpus are trained on a model of HyperCLOVA [24], a type of PLM decoder, performance differences of up to 20 percent can occur in few-shot in-context learning, and consequently, pre-training corpus is an important factor. [28]

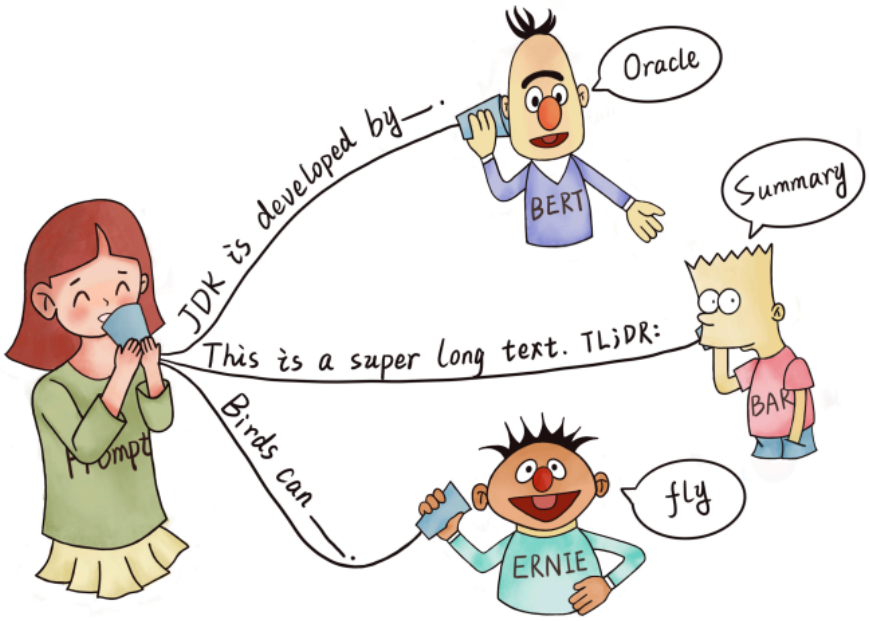

Fig. 1: An Illustrated overview of "Pretrain, Prompt, and Predict Paradigm."

2.2 Natural Language Understanding

2.2.1 NLU

Natural Language Understanding (NLU) refers to a task that allows a model to extract and predict the correct label, index, etc. when a natural language text input is received. Modern NLU understands data by splitting natural language into tokens and sequentially entering the model through deep learning-based modeling, from a modeling perspective, understanding natural language text inputs through RNN, LSTM, or transformer encoder, which finally returns labels or indexes through linear layers.

This can be completely distinguished from Natural Language Generation (NLG), which understands natural language through the same encoder and sends it to hidden vectors, but then learns a mechanism to generate natural language by constructing RNN, LSTM, or Transformer-based decoders rather than predicate MLP layers.

Typical tasks of NLU include spam mail filtering, sentimental analysis, topic/category classification, natural language inference, semantic textual similarity; There are also question answering, named entity recognition, and dialogue state tracking, which belong to token classification.

Sentiment Analysis, one of the simplest classification tasks, is a task in which the model learns to predict whether the text is 'positive' or 'negative' when given as an input, such as a movie review text.

In the case of Question Answering, which is considered to be more difficult, the typical case is the Stanford Question Answering Dataset (SQuAD) and KorQuAD (Korean Question Answering Dataset) Dataset, when the model is given a hint to understand a question and the question to find an answer, it is given a text[29], [30].

2.2.2 BERTology

These tasks can be performed to very high levels when applying Bidirectional Encoder Presentations from Transformers (BERT), a type of PLM (in which multiple Transformers are stacked without using RNNs for modeling). Therefore, Question Answering, which was previously one of the challenge tasks, predicts

an accurate answer of 87 percent of the test data when fine-tuning is performed through the current BERT. Because of these things, BERT-based language modeling has become the best strategy to perform NLU at a very high level.

Devlin et al's successful research achievements in BERT have changed the NLP paradigm to the way the PLM becomes the main. Various applications and improvement versions of BERT were actively studied, and studies such as RoBERTa[9], ALBERT[10], DeBERTa[11], and SpanBERT[31], and versions of SciBERT[32], MathBERT[33], and Code learned by optimizing BERT's pre-training method to task specific data were studied. As a result, there are people who consider BERT itself a discipline, and the term 'BERTology' has emerged and is frequently used in the open-source world.

2.3 Large-Scale Pre-trained Language Models

2.3.1 Transformer Architecture

In this section, only the structure and principle of transformer architecture [3] are briefly described. Transformer is a new sequence-to-sequence model introduced by Google Brain in mid-2017 based solely on attention mechanisms. The transformer structure requires less time to learn through parallel processing of input text sequences, and the multi-head self-attention layer with attention mechanisms provides better task scores such as natural language translation than existing models. As can be seen in Figure 2, the encoder has an encoder and a decoder, which has a feed forward layer that is simply fully connected to the multi-head self-attention as sub-layers, and the decoder has one more multi-head

shelf that receives output from both sub-layers of the encoder.

