KIPS Trans. Softw. and Data Eng.
Vol.11, No.10 pp.437~446
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2022.11.10.437

Exploiting Korean Language Model to Improve Korean Voice Phishing Detection

Milandu Keith Moussavou Boussougou† · Dong-Joo Park††

ABSTRACT

Text classification task from Natural Language Processing (NLP) combined with state-of-the-art (SOTA) Machine Learning (ML) and Deep Learning (DL) algorithms as the core engine is widely used to detect and classify voice phishing call transcripts. While numerous studies on the classification of voice phishing call transcripts are being conducted and demonstrated good performances, with the increase of non-face-to-face financial transactions, there is still the need for improvement using the latest NLP technologies. This paper conducts a benchmarking of Korean voice phishing detection performances of the pre-trained Korean language model KoBERT, against multiple other SOTA algorithms based on the classification of related transcripts from the labeled Korean voice phishing dataset called KorCCVi. The results of the experiments reveal that the classification accuracy on a test set of the KoBERT model outperforms the performances of all other models with an accuracy score of 99.60%.

Keywords : KoBERT, NLP, Text Classification, Machine Learning, Deep Learning

한국어 언어 모델을 활용한 보이스피싱 탐지 기능 개선

Milandu Keith Moussavou Boussougou† · 박 동 주††

요 약

보이스피싱 통화 내용을 탐지하고 분류하는데 핵심 엔진으로 최신 머신러닝(ML) 및 딥러닝(DL) 알고리즘과 결합된 자연어 처리(NLP)의 텍스트 분류 작업이 널리 사용된다. 비대면 금융거래의 증가와 더불어 보이스피싱 통화 내용 분류에 대한 많은 연구가 진행되고 양호한 성과를 보이고 있지만, 최신 NLP 기술을 활용한 성능 개선의 필요성이 여전히 존재한다. 본 논문은 KorCCVi라는 레이블이 지정된 한국 보이스 피싱 데이터의 텍스트 분류를 기반으로 여러 다른 최신 알고리즘과 비교하여 사전 훈련된 한국어 모델 KoBERT의 한국 보이스 피싱 탐지 성능을 벤치마킹한다. 실험 결과에 따르면 KoBERT 모델의 테스트 집합에서 분류 정확도가 99.60%로 다른 모든 모델의 성능을 능가한다.

키워드 : KoBERT, 자연어 처리, 텍스트 분류, 머신러닝, 딥러닝

1. Introduction

Artificial intelligence (AI) is an extensive branch of computer science aiming to build machines capable of simulating human intelligence. AI includes several fields such as Natural Language Processing (NLP) concerned with enabling machines to understand human languages in their spoken and written forms. There is a variety of tasks included in NLP, namely text classification, which is a fundamental task. By using these NLP tasks, machines are thus capable of reading, analyzing, understanding, and cognitively deriving human languages. Some common real-life applications and services involving text classification are spam email filtering, sentiment analysis [1, 2], text intent detection, or topic labeling. When performing text classification tasks, a combination of NLP techniques with machine learning (ML) or deep learning (DL) techniques are widely used to assign the inputted textual data (e.g., documents, emails, comments, surveys, etc.) to predefined categories. It is an automatic classification made using the ML or DL model that was trained on a labeled training dataset. Fig. 1 presents the common workflow used to perform text classification tasks.

※ 이 논문은 2021년 한국정보처리학회 ACK 2021의 우수논문으로 "머신러닝 기법을 이용한 한국어 보이스피싱 텍스트 분류 성능 분석"의 제목으로 발표된 논문을 확장한 것임.

† 준 회 원 : 숭실대학교 컴퓨터학과 박사과정
†† 정 회 원 : 숭실대학교 컴퓨터학부 교수

Manuscript Received : December 29, 2021
First Revision : March 15, 2022
Second Revision : May 6, 2022
Accepted : May 12, 2022

* Corresponding Author : Dong-Joo Park(djpark@ssu.ac.kr)

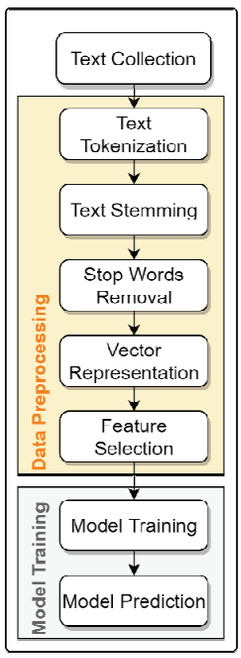

Fig. 1. Text Classification Workflow

Furthermore, language model (LM) pre-training has also proven its effectiveness in improving various NLP tasks, especially with the flexibility of fine-tuning the resulting pre-trained LM for the targeted downstream NLP task. The pre-trained LM BERT [3] has achieved the state-of-the-art (SOTA) performance on several NLP tasks and provides the flexibility to fine-tune its parameters. However, because the BERT model was mostly pre-trained on an English dataset, its performance in other languages such as Korean was revealed poor. Consequently, the Korean BERT (KoBERT) pre-trained LM [4] emerges to overcome the limitations of the BERT model for Korean language representation.

