Enhancing Text Classification:
A Comprehensive Approach with the DKTC Dataset

Injoon Hwang *injoonh96@gmail.com*

Abstract

In this study, we explored the classification of Korean threatening conversations using the DKTC dataset. Our team, based at Aiffel, participated in the DL-Thon event, where we initially achieved an accuracy of 0.92 using transformer-encoder-based models. Post-event enhancements involved a combination of deep learning and traditional machine learning models, data augmentation, refined preprocessing, and ensemble techniques. By meticulously optimizing model weights in our ensemble, we achieved an improved accuracy of 0.9275. Our findings highlight the significance of diverse modeling approaches, comprehensive data preparation, and the synergistic effects of ensemble methods in text classification tasks.

1 Introduction

Our team, currently studying at *Aiffel*, a renowned as AI school(a.k.a boot camp) in South Korea, participated in an event titled **DL-Thon** organized as part of the deep learning curriculum. This event, reminiscent of a deep learning hackathon, encouraged participants to deep dive into text classification using the DKTC dataset. Preliminary attempts with Transformer encoder-based models(Bert) achieved an impressive accuracy of 0.92. Nonetheless, there's a belief that further refinement and augmentation of the dataset or ensembling with other models might lead to overcome outcomes.

1.1 DKTC (Dataset of Korean Threatening Conversations)

The DKTC dataset was curated by TUNiB for the 4th AI Grand Challenge in 2021 for speech recognition. Participants were tasked with classifying conversational audio into threatening subclasses or generic conversations, but no formal training data was provided. To meet this challenge, TUNiB crowd-sourced their data. After securing the 3rd position and receiving the IITP Director's Prize, TUNiB released the dataset for public non-commercial use.

The dataset, in its entirety, is bifurcated into:

- **Training Data:** Incorporating roughly 1,000 conversations for each of the four labels: 'Threats', 'Extortion', 'Workplace Harassment', and 'Other Harassments'.
- **Testing Data:** Encompassing 100 dialogues for the five labels: 'Threats', 'Extortion', 'Workplace Harassment', 'Other Harassments', and 'General Conversations'. It's noteworthy that for the DL-Thon, instances from the 'General Conversations' category in the testing data were deliberately excluded.

2 Methods

In this section, we will systematically compare the methodologies employed during the DL-Thon with those adopted post-event. Our intention is to clearly highlight the evolution of our approach and the subsequent refinements made.

2.1 Data Processing

2.1.1 During DL-Thon

One of the critical considerations when working with BERT-based models is the maximum input token length, typically capped at 512. Given the DKTC dataset's nature, we were uncertain about the optimal token length for the best results. To gain empirical insights, our team decided to conduct parallel experiments using the ko-ELECTRA model at various input lengths.

We began experimenting with the standard maximum length of 512. However, to explore performance at reduced token lengths, we sequentially tested lengths of 256, 128, 320, and 350. Surprisingly, a length of 256 outperformed the default 512. This finding prompted further exploration around this range, leading to tests at lengths of 320 and 350.

Based on these experiments, we determined that an input length of 350 tokens was most effective for our task. Consequently, we adopted this length for subsequent experiments with other models.

In addition to token length considerations, we aimed to fit as much meaningful information as possible within this constraint. We sought a ChatGPT4 model's assistance to recommend a aggressive list of Korean stop-words for removal. By incorporating this refined list, we ensured that the most relevant content was retained within the limited token length, potentially enhancing the model's performance.

2.1.2 Post DL-Thon

Update Stop-words dictionary In our initial approach to removing stopwords, we inadvertently eliminated numerous words with negative connotations. This unintentional removal seemed to adversely affect the predictive capabilities of our models. To address this issue, we revisited and revised our stopwords list, ensuring that crucial words with potential predictive power were not excluded.

Input Token Length Optimization While our initial approach during DL-Thon was guided by intuition and empirical testing, our subsequent experiments were more systematic. Instead of arbitrarily selecting token lengths, we adopted a more exhaustive approach. Starting from a length of 128, we incremented the token length by steps of 16, iterating up to the 512 limit. For each iteration, we trained the model and evaluated its performance. The highest-performing model across these iterations was saved and selected for further use.

Data Augmentation To further enhance our model's robustness and generalization, we explored various data augmentation techniques.

1. **Large Language Model Augmentation:**
   - *LLaMa2:* Utilized this model to generate augmented data, leveraging its capability to produce semantically coherent and contextually relevant sentences.
   - *KoAlpaca:* Used KoAlpaca for data augmentation, benefiting from its proficiency in understanding and generating Korean text.
2. **Traditional Text Augmentation:**
   - *KoEDA (Korean Easy Data Augmentation):*
     - *SR (Synonym Replacement):* This method replaces words in the original text with their synonyms, retaining the original context but introducing lexical variations.
     - *RI (Random Insertion):* Inserts random words at random positions in the text, introducing noise and ensuring the model's resilience to such perturbations.
3. **Word Vector Augmentation:** Using the Kyubyong/wordvectors, we identified the most similar words for lexical substitution. By replacing 25% of the words in the original data with their closest counterparts, we aimed to introduce variance while preserving the overall context and meaning.