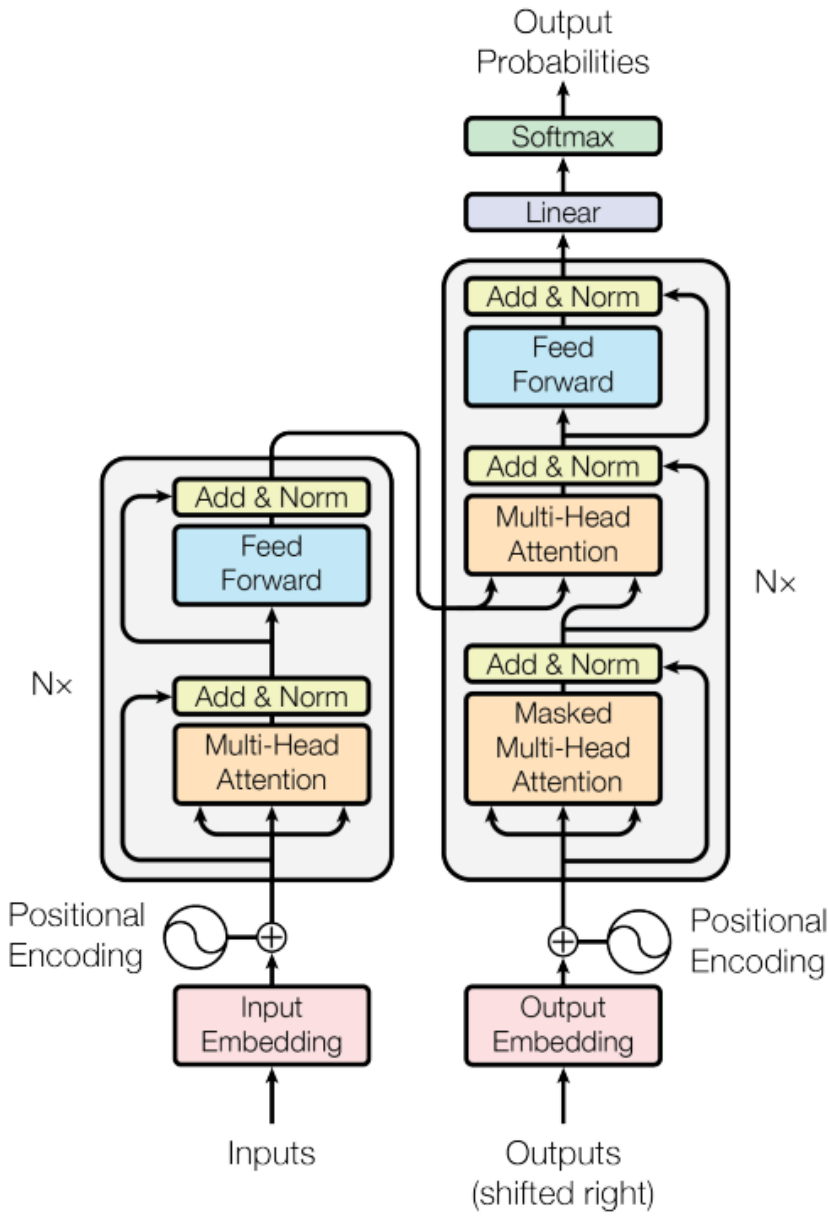

Fig. 2: The Transformer Architecture

2.3.2 Pre-trained Language Models

The common background of the pre-trained language models (PLMs) such as BERT-Family, GPT-Family that implemented by self-supervised pre-training with transformer architecture is that, from a data perspective, 'unlabeled text data' such as web Wikipedia data is very abundant, while the number of 'labeled text data' required to perform NLU is not abundant. To handle this gap, researchers presented a method of 'pre-training' these language models through a wide variety of unlabeled text corpus and subsequently fine-tuning each task data according to the task, and show that the performance increases by a high margin.[7], [8]

As these pre-training techniques continue to improve, BERT has been further improved with RoBERTa [9], ALBERT [10], and GPT with GPT-2, GPT-3. Most of these improvements can be seen as reflecting new techniques to make the dataset for pre-training better or to improve the shortcomings of pre-training techniques [34].

2.3.3 Large-Scale Pre-trained Language Models

Models called large-scale pre-trained language models include GPT, BERT, T5, and BART. Recently, as the number of model parameters and the number of unlabeled text corpus used for pre-training increase, it is becoming increasingly obvious that the performance of the NLP task improves in proportion to that. [14] Of course, in studies such as ALBERT[10], it is mentioned that large models can cause many disadvantages, but selecting a model with an appropriate number of

large-scale parameters is an essential process. Therefore, this subsection presents the background of how the model parameter of the transformer-based model is calculated and the model parameter size of the major large-scale PLMs.

How to calculate the number of PLM's parameters?: Figure 3 shows that how to calculate the number of model parameter of BERT model structure. Shape represents the model size of the BERT-base-uncased model released through Google Brain's github. The BERT-based-uncased model is known to have a model parameter of 110M, and you can see how it is calculated through Figure 3. BERT's layer consists of an embedding layer that first embeds text input data as a vector, 12 transformer layers, which are key layers for advanced NLU, and finally a pooler layer for finalizing the output of the transformer encoder layer. The 'Embedding' layer is based on the shape of [30522,768], which indicates that the input text sequence is embedded in 768 dimensions by word vectors through 30522 word vocabulary look-up tables. By adding the shapes of all layer keys in this way, I can calculate that the model parameter of BERT-based-uncased is 109,482,240. In general, when a model is released through open source, it tends to be released in 10 million units for researchers to easily remember, such as 110M rather than 109M.

BERT Layer	Key	Shape	Count
Embedding	embedding.word_embeddings.weight + position_embeddings.weight + token_type_embeddings.weight +LayerNorm.weight	[30522, 768] +[512,768] +[2,768] +[768] +[768]	23,837,184
Transformer Encoder x 12	encoder.layer.attention.self.query[weight,bias] +self.key[w,b] +self.value[w,b] +self.dense[w,b] +encoder.layer.intermediate.dense[w,b] +output.dense[w,b] +output.LayerNorm[w,b]	[768,768]+[768] x 4 [3072,768] + [3072] [768,3072] + [768] [768] x 2	7,087,872 x 12 = 85,054,464
Pooler	pooler.dense[w,b]	[768,768] +[768]	590,592

Fig. 3: Table showing how model parameter count by layer of BERT model is calculated

Model parameters of Most popular PLMs

The model parameters of pre-trained language models tend to be released through open source platforms (github, hugging face), as shown in Table 1 of the popular and mainly-used models. For each model, nparams means model parameters, nlayers means the number of transformer layers, and dmodel means hidden size. [35]

Table 1: Parameter information for major PLM

Models	nparams	nlayers	dmodel
GPT-3 6.7B	6.7B	32	4096
GPT-3 (Main, 175B)	175B	96	12288
GPT-2 XL	1.5B	48	1600
GPT-2	117M	12	768
BERT-base	110M	12	768
BERT-large	340M	24	1024
RoBERTa-large	340M	24	1024

2.4 An Overview of Korean NLP

Natural language processing is highly influenced by the nature of the field. In other words, because the characteristics vary from language to language, there is a high demand for research on how to understand the characteristics of the language and improve performance. Influenced by the development of English-based NLP, which has achieved successful results in the order of Transformer-GPT-BERT-(Getting bigger) since 2017, Korean NLP has independently developed into an independent field of Korean NLP itself in the same way as Korean BERT and Korean GPT.

This section provides a background to help you briefly understand the current trends of Korean NLP.

Technical status: Most Korean NLP researches tend to focus on successfully applying international research cases that have proven successful results in international societies to Korean language domains. This trend allows studies introducing new methods of NLP to give a glimpse into the focus of the standard language on English. SKTBrain's KoBERT [36] and KoGPT2 are the

best examples of research showing this Korean NLP trend, which are the same specifications as BERT and GPT-2 model parameters, and all options as the pre-training objective function, but only the pre-training text corpus is collected and distributed in Korean. According to information released on the pre-training dataset, 5 million wiki corpus and 20 million news sentiments were used for KoBERT's learning. [36] In addition, recently, HyperCLOVA [24] and KoGPT [25] were successfully released to the academic world and open source with the motif of Korean-based GPT-in order to apply successful cases of GPT-3 to Korean natural language processing.