The usage of text classification has already proved its efficiency in solving real-life problems, and currently, several researchers are leveraging its power with ML and DL algorithms to propose solutions to mitigate the worldwide issue of voice phishing. Indeed, in South Korea in particular, voice phishing is still a topical issue that needs to be tackled. In addition, with the worldwide pandemic of COVID-19, non-face-to-face financial transactions have increased, and therefore, increased at the same time potential voice phishing victims. For these reasons, despite the progress made by recent publications, new approaches to classifying Korean voice phishing texts are necessary to come out with new detection techniques for voice phishing calls.

This paper is an extension of our prior study on the performance analysis of ML techniques in classifying Korean voice phishing text [5], and no novel technique is introduced in this work's contribution. Nevertheless, one major contribution of this paper is the analysis of the KoBERT model's performance in the classification of Korean voice phishing text. Considering the SOTA ML and DL algorithms [6] and pre-trained LMs, multiple existing and well-known ML and DL algorithms are selected for this new benchmarking of their classification performances on Korean Call Content Vishing (KorCCVi) [7], which we claim to be the first known labeled dataset of Korean voice phishing call transcript. The algorithms benchmarked include CatBoost [8], Gradient XGBoost [9, 10], LGBM [11, 12] for the ML approach, and Recurrent Neural Networks (RNNs) [13, 14], Bidirectional Long Short-Term Memory units (BiLSTM) [15, 16], Gated Recurrent Units (GRUs) [17], and KoBERT [4] for the DL approach. Code is available at: https://github.com/selfcontrol7/Korean_Voice_Phishing_Detection/tree/main/KoBERT.

2. Background and Related work

2.1 Deep Learning baseline models

RNNs are types of artificial neural networks using sequential data (time-series, text, audio, video, etc) to build deep learning models capable of solving various temporal problems which can be related to NLP such as speech recognition, and machine translation, or handwriting recognition. Compared to other deep learning neural networks types, RNNs are particular as they are endowed with an internal memory, allowing the network to keep information from the previous inputs node which is passed to the next input and highly influences the latter input's result. By way of explanation, in an RNNs architecture, the output of the network depends on the prior element within the sequence learned.

LSTM is a popular variant of the RNNs architecture, which emerged to solve RNNs' vanishing gradients problem. The approach behind the LSTM variant is that it introduced cells state in the hidden layers of the network. LSTM has three gates, an input gate, an output gate, and a forget gate, which manages the flow of information going through each layer to predict an accurate output. BiLSTM is a neural network composed of two LSTM units in parallel that operate

in two directions. One unit receives the information and propagates in the forward direction (left-to-right), and the other unit propagates in the backward direction (right-to-left) to capture the long-term dependencies in the left and right context. Therefore, it provides better performances for text classification tasks and is more robust than using a unary LSTM unit.

The GRUs and the LSTM variants are similar, and the GRUs variant appears additionally as a solution to RNNs' short-term memory problem. Compared to the LSTM variant, GRUs are a gating mechanism in RNNs. They use hidden states in the hidden layers and have two gates instead of three, a reset gate and an update gate. The two gates present in the GRUs variant play the exact same function as in the LSTMs variant, controlling the information to retain to obtain an accurate output.

2.2 Related work

1) BERT pre-trained language model

Bidirectional Encoder Representations (BERT) from Transformers is a pre-trained LM developed by Google and trained on the huge corpus, including Book Corpus and Wikipedia text data [3]. It is the first language representation model, allowing fine-tuning that achieved SOTA performance on a large suite of token-level and sentence-level tasks. The well-known performance of the BERT model resides in two specific steps, which are pre-training and fine-tuning. In the pre-training step, Masked Language Model (MLM) is used to randomly mask some of the tokens in the sequence inputted, and thereafter, the Next Sentence Prediction (NSP) task is used to understand the sentences relationships and predict the masked token in the sequence. The fine-tuning step gives the flexibility to fine-tune the model parameters for the targeted tasks. It, therefore, reduces the need for network architecture modifications for the targeted NLP task.

However, although BERT demonstrated an impressive multilingual performance on NLP tasks, it has limitations with tasks related to specific fields such as Bioscience or finance. In addition, the BERT model has also shown low performance in languages such as Korean leading researchers to conduct study aiming to propose different methods to overcome BERT's limitations.

2) KoBERT pre-trained language model

T-Brain from SK AI Center [4] developed and released the Korean BERT (KoBERT) pre-trained LM to overcome the low performance of the BERT model in the Korean language. Similar to the BERT model, KoBERT is a transformer-based LM pre-trained on the Korean Wiki text. It has, therefore, all the advantages of the BERT model and it is specialized for the NLP tasks in the Korean language.

