J Inf Process Syst, Vol.18, No.5, pp.650~664, October 2022
https://doi.org/10.3745/JIPS.02.0182

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS

Feature Analysis for Detecting Mobile Application Review Generated by AI-Based Language Model

Seung-Cheol Lee, Yonghun Jang, Chang-Hyeon Park, and Yeong-Seok Seo*

Abstract
Mobile applications can be easily downloaded and installed via markets. However, malware and malicious applications containing unwanted advertisements exist in these application markets. Therefore, smartphone users install applications with reference to the application review to avoid such malicious applications. An application review typically comprises contents for evaluation; however, a false review with a specific purpose can be included. Such false reviews are known as fake reviews, and they can be generated using artificial intelligence (AI)-based text-generating models. Recently, AI-based text-generating models have been developed rapidly and demonstrate high-quality generated texts. Herein, we analyze the features of fake reviews generated from Generative Pre-Training-2 (GPT-2), an AI-based text-generating model and create a model to detect those fake reviews. First, we collect a real human-written application review from Kaggle. Subsequently, we identify features of the fake review using natural language processing and statistical analysis. Next, we generate fake review detection models using five types of machine-learning models trained using identified features. In terms of the performances of the fake review detection models, we achieved average F1-scores of 0.738, 0.723, and 0.730 for the fake review, real review, and overall classifications, respectively.

Keywords
Artificial Intelligence, Fake Review, GPT-2, Language Model, Machine Learning, Software Engineering

1. Introduction

Generally, Internet users access communities on social media to obtain the data required, and they can propagate information to millions of users in a relatively short time [1]. Previously, users have primarily used the Internet through desktops. However, since the launch of the iPhone 3GS by Apple in 2007, the demand for smartphones with mobile phones and Internet functions has increased significantly, resulting in the popularization of the smartphone. In addition, the rate of Internet access through smartphones has exceeded that through desktop owing to the popularization of smartphones worldwide. Currently, mobile applications installed on smartphones are used extensively to share information and perform efficient work, resulting in a rapidly developing mobile application market [2]. Mobile applications are generally distributed through application markets, and Google Play Store and Apple App Store are the primary application markets used. Application markets provide reviews and place popular applications in the top list, in which the reviews contain opinions of users and rating scores. Users verify the reliability of the

Manuscript received September 15, 2020; first revision December 30, 2020; second revision February 1, 2021; accepted February 3, 2021.
* **Corresponding Author:** Yeong-Seok Seo (ysseo@yu.ac.kr)
Dept. of Computer Engineering, Yeungnam University, Gyeongsan, Korea (fatalist316@gmail.com, killerwise2@ynu.ac.kr, park@yu.ac.kr, ysseo@yu.ac.kr)

application by referring to the review and rating scores of applications [3]. They check the app's reviews and app ratings because people with malicious purposes can acquire personal information or expose false advertisements in the app [4]. The higher the rating of the application, due to an application is placed on the top of the list, the more likely are companies to distribute the application rating scores and perform reviews in real time. In addition, an updated version of the application can be planned by reflecting the opinions and improvements of users through reviews.

Typically, reviews contain positive or negative opinions regarding applications, and these opinions can significantly affect application updates. However, reviews contain a number of false evaluation that cannot be disregarded, and these reviews can severely damage application updates [5]. The updates of applications reflecting false reviews degrade the applications. In addition, reviews unrelated to applications and those that are criticized for malicious purposes exist; such reviews are known as fake reviews. The main purpose of fake reviews is to place a specific application at the top of the product list of the market or to reduce the reliability and utilization of the application by manipulating the application review and rating score. In fact, some companies hire fake reviewers for malicious purposes [6]. Furthermore, fake reviews written through programs such as social bots have been reported [7,8].

Recently, advanced artificial intelligence (AI) technology has proven its excellence in many fields [9], and article writing at a level similar to that of humans has been enabled [10,11]. However, AI can be used for malicious purposes such as fake news or synthetic photographs, and such cases are increasing [12]. AI technology is likely to be exploited in application markets, and if exploited, will cause considerable confusion to users and service providers by several fake reviews generated in a short time. In addition, it has been regarded as a serious problem, in which fake reviews cannot be identified easily among many reviews [13]. Herein, we analyze the linguistic features of fake reviews generated from AI-based language models and research to identify fake reviews. In addition, to distinguish reviews written by fake reviewers, reviews generated by AI-based language models are named machine reviews, and the definition of a review is presented in Fig. 1.

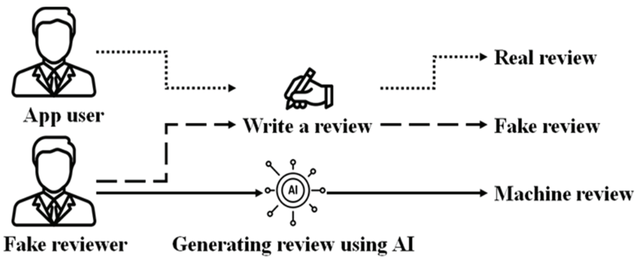


Fig. 1. Definition of three application reviews.

The contributions of this study are as follows:

- Linguistic features of reviews are extracted and defined
- Features of real and machine reviews are compared via statistical analysis
- Machine-learning-based machine review detection models are created using identified linguistic features of machine review
- Identified features are verified as significant using machine review detection models

This paper is organized as follows. Section 2 introduces the background of AI-based language models. Section 3 introduces relevant studies of AI-based language models and fake reviews. Section 4 presents the experimental preparation and linguistic feature analysis of reviews. Section 5 describes the machine review classification model, and Section 6 presents the conclusions.