Task and Benchmarks: I introduce representative examples of Korean NLP benchmark datasets for evaluating the performance of these models. There has traditionally been a formalized Korean NLP benchmark, mainly in terms of natural language understanding, with the most representative benchmarks being text classification's NSMC and question answering's KorQuAD [30]. In addition, with the release of the Korean Language Understanding Evaluation (KLUE) Dataset, benchmarks for previously lacking Korean language understanding tasks such as natural language inference (NLI), named entity recognition (NER), and dialogue state tracking (DST) can be formalized. [37]

3 Methods

3.1 Prompt-Based Few-shot Learning

This subsection covers prompt-based learning to be used as the main technique in this paper. Prompt-based few-shot learning, also known as ”in-context learning,” is a new type of inference that induces the model to predict output by giving input from data rather than skipping the process of learning data, such as [Figure 3.1.]. At this time, the model learns the data as a few shot through this instruction, as the input instruction includes a small number of datasets together with an appropriate inductive sentence called ’prompt’. The reason why this is possible is unclear, but in terms of methodology, the model is similar to meta-learning, which automatically learns what the task is by itself by looking only at the data[38].

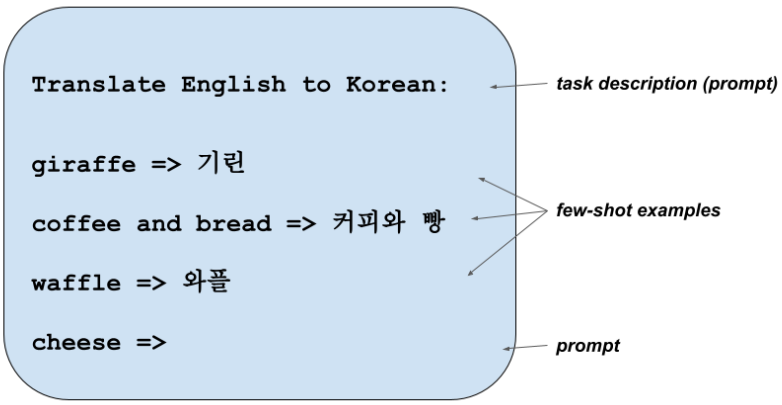

Fig. 4: Example of prompt-based few-shot learning for translation task

In conclusion, the main role of 'prompt' in prompt-based few-shot learning is to wrap natural language-based text around the few-shot data, leading the model to better understand the data only with input instruction without learning. For this reason, selecting appropriate prompts to help the model better understand the data held can be a factor that increases performance under model conditions of the same parameters. In other words, presenting the prompt as a worst case is a very big obstacle to triggering the model's few-shot performance, no matter how well the model is equipped. Also, in terms of the data for few-shot, of course, as various previous studies indirectly demonstrate these results, it is highly likely that performance variances can occur even in the same model parameters depending on the number of few-shot data for few-shot learning. In addition, even if k pieces of few-shot data are selected, it will be very likely that poor task performance can occur if it is selected only as extremely biased data. That is, a process of proving performance differences in how well k data are selected may be required.

3.2 Fine-tuning vs Prompt-based Few-shot Learning

In this subsection, we look at the differences between prompt-based learning paradigm and existing fine-tuning. As described in Figure 5, fine-tuning and prompt-based few-shot learning have the greatest difference in the training stage.

In conventional fine-tuning-based training, model parameters are continuously updated by learning a large amount of learning datasets, while prompt-based few-shot learning is a 'model freeze' method that does not modify model parameters θ that only include data in input instruction but do not perform any gradient updates. In addition, in terms of the number of data required for task

learning, prompt-based few-shot learning can learn tasks sufficiently through prompt engineering even with a small amount of data compared to fine-tuning parameters that require sufficient amounts of data.

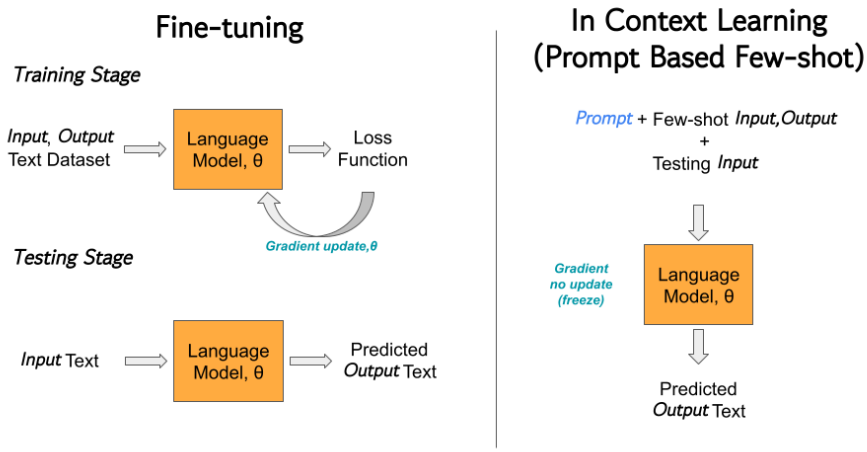

Fig. 5: A figure comparing Fine-tuning and Prompt-based Few-shot Learning

3.3 Target Tasks: Korean NLU Dataset

In general, the most commonly applied benchmark datasets to measure the performance scale of the natural language understanding of language models are SST-2 (Sentimental Analysis) [39], SQuAD (Question Answering) [29]. In Sentimental Analysis, SST-2 is constructed as shown in Figure 6 and is mainly applied to evaluate how well the model performs 'positive' or 'negative' binary classification. However, because these benchmarks are English-centric, these cannot evaluate the performance of the Korean-specific task of the language model. Fortunately, the NAVER Sentimental Movie Corpus (NSMC) dataset is labeled positive/negative with reviews of Korean-based films, and the amount is

appropriate, so it can be a good alternative. An example of the NSMC dataset is shown in Figure 6.

Similarly in the Question Answering task, the SQuAD dataset, which predicts the correct answer through a given context and question, is one of the most commonly used datasets as a benchmark. KorQuAD [30], which emerged due to the demand for datasets to evaluate the model's performance for Korean-based Question Answering, is a dataset built by LG CNS and maintains the format the same as SQuAD, but source data is in Korean. Examples are illustrated in Figure 7.