Recently, other new Korean pre-trained LMs are rising, and few studies [18-21] have conducted surveys or performance comparisons of these Korean LMs on various downstream NLP tasks. As the KoBERT is currently the pre-trained LM mostly used for Korean NLP tasks because of its acceptable performance, it therefore motivated us to evaluate its performance on the KorCCiv dataset.

3) Sentiment Analysis of Korean text using KoBERT

Since the release of the KoBERT pre-trained LM, in recent years, several studies have been conducted to apply it to solve Korean text classification problems. In [22], the authors used a fine-tuned KoBERT model to understand the emotion behind the Korean comments on YouTube videos to evaluate their negativity or positivity and classify them. In this study, the KoBERT model fine-tuned on the NSMC dataset demonstrate a slightly higher classification accuracy score of 81% on the real YouTube comment dataset in comparison to accuracy on the learning dataset used. In [23], classifying Korean daily conversation topics, the authors have shown a high accuracy score of 85% for the KoBERT model outperforming the Random Forest and XGBoost model's accuracy with 82% and 83%, respectively. The F_1 scores of the KoBERT model were also better than the other algorithms on 9 topics out of 15.

3. Methodology

To conduct this new benchmark, we followed the same methodology from our prior work [5], which is presented in Fig. 2. This paper, therefore, benchmarks the Korean voice phishing detection performances of the pre-trained LM KoBERT against other SOTA baseline algorithms.

In step number one, the Korean voice phishing da-

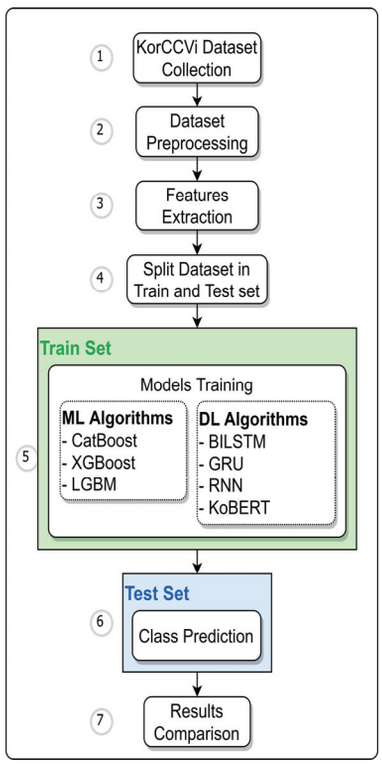

Fig. 2. Workbench Methodology

taset, KorCCVi, was collected from our prior existing work. This dataset was built using the transcripts of genuine voice phishing call conversations and transcripts of general Korean conversations on various daily topics.

The next step, step number two, consists of preprocessing all the textual data collected from the dataset to only remain with the relevant learning features that will enable the model to understand the difference between a voice phishing text and a non-voice phishing text. The raw data were preprocessed through different cleaning steps, such as deleting from the transcripts, unwanted characters, all the special characters and punctuation, the telephone numbers, and the Korean stop words. To delete the Korean stop word from the transcripts, each inputted sentence was first tokenized using the subword tokenization method, then with a morphological analysis of the created tokens, the stop words were detached from existing tokens to create separate tokens that can be easily removed. For example, the Korean sentence *"고객이 자신의 은행 계좌 정보를 알려주었습니다." (The customer gave me his bank account information.)* will become ['고', '객', '이', '자신', '의', '은행', '계좌', '정보', '를', '알려', '주', '었', '습니다', '.'] after tokenization and the stop word being detached. Indeed, the Korean language is an agglutinative language in which, most of the time, nouns and verbs in sentences have a postposition attached to them. Therefore, to separate the agglutinating words, which are the stop words, we used the morphological analyzer MeCab. In the task of text classification, data preprocessing is one of the most crucial steps that need particular attention, as it has a huge impact on the model performance [24]. The reason why data preprocessing is important is that this step also aims to reduce the dimensionality problem when creating the feature matrix during the feature engineering step. Table 1 indicates a summary of the content removed from all the dataset transcripts during this preprocessing part.

