Professional Portfolio

Jason Heesang Lee

List of Contents

Timeline

GLION INSTITUTE OF HIGHER EDUCATION
2015.08 ~ 2021.02
Bachelor of Business Administration
Hospitality Management
With Luxury Specialization

FOUR SEASONS HOTEL SEOUL
2021.04 ~ 2021.09
People & Culture
Coordinator
Apr

Atheneum
2021.09 ~ 2023.02
Associate
Sep
Sr. Associate
Jun
Team Lead
Aug

Fast campus × KOSME
이어드림스쿨 YEAR-DREAM SCHOOL
2023.04 ~ 2023.12
My First
Hello World!
Mar
My First
Kaggle Submit
kaggle
Apr
Achieved
Notebook Expert
Aug
Achieved
Notebook Master
Oct
Released
DanGam
Dec

2020 2021 2022 2023

Career Transition

FOUR SEASONS HOTEL SEOUL

After graduating from Glion Institute of Higher Education, a Hotel School located in Switzerland, I joined at Four Seasons Hotel Seoul as a People & Culture (P&C) Coordinator.

I was mostly responsible for the Human Resources tasks :
General Administration, Enrollment & Termination Interviews,
Organizing Employee Events and Job Fair.

Atheneum

I joined Atheneum Partners after being scouted by a director I met previously.

I was responsible for about 15 projects per month as a Project Manager.
Exceeded target every month by around 250%, best achievement being 425%.
Thankfully, I was able to be promoted as a Team Lead,
which was the fastest promotion in all global Atheneum offices.

Main Industries Covered : Artificial Intelligence, Semiconductor and Digital Transformation.

Fast campus × KOSME 중소벤처기업진흥공단

이어드림스쿨

스타트업-청년인재 YEAR-DREAM SCHOOL

After being being responsible for number of Artificial-Intelligence related projects at Atheneum Partners, I got more interested in developing AI.

Coincidently, I happened to bump into a news article that Korea SMEs and Startups Agency is collaborating with Fast Campus,
one of the top computer science education company in Korea, to launch a government-funded AI education course : Year-Dream School.

At Year-Dream School, I learned the basics required for Machine Learning and Deep Learning.
Also, I was introduced to Kaggle, which I later found my enthusiasm in.

× Avocadoland

Industry Partnership Project

Date : 2023. 11. ~ 2023.12
Contribution : 40%

Project Overview

Avocadoland

We were required to work with a company, pre-selected by the Ministry of SME for the final project at Year-Dream School.

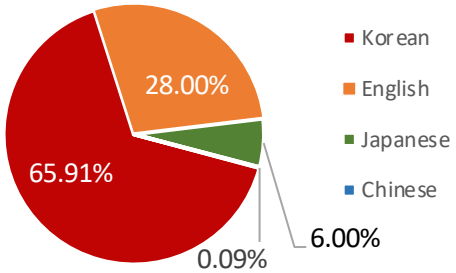
Among 12 different companies, I have decided to work with Avocadoland, a company that services an application named “Momory” where users record their daily lives and earn gems in return.

The purpose of this mobile application is to help self-diagnose the users’ emotional status.

Task

Type of Data: Daily Records of users, recorded in 4 different languages.
Objective: 1. Distinguish the emotion of each record.
2. Find acceptable keywords that mostly influenced the emotion of each record.
3. Accumulate the weekly records, showing the most frequent keywords to the users.
Provided Codes: - Simple Text Preprocessing Code
- Sentiment Analysis with TweetNLP
- Unicode based Language Detection Code

Language Distribution



Questions

Below are the questions we had after taking a brief look at the data and codes.