SST-2 [English]

contains very few laughs and even less surprises	**Negative (0)**
generous and subversive artworks!	**Positive (1)**
shot on ugly digital video	**Negative (0)**

NSMC [Korean]

원작의 긴장감을 제대로 살려내지 못함	**Negative (0)**
사이먼 페그의 명연기가 돋보였던 영화!	**Positive (1)**
아 더빙 진짜 짜증나네요	**Negative (0)**

Fig. 6: An example of SST-2 and NSMC Dataset

SQuAD [English]

The Lobund Institute grew out of pioneering research in germ-free-life which began in 1928 (...)	***Context***
When did study of a germ-free-life begin at Notre Dame?	***Question***
1928	***Answer***

KorQuAD [Korean]

1955년 프랑스의 탐험가인 Fernand Navarra가 발견한 목재 파편의 경우, 스페인의 임업 연구소에서 목재의 특성을 토대로 5000년 전의 것이라고 밝히긴 했으나 (...)	***Context***
1955년 프랑스 탐험가가 발견한 목재파편은 몇 년 전 것이라고 밝혀졌는가?	***Question***
5000년 전	***Answer***

Fig. 7: An example of SQuAD and KorQuAD

3.4 Large-scale Language Models

In this subsection, we consider large-scale language models to select the suitable large-scale pre-trained language models (PLMs) required for prompt-based few-shot learning in this paper. First, in Table 2, the model parameters, layer number, and model dimension of the large-scale pretrained language models we considered are tabulated. As already mentioned in Section 2, the appropriate PLM for in-context learning, our target methodology, prompt-based few-shot learning, is when the model parameter is greater than 1 Billion. GPT-3 is available in a wide variety of sizes, and the first model with a model parameter of 1 Billion or higher is the GPT-3 XL, consisting of 24 Transformer Decoder Layers, with a model parameter of 1.3B. It should be noted that the performance of the 1.3B version is quite unstable [14], [27], and therefore the most reasonably used model parameter is the GPT-3 6.7B version. In addition, GPT-J 6B, released

by Eleuther AI, is based on a method called 'mesh transformer' and is often used because it also shows excellent and stable in-context learning performance.

Meanwhile, in Korean-based PLMs, there are two model options for the GPT system that can be selected with model parameters of 1B or more: KoGPT (Kakao Brain) and HyperCLOVA (NAVER clova). It can be inferred that both models have approximately 6 Billions model parameters and are based on the GPT-3 6.7B version and GPT-J 6B. However, since HyperCLOVA is not fully disclosed as an open source, I have chosen KoGPT 6.16B as the PLM for prompt-based few-shot learning in the most reasonable Korean tasks.

Table 2: Parameter information for major PLM

Models	**nparams**	**nlayers**	**dmodel**
GPT-3 XL	1.3B	24	2048
GPT-J 6B	6B	28	4096
GPT-3 6.7B	6.7B	32	4096
GPT-3 175B	175B	96	12288
KoGPT	6.16B	28	4096
HyperCLOVA	6.9B	28	4096

3.5 GPT Understand too

This section briefly explains why models of the GPT system, not BERT, are considered, selected, and applied even though it is a natural language understanding task. If a fully-connected layer is attached to BERT, which consists only of Transformer's encoders, it can achieve high performance in natural language understanding.

However, this corresponds to a fine-tuning paradigm, and from the point of

view of prompt-based learning, the next token cannot be predicted through natural language. This is because BERT's pre-training method is based on masked language modeling that predicts intermediate words.

On the other hand, GPT (consisting only of transformer's decoder) can predict the next word very well given its text because the prior learning method is Causal Language Modeling, which repeatedly predicts the next word. And since the first successful solution to NLU problems through in-context learning in the GPT-3 research paper was presented, it is very natural to use a model of the GPT system, not BERT, to solve the NLU problem in a paradigm that applies prompt-based few-shot learning. Of course, proper prompt engineering is essential because the GPT model cannot understand the task if the prompt is not given.[14], [19]

3.6 Prompts for In-Context Learning

In this subsection, we introduce the prompt engineering tasks we have configured to actually perform prompt-based few-shot learning (In-context learning).

There are two points to be designed in this section.

First, we experiment with an appropriate prompt to actually apply prompt-based few-shot learning to NSMC, KorQuAD data, two Korean task benchmarks mentioned in section 3.3. And through this, when the model receives instruction combining few-shot data and prompt as input, it is presented to understand the task and make good predictions.

Second, prompt-based few-shot learning tends to have unstable performance

depending on the two requirements: prompting (=prompt engineering), which is the task of selecting appropriate prompts, and selecting data for few-shot learning. Therefore, we conducted a comparative study by properly converting these two requirements to suggest whether the selection of prompt engineering and few-shot data is critical for the task's score results even in the same model parameters.

Specifically, in both the case of the Korean Sentiment Analysis (=Binary Classification) task through the NSMC dataset and the Korean Question Answering task through the KorQuAD, four and three directly selected prompt formats were prepared, respectively. The prompt format is tabulated in Figure 8 and Figure 9 for NSMC and KorQuAD, respectively. In addition, an example of the KorQuAD instruction input used in the experiment by combining Prompt and few-shot data is shown in Figure 10.

In this paper, we analyze the results through performance comparison experiments in Section 4 by constructing several few-shot data settings through prompt engineering in this manner. In conclusion, what we want to get from this comparative experimental process is to get an answer to what points and factors act as important points in prompt-based learning.

Prompt #1	{review} ({label})
Prompt #2	리뷰: {review} \n 감정: {label} \n
Prompt #3	리뷰: {review} \n 감정: {label} \n\n
Prompt #4	{{{review}}}\n 이 리뷰는 {{긍정,부정}}중 무엇?\n {label} \n\n

Fig. 8: Examples of 4 Prompts used for Prompt-based learning of NSMC datasets

Prompt #1	아래 지문을 읽고 물음에 답을 유추하시오. \n {context} \n 질문: {question} 답변: {answer}
Prompt #2	질의응답 태스크 수행: \n {context} \n 질문: {question} \n 답변: {answer}
Prompt #3	{context} \n {question}에 대해 답이 뭔지 아니? \n {answer} \n

Fig. 9: Prompt texts for Prompt-based learning for KorQuAD

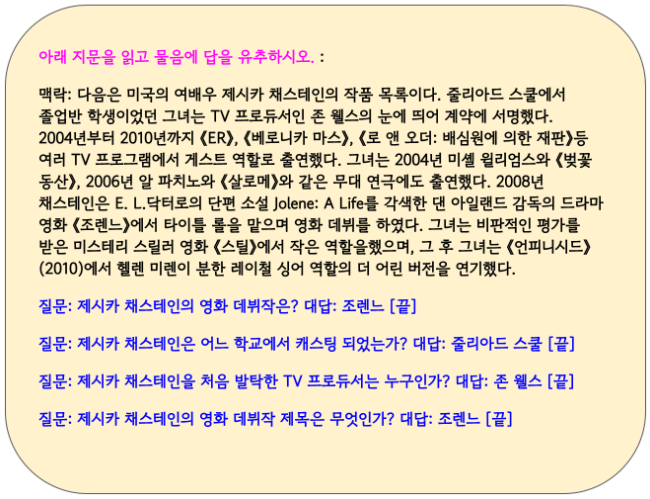

Fig. 10: Example of applying prompt engineering to the few-shot setting of the KorQuAD Dataset