Table 1. Summary of Content Removed from the Dataset During Data Preprocessing

Unwanted Characters	Punctuation	Korean stop Words
o/, b/, n/, \n	!\"#$%&(")*,+-./::〈=〉?@¡ØØ¤ ý[\]^_`{¡®|'}~\n¡¦¡»,	을, 를, 이, 가, —, 은, 는, 의

Step number three is the features extraction or features engineering part where the preprocessed and cleansed data are used to create and select the features to train our ML and DL models in the experimental part. In this paper, three different techniques are used for feature engineering: Bag-of-Words, word embedding, and LM. For the Bag-of-Words technique, using the Tf-Idf vectorizer method with a maximum vocabulary limit of 2000 words as a parameter, a features matrix of the term frequency-inverse document frequency (TF-IDF) score of the dataset's words is created. This matrix of TF-IDF features is used as input features for training our ML models. The TF-IDF method evaluates how important a word is to a given document in a corpus, which is a collection of documents. The TF-IDF score is obtained by multiplying two metrics: the number of times a term appears in a given document divided by the total number of terms in that document (TF), and the inverse document frequency of the term across the corpus, which means how common or rare a term is in the entire corpus. Equation (1) defines the TF-IDF score for a word in a document.

$$TFIDF(t,d) = TF(t,d) \times \log \frac{N}{DF(t)} \quad (1)$$

Table 2. Summary of the Features Engineering Process

Algorithm Type	Input Features
Machine Learning (ML)	Features matrix of TF-IDF values (Bag-of-Words)
Deep Learning (DL)	Sparse embedding matrix (word embedding)
KoBERT (DL)	Dense embedding matrix (word embedding)

In equation (1), the terminology is as follows: t represents the term (word in a document), d the document (set of words), TF(t) the term frequency of t in d, N the number of documents in the corpus and DF(t) the number of documents in the corpus containing the term t.

Concerning the word embedding technique, we use the FastText model [25] to produce feature vectors of the dataset's words that can be represented in a vector space where words of the same context are represented close to one another. The word vectors are then transformed into a sparse embedding matrix, which is used as weights in the neural network's first embedding layer when training the DL models. In the feature engineering with the LM, following the Transformer modeling method, the raw texts from the dataset are transformed into sequences containing text tokens and special tokens, then for each token of the sequence, a mask and segment are generated. We also followed the maximum sequence length of 64, which is supported by this KoBERT model as mentioned in its documentation. The final features matrix is a dense embeddings matrix used in the embedding layer when training the pre-trained KoBERT model. Table 2 presents a summary of the feature engineering process with the input features used by each model.

As presented in Table 3, step number four concerns splitting the extracted features into two sets, training and a test set. Thereafter, using the training set, we trained different models with the selected ML and DL algorithms in step number five using. The TF-IDF scores of all the tokens generated during the preprocessing part are used to train the ML models, while embedding matrices representing the sentences in the training set are used to train the DL models.

The final trained models are used to predict the class of the inputted transcript from the test set in step number six.

Table 3. Sampling of the Dataset in Training and Test Set

Algorithm	Training Set	Test Set	Total
Machine Learning (ML)	852	366	1218
Deep Learning (DL)	852	366	1218
KoBERT (DL)	852	366	1218

In the benchmarking of the model's classification, we evaluated their performance in determining whether the inputted Korean conversation text is a voice phishing or non-voice phishing related text. In this paper, there are four different metrics used to evaluate the models' performance: accuracy, precision, recall, and F_1 score. The precision, the recall, and the F_1 score metrics are designated in Equations (2), (3), and (4), respectively. TP in Equations (2) and (3) is the number of true positives, FP in Equation (2) is the number of false positives, and FN in Equation (3) is the number of false negatives.

$$\mathrm{Precision} = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F_1 score = \frac{\mathrm{Precision} \times recall}{\mathrm{Precision} + recall} \tag{4}$$

Among the selected metrics, the benchmark in this paper focuses more on the accuracy and F_1 scores of the models as they provide a better evaluation of classification models. In addition, for further performance comparison, we also considered the training time of each model in the benchmarking. Looking at the metrics results of each model, we confronted the KoBERT model's performance with the other baseline models' performance and voted the most efficient model for our Korean voice phishing text classification task. This is the last step of the benchmark methodology used in this paper, step number seven.

4. Experiments

4.1 Korean voice phishing Dataset

In this paper, the dataset used is KorCCVi, which contains 1218 samples distributed as 609 voice phishing phone call conversation transcripts and 609 general conversation transcripts. KorCCVi has two differ-