2. Background

This section describes studies regarding fake reviews used for malicious purposes and AI-based language models.

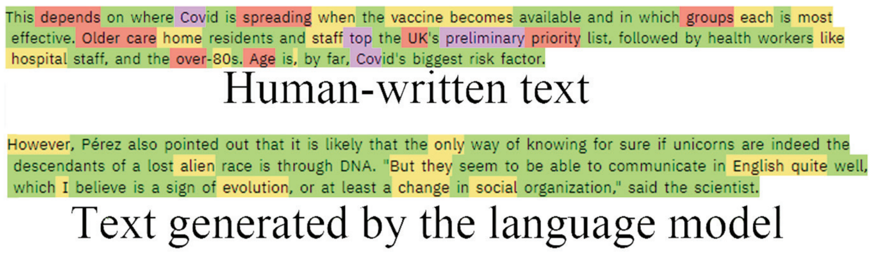
2.1 AI-based Language Models

Until recently, Internet users have shared information efficiently through a macro program known as "social bot." However, social bots are highly likely to be abused for purposes such as political propaganda or spreading of false advertising. Existing social bots can quickly generate short sentences; however, sentences written by social bots can be easily identified by humans because the quality of text is low.

In recent years, the quality of text by AI technology, which has improved rapidly, has progressed to the extent that it becomes difficult for humans to distinguish different qualities. In 2018, Generative Pre-Training-1 (GPT-1) was announced by Open AI, a non-profit research institute, followed by GPT-2 in 2019 [11,14]. In addition, the closed beta of GPT-3 was implemented in May 2020 [15]. In particular, GPT-2 has garnered significant attention from many researchers hitherto; consequently, models such as Grover was derived. GPT-2 is a language model based on AI that trains big data and writes texts when it receives initial texts. Existing social bots that repeat simple patterns are easily distinguishable; however, GPT-2 based social bots will write countless high-quality texts that are difficult for humans to identify. In addition, GPT-2 can be exploited to write fake reviews, and fake reviews by GPT-2 will adversely affect the mobile application market. In 2018, bidirectional encoder representations from transformers (BERT), a language model released by Google, trained big data including Wikipedia and book data on the web [16]. Compared with generative pre-training, BERT proved its superiority in an experiment, demonstrating unrivaled performance in natural language processing.

2.2 Giant Language Model Test Room

Giant Language Model Test Room (GLTR) is a technique for identifying texts generated from language models and is based on probability distributions [17]. A GLTR generates word candidates based on a probability distribution $\mathrm{P}(X_N|X_{1:N-1})$ affected by first words $(N-1)$ to generate the N-th word in the sentence. The top K candidates with high probability are known as "top-k," and the language model generates sentences from the top-k to the top word or any word. A GLTR uses the top-k to identify the text generated by the language model, identifies the top-k of the word in a sentence, and determines the number of words in the candidate group. In addition, it overlays colors by the range of top-k above the corresponding word in the sentence. The top-k range used is green if k is less than 10, yellow if k is less than 100, red if k is less than 1000, and purple if k is more than 1000. This is measured as a word with a higher probability of being generated by the language model as the top-k is lower. Meanwhile, the higher the top-k, the more difficult it is to distinguish it from humans. Therefore, when performing color overlay via a GLTR for texts generated via a language model, the more green areas appear in a sentence, the more likely it is that the language model has generated it. Fig. 2 shows the results of color overlays for human-written text using a GLTR and text generated through a language model. For the two sentences shown in Fig. 2, the first sentence was written by humans, and the second sentence was generated through the language model. As a result of the visualization, most of the sentences generated from the language model were displayed in green; however, in the human-written text, yellow, red, and purple were expressed variously. Hence, the GLTR can identify texts generated by language models through top-k ratios instead of using simple statistical techniques and machine learning-based analysis. However, studies regarding the detection of texts that cannot be detected by GLTR is necessitated. Experiments and analyses were conducted in this study to generate machine reviews through a linguistic model and classify high-quality machine reviews validated through a GLTR.


This depends on where Covid is spreading when the vaccine becomes available and in which groups each is most effective. Older care home residents and staff top the UK's preliminary priority list, followed by health workers like hospital staff, and the over-80s. Age is, by far, Covid's biggest risk factor.

However, Pérez also pointed out that it is likely that the only way of knowing for sure if unicorns are indeed the descendants of a lost alien race is through DNA. "But they seem to be able to communicate in English quite well, which I believe is a sign of evolution, or at least a change in social organization," said the scientist.

Fig. 2. GLTR visualization of human-written and language-model-generated texts.

3. Related Work

This section describes studies regarding fake reviews used for malicious purposes and AI-based language models.

3.1 Studies regarding Fake Reviews

Zhang et al. [18] reported the difficulty encountered by developers in identifying problems, even though users request errors and improvements in applications with reviews. Therefore, they analyzed reviews and then classified the problem into 17 types. Martens and Maalej [19] received 60,000 fake reviews from 43 fake reviewers using camouflage questionnaires and collected 66,000,000 reviews on the App Store. Subsequently, they analyzed the fake reviews and created a fake review detection classifier. Their performance evaluation showed that the classifier achieved an AUC/ROC value of 98%. He et al. [8] proposed a detection method based on a positive and unlabeled (PU) learning to prevent the spread of malicious reviews.