3.7 Prompt-based Few-shot Learning for New Custom Task

Sections 3.3 to 3.6 of this paper aimed to make the Korean language model work well for typical natural language tasks. In this subsection 3.7, I constructed a simple experiment to apply it to a new task called ”Arbitrary Text Style Transfer” using the method of prompt-based few-shot learning described in 3.1 to 3.6 to demonstrate that it can maximize the benefits (= Extremely small amount of data works fine) of few-shot learning. The ”Arbitrary Text Style Transfer” task can be

viewed as a task that converts the text style we want, not a typical dataset. By constructing a data pair with the desired pattern into a few-shot setting including a prompt, a new input text can be defined as a task to see if the model can produce a converted output from the text style intended by the few-shot setting. [40]

I tried to link this learning method of text style transfer (TST) with a text-to-image generation model. Specifically, the input text method of the Text-to-Image Generation model is characterized by a command format, and I constructed prompt-based few-shot learning to implement a TST that aims to convert such dialogue-type text into a command format, assuming dialogue text is given.

For the used few-shot data, four data pairs were arbitrarily written. Since it is not aimed at quantitative performance comparisons, only one prompt format is used. Figure 11 is a diagram showing an overview of a kind of *Textchat-to-Prompt* task configured to apply arbitrary text style transfer to text-to-image modeling. In addition, Figure 12 shows examples of prompt engineering and few-shot data pair settings used.

The difference in results for this is presented in Section 4.

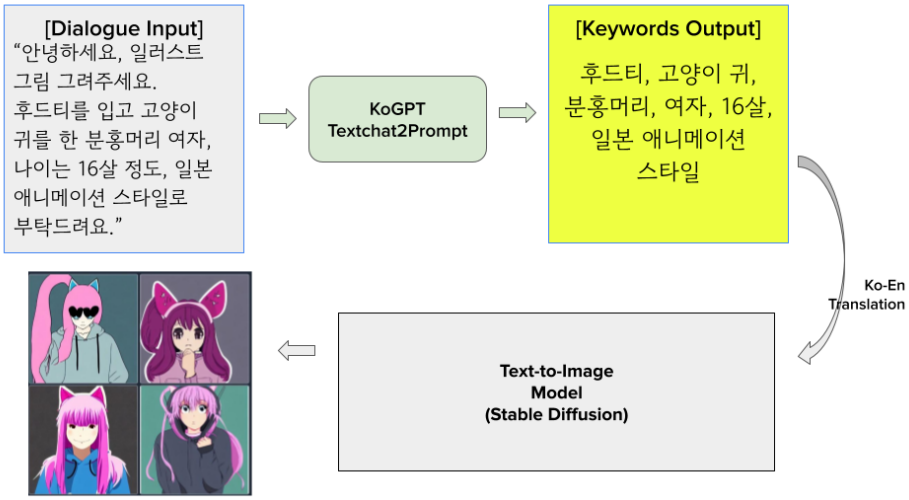

Fig. 11: Overview of the entire process of configuring Arbitrary Text Style Transfer as an interactive text-based image generation modeling

Used Prompt	입력:{input}\n 출력:{output}\n\n
Paired Data Example	**input:**웹툰 표지로 일러스트를 쓰고싶은데 그림 그려줘. 웹툰은 좀비 던전과 보물에 얽힌 비밀들에 관한 내용이고, 내가 필요 한 일러스트는 좀비 던전을 지나는 칼을 든 금발의 5살 여자 아이야. **output:**일러스트레이션, 웹툰, 좀비 던전, 칼을 든, 금발, 5살, 여자 아이

Fig. 12: Prompt text and few-shot data example for prompt-based arbitrary text style transfer (Textchat2prompt)

4 Experimental Results

4.1 Experimental Setups

This section describes in detail the experimental setup used in this study. For all GPU computing environments used, Colab Pro's GPU version was used.

In addition, in the case of the pre-trained language model, only the KoGPT 6.16B version [25] was used based on the basis presented in section 3.4. It took about 5 to 7 minutes to load the corresponding model (size: 12.3GB) based on the Colab Pro.

To load the pre-trained weights, the Transformers [35] library was used, and the transformers used the 4.22 version via pip install. Figure 13 shows a diagram of the code for loading KoGPT with the transformers library. PyTorch was used as a deep learning framework.

Because it does not learn or train, the hyperparameter for the training stage was not included in the experimental setup. In other words, an experimental environment that does not use any hyperparameters such as loss function and epochs was used.

```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

tokenizer = AutoTokenizer.from_pretrained(
  'kakaobrain/kogpt', revision='KoGPT6B-ryan1.5b-float16',  # or float32 ver
  bos_token='[BOS]', eos_token='[EOS]', unk_token='[UNK]', pad_token='[PAD]'
)
model = AutoModelForCausalLM.from_pretrained(
  'kakaobrain/kogpt', revision='KoGPT6B-ryan1.5b-float16',  # or float32 ver
  pad_token_id=tokenizer.eos_token_id,
  torch_dtype='auto', low_cpu_mem_usage=True
).to(device='cuda', non_blocking=True)
```

Fig. 13: Example of code to load Pre-trained KoGPT into huggingface's transformers

Instead, setting the generate parameter of the model becomes a very important factor, so in the experiment of this study, some hyperparameters of the model.generate() process are presented as set-up as follows.

model generate hyperparameters

- tokenizer: 'KoGPT6B-ryan1.5b-float16' tokenizer - tokenizer max length: input token length + 6(NSMC), 6(KorQuAD)

- temperature[41]: 0.8

In the process of setting the prompt, after referring to various related studies [21], [24], [26], a prompt suitable for the task was directly written. In addition,Only simple Python grammar such as f-print, 'str' + 'str' was used for prompt engineering, which merges Prompt with few-shot data in an appropriate form, based on python-string format.

4.2 Prompt-based Few-shot learning for Korean Sentiment Analysis

In this section, the results of applying the methodology presented in section 3 to the Korean Sentiment Analysis Task are presented and analyzed.

First, the fixed setting is summarized as follows.

● Train Strategy: Prompt-based Few-shot Learning

● Used Pre-trained Language Model: Kakaobrain/KoGPT (6.16B ver)

● Used Benchmark Dataset: Naver Sentiment Movie Corpus (NSMC)

● Used Prompt formats: 4 prompt formats are used. (Details are same as section 3.6's Table)

The variable conditions are as follows. We show performance differences when the criteria below change.