Table 4. Extract of KorCCVi Dataset with English Translation

id	Transcript	Label
417	진통제 계십니까 다름 아니 얼마 저희 검찰 에서 김승대 주범 으로 금융 범죄 사기 식당 제일 걸로 현장 에서 지금 본인 명의 기업 통장 근데 으시 본인 으로 연락 드린 통장 본인 직접 으로 계산 셔서 서대 선물 총장 넘긴 건지 아니 본인 개인 정보 졸리 면서 당하 피해자 신지 부분 결과 려고 연락 드렸 습니다 무슨 말씀	1
	Do you have painkillers? The prosecution said Kim Seung-dae was the main culprit in the financial crime fraud restaurant. He paid for the bank account and handed it over to the president of Seodae Gift. I'm calling to see the result What are you talking about?	
784	지금 언니 시간 보내 도착 체크인 근데 근까 우리 시간 보다 한참 그서 그니까 몰랐 해서 그러면은 라도 그래야 아닌가 그래서 놓쳐 그리고 일본 체크인 빨리 몰랐 면서 비행기 놓쳐서 그냥 끝나	0
	Send your time now. Arrived, checked in. But it was way beyond our time. I didn't know. So I thought maybe I should. So I missed it. I missed the flight. I missed it.	
318	첨단 범죄 수사 박동석 느라고 다름 아니 본인 대신 명의 도용 사건 어서 에서 확인 부탁 연락 드렸	1
	I contacted Park Dong-seok, a high-tech criminal investigation, and asked him to confirm the theft of his name instead of him.	
1075	어떻 명절 다가오 그럴 조금 남편 짠하 자기 요즘 아무리 형제 다고 더라도 부모 돌아가 그렇게 모이 잖아 더더군다나 형제 아니 아니 아들 으니까 각자 서로 기름 처럼 드라고 명절 우리 신랑 조금 그런 느낌 대신 인제 처가 으로	0
	How could the holidays be coming? A little husband, a little sad. No matter how many brothers you are, your parents go back and get together like thatI'm not a brother. I'm a son, so each of us should hold it like oil. Instead, let's go to the wife's house	

ent classes which are voice phishing, represented as '1', and non-voice phishing represented as '0'. An extract of KorCCVi dataset is presented in Table 4 with English translation of each sample for understanding purpose only.

4.2 Training of the ML and DL Classifier

The experiments of this paper were conducted using Python programming language in the Google Colab Jupyter environment. The training of all the ML (CatBoost, Gradient XGBoost, and LGBM) and DL (RNN, BiLSTM, and GRU) models for the detection of voice phishing by classifying related phone call transcripts has been implemented mostly using default hyperparameters. In other words, there are only few parameters that have been fine-tuned with random values during the training of the model. No optimal hyperparameters were used.

The DL models were all implemented with Keras from TensorFlow, trained over ten epochs using 32 as batch size. To optimize the models' training, we use the Adam optimizer, added dropout layers to the network architecture, and applied the Softmax function before the output layer containing two nodes.

4.3 Training of KoBERT Classifier

The training of the fine-tuned KoBERT model was implemented using PyTorch API. From the documentation of the pre-trained LM KoBERT [4], the model is an encoder-decoder attention architecture with 12 self-attention hidden layers and an output layer number that can be customized. Since this study addresses a binary classification case, the output layers were set to two. One important standard parameter of the model is the maximum sequence (sentence) length that this pre-trained LM can handle. It is a parameter that was considered in the feature engineering part of this paper when creating the feature matrix. As an input feature for this model, the dense matrix of embeddings previously created based on the KorCCVi dataset is used to perform a transfer learning from the pre-trained KoBERT model and fine-tune the final model for our classification task. Concerning the hyperparameters used, a learning rate of 5e-5 was used during the training of the model over 10 epochs with 32 as batch size. Moreover, to regulate the model's training, a drop rate value of 0.5 was set on the final dropout layer. We used the AdamW optimizer, which implements the Adam algorithm with weight decay and performs very well for BERT-based models in particular and with NLP data.

Fig. 3 demonstrates the process of fine-tuning the pre-trained KoBERT LM using a sample of the KorCCVi dataset. In this illustration, the [CLS] token stands for classification and it is added at the beginning of each sentence to indicate that the training task here is sentence-level classification. The token [SEP] instead represents a separation between two in-