3.2 Application of AI-based Language Models

Ouazzane et al. [20] proposes ALMIL (adaptive language modeling intermediate layer), a framework that provides text prediction and text correction functions to corresponding users by analyzing typing stream data that reflects typing habits and vocabulary skills of users who are typing on keyboard. They applied the proposed framework to the QWERTY keyboard to create IK (Intelligent Keyboard), and achieved more enhanced text prediction and text correction performance. Zellers et al. [21] introduced a model for generating speech using the training speech of the UN General Assembly and announced the necessity for a policy to prevent language models. Kieuvongngam et al. [22] proposed a model that can provide abstract and comprehensive information using keywords extracted from the output of GPT-2 to collect the latest information regarding coronavirus disease 2019 (COVID-19) in recent years. Barrio [23] proved that creative song lyrics can be created using a GPT-2 model that trained the lyrics of Taylor Swift. Adelani et al. [24] generated a high-quality review based on sentiment using GTP-2 and classified reviews with specific sentiments with 98% accuracy using BERT. Nishi et al. [25] analyzed the effect of news written by language models on the financial market. They created a news evaluation system for financial market analysis using language models. Kreps et al. [26] warned that AI-based language models can generate news and spread them quickly. They compared the fake news written by language models with real news and discovered that both fake and real news demonstrated the same reliability. Destine-DeFreece et al. [27] conducted a project to film the famous TV program "Sex and the City," in which the script written was by GPT-2. Liao et al. [28] attempted to write classical Chinese poetry using GPT-2. The proposed method was much simpler compared with the existing RNN-based method, and in particular, the classical poetry achieved high performance in a specific area. Fangi et al. [29] investigated the generation of deep fake tweets. In their study, 23 language models that trained collected tweets generated short texts that were difficult to detect. Horvitz et al. [30] conducted a study to generate a satirical headline. Headlines generated with GPT-2 achieved scores higher by 73% compared with headlines written by humans. Lee and Hsiang [31] proposed a classification model based on BERT for patent classification. Huang et al. [32] generated legal texts using a language model and proposed ColMQA to solve logical legal problems. Peng et al. [33] proposed CMedGPT2 to train electronic medical records and generate electronic medical records for China. Salminen et al. [34] investigated the detection of hatred-related texts in social media. They confirmed that the text written by BERT was the most influential among the language models.

4. Feature Identification for Machine Review Detection

In this section, we analyze the linguistic features of machine reviews generated through AI-based language models. The overall analysis and model evaluation process are shown in Fig. 3.

4.1 Extracting Linguistic Features of Application Review

Generating a machine review requires a real review of the mobile application. We used the Kaggle website to collect real reviews written on Google Play Store and Apple App Store. GPT-2, an AI-based language model, was trained using the collected real review. Subsequently, a mobile application review style machine review was generated using the trained model. Fig. 4 shows some of the real and machine reviews.

After generating a machine review through the trained model, we verified the quality of the generated machine review using a GLTR. Consequently, a sufficiently identifiable review was created through the GLTR, and to improve the quality of the review, reinforcement learning was performed by setting the hyper-parameters of top-k and the temperature of GPT-2 to 10 and 0.4, respectively. The previously trained GPT-2 was repeated in 1,000 epochs to train the real review, and after comparing the generated review to GLTR, repeated learning was conducted for 8,000 epochs. An excellent performance machine review can be generated through iterative learning, and the result of comparing the real review and the generated machine review using GLTR features (shown in Table 1) is shown in Fig. 5.

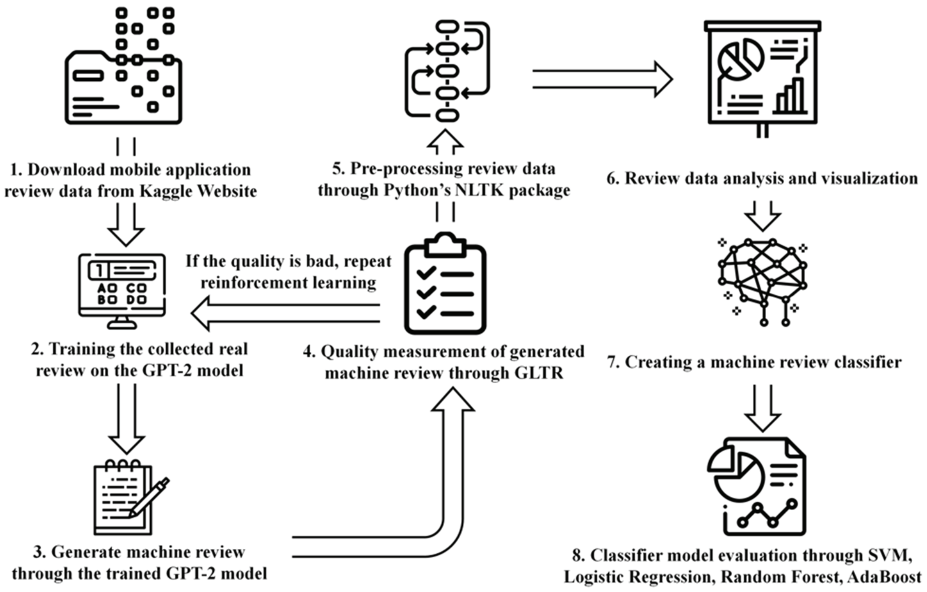
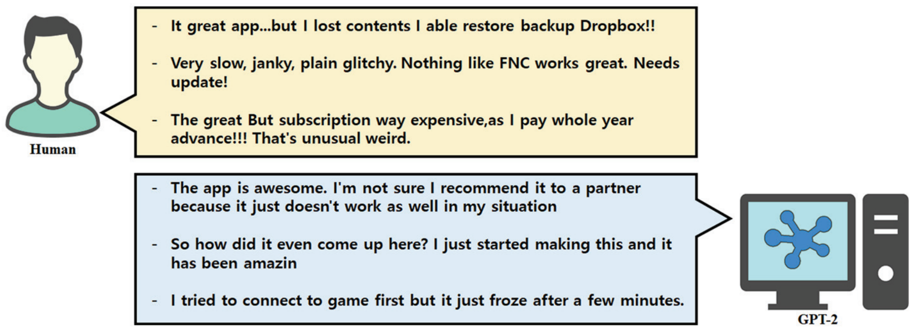