Language Detection	Keyword Extraction	Improving Current Model	Building Model Evaluation Metric
Is there any error in language detection in the current code?	Should we correct newly coined terms and indiscriminate abbreviations?	Is preprocessing reasonably done?	How can we set the human evaluation standard when checking whether the extracted keywords are acceptable?
	How much weight should we put on each keyword based on the emotional state to make most users accept?	While our priority is to create more precise and explicit keyword extraction model, is there a way to make the model work efficiently and lightly?	Is there any other metric that can be an objective standard?

Initial Obstacle

The company didn’t answer our questions and disappeared until the final presentation.

Therefore, we were on our own and had to find our way out.

Resolving Questions

이어드림스쿨 × Avocadoland

Language Detection

Is there any error in language detection in the current code?
In the baseline code provided by the company, language detection was done using a heuristic approach based on Unicode. Therefore, it performed great for English and Korean but didn't work well for Chinese and Japanese, which share Chinese characters.

How did we solve?
We have used a library called Lingua that work based on dictionaries, which was quite accurate.

Keyword Extraction

We were required to work on all the languages; however, we began with Korean, which took the largest portion of the entire data.

Should we correct newly coined terms and indiscriminate abbreviations?
The most frequent problem in Korean texts is that, as Korean words are combinations of vowels and consonants, there are hundreds of thousands of ways to modify the original form of the word, purposely or mistakenly.

How did we solve?
I have developed a module called JsonSpeller that maps unregistered words with registered words
The logic is explained in detail here.

How much weight should we put on each keyword based on the emotional state to make most users accept?
We have followed the weight predefined by a renowned Psychiatry lab in Seoul.

JsonSpeller: https://github.com/jasonheesanglee/JsonSpeller

Improving Current Model

Is preprocessing module in the baseline code reasonably done?
The baseline code showed how to retrieve the data from the server, but did not do much of preprocessing.

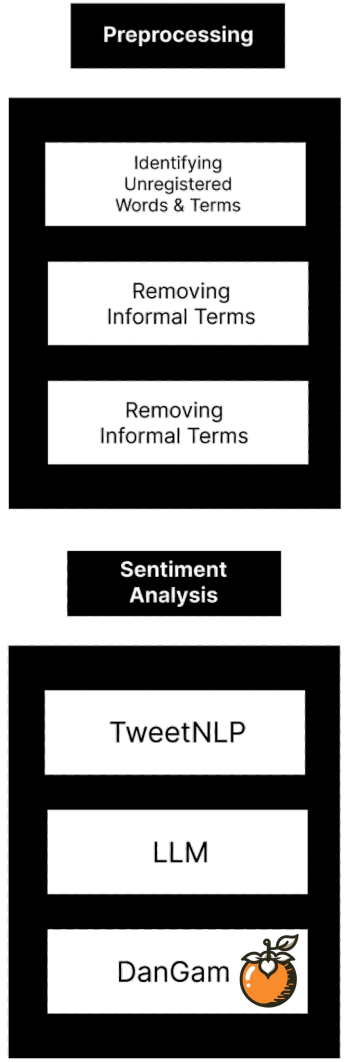
While our priority is to create more precise and explicit keyword extraction model, is there a way to make the model work efficiently and lightly?
TweetNLP was used as an example.
It was light and efficient, but it only performed sentiment analysis on sentence-level.
I have developed a brand-new package called **DanGam** for word-level sentiment analysis.
Its logic is briefly explained in the next slide.

Building Model Evaluation Metric

How can we set the human evaluation standard when checking whether the extracted keywords are acceptable?

Is there any other metric that can be an objective standard?

Unfortunately, these questions remain unsolved.
As each person has a different standard for the level of emotion, it is nearly impossible to create objective standards.



DanGam

이어드림스쿨 × Avocadoland

Compared with other existing research

- **TweetNLP**
  - TweetNLP supports multiple languages including Korean and works well at the sentence level. However, it does not support word-level sentiment analysis.
- **HuggingFace Text Classification Models**
  - Similar to TweetNLP, they support sentence-level sentiment analysis, but not word-level.
- **Word-Level Sentiment Analysis with Reinforcement Learning**
  - This research is similar to DanGam, but DanGam offers sentiment analysis for all the words in each sentence.
- **Word-Level Contextual Sentiment Analysis with Interpretability**
  - The result of this research research is similar to those of DanGam. However, DanGam is an inference tool, in contrast to a Deep-Learning Model that requires training..