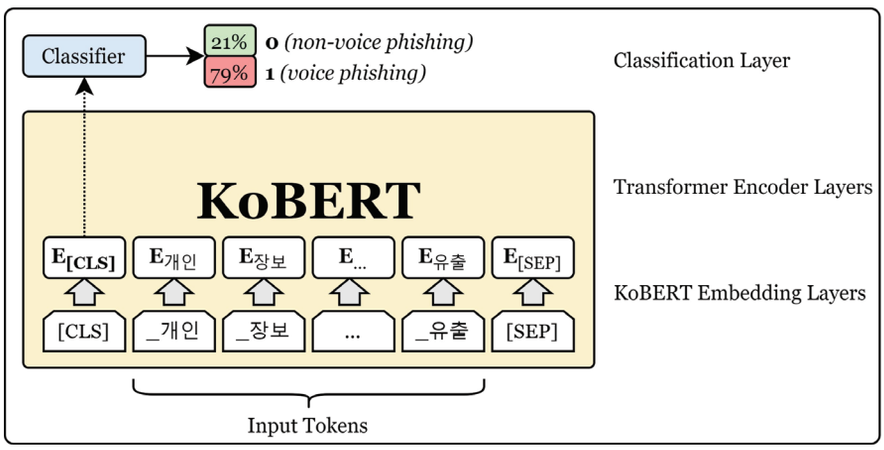

Fig. 3. Fine-tuning KoBERT Classification Model for Voice Phishing Detection

Algorithm for Fine-tuning KoBERT model

```
Initialize PyTorch pre-trained KoBERT-based model and vocabulary
Initialize End-to-end SentencePiece tokenization for BERT models

def BERTClassifier(PyTorch nn module):
  …
  Set the number of output classes to 2
  …
  return attention mask
  …
  return classifier output

#Transform the training and test set sequences
for data in dataset do
  Tokenize the input sequences
  Insert [CLS], [SEP] tokens as necessary
  Encode token into embedding ids in NumPy int32
  Generate valid length using maximum sequence length
  Pad sentences to maximum length
  Store transformed sequence in array
  Change data's label to NumPy int32
  Store transformed sequences in tuple of arrays
end for

Create torch-type training and test set from transformed datasets
Initialize BERT classifier with KoBERT base and dropout rate arguments
Prepare model's optimizer and schedule (linear warmup and decay)
Initialize AdamW optimizer with optimizer grouped parameters
Initialize cross-entropy loss function

#Train the KoBERT model
for each training epoch do
  Train KoBERT model with training set
  Validate trained KoBERT model
end for

#Test the trained KoBERT model
Load trained KoBERT model
Predict new samples classes from test set
```

Fig. 4. Pseudocode of the Algorithm for Fine-tuning the KoBERT Model

put sentences. In addition, for a deeper understanding of the different inner steps involved in this fine-tuning process, Fig. 4 presents the pseudocode of the KoBERT classifier's training algorithm.

4.4 Experimental Results

Table 5 presents the results of the Korean voice phishing text classification experiments. It shows the scores of the different performance evaluation metrics selected for the KoBERT model, the ML and DL models. During the models' performance comparison, we focused on the accuracy and F_1 scores.

The accuracy score of the KoBERT model was 99.60%, while the F_1 score was 99.57%. From this performance score, we can seet hat KoBERT outperforms all the other ML and DL models. The second closest accuracy score is from the LGBM model, which achieved an accuracy score of 99.45%, for a F_1 score of 99.43%. These performances are slightly lower than KoBERT's score. The modeling time of the KoBERT model is higher compared to LGBM one, which appeared to be the fastest model as shown in Fig. 5.

Table 5. Benchmarking Results of the ML and DL models' performances

Model	Accuracy	F_1	Precision	Recall
XGBoost	95.35	95.04	98.19	92.09
LGBM	99.45	99.43	99.43	99.43
CatBoost	98.08	97.99	99.41	96.61
BiLSTM	96.99	96.99	97.01	96.99
GRU	95.36	95.35	95.52	95.36
RNN	87.43	87.39	88.00	87.43
KoBERT	**99.60**	**99.57**	**1**	**99.14**

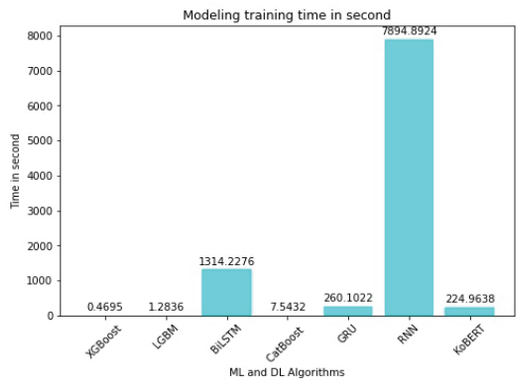

Fig. 5. Modeling Time Comparison

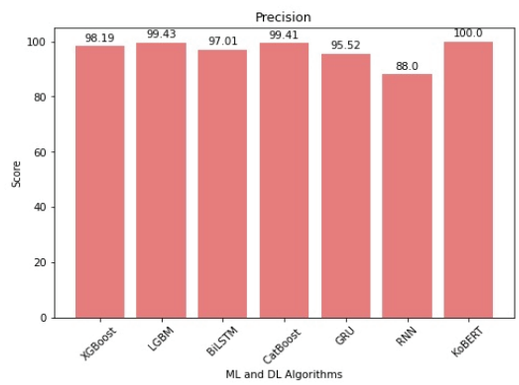

Fig. 6. Precision Score Comparison

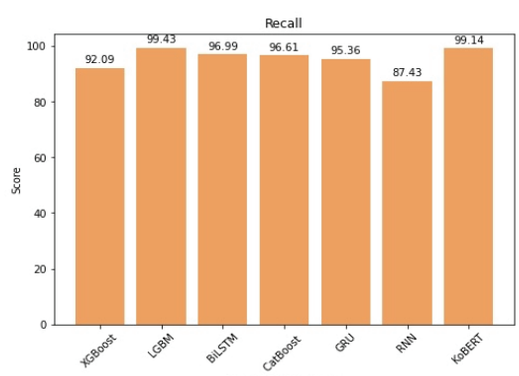

Fig. 7. Recall Score Comparison

However, because the KoBERT model uses neural networks through the Transformer architecture, its modeling time was faster than all the other DL models, which is a considerable advantage knowing the benefits that a model trained with neural network algorithm provides. We can see that LGBM and XGBoost models have similar modeling times, while the LGBM model outperforms the XGBoost model on all the other metrics. Fig. 6 and Fig. 7 indicate the comparison of the scores for precision and recall metrics, respectively.