● Selection of prompt format : prompt #1 to prompt #4 are same as Sec 3.6 (Sec 4.2.1)

● Selection of Index of Example Data: Random select #1, #2, #3 (Sec 4.2.2)

● Selection of the number of Shots : 32,16,8,4,1 (Sec 4.2.3)

4.2.1 Selection of prompt format

The subsection presents the results of a direct experiment on the performance change when only four prompt formats are changed. Additional main settings are

as follows.

● The number of shots: k=32

● Few-shot data picking strategy: randomly picked.

● The number of test data that we used: 1000 (Test data[0:1000])

We present the results. The results are shown in Table 3, and the main points considered are as follows.

● The result was calculated as accuracy.

● Since the Prompt-based Few-shot learning method is generation-based label prediction, the range of labels cannot be completely controlled. Therefore, we first present 'Matching', a measure of whether the model predicted the correct label of positive/negative well or the wrong label. This is a ratio indicating how much one of the positive/negative data has been output.

● Second, in presenting the accuracy, the accuracy considering matching and the entire accuracy not considering matching are provided together. Through this, it is possible to grasp the strengths and weaknesses of the model to some extent.

● All accuracy was rounded to the third decimal place.

Table 3: Results: Selection of Prompt Format

Prompt	Matching Accuracy	Acc (only Matching)	Acc (Total)
Prompt #1	0.660	**0.799**	0.460
Prompt #2	0.378	0.610	**0.242**
Prompt #3	**0.990**	0.640	**0.630**
Prompt #4	**0.998**	0.645	**0.643**

The results were very interesting. First of all, although all the conditions other than the format of the prompt were the same, the minimum value of the total acuity, which can be seen as the performance of the task, was 0.242 maximum value of 0.630, which was different by up to about 39%, In addition, Prompt #3 and #4 showed more than 10% higher accuracy performance than other Prompt formats (#1).

As a result, it has been proved that proper prompt engineering in the prompt-based few-shot learning of the Korean Sentiment Analysis task is a key element that is very disruptive to the performance of the model.

In addition, as you can see from the table in Section 3.6, Prompt 2 and Prompt 3 are almost the same, and the only difference is that there is one more 'n' that goes in at the end. However, Prompt 3 correctly predicted labels up to 0.990, while Prompt 2 was only 0.368. Intuitively, it can be seen that open strings such as 'n' contribute greatly to letting the model learn the few-shot data, resulting in the performance of Prompt 2, and prompt 3 reaching 39%.

As a result, it can be seen that on the contrary, if the correct prompt is not given, the performance may plunge.

It may also be inferred that k=32 may have been insufficient in the case of

Prompt 1, which is relatively simple. For example, one of the related studies [24], [28], applied k = 70 shots to NSMC to HyperCLOVA.

4.2.2 Selection of Index of Example data

The relevant subsection presents the results of a direct experiment on the performance change when only Random select 1, 2, and 3 are changed in the process of selecting the few-shot data for k-few shot. Additional main settings are as follows.

- The number of shots: k=32
- Few-shot data picking strategy: randomly picked.
- The number of test data that we used: 1000 (Test data[0:1000])
- Used Prompt format: Prompt 1 (Matching Acc: 0.660)

The results were as shown in Table 4. This result shows that the performance difference can widen to 0.048 with multiple random choices of few-shot data in the total Accuracy, the task performance of the model. In addition, it is significant that there is a difference of up to 8.5% in Matching Accuracy, and related research can be used as an example to analyze the reason for this, Zhao et al's study [27] suggested that setting the few-shot data randomly could break the balance of the number of labels, which could lead to a performance difference.

Also, one of the reasons why it can be intuitively identified is that in the case of randomly extracted data, the probability of mixing data that is abundant or not abundant is very unstable, so this performance difference occurs only in the

process of selecting k-shot data.

Table 4: Results: Selection of Index of Example data

Select	Matching Accuracy	Acc (only Matching)	Acc (Total)
Random Select #1	0.660	0.799	0.460
Random Select #2	0.702	0.806	**0.508**
Random Select #3	0.617	**0.859**	0.476

4.2.3 Selection of the number of Shots

This section presents the results of a direct experiment on the performance change when only Selection of the number of shots is changed. Additional main settings are as follows.

● The number of shots: k=32, 16, 8, 4, 1

● Few-shot data picking strategy: first randomly 32 $\rightarrow$ Then, randomly select k data from within 32

● The number of test data that we used: 1000 (Test data[0:1000])

● Used Prompt format: Prompt 4 (Matching Acc: 0.998)

The results are shown in the table 5 below. It can be seen that there is little difference up to k=32,16,8, but it can be seen that the result falls as k rapidly decreases to 4 and 1. In fact, it can be seen that it has decreased by up to 36% at k=1 (0.268) compared to the best total Accuracy of 0.643 (k=32).

In addition, the key to how far the stable matching acuity is maintained by applying Prompt 4, which previously showed a matching acuity of 0.990, but it

can be seen that it fell to the 50% range as k=1.

This consequently suggests that k=1 is not a suitable few-shot number at all for smooth binary classification with two labels, and that the process of setting a suitable amount of k to make the label understandable is a key factor in stabilizing performance.

Table 5: Results: Selection of the number of Shots

Shots	Matching Accuracy	Acc (only Matching)	Acc (Total)
k=32	0.998	0.645	0.643
k=16	0.997	0.620	0.616
k=8	0.983	0.644	0.627
k=4	0.940	0.578	0.518
k=1	**0.509**	**0.759**	**0.268**

4.3 Prompt-based Few-shot learning for Korean QA

In this section, the results of applying the methodology presented in section 3 to the Korean QA (Question Answering) Task are presented and analyzed.

First, the fixed setting is summarized as follows.

● Train Strategy: Prompt-based Few-shot Learning

● Used Pre-trained Language Model: Kakaobrain/KoGPT (6.16B ver)

● Used Benchmark Dataset: LG-CNS/ Korean Question Answering Dataset ver 1.0 (KorQuAD)

● Used Prompt formats: 3 prompt formats are used. (Details are same as section 3.6's Table)

The variable conditions are as follows. We show performance differences when the criteria below change.

● Selection of prompt format (prompt #1 to prompt #3, same as Sec 3.6) (Sec 4.3.1)

● Selection of label: Gold label vs random label (Sec 4.3.2)

4.3.1 Selection of prompt format

Much like section 4.2.1 of NSMC, this section constructs an experiment to verify the performance of the Korean QA Task and demonstrate how different the selection of prompt formats can make a difference in task performance, and presents the results.