1. Download mobile application review data from Kaggle Website
2. Training the collected real review on the GPT-2 model
3. Generate machine review through the trained GPT-2 model
4. Quality measurement of generated machine review through GLTR
If the quality is bad, repeat reinforcement learning
5. Pre-processing review data through Python's NLTK package
6. Review data analysis and visualization
7. Creating a machine review classifier
8. Classifier model evaluation through SVM, Logistic Regression, Random Forest, AdaBoost

Fig. 3. Overall process for analyzing linguistic features of two reviews.



- It great app...but I lost contents I able restore backup Dropbox!!
- Very slow, janky, plain glitchy. Nothing like FNC works great. Needs update!
- The great But subscription way expensive,as I pay whole year advance!!! That's unusual weird.

- The app is awesome. I'm not sure I recommend it to a partner because it just doesn't work as well in my situation
- So how did it even come up here? I just started making this and it has been amazin
- I tried to connect to game first but it just froze after a few minutes.

Fig. 4. Examples of human-written review and examples of review generated through GPT-2 model.

Table 1. Schema of preprocessed table for machine review analysis

Column name	Description
top_k_green	top-k ≤ 10
top_k_yellow	10 < top-k ≤ 100
top_k_red	100 < top-k ≤ 1,000
top_k_purple	1,000 < top-k

In Fig. 5, the x-axis represents the features of GLTR, and the y-axis represents the real review and machine review. Additionally, top_k_green indicates that k is less than 10. If these features appear in large numbers, then it is likely that the language model has generated text. In addition, the k in top_k_yellow, top_k_red, and top_purple represent less than 100, less than 1000, and more than 1,000, respectively. A larger number of these features render them more difficult to be distinguished from humans. The machine review generated through this method yielded a slightly higher top_k_green than the real review; however, text similar to that of humans was used. In this study, as shown in Fig. 5, a machine review analysis with improved text quality was conducted.

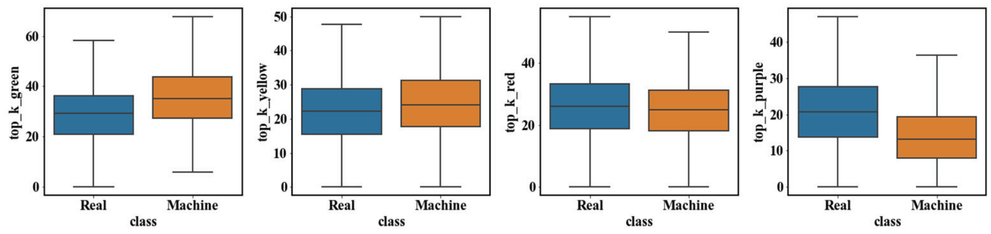


Fig. 5. Difference between GLTR features of real review and GLTR features of machine review generated by GPT-2 model with 8,000 epochs.

To extract the linguistic features of real and machine reviews, we used NLTK, Python's natural language processing support module. NLTK is a package that is widely used in natural language processing and is primarily used for morpheme analysis. The four linguistic features were composed of basic, word-based sentiment, sentence-based sentiment, and part-of-speech (POS)-based features. Table 2 shows groups of attributes of the basic features based on linguistic features. Table 3 shows groups of attributes for word-based sentiment features, and Table 4 shows groups of attributes for sentence-based emotion features. Table 5 shows the properties of POS-based features by grouping.

Table 2. Definition of the basic features of the four linguistic features

Attribute name	Description
num_count	Number of numbers
word_count	Number of words
sentence_count	Number of sentences
special_character_count	Number of special characters (!, ?, #, @, $, %, & …)
link_count	Number of links

Table 3. Definition of word-based sentimental features among four linguistic features

Attribute name	Description
word_strong_pos	Number of strong positive words
word_weak_pos	Number of weak positive words
word_strong_neg	Number of strong negative words
word_weak_neg	Number of weak negative words
word_sentimental_score_1	Word-based sentimental score 1: Refer to Equation 2
word_sentimental_score_2	Word-based sentimental score 2: Refer to Equation 3

Table 4. Definition of sentence-based sentimental features among four linguistic features

Attribute name	Description
sentence_pos	Number of positive sentences
sentence_neg	Number of negative sentences
sentence_sentimental_score_1	Sentence-based sentimental score 1: Refer to Equation 5
sentence_sentimental_score_2	Sentence-based sentimental score 2: Refer to Equation 6

Table 5. Definition of the part-of-speech features among four linguistic features

Attribute name	Description
noun_count	Number of nouns
pronoun_count	Number of pronouns
verb_count	Number of verbs
auxiliary_verb_count	Number of auxiliary verbs
adjective_count	Number of adjectives
adverbs_count	Number of adverbs
interjection_count	Number of interjections
questions_count	Number of questions
conjunctions_count	Number of conjunctions
etc_count	Number of other part-of-speech features
unknown_count	Number of typos and number of unknown nouns

Among the linguistic features, the basic features were merely basic features of the language for sentences and words. Sentimental features of words and sentences were preprocessed using NLTK's VADER sentiment analyzer, which is a Python natural language processing support package, among sentiment vocabulary dictionaries such as SentiWordNet, VADER, and Pattern. Among the modules of VADER, SentimentIntensityAnalyzer has a polarity_score function that calculates positive, negative, and neutral sentiment scores, and the sentiment score was measured using the function. The higher the value returned through the polarity_score function, the stronger is the positive sentiment; conversely, the lower the value, the stronger is the negative sentiment. In this study, a sentiment score of 0.2 or higher was assumed to be a strong positive, whereas a score of less than -0.2 was assumed to be strong negative. In addition, if the sentiment score is 0, then it is considered as neutral. Furthermore, “wsNormalize,” a function that normalizes word-based sentiment scores by intensity, is expressed as shown in Eq. (1).