2.2 Models and Ensemble

2.2.1 During DL-Thon

During our DL-Thon, while models like RNN, LSTM, and Seq2Seq were plausible choices for text classification, we anticipated that transformer-based models would outperform them in classification tasks. Our initial choice was BERT, primarily due to its bidirectional mechanism, which reads and classifies sentences, potentially enhancing prediction accuracy. We began with ko-BERT.

ko-BERT:

- **Origin:** BERT is a model proposed by Google.
- **Methodology:** Employs a transfer learning approach where a language model is first trained on a large amount of unlabeled data and then fine-tuned using a smaller amount of labeled data for specific tasks.
- **Pros:** Offers pre-trained word embeddings, which allows tasks to be executed with relatively fewer resources.
- **Details:** Released by SKT, this model was trained on about 50 million sentences sourced from Wikipedia, news articles, and more. It utilizes data-based tokenization through the SentencePiece tokenizer to accommodate the irregularities of the Korean language. It boasts a vocab size of 8,002 and a model parameter size of 92M.
- **Performance on our Task:** Regrettably, it underperformed, achieving an accuracy of just around 0.6.

Due to ko-BERT's unsatisfactory performance, we shifted our focus to ko-ELECTRA.

ko-ELECTRA:

- **Methodology:** During its training phase, a generator produces tokens, and a discriminator then decides whether each token is "real" or "fake".
- **Pros:** All input tokens contribute to learning, resulting in enhanced performance in contrast to models like BERT.
- **Details:** Trained on 34GB of Korean text, it employs Wordpiece and is compatible with the Transformers library installation.
- **Performance on our Task:** After conducting over 50 experiments (preserving the best model for later ensemble methods), we observed a substantial uptick in accuracy, touching 0.9025. This marked improvement from BERT's 0.6 is credited to the fact that, unlike BERT which uses some tokens solely for pre-training, ELECTRA engages all tokens for both pre-training and fine-tuning.

During the competition, the top-performing single model was klue/bert-base.

klue/bert-base:

- **Details:** This model served as a baseline in the KLUE benchmark. It underwent training on 63GB of data extracted from a variety of sources, including the universal corpus, CC-100-Kor, Namuwiki, news, petitions, etc. The model employs a Morpheme-based Subword Tokenizer.

Lastly, our team dabbled with the funnel-transformer(kor) model.

funnel-transformer(kor):

- **Misconception:** We were initially under the impression that this model encompassed both the encoder and decoder sections of the transformer. However, it dawned on us later that it only contained the encoder component.
- **Details:** The FUNNEL-TRANSFORMER model is rooted in the ELECTRA model, integrating both the generator and discriminator. It was trained on an expansive dataset, including 100 million e-commerce reviews, 20 million blog-styled website entries, the universal corpus, Wikipedia, and Namuwiki, summing up to 99GB. It uses a vocabulary spanning 42,000 words.

After evaluating the individual performance of the single models, we also delved into ensemble methods to potentially enhance accuracy. Specifically, we employed both soft voting and weighted soft voting techniques. By aggregating the predictions of different models, we aimed to capitalize on their complementary strengths, thereby achieving a more robust and accurate classification. And ensemble methods showed improvements in accuracy.

2.3 Post DL-Thon

2.3.1 Machine Learning

In addition to deep learning models, we also explored traditional machine learning models to benchmark and to improve ensemble model.

Preprocessing: We utilized an updated version of stopwords removal(prevent to remove negative stopwords). After this preprocessing step, to convert the text data into a format suitable for machine learning models, we applied the TF-IDF (Term Frequency-Inverse Document Frequency) technique. This transformation allowed us to capture the importance of words relative to the entire corpus, making it a suitable representation for our classification task.

Modeling and Hyperparameter Tuning: To systematically search for the optimal hyperparameters for each model, we employed a grid search approach. We selected the model with the highest valid F1 score. If this model demonstrated an accuracy of 0.83 or higher when compared to the answer, it was saved. Following this criterion, three models – SVM, Random Forest, and Logistic Regression – were saved for further analysis.

- **Random Forest:**
  - Model: `RandomForestClassifier()`
  - Parameters:
    - n_estimators: [10, 100]
    - max_depth: [None, 10]
    - min_samples_split: [2, 5]
- **Logistic Regression:**
  - Model: `LogisticRegression(solver='liblinear', max_iter=1500)`
  - Parameters:
    - C: [10, 100]
    - penalty: ['l1', 'l2']
- **Gradient Boosting (Not meeting the F1-score threshold):**
  - Model: `GradientBoostingClassifier()`
  - Parameters:
    - n_estimators: [50, 100]

        * learning_rate: [0.01, 0.05]
        * max_depth: [3, 5]

- **SVM:**
  - Model: `SVC(probability=True)`
  - Parameters:
    * C: [1, 100]
    * kernel: ['linear', 'rbf']
    * gamma: ['scale', 'auto']
- **KNN (Not meeting the F1-score threshold):**
  - Model: `KNeighborsClassifier()`
  - Parameters:
    * n_neighbors: [3, 5]
    * weights: ['uniform', 'distance']
    * metric: ['euclidean', 'manhattan']

By comparing these machine learning models against our deep learning models, and employing TF-IDF for feature extraction, we aimed to understand the strengths and weaknesses of each approach and potentially harness them in a complementary manner.