How does it work

- DanGam takes a sentence as an input and identifies the general emotion as well as specific emotions within that sentence.
- DanGam calculates the cosine similarity between the sentence and the emotion, and between the sentence and the specific emotion.
- It combines sentence embedding, emotion embedding and specific emotion embedding with weights based on the calculated similarities.
- Then it calculates the cosine similarity between word embedding and the combined embedding.
- If the similarity is high, it suggests that the word has an emotion similar to that of the combined embedding.

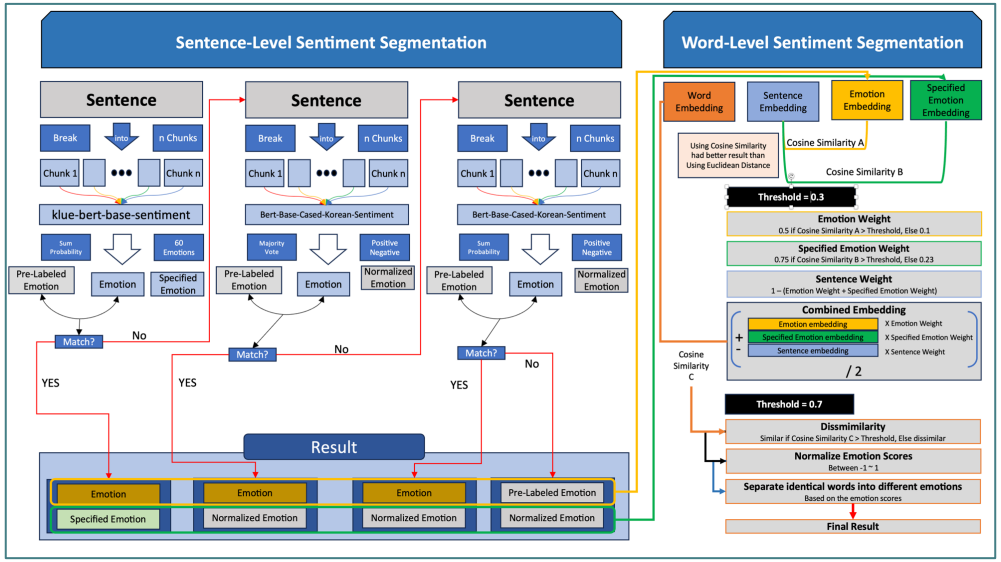
DanGam: https://github.com/jasonheesanglee/dangam

Output Example

- Example Sentence :
  "나는 방금 먹은 마라탕이 너무 좋다.
  적당한 양념에 알싸한 마라향이 미쳤다.
  그런데 고수는 진짜 싫다"
- The resulted output is in a range as below.
  Positive Emotion (1) ~ Negative Emotion (-1)

```
# {'나는': 1.0,
# '방금': 0.8419228076866834,
# '먹은': 1.0,
# '마라탕이': 0.8522973110543406,
# '너무': 1.0,
# '좋다': 1.0,
# '적당한': 0.965806179144829,
# '양념에': 0.7151325862316465,
# '알싸한': 0.4678710873322536,
# '마라향이': 0.328179239525493,
# '미쳤다': 0.34263925379014165,
# '그런데': -0.07491504014905744,
# '고수는': -0.7992964009024587,
# '진짜': -0.9295882226863167,
# '싫다': -0.9120299268217638}
```

Visualized Diagram



Final Output and Recap

이어드림스쿨 × Avocadoland

Final Output

"태형이와 시오니와 찬혁기, 배나영, 이희상과 함께 롯데월드에 갔다.
롯데월드 좋아졌더라. 마라탕을 먹고 오꼬노미야끼도 먹고, 탕후루도 먹었다.
나영이와 희상이가 싸우게 되어 분위기가 좀 그랬다."