The conditions introduced for the relevant subsection are as follows.

● The number of shots: k=4

● Few-shot data picking strategy: (i) context: randomly → (ii) 4 QA pairs on the context are randomly selected.

● The number of test data that we used: 5774 (All Test data are used)

● Evaluation Metrics: F1 Score / Exact Match

The prompt format used is the same as the KorQuAD table in Section 3.6, and we experimented with how much performance can be changed when three completely different formats are presented.

We used two evaluation metrics: exact match (EM) and F1 scores. The exact match (EM) score calculates when the predicted answer exactly matches the label answer, and F1 score measures the similarity between 0 and number of how much the predicted answer matches the label answer after removing unnecessary expressions, such as special characters and punctuation through preprocessing.

The results are shown in the Table 6 below.

First, in the KorQuAD benchmark, a QA task that is more difficult to predict than the NSMC, the Sentimental Analysis, the model predicts an output containing the correct label by 62.889%. In addition, it was shown that when the Prompt was changed, the F1-score had a symbolic difference of up to about 9% and the Exact Match score could also be reduced by up to about 11%.

This suggests that, like the Sentiment Analysis Task, the Korean Question Answering task requires the proper format of the prompt to reliably extract performance.

Table 6: Results: Selection of prompt format for KorQuAD

Prompt	**F1**	**Exact Match (EM)**
Prompt #1	**62.889**	**41.756**
Prompt #2	59.387	35.036
Prompt #3	53.836	30.880

4.3.2 Selection of label: Gold label vs random label

In this section, we constructed an experiment through the Korean QA Task to prove whether 'gold label', which means the correct label in prompt-based few-shot learning, is really important or if the model can learn the task well enough if

it is formatted.

The experimental conditions introduced for the relevant subsection are as follows.

● The number of shots: k=4

● Few-shot data picking strategy: (i) context: randomly → (ii) 4 QA pairs on the context are randomly selected.

● The number of test data that we used: 5774 (All test data are used)

● Label → Gold label, Random label, zero-shot (k=0, only prompt)

● Used Prompt format: Prompt 1 (F1/EM:62.889, 41.756)

The results are as shown in Table 7.

In conclusion, when the 'gold label' was replaced with a 'random label', the F1 score decreased by 6.8%, and the Exact match score decreased by about 9%, showing a very meaningful performance drop. In addition, if the data is not given by replacing the gold label with zero-shot at all, the performance of F1 score decreases 21.5% compared to the gold label, and EM score decreases very significantly by about 30%.

On the other hand, compared to the random label, the F1/EM of the zero-shot label shows a significant decline with a decrease of about 15% and 21%, respectively. This suggests that even if the gold label is replaced with a random label, task learning can be smoother than zero-shot, where the format of the data is completely unequipped, In other words, we show that not only can the gold

label be a meaningful factor in performance to some extent, but also the format of the few-shot data in prompt engineering itself can be meaningful even if the label is damaged randomly. This suggests that performing prompt-based few-shot learning on tasks by randomly labeling unlabeled data may be better than applying it as zero-shot at all.

Table 7: Results: Selection of label: Gold label vs random label for KorQuAD

prompt	Label	F1	Exact Match (EM)
Prompt #1	gold label	62.889	41.756
Prompt #1 (same)	random label	56.064	32.577
Prompt #1 (same)	zero-shot	41.300	11.552

4.4 Prompt-based Learning for Arbitrary Text Style Transfer

In this section, we construct an experiment that applies the Arbitrary Text Style Transfer experiment presented to the Text-to-Image Generation model to explore the additional possibilities of Prompt-based Few-shot learning introduced in Section 3.7.

The specific conditions used are as follows.

● Main Task: (Input= Dialogue Text) →[Few-shot Text style Transfer] → (Output= Command-type Text) → (Better) Text to Image Generation

● Used Pre-trained Language Model: KoGPT 6.16B

● Used Training Strategy: Prompt-based Few-shot Learning

● The number of shots: k=4

● Few-shot data: manually generated data pairs (same as Section 3.7)

● Used Text-to-Image Generation Model: Stability AI's Stable Diffusion [42]

● Evaluation Metric: Qualitative evaluation (based on qualitative evaluation of image generation because there is a limitation that there is no suitable metric because it is a custom task)

The model has the same conditions as above We construct two example interactive text inputs and present the results of the model transforming them.

First, Figure 14 shows the output when two interactive input examples are entered to see if the model learns the arbitrary text style transfer task well through prompt-based learning. It was seen that keyword-based text was extracted very well.

In addition, Figure 15 shows a tendency to fail to grasp the exact requirements as shown in the figure above when the interactive text is entered as it is.

On the other hand, as noise such as 'Please draw', 'Mmm', 'Please', and 'Please!' contained in the interactive text in Example 1 was removed through the arbitrary text style transfer (Textchat2Prompt), the figure below in Figure 15 could more clearly understand the requirements of the interactive text and output the output.

To address the problem of poor interactive text input that may arise in the process of using image-text multi-modal technology, these results are

transformed into a keyword-type input based on a direct design of Arbitrary Text Style Transfer, which enables users to study text-based images at a higher level It implies that there may be potential.

Results: Arbitrary Text Style Transfer, Dialogue to Prompt input

	Text
Example #1 Input Dialogue Text	"그림을 그려주세요. 음 기타를 든 잘생기고 젊은 가수로 부탁하고, 배경은 대학교 축제로 해주세요!"
Example #1 style-changed output	**대학교 축제, 기타를, 든, 잘생기고, 젊은, 가수, 배경**
Example #2 Input Dialogue Text	안녕하세요, 일러스트 그림 그려주세요. 후드티를 입고 고양이 귀를 한 분홍머리 여자, 나이는 16살 정도, 일본 애니메이션 스타일로 부탁드려요.
Example #2 style-changed output	**후드티, 고양이 귀, 분홍머리, 여자, 16살, 일본 애니메이션 스타일**

Fig. 14: Results: Arbitrary Text Style Transfer, Dialogue to Prompt input

Fig. 15: Image created without Arbitrary Text Style Transfer (above) and Image created by applying Arbitrary Text Style Transfer (Textchat2Prompt) (below) results

5 Conclusions

In this paper, I tested how to apply the Prompt-based Few-shot learning method, which is a model freeze training method aimed at the GPT-3 model, which can be seen as a cutting-edge technology in natural language processing, to the Korean pre-trained language model. The optimal experimental environment was formed to apply related studies on Prompt-based Learning, which was previously conducted mainly in English, to Korean tasks, and various meaningful results were derived from Sentimental Analysis and Question Answering tasks, and factors affecting performance were presented with experimental data. Specifically, the KoGPT 6.16B Large-scale pre-trained language model directly applied the methodology for melting into the Korean NLU task, and based on this, it was possible to confirm some increase in qualitative performance in the new task, Text Style Transfer task.