Among the attributes, word-based sentimental scores 1 and 2 and sentence-based sentimental scores 1 and 2 were added to observe sentimental scores from various perspectives, and several equations were used to calculate them. Moreover, “ws1,” which is a word-based sentiment score of 1, can be obtained through Eq. (2), and “ws2,” which is a word-based sentiment score 2, can be obtained using Eq. (3). In addition, the “ssNormalize” function for generalizing the sentiment score of a sentence is expressed as shown in Eq. (4), and the sentence-based sentiment score 1 is obtained using Eq. (5). Eq. (6) shows an equation for obtaining a sentence-based sentiment score of 2. Finally, POS-based features were extracted using the pos_tag of the NLTK package. For data analysis, preprocessing was performed using the extracted features, and the schema of the preprocessed table included the following GLTR features: top_k_green, top_k_yellow, top_k_red, and top_k_purple.

$$wsNormalize(x) = \begin{cases} 1, & if\ \ polarity\_score(x) \geq 0.2 \\ 0.5, & if\ \ polarity\_score(x) < 0.2\ and\ if\ \ polarity\_score(x) > 0 \\ -1, & if\ \ polarity\_score(x) \leq -0.2 \\ -0.5, & if\ \ polarity\_score(x) > -0.2\ and\ if\ \ polarity\_core(x) < 0 \\ 0, & otherwise \end{cases} \tag{1}$$

$$ws1 = \sum_{k=1}^{n} wsNormalize(word_k) \tag{2}$$

$$ws2 = \sum_{k=1}^{n} polarity\_score(word_k) \tag{3}$$

$$ssNormalize(x) = \begin{cases} 1, & if\ \ polarity\_score(x) > 0 \\ -1, & if\ \ polarity\_score(x) < 0 \\ 0, & otherwise \end{cases} \tag{4}$$

$$ss1 = \sum_{k=1}^{n} ssNormalize(sentence_k) \tag{5}$$

$$ss2 = \sum_{k=1}^{n} polarity\_score(sentence_k) \tag{6}$$

4.2 Analysis of Features of Machine Review

We performed a statistical analysis to confirm the difference between real and machine reviews. First, we conducted an F-test to determine equality of variances for all features. The results show that question_count, num_count, unknown_count, sentence_neg, word_strong_neg, word_weak_neg, and word_sentimental_score_1 satisfied the equality of variances. However, they did not present a p-value <0.05 on an independent T-test; hence, they were regarded as meaningless features. Table 6 shows the features satisfying the equality of variances and their p-values on an independent T-test. Among the

Table 6. Results of T-test for features that do not satisfy equal variance

Attribute name	*p*-value
special_character_count	3.501997750080279e-05
noun_count	2.888123643568305e-06
pronoun_count	6.389010497026173e-39
verb_count	1.1831328824615509e-14
auxiliary_verb_count	0.0022774726180378236
adjective_count	7.114495403144986e-06
adverb_count	2.0458729106901304e-07
interjection_count	0.00030610638480726006
conjunction_count	2.0644178039054096e-06
etc_count	2.4514895681089804e-16
sentence_count	3.32933521295455e-13
sentence_pos	8.715056931697415e-15
sentence_sentimental_score_1	1.3697905726638073e-09
sentence_sentimental_score_2	1.2403493784116909e-08
word_count	1.0939986341096128e-19
word_strong_pos	7.376694649753007e-13
word_weak_pos	3.997635870522615e-08
word_sentimental_score_2	0.0008491579669236737

features in Table 6, word_count, sentence_count, sentence_sentimental_score_1, sentence_sentimental_score_2, noun_count, pronoun_count, verb_count, etc_count, and special_character_count indicated a *p*-value <0.05. To visualize these features, we removed outliers and then represented them in a boxplot. The visualization results, as depicted in Fig. 6, shows that in terms of noun_count and sentence_count, machine reviews use more words than real reviews. Generally, when machine reviews are written, it is predicted whether they contain many words and sentences. Meanwhile, real reviews often use special_character_count, whereas machine reviews do not use it. Comparing noun_count and sentence_sentimental_score_1, machine reviews use more positive sentiments than real reviews. In etc_count, both reviews exhibit a similar pattern, but a different median value. It is predicted that such words are used slightly more frequently in machine reviews.

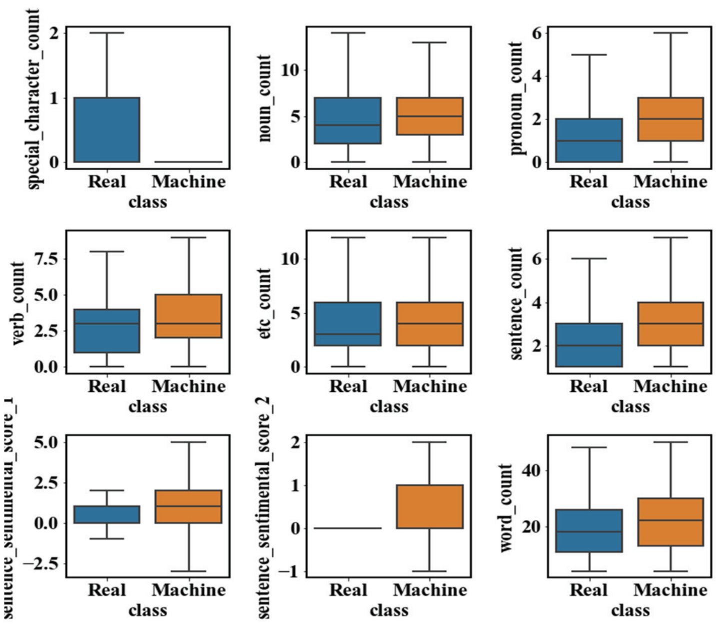


Fig. 6. Boxplot showing significant properties of *p*-value <0.05.