Moreover the analysis of the models' performances, a further analytical comparison of the models' complexity can be performed through the parameters of the model presented in Table 6, which are to bring more insight into the performances compared above. Indeed, looking at the complexity of the ML models, default values were used for the learning rate during the training process. As a result, we can observe that the lower the learning rate, the slower is the model to train on the training set since the learning rate has a significant impact on each tree of these ML models, Looking closer at the CatBoost model which has a maximum depth of 6 trees, it appeared to be the slowest ML models despite the random value used to fine-tune the subsample parameter to accelerate the training process and avoid overfitting. An important factor to consider is that the more a ML model is deep in trees, the more its complexity and its memory consumption increase. On the contrary, the XGBoost model has a faster training time with adequate parameters, yet its performance does not outperform LGBM one or CatBoost one. Surprisingly, although the maximum depth of the tree is unlimited on the LGBM model and has a high risk to increase in complexity, its performance among the ML and DL model is ranked among the best.

Table 6. Comparison of the ML and DL models' Complexity

Model		n estimators	Subsample	Max depth	Learning rate	Num layers	Trainable params	Batch size	Num epochs	Modeling time in second
ML	XGBoost	2000	0.67	3	1e-1	-	-	-	-	0.46
	LGBM	2000	0.67	-1†	1e-1	-	-	-	-	1.28
	CatBoost	2000	0.67	6	2e-2	-	-	-	-	7.54
DL	BiLSTM	-	-	-		10	801,538	32	10	1314.22
	GRU	-	-	-		12	234,702	32	10	260.10
	RNN	-	-	-		12	111,586	32	10	7894.89
	KoBERT	-	-	64	5e-5	12	-	32	10	224.96

† <=0 means no limit

The complexity of the DL model is already well-known, as it implements a neural network that requires a huge quantity of training data to train the model. The four DL models in this benchmark have fairly the same number of layers in their architectures and this results in a deep architecture with thousands of parameters to train. The training time of the DL model is relative to the depth of the model, and we assume that the overall computation complexity is high because of the huge consumption of memory and GPU power required. However, though having a high complexity, the KoBERT model stands out from the other DL models in terms of training speed and accuracy performance.

5. Conclusion and Future work

In this work, we conduct a benchmark of the popular Korean language model, KoBERT, against several other ML and DL algorithms in the binary classification task of Korean voice phishing text to detect voice phishing calls. The experimental results reveal that KoBERT outperformed all the other models with an accuracy score of 99.60%. From this performance comparison, KoBERT is the most efficient model in the classification of voice phishing text from the KorCCVi dataset. It has also been observed that the KoBERT model and LGBM model accuracies are slightly different, with 99.60% and 99.45%, respectively. However, further comparing the models' complexity, despite the risk of complexity increase caused by its unlimited maximum depth parameter, the advantage of the LGBM model was its fast modeling time of 1.28 seconds making it the fastest model, and its highest F_1 score of 99%. Moreover, regardless of its high complexity and the small size of the dataset used to train the KoBERT model with mostly its default hyperparameters, it is the most accurate model in this study.

Future research aims to use a larger dataset and fine-tune all the hyperparameters of all the ML and DL algorithms to improve the performances. In addition, with the different emerging Korean language models, another future contribution can be a performance comparison of these language models in detecting Korean voice phishing and deploying the most efficient model in a mobile application to evaluate its efficiency in the real-world setting.

References

[1] K. Kowsari, D. E. Brown, M. Heidarysafa, K. J. Meimandi, M. S. Gerber, and L. E. Barnes, "HDLTex: Hierarchical deep learning for text classification," *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp.364-371, 2017. doi: 10.1109/ICMLA.2017.0-134.

[2] C. C. Aggarwal and C. X. Zhai, "A survey of text classification algorithms," *Mining Text Data*, pp.163-222, Aug. 2012. doi: 10.1007/978-1-4614-3223-4_6.

[3] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp.4171-4186, 2018, Accessed: Dec. 27, 2021. [Online]. Available: https://arxiv.org/abs/1810.04805v2.

[4] SKTBrain/KoBERT: Korean BERT pre-trained cased (KoBERT) [Internet], https://github.com/SKTBrain/KoBERT (accessed Dec. 27, 2021).

[5] M. K. M. Boussougou, S. Jin, D. Chang, and D.-J. Park, "Korean voice phishing text classification performance analysis using machine learning techniques," *Proceedings of the Korea Information Processing Society Conference*, pp. 297-299, 2021. doi: 10.3745/PKIPS.Y2021M11A.297.