	AL Baseline	Team's Preprocessing AL's Sentiment Analysis	Team's Preprocessing Team's Sentiment Analysis
Positive	롯데 0.6658 탕후루 0.5111 월드 0.4804	롯데월드 0.5291 시오니와 0.466 마라탕 0.4518 찬혁기 0.4106 탕후루 0.4104 배나영 0.3737 이희상 0.3714	마라탕 : 0.3355 오꼬노미야끼 : 0.2269
Neutral	분위기 0.0 상이 0.0 나영이 0.0	나영이 0.0 나영 0.0 희상 0.0 희상이 0.0 분위기 0.0	태형 : -0.1958 , 배나영 : -0.1226, 이희상 : 0.068, 롯데월드 : -0.0746, 탕후루 : -0.175, 나영 : -0.1226, 희상 : 0.068
Negative			나영 : -0.4526, 희상 : -0.3573, 분위기 : -0.6941

The final output was better than how we initially expected it to be.

The initial model did not distinguish well between positive and neutral words and even failed to distinguish words with negative emotions.

However, the new model provides a more accurate determination of each word's emotion. It also distinguishes negative words well.

Hardships

As this was our first time collaborating with others as data scientists, it was quite hard to align our methods of approach.
For instance, some of the team wanted to simply count the number of keywords for the final display, while others wanted to calculate the level of impact that the keywords have on the overall context.
Moreover, lack of guidance from the company played a main role on our hardship.

However, with this experience, we learned a lot on how to collaborate as a data scientist, professionally exchanging our thoughts and opinions.

What could be improved

- When extracting keywords, there was too much information loss.
- There were some cases where there are investigations and other parts of speech, so if we had more time, we could have prevented this.
- We wanted to find a better way to stack words using the features of each morphological analyzer.
  We tried it based on a model named "Kkma", which decomposes the sentence into smaller parts, but it could have been better if we could try on other models as well.
- We tried Japanese and Chinese sentiment classification.
  However, if we had more time, we could have developed and introduced it.

Competitions

AI CONNECT

Generating Seamless En-Ko Translation

- **Date :** 2023-10-27 ~ 2023-11-08
- **Type of Data :** NLP / LLM / Machine Translation
- **Rank :** 2nd
- **Used Models :** Mistral & LLaMa2 based pretrained models
- **Focus :** Utilizing LLM models.

DACON

- **Judicial Precedent Prediction**
  - **Date :** 2023-06-05 ~ 2023-07-03
  - **Type :** NLP / Classification
  - **Rank :** Public : 15% / Private 18%
  - **Used Models :** Sentence-BERT / Legal-BERT
  - **Focus :** Getting familiar with Text data
- **Sound Emotion Recognition**
  - **Date :** 2023-05-07 ~ 2023-06-05
  - **Type :** Acoustic / Emotion Recognition / Classification
  - **Rank :** Public : 43% / Private 40%
  - **Used Models :** Librosa / RandomForest / DecisionTree / XGBoost / LightGBM
  - **Focus :** Getting familiar with Acoustic data

kaggle

Linking Writing Processes to Writing Quality

- **Date :** 2023-10-03 ~ 2024-01-10
- **Type :** Tabular / Classification
- **Rank :** Public : 17% / Private 10% / Top 176th
- **Used Models :** Rule-Based / XGBoost / TabPFN
- **Focus :** Reconstructing essays using given dataset.

CAFA 5 Protein Function Prediction

- **Date :** 2023-04-18 ~ 2023-12-21
- **Type of Data :** Tabular / Biology
- **Rank :** Public 4% / Private 4% / Top 63rd
- **Used Models : ProtBERT, Prot-T5, ESM2**.
- **Focus :** Solving given problem with Protein Language Models.