In particular, it is assumed that other words not in real reviews appear in machine reviews because our proposed GPT-2 model is based on the 117M GPT-2 small model. In terms of noun_count, pronoun_count, and verb_count, they are a concept of word units. Because machine reviews use more words than real reviews, POS counts appear more in machine reviews. In real reviews, a reviewer writes more noun-oriented texts, whereas machine reviews are written using fewer nouns.

5. Feature Evaluation using Machine Review Classification Model

To design our experiment, we collected 30,000 real human-written reviews from Kaggle and 5,000 machine reviews generated by GPT-2. We built and evaluated a machine review classifier based on five representative classifiers using the features analyzed in Section 4.2. The classifiers were as follows: logistic regression, random forest, support vector machine (SVM), neural network, and AdaBoost. We used the scikit-learn package to apply them. To evaluate our model, we used the accuracy and F1-score of the machine and real reviews, as well as the macro F1-score. Five-fold cross-validation was applied in each experiment for a balanced evaluation.

5.1 Hyper-parameters of Model used for Machine Review Classification

Table 7 shows the hyper-parameters of each model used in the experiment. In Table 7, "c" hyper parameter of SVM, determines the number of data samples allowed to be placed in different classes. "Gamma" is the effect of one element in SVM, and if this value is set to scale, then the effect is calculated as 1/(count of features×variance of elements). In logistic regression, "solver" means the optimization algorithm, and when the solver is set to lbfgs, the limited memory-BFGS algorithm searches for the optimization value. In a random forest, max_depth is the maximum depth of the tree, and if max_depth is "none," the random forest will differentiate the samples without limitation until only one sample is entered in the leaf node. In AdaBoost, n_estimators mean the number of child learners to be created while AdaBoost is executing. The learning_rate of the machine network means the step size of the loss function per iteration, and loss is a parameter that designates a function to calculate the loss value. The optimizer designates an algorithm for optimizing the loss, and the hidden_node denotes the number of nodes in the hidden layer of the neural network. Additionally, hidden_layer_count is the number of hidden layers used for learning.

Table 7. Hyper-parameter definition of machine review classification model

Model name	Hyper parameter	Value
SVM	c	1
	gamma	Scale
Logistic regression	c	1
	solver	lbfgs
Random Forest	n_estimators	100
	max_depth	None
AdaBoost	base_estimators	DecisionTreeClassifier(max_depth=5)
	n_estimators	50
	learning_rate	0.01
Neural network	learning_rate	0.001
	loss	binary_crossentropy
	optimizer	RMSprop
	hidden_node	32
	hidden_layer_count	1

5.2 Evaluation of Machine Review Classification Model by Features

Table 8 shows the average performance with five-fold cross-validation of the models using GLTR features, the models using linguistic features analyzed in Section 4, and the models using GLTR features and linguistic features simultaneously. In the table, the underlined numbers indicate the maximum values, and the minimum values are indicated in parentheses. Among the classification models for all features, the random forest classification model performed the best, whereas the SVM classification model performed the worst. In general, the models using linguistic features as inputs outperformed the models using GLTR features as inputs. In particular, when GLTR and linguistic features were used simultaneously, the performance improved by 5%–6% compared with the case where only GLTR features were used. In addition, except for machine networks, real reviews written by humans were better classified than machine reviews. This was because the parameters of top_k_green, top_k_yellow, top_k_red, top_k_purple were applied in a balanced manner. Meanwhile, in the machine review, it was predicted that the four features of GLTR appeared differently in each sentence.

When using GLTR features alone and GLTR features and linguistic features simultaneously, the difference in performance by model was insignificant. However, when using only linguistic features, the maximum difference in performance between the random forest and SVM was 13%. The performance difference between the random forest and AdaBoost was more significant than those of other models, and the use of an ensemble model was appropriate when linguistic features were used in the classification model.

Table 8. Average performance of classification models by features

Feature type	Model name	Accuracy	Real-F1	Machine-F1	Macro-F1
Use GLTR feature	SVM	(0.620)	(0.640)	(0.606)	(0.620)
	Logistic regression	0.634	0.640	0.628	0.630
	Random forest	<u>0.680</u>	<u>0.686</u>	<u>0.678</u>	<u>0.680</u>
	AdaBoost	0.636	0.642	0.638	0.636
	Neural network	0.638	0.652	0.620	0.634
	Average	0.642	0.652	0.634	0.640
Use language feature	SVM	(0.630)	(0.662)	(0.590)	(0.626)
	Logistic regression	0.638	0.662	0.612	0.638
	Random forest	<u>0.726</u>	<u>0.726</u>	<u>0.724</u>	<u>0.726</u>
	AdaBoost	0.706	0.704	0.704	0.704
	Neural network	0.642	0.630	0.644	0.636
	Average	0.668	0.677	0.655	0.666
Use GLTR+language feature	SVM	(0.704)	(0.716)	(0.692)	(0.704)
	Logistic regression	0.714	0.722	0.704	0.714
	Random forest	<u>0.782</u>	<u>0.784</u>	<u>0.778</u>	<u>0.778</u>
	AdaBoost	0.736	0.736	0.736	0.736
	Neural network	0.718	0.734	0.704	0.718
	Average	0.731	0.738	0.723	0.730

The numbers in parentheses indicates the lowest performance in the evaluation criteria of each feature type.
The numbers underlined indicates the highest performance in the evaluation criteria of each feature type.