[6] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, L. E. Barnes, and D. E. Brown, "Text classification algorithms: A survey," *Information (Switzerland)*, Vol.10, No.4, Apr. 2019, doi: 10.3390/info10040150.

[7] M. K. M. Boussougou and D.-J. Park, "A real-time efficient detection technique of voice phishing with AI," in *Proceedings of Korea Software Congress 2021*, pp.768-770, 2021, Accessed: Oct. 13, 2021. [Online]. Available: https://www.dbpia.co.kr/Journal/articleDetail?nodeId=NODE10583070.

[8] Catboost, "CatBoost - open-source gradient boosting library," 2021. [Internet], https://catboost.ai/ (accessed Mar. 11, 2022).

[9] XBGoost, "XGBoost Documentation - Introduction to Boosted Trees," 2020. [Internet], https://xgboost.readthedocs.io/en/latest/tutorials/model.html (accessed Mar. 11, 2022).

[10] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.785794, 2016. doi: 10.1145/2939672.2939785.

[11] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, pp.3147-3155, 2017, Accessed: Mar. 11, 2022. [Online]. Available: https://github.com/Microsoft/LightGBM.

[12] "Welcome to LightGBM's documentation! — LightGBM 3.3.2.99 documentation." [Internet], https://lightgbm.readthedocs.io/en/latest/index.html (accessed Mar. 11, 2022).
[13] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, Accessed: Mar. 11, 2022. [Online]. Available: http://arxiv.org/abs/1506.00019.
[14] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Physica D: Nonlinear Phenomena*, Vol.404, 2020, doi: 10.1016/j.physd.2019.132306.
[15] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, Vol.45, No.11, pp.2673-2681, 1997. doi: 10.1109/78.650093.
[16] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, Vol.18, No.5-6, pp.602-610, 2005, doi: 10.1016/j.neunet.2005.06.042.
[17] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *EMNLP 2014 - 2014 In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp.1724-1734, 2014. doi: 10.3115/v1/d14-1179.
[18] H. Cho, H. Im, Y. Yi, and J. Cha, "Comparison of Korean classification models' Korean essay score range prediction performance," *KIPS Transactions on Software and Data Engineering*, Vol.11, No.3, pp.133-140, 2022. doi: 10.3745/KTSDE.2022.11.3.133.
[19] S. Choi, M.-K. Park, and E. Kim, "How are Korean neural language models 'surprised' layerwisely?," *Journal of Language Sciences*, Vol.28, No.4, pp.301-317, 2021. doi: 10.14384/kals.2021.28.4.301.
[20] K. Yang, "Transformer-based Korean pretrained language models: A survey on three years of progress," 2021, Accessed: Mar. 11, 2022. [Online]. Available: http://arxiv.org/abs/2112.03014.
[21] K. Yang, W. Jang, and W. I. Cho, "APEACH: Attacking pejorative expressions with analysis on crowd-generated hate speech evaluation datasets," 2022, Accessed: Mar. 13, 2022. [Online]. Available: http://arxiv.org/abs/2202.12459.
[22] S. Park, H. Yang, M. Choe, M. Ha, K. Chung, and M. Koo, "Sentimental Analysis of YouTube Korean Comments Using KoBERT," in *Proceedings of Korea Software Congress 2020*, pp.1385-1387, 2020. [Online]. Available: http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE10529995
[23] K.-H. Park, Y.-S Jeong, "Korean daily conversation topics classification using KoBERT," in *Proceedings of Korea Software Congress 2021*, pp.1735-1737, 2021. Accessed: Dec. 27, 2021. [Online]. Available: https://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE10583420.
[24] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Information Processing & Management*, Vol.50, No.1, pp.104-112, 2014. doi: 10.1016/J.IPM.2013.08.006.
[25] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning word vectors for 157 languages," 2019. Accessed: Nov. 15, 2020. [Online]. Available: https://fasttext.cc/

Milandu Keith M. B.

https://orcid.org/0000-0002-2407-6750
e-mail : mbmk92@soongsil.ac.kr

He graduated in 2014 with a BS degree in Computer Science major Network, Database and Web from the Institut Superieur de Technologie of Libreville, Gabon. In February 2021, he received his M.S. degree in Computer Science and Engineering from Soongsil University, Seoul, South Korea. He is currently pursuing the Ph.D. degree in Computer Science and Engineering at Soongsil University, Seoul, South Korea. Milandu's research interests include the application of AI and NLP to build solutions for security and cybersecurity.

Dong-Joo Park

https://orcid.org/0000-0002-2724-1520
e-mail : djpark@ssu.ac.kr

He received his BS and MS degrees in the Computer Engineering Department at Seoul National University in February 1995 and February 1997, respectively, a Ph.D. in School of CS&E from Seoul National University in August 2001. He is currently an associate professor in the Department of Computer Science and Engineering at Soongsil University. His research interests include flash memory-based DBMSs, multimedia databases, and database systems.