- **Kaggle – LLM Science Exam**
  - **Date :** 2023-07-12 ~ 2023-10-11
  - **Type of Data :** NLP / LLM / Question Answering
  - **Rank :** Public : 15% / Private 15%
  - **Used Models :** T5, DeBERTa, LLaMA2, Platypus2, Alpaca
  - **Focus :** Getting familiar with Large Language Models.
- **CommonLit – Evaluate Student Summaries**
  - **Date :** 2023-07-13 ~ 2023-10-12
  - **Type of Data :** NLP / LLM / Text Summary Evaluation
  - **Rank :** Public : 7% / Private 30%
  - **Used Models :** MobileBERT, DeBERTa, Numerous BERT family models
  - **Focus :** Transformer-based Deep Learning model compression
- **ICR – Identifying Age-Related Conditions**
  - **Date :** 2023-05-12 ~ 2023-08-11
  - **Type :** Tabular / Classification
  - **Rank :** Public : 7% / Private 48%
  - **Used Models :** Rule-Based / TabPFN / XGBoost / LightGBM
  - **Focus :** Finding relations between each column and the meta data

Evaluate Student Summaries

Purpose

I participated in the CommonLit – Evaluating Student Summary Competition, where the objective was to develop an automated system for evaluating the quality of summaries written by students in grades 3-12.

This involved building a Deep Learning model capable of assessing how effectively a student captures the main idea and details of a prompt text, along with the clarity, precision, and fluency of their written summary. A comprehensive dataset of real student summaries were given to the participants to train and refine the model.

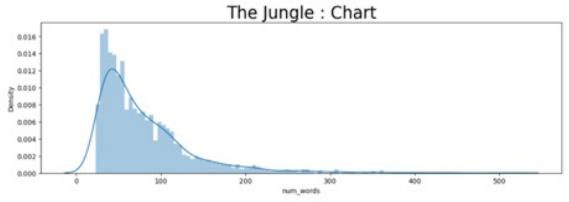
Competition Review

Date : 2023. 07. ~ 2023. 10.
Contribution : 60%
Link : Github / PyPi

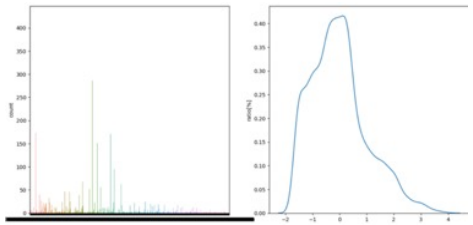
Reviews on other competitions can be found from below.
https://github.com/jasonheesanglee/kaggle

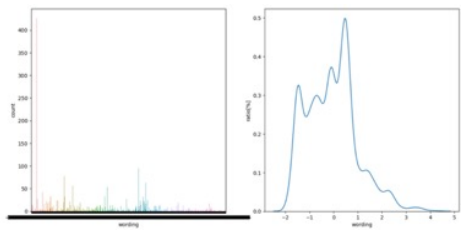
Data Overview

Exploratory Data Analysis



	num_words	num_unique_words	num_chars	num_stopwords	num_punctuations	num_words_upper	num_words_title	mean_word_len
count	1996.0000	1996.0000	1996.0000	1996.0000	1996.0000	1996.0000	1996.0000	1996.0000
mean	79.4509	54.5040	420.7560	42.1242	10.3472	0.0521	3.6844	4.2630
std	55.6905	30.2666	302.9265	29.4349	9.2133	0.3155	3.3914	0.3110
min	23.0000	18.0000	115.0000	8.0000	0.0000	0.0000	0.0000	3.2581
25%	41.0000	32.0000	214.0000	22.0000	4.0000	0.0000	2.0000	4.0522
50%	63.0000	47.0000	330.0000	33.0000	8.0000	0.0000	3.0000	4.2500
75%	100.0000	68.0000	527.2500	53.2500	14.0000	0.0000	5.0000	4.4458
max	509.0000	252.0000	2775.0000	257.0000	88.0000	8.0000	30.0000	5.8947