3 Results & Discussion

3.1 Results During DL-Thon

During the DL-Thon, our models exhibited the following performances when considered individually:

- ko-ELECTRA: 0.9025
- KLUE: 0.9125
- Funnel-Transformer: 0.9075

By employing a soft voting ensemble technique that combined these three models, we managed to elevate our accuracy, achieving a score of 0.92.

3.2 Post DL-Thon Enhancements

Post DL-Thon, we were resolute in our attempts to further refine our model's accuracy. By incorporating strategies like data augmentation, revamped stop-words removal, and other preprocessing enhancements, our single-model performances observed improvements:

- KLUE: 0.92
- Funnel-Transformer: 0.9225

Despite our efforts in merging these two models using ensemble techniques, we found ourselves plateauing at the 0.9225 accuracy mark.

Recognizing the potential benefits of diversifying our approach, we then incorporated traditional machine learning models into our arsenal. Their performances were as follows:

- SVM: 0.8725

- Random Forest: 0.855
- Logistic Regression: 0.8625

Although our initial inclination was to prioritize deep learning models during the ensemble due to their perceived superiority, such attempts did not result in surpassing the 0.9225 accuracy threshold.

In our pursuit to optimize the ensemble further, we embarked on a meticulous search for the most effective weight combinations. Even though this approach might seem to be cheating or overfitting, our primary intent was to derive meaningful insights into model ensembling. Our optimal configuration was:

- KLUE: 0.05
- Funnel-Transformer: 0.25
- Logistic Regression: 0.05
- Random Forest: 0.5
- SVM: 0.15

Implementing these weights, our ensemble model achieved a commendable accuracy of 0.9275.

4 Conclusion

In our study, we looked closely at different deep learning and traditional machine learning models to classify Korean threatening conversations using the DKTC dataset.

1. **Transformer Preference:** Transformer-based models, especially those related to BERT, performed really well. Models like KLUE, ko-ELECTRA, and Funnel-Transformer were especially good for this task.
2. **Data Augmentation & Preprocessing:** We spent a lot of time improving our data. This included adding more data, removing stop-words, and using TF-IDF to turn text into numbers. These steps made sure our models understood the important parts of the conversations.
3. **Ensemble Models:** We found that using multiple models together gave better results than any single model on its own. We mixed their predictions in a way that gave the best results. The best combination used both deep learning and traditional machine learning models.
4. **Hyperparameter Exploration:** We tried many settings to find the best ones for our models. This included finding the best way to combine different models' predictions.
5. **Diversity Approaches:** We started with the models from the DL-Thon and then tried other models and techniques. Adding traditional machine learning models to our deep learning ones helped improve our results.

In short, our work shows the importance of trying different things, using different models together, and preparing data well for text classification. As we continue to push the boundaries of AI, these methods will be key to solving complex language problems.

5 Reference

References

[1] Cho, Soyoung and Ha, Sangchun and Ryu, Myeonghyeon and Keum, Bitna and Park, Kyubyong. *DKTC, Dataset of Korean Threatening Conversations.* `https://github.com/tunib-ai/DKTC`, 2022.

[2] facebookresearch. *Inference code for LLaMA models.* `https://github.com/facebookresearch/llama`, 2022.

[3] Beomi. *KoAlpaca: 한국어 명령어를 이해하는 오픈소스 언어모델.* `https://github.com/Beomi/KoAlpaca`, 2023.

[4] Beomi. *KcBERT: Korean comments BERT.* `https://github.com/Beomi/KcBERT`, 2023.

[5] toriving. *KoEDA: Korean Easy Data Augmentation.* `https://github.com/toriving/KoEDA`, 2023.

[6] Kyubyong. *Pre-trained word vectors of 30+ languages.* `https://github.com/Kyubyong/wordvectors`, 2023.

[7] SKTBrain. *Korean BERT pre-trained cased (KoBERT).* `https://github.com/SKTBrain/KoBERT`, 2023.

[8] Park, Jangwon. *KoELECTRA: Pretrained ELECTRA Model for Korean.* `https://github.com/monologg/KoELECTRA`, 2020.

[9] Sungjoon Park et al. *KLUE: Korean Language Understanding Evaluation.* `https://huggingface.co/klue/bert-base`, 2021.

[10] Kim, Kiyoung. *Pretrained Language Models for Korean.* `https://github.com/kiyoungkim1/LMkor`, 2020.