I hope that my small but meaningful research results will help related researchers.

Abstract in Korean

자연어 처리 기술은 자연어 이해와 자연어 생성으로 나눌 수 있는데, 양 방향에서 자연어 이해 분야에서는 BERT, 자연어 생성 분야에서는 GPT 모델이 기존 자연어 처리 모델, 방식들의 패러다임을 바꿔 놓았다. 이들은 트랜스포머 기반의 모델이자 Large corpus로 사전학습된 Pretrained Language Model이라는 공통점이 있으며, BERT는 Transformer의 Encoder, GPT는 Decoder만으로 나누어졌다.

그러나 GPT-3 논문에서 발표된 프롬프트 기반의 퓨 샷 러닝(few-shot learning) 기법인 인 컨텍스트 러닝(in-context learning)은 자연어 생성 혹은 디코더로도 자연어 이해 태스크의 레이블을 생성하는 방식을 통해 매우 높은 수준의 성능을 거둘 수 있음을 증명하였다. 이러한 흐름은 기존의 BERT,GPT에서 한단계 더 발전한 프롬프트 기반 퓨샷 학습(prompt-based learning)이라는 새로운 패러다임을 불러왔다. 그러나 프롬프트 기반 퓨샷 학습의 연구들은 아직 초창기에 접어들어 연구의 양이 부족한 편이며, 한국어 기반의 프롬프트 기반 퓨샷 학습 관련 연구는 아직 많이 이루어지지 못하고 있다.

이 논문에서는, 이러한 흐름을 반영하기 위해 한국어 GPT의 일종인 모델을 이용하여 두 가지 한국어 자연어 이해 태스크에 직접 적용해본다. 또한 다양한 비교 실험을 적용해보며 중요한 점들을 실험 결과를 통해 입증할 수 있도록 하였다.

결과적으로 실험에서 많은 성능 차이를 야기시키는 몇 가지 요소들을 발견하고 Section 4에서 제시하였다. 구체적으로 우리는 KoGPT 6.16B라는 모델 파라미터를 사용하며, Prompt-based Learning을 Korean Sentiment Analysis, Korean Question Answering 태스크의 벤치마크 데이터셋인 NSMC, KorQuAD에 각각 적용해본다. 그리고 마지막으로 새로운 Arbitrary Text Style

Transfer 태스크를 새로 설계하여 적용해본다.

References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[3] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[4] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132 306, 2020.

[5] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[6] M. E. Peters, M. Neumann, R. L. Logan IV, *et al.*, "Knowledge enhanced contextual word representations," *arXiv preprint arXiv:1909.04164*, 2019.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[8] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[9] Y. Liu, M. Ott, N. Goyal, *et al.*, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[10] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[11] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," *arXiv preprint arXiv:2006.03654*, 2020.

[12] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[13] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[14] T. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[15] Y. Zhang, S. Sun, M. Galley, *et al.*, "Dialogpt: Large-scale generative pre-training for conversational response generation," *arXiv preprint arXiv:1911.00536*, 2019.

[16] D. Araci, "Finbert: Financial sentiment analysis with pre-trained language models," *arXiv preprint arXiv:1908.10063*, 2019.

[17] J. Lee, W. Yoon, S. Kim, *et al.*, "Biobert: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[18] M. Chen, J. Tworek, H. Jun, *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021.

[19] X. Liu, Y. Zheng, Z. Du, *et al.*, "Gpt understands, too," *arXiv preprint arXiv:2103.10385*, 2021.

[20] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *arXiv preprint arXiv:2107.13586*, 2021.

[21] V. Sanh, A. Webson, C. Raffel, *et al.*, "Multitask prompted training enables zero-shot task generalization," *arXiv preprint arXiv:2110.08207*, 2021.

[22] S. Ruder, *ML and NLP Research Highlights of 2021*, `http://ruder.io/ml-highlights-2021/`, 2022.

[23] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer.," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[24] B. Kim, H. Kim, S.-W. Lee, *et al.*, "What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers," *arXiv preprint arXiv:2109.04650*, 2021.

[25] I. Kim, G. Han, J. Ham, and W. Baek, *Kogpt: Kakaobrain korean(hangul) generative pre-trained transformer*, `https://github.com/kakaobrain/kogpt`, 2021.

[26] S. Min, X. Lyu, A. Holtzman, *et al.*, "Rethinking the role of demonstrations: What makes in-context learning work?" *arXiv preprint arXiv:2202.12837*, 2022.

[27] Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, “Calibrate before use: Improving few-shot performance of language models,” in *International Conference on Machine Learning*, PMLR, 2021, pp. 12 697–12 706.

[28] S. Shin, S.-W. Lee, H. Ahn, *et al.*, “On the effect of pretraining corpora on in-context learning by a large-scale language model,” *arXiv preprint arXiv:2204.13509*, 2022.

[29] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” *arXiv preprint arXiv:1606.05250*, 2016.

[30] 임승영, 김명지, and 이주열, “Korquad: 기계독해를 위한 한국어 질의응답 데이터셋,” *한국정보과학회 학술발표논문집*, pp. 539–541, 2018.

[31] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, “Spanbert: Improving pre-training by representing and predicting spans,” *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64–77, 2020.

[32] I. Beltagy, K. Lo, and A. Cohan, “Scibert: A pretrained language model for scientific text,” *arXiv preprint arXiv:1903.10676*, 2019.

[33] S. Peng, K. Yuan, L. Gao, and Z. Tang, “Mathbert: A pre-trained model for mathematical formula understanding,” *arXiv preprint arXiv:2105.00377*, 2021.

[34] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, “Pre-trained models for natural language processing: A survey,” *Science China Technological Sciences*, vol. 63, no. 10, pp. 1872–1897, 2020.

[35] T. Wolf, L. Debut, V. Sanh, *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[36] J. Park, *Distilkobert: Distillation of kobert*, `https://github.com/monologg/DistilKoBERT`, 2019.

[37] S. Park, J. Moon, S. Kim, *et al.*, "Klue: Korean language understanding evaluation," *arXiv preprint arXiv:2105.09680*, 2021.

[38] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi, "Metaicl: Learning to learn in context," *arXiv preprint arXiv:2110.15943*, 2021.

[39] R. Socher, A. Perelygin, J. Wu, *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[40] E. Reif, D. Ippolito, A. Yuan, A. Coenen, C. Callison-Burch, and J. Wei, "A recipe for arbitrary text style transfer with large language models," *arXiv preprint arXiv:2109.03910*, 2021.

[41] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.

[42] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.