Train content description:

count	7165.000000
mean	-0.014853
std	1.043569
min	-1.729859
25%	-0.799545
50%	-0.093814
75%	0.499660
max	3.900326

Name: content, dtype: float64

Train wording description:

count	7165.000000
mean	-0.063072
std	1.036048
min	-1.962614
25%	-0.872720
50%	-0.081769
75%	0.503833
max	4.310693

Name: wording, dtype: float64

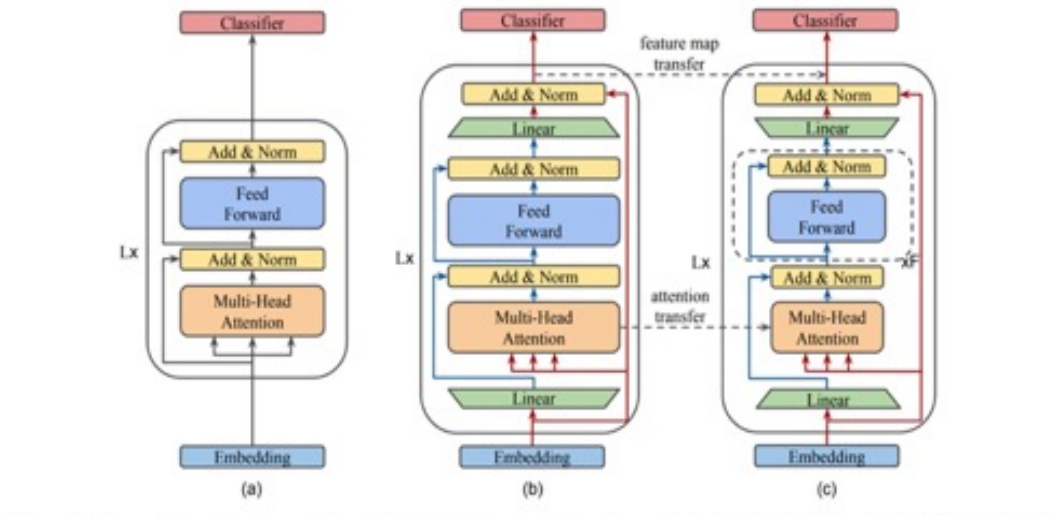
Preprocessing & Spellcheck

- Cleansing the text is a must to run the NLP model effectively
- Many misspelled words were in text:
  ex) “Pharaoh” -> “Pharoh” or “Pharoah”.
- Accuracy : SymSpellPy > PySpellChecker > TextBlob > AutoCorrect
- Speed : AutoCorrect > PySpellChecker > SymSpellPy > TextBlob

Tool	Detected	Correctly Replaced	Misjudged	Time
Grammarly	16 words	16 words	Less 1 word	-
TextBlob	20 words	14 words	2 words 2 names	4 sec
PySpellChecker	18 words	16 words	2 words	1 sec
SymSpellPy	18 words	17 words	1 name	2 sec
AutoCorrect	15 words	12 words	2 words 1 name	0 sec