6. Conclusion

In this study, we analyzed machine reviews generated from AI-based linguistic models and real reviews written by humans. We collected a real review from Kaggle. Subsequently, we trained a GPT-2 model using a real review. The trained GPT-2 model generated texts in a mobile application review manner. In addition, the generated machine review was compared with GLTR; subsequently, iterative learning was performed to create a more human-like machine review. We used Python’s natural language processing package, NLTK, to extract linguistic features from the mobile application review. In addition, we analyzed the differences between machine and real reviews using T-tests and boxplots. Consequently, we confirmed the linguistic features of the machine review. Based on the analysis results, a machine review classifier model was created using five machine learning techniques. In addition, we conducted a model evaluation using the existing linguistic features and four features of GLTR. Our results showed that the F1-scores for the machine and real review classifications of the model were 0.723 and 0.738, respectively, whereas the macro F1-score was 0.730. However, when training GPT-2 language generation model, it seems that the classification of machine reviews will become difficult if more data is trained. In addition, an analysis using language models other than GPT-2 is necessary, and analysis techniques that are widely used in natural language processing should be applied, and we plan to proceed with this for future research.

Acknowledgement

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. NRF-2020R1I1A3073313).

References

[1] Y. Liu, Z. Bao, Z. Zhang, D. Tang, and F. Xiong, "Information cascades prediction with attention neural network," *Human-centric Computing and Information Sciences*, vol. 10, article no, 13, 2020. https://doi.org/10.1186/s13673-020-00218-w

[2] Z. Zhang, J. Jing, X. Wang, K. K. R. Choo, and B. B. Gupta, "A crowdsourcing method for online social networks security assessment based on human-centric computing," *Human-centric Computing and Information Sciences*, vol. 10, article no, 23, 2020. https://doi.org/10.1186/s13673-020-00230-0

[3] M. Harman, Y. Jia, and Y. Zhang, "App store mining and analysis: MSR for app stores," in *Proceedings of 2012 9th IEEE Working Conference on Mining Software Repositories (MSR)*, Zurich, Switzerland, 2012, pp. 108-111.

[4] M. Talal, A. A. Zaidan, B. B. Zaidan, O. S. Albahri, M. A. Alsalem, A. S. Albahri, et al., "Comprehensive review and analysis of anti-malware apps for smartphones," *Telecommunication Systems*, vol. 72, no, 2, pp. 285-337, 2019.

[5] S. Y. Choi, C. G. Lim, and Y. M. Kim, "Automated link tracing for classification of malicious websites in malware distribution networks," *Journal of Information Processing Systems*, vol. 15, no, 1, pp. 100-115, 2019.

[6] H. Chen, D. He, S. Zhu, and J. Yang, "Toward detecting collusive ranking manipulation attackers in mobile app markets," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, Abu Dhabi, United Arab Emirates, 2017, pp. 58-70.

[7] N. Genc-Nayebi and A. Abran, "A systematic literature review: opinion mining studies from mobile app store user review," *Journal of Systems and Software*, vol. 125, pp. 207-219, 2017.

[8] D. He, M. Pan, K. Hong, Y. Cheng, S. Chan, X. Liu, and N. Guizani, "Fake review detection based on PU learning and behavior density," *IEEE Network*, vol. 34, no. 4, pp. 298-303, 2020.

[9] Y. S. Jeong and J. H. Park, "Learning algorithms in AI system and services," *Journal of Information Processing System*, vol. 15, no, 5, pp. 1029-1035, 2019.

[10] A. See, A. Pappu, R. Saxena, A. Yerukola, and C. D. Manning, "Do massively pretrained language models make better storytellers?," 2019 [Online]. Available: https://arxiv.org/abs/1909.10705.

[11] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019 [Online]. Available: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

[12] Y. Jang, C. H. Park, and Y. S. Seo, "Fake news analysis modeling using quote retweet," *Electronics*, vol. 8, no, 12, article no. 1377, 2019. https://doi.org/10.3390/electronics8121377

[13] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, "Finding deceptive opinion spam by any stretch of the imagination," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, OR, 2011, pp. 309-319.

[14] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018 [Online].
Available: https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf.

[15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, et al., "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877-1901, 2020.

[16] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," 2019 [Online]. Available: https://arxiv.org/abs/1810.04805.

[17] S. Gehrmann, H. Strobelt, and A. M. Rush, "GLTR: statistical detection and visualization of generated text," [2019]. Available: https://arxiv.org/abs/1906.04043.

[18] L. Zhang, X. Y. Huang, J. Jiang, and Y. K. Hu, "CSLabel: an approach for labelling mobile app reviews," *Journal of Computer Science and Technology*, vol. 32, no. 6, pp. 1076-1089, 2017.

[19] D. Martens and W. Maalej, "Towards understanding and detecting fake reviews in app stores," *Empirical Software Engineering*, vol. 24, no. 6, pp. 3316-3355, 2019.

[20] K. Ouazzane, J. Li, H. B. Jun, Y. Jing, and R. Boyd, "An artificial Intelligence-based language modeling framework," *Expert Systems with Applications*, vol. 39, no, 5, pp. 5960-5970, 2012.

[21] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news," *Advances in Neural Information Processing Systems*, vol. 32, pp. 9054-9065, 2019.

[22] V. Kieuvongngam, B. Tan, and Y. Niu, "Automatic text summarization of COVID-19 medical research articles using BERT and GPT-2," 2020 [Online]. Available: https://arxiv.org/abs/2006.01997.

[23] S. Barrio, "Writing the next American hit: using GPT-2 to explore the possibility of creating successful AI-generated song lyrics," 2020 [Online].
Available: https://digital.kenyon.edu/cgi/viewcontent.cgi?article=1011&context=dh_iphs_prog.

[24] D. I. Adelani, H. Mai, F. Fang, H. H. Nguyen, J. Yamagishi, and I. Echizen, "Generating sentiment-preserving fake online reviews using machine language models and their human- and machine-based detection," in *Advanced Information Networking and Applications*. Cham, Switzerland: Springer, 2020, pp. 1341-1354.

[25] Y. Nishi, A. Suge, and H. Takahashi, "Construction of news article evaluation system using language generation model," in *Agents and Multi-Agent Systems: Technologies and Applications 2020*. Singapore: Springer, 2020, pp. 313-320.

[26] S. Kreps, R. M. McCain, and M. Brundage, "All the news that's fit to fabricate: AI-generated text as a tool of media misinformation," *Journal of Experimental Political Science*, vol. 9, no. 1, pp. 104-117, 2022.

[27] A. Destine-DeFreece, S. Handelsman, T. Light Rake, A. Merkel, and G. Moses, "Can GPT-2 replace a Sex and the City writers' room?," 2019 [Online]. Available: https://digital.kenyon.edu/dh_iphs_ai/15/.

[28] Y. Liao, Y. Wang, Q. Liu, and X. Jiang, "GPT-based generation for classical Chinese," 2019 [Online]. Available: https://arxiv.org/abs/1907.00151.

[29] T. Fagni, F. Falchi, M. Gambini, A. Martella, and M. Tesconi, "TweepFake: about detecting deepfake tweets," 2021 [Online]. Available: https://arxiv.org/abs/2008.00036.

[30] Z. Horvitz, N. Do, and M. L. Littman, "Context-driven satirical headline generation," in *Proceedings of the 2nd Workshop on Figurative Language Processing*, Virtual Event, 2020, pp. 40-50.

[31] J. S. Lee and J. Hsiang, "Patent classification by fine-tuning BERT language," *World Patent Information*, vol. 61, article no. 101965, 2020. https://doi.org/10.1016/j.wpi.2020.101965

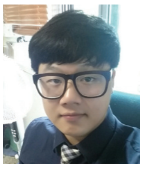
[32] W. Huang, X. Liao, Z. Xie, J. Qian, B. Zhuang, S. Wang, and J. Xiao, "Generating reasonable legal text through the combination of language modeling and question answering," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, Virtual Event, 2020, pp. 3687-3693.

[33] J. Peng, P. Ni, J. Zhu, Z. Dai, Y. Li, G. Li, and X. Bai, "Automatic generation of electronic medical record based on GPT2 model," in *Proceedings of 2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, 2020, pp. 6180-6182.

[34] J. Salminen, M. Hopf, S. A. Chowdhury, S. G. Jung, H. Almerekhi, and B. J. Jansen, "Developing an online hate classifier for multiple social media platforms," *Human-centric Computing and Information Sciences*, vol. 10, article no. 1, 2020. https://doi.org/10.1186/s13673-019-0205-6

Seung-Cheol Lee **https://orcid.org/0000-0001-8995-7463**

He received the B.S. degree in computer engineering from the National Institute for Lifelong Education, Republic of Korea, in 2018. He is currently a M.S. student in the Department of Computer Engineering, Yeungnam University, Republic of Korea. His research interests include machine learning, big data analysis, natural language processing, and software engineering.



Yonghun Jang **https://orcid.org/0000-0003-1987-6558**

He received the B.S. degree in computer engineering from the National Institute for Lifelong Education, Republic of Korea, in 2012, and M.S and Ph.D. degrees in computer science from the University of Yeungnam, Republic of Korea, in 2020. He is currently a post-doctoral researcher at the Department of Computer Engineering, Yeungnam University, Republic of Korea. His research interests include data analysis, machine learning, and vision systems.

Chang-Hyeon Park **https://orcid.org/0000-0001-6903-3626**

He received the B.S. degree in electronics engineering from Kyungpook University, in Republic of Korea, in 1986, and M.S and Ph.D. degrees in computer statistics from Kyungpook University, Republic of Korea, in 1989 and 1992, respectively. From 1998 to 1999, he was a visiting researcher at the University of Maryland, Institute of Advanced Computer System. From 2009 to 2010, he was a visiting professor at the University of Washington, Department of Electronic Engineering. He is currently a professor at the Department of Computer Engineering, Yeungnam University, Republic of Korea. His research interests include AI, data mining, big data, and vision systems.

Yeong-Seok Seo **https://orcid.org/0000-0002-5319-7674**

He received the B.S. degree in computer science from Soongsil University, Korea, in 2006, and M.S. and Ph.D. degrees in computer science from the Republic of Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea, in 2008 and 2012, respectively. From September 2012 to December 2013, he was a post-doctoral researcher at the KAIST Institute for Information and Electronics. From January 2014 to August 2016, he was a senior researcher at the Republic of Korea Testing Laboratory, Republic of Korea. He is currently an assistant professor at the Department of Computer Engineering, Yeungnam University, Republic of Korea. His research interests include software engineering, artificial intelligence, Internet of Things, and data mining.