Hindawi
Mobile Information Systems
Volume 2021, Article ID 9987462, 9 pages
https://doi.org/10.1155/2021/9987462

*Research Article*

# KoRASA: Pipeline Optimization for Open-Source Korean Natural Language Understanding Framework Based on Deep Learning

**Myeong-Ha Hwang, Jikang Shin, Hojin Seo, Jeong-Seon Im, and Hee Cho**

*Digital Solution Laboratory, Korea Electric Power Research Institute (KEPRI), Daejeon, Republic of Korea*

Correspondence should be addressed to Myeong-Ha Hwang; mh.hwang@kepco.co.kr

Received 15 March 2021; Revised 27 May 2021; Accepted 14 June 2021; Published 24 June 2021

Academic Editor: Muhammad Bilal



Since the emergence of deep learning-based chatbots for knowledge services, numerous research and development projects have been conducted in various industries. A high demand for chatbots has drastically increased the global market size; however, the limited functional scalability of open-domain chatbots is a challenge to their application to industries. Moreover, as most chatbot frameworks employ English, it is necessary to create chatbots customized for other languages. To address this problem, this paper proposes KoRASA as a pipeline-optimization method, which uses a deep learning-based open-source chatbot framework to understand the Korean language. KoRASA is a closed-domain chatbot that is applicable across a wide range of industries in Korea. KoRASA's operation consists of four stages: tokenization, featurization, intent classification, and entity extraction. The accuracy and $F1$-score of KoRASA were measured based on datasets taken from common tasks carried out in most industrial fields. The algorithm for intent classification and entity extraction was optimized. The accuracy and $F1$-score were 98.2% and 98.4% for intent classification and 97.4% and 94.7% for entity extraction, respectively. Furthermore, these results are better than those achieved by existing models. Accordingly, KoRASA can be applied to various industries, including mobile services based on closed-domain chatbots using Korean, robotic process automation (RPA), edge computing, and Internet of Energy (IoE) services.

## 1. Introduction

Chatbots are communication tools that can be used to achieve a goal through automated dialog without human intervention [1]. Similar to a personal assistant, chatbots have been used in various applications, such as booking restaurant reservations, assisting with shopping, and conducting web searches. In recent years, many businesses around the world have strengthened their competitiveness by using chatbots and, as a result, many industries in Korea have also introduced chatbots in their services [2]. The global market size of chatbots is increasing at a rapid pace, as the benefits they provide have produced a high demand. Accordingly, large companies such as Facebook, Google, and Naver have started providing chatbot services as part of their professional services [3]. For example, in April 2016, eight years after Steve Jobs introduced the App Store, Mark Zuckerberg of Facebook announced the launch of Facebook Messenger platform, which includes chatbots. The number of users on that platform has reached over one billion, which exceeds the number of iPhone users that existed at the launch of the App Store.

However, application of open-domain chatbots is not effective in every industry because of the limited functional scalability of these chatbots. Moreover, as most chatbot frameworks employ English, it is necessary to create chatbots customized for other languages. To this end, this paper proposes KoRASA as a pipeline-optimization method using a deep learning-based open-source chatbot framework to understand the Korean language. KoRASA, which is a closed-domain chatbot, consists of four stages: (1) a tokenization step, which separates the corpus into tokens, i.e., language elements that cannot be further grammatically divided; (2) a featurization step, which extracts the features of each token; (3) an intent classification step, which classifies the intent of each question posed to the chatbot; and

(4) an entity extraction step, which extracts the relevant entity from the question sentence. In the present study, an optimization experiment was performed at each stage with respect to the Korean language understanding. Based on datasets reflecting common tasks in industry, the accuracy and $F1$-score of our proposed method were 98.2% and 98.4% for intent classification and 97.4% and 94.7% for entity extraction, respectively. Accordingly, KoRASA can be used not only for developing a closed-domain chatbot for the Korean language, but also for various industrial purposes such as mobile chatbot services, edge computing services, and robotic process automation (RPA) services.

The remainder of this paper is organized as follows. In Section 2, related work is discussed. The method of our proposed work is explained in Section 3 and we provide experimental results in Section 4. Finally, we conclude the paper with future research directions in Section 5.

## 2. Related Work

*2.1. Chatbot Framework.* As numerous companies have started introducing and utilizing chatbots in recent years, various research and development projects have been conducted to improve the functionality of chatbots. Adamopoulou et al. [4] introduced the history, technologies, and applications of chatbots, and Cahn [5] proposed the architecture and design of chatbots and the process of chatbot development, thereby presenting guidelines for chatbot researchers and developers to follow. Figure 1 shows the architecture of a typical chatbot. When a user inputs a message, the intent is classified, and an entity is extracted by the natural language understanding (NLU) component. A tracker maintains a conversation state, while the intent and entity (which have been classified and extracted by NLU) are put into slots to await further processing. A dialog management system then designates the next action based on previous actions, action results, and the contents of the slots. The message generator then provides the user with a corresponding response. If it is necessary to extract data from a database or URL, an application programming interface (API) is used for communication, transmission, and reception.

Gwendal et al. [6] suggested a Xatkit framework to develop a multimodal low-code chatbot. Xatkit comes with a runtime engine that automatically deploys the chatbot application and manages the defined conversation logic over the platforms of choice. Addi et al. [7] developed a chatbot based on knowledge graphs, which led to the emergence of chatbots capable of learning diverse data types. Furthermore, Park et al. [8] developed a framework for developing dialog systems in a smart home environment. The framework ontologically presents the knowledge required for the task-oriented dialog system's process and can build a dialog system by editing the dialog knowledge. Reshmi and Balakrishnan [9] proposed an interface framework for providing customer service chatbots. More specifically, they proposed an integration method with big data as a knowledge base for the chatbots that can enable the generation of dynamic responses to user queries and improve the analytical capability of chatbots with data from a distributed environment. By using chatbots in this manner, developers can create services that are tailored to specific fields and increase the productivity of companies. However, because the chatbots proposed by these studies are based on frameworks that were designed for specific purposes, their development and use in a broad range of industries are limited.

*2.2. Natural Language Understanding (NLU).* The key technology used in chatbots is NLU, which involves transforming