Model Application

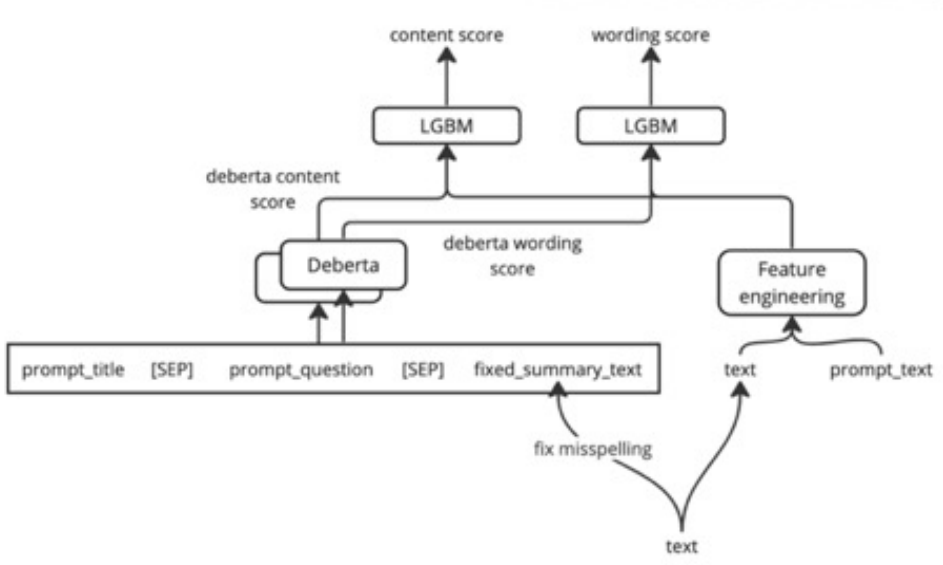
Solving a Real-World Problem with MobileBERT



MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices (Sun et al., 2020)

- As the purpose of this competition is to enhance the teaching and scoring experience of the teachers, I believed making this model lightweight is the key.
- By implementing MobileBERT, the model would run on individual mobile devices, making it accessible to teachers in environments where computer access is difficult.

LGBM – Post-Processing

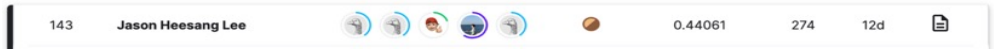


- After getting the result with NLP Deep Learning models, I used LightGBM to improve the score by feature engineering, as introduced in this notebook.
- The original author of the notebook found the hyperparameters through experimentation but added Optuna, a hyperparameter optimization module, as a last step, thinking that it would find better-performing hyperparameters.

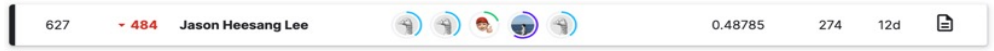
Result & Reflection

Result

- We were placed at the top 143 in the Public Leaderboard, so we believed that we could make it to the Bronze medal when the Private Leaderboard was published.
- Therefore, we tried our best until the very end to make minor changes to our existing solution.



- When the Private Leaderboard was published, we were pretty shocked with the result.



What could be improved

- **First** : The proportion of the Public Data.
  The organizer stated that the test data provided to the competitors is only 13% of the whole test data.
  Data augmentation could have been done as the 1st place team have.
- **Second** : Using SymSpellPy for spellchecking.
  Most competitors used PySpellCheck and AutoCorrect.
- **Third** : Focused too much on MobileBERT.
  If it was to aim for higher rank, MobileBERT should have been neglected earlier.
  However, as it was done on purpose, I have no regret.
- **Fourth** : Need more and better modeling techniques.
  It would have been better if we could also try T5 and BART, as no one else was trying that measure.
  Only if we had a better modeling technique, we could have also tried these two models.

1st Place Solution

- Surprisingly or not, the first-place holder turned out to be using the same baseline that we were using.
- The differences between our solution and the first-place solution are:
  1. They have barely modified the logic of the model.
  2. They not only have used the provided four prompts and generated even more by effectively using Large Language Model.
  3. They have excluded using LightGBM and solved the problem solely with DeBERTa-v3-Large.
- Per their discussion, they have spent more time generating new prompts and summaries with LLM than tuning the DeBERTa model.
- I believe this was quite a smart move, as the competition overview – context has already mentioned utilizing LLM.
  - Even though I don't think this is not the way the organizer expected the LLM to be used, they were the only team that actually used LLM for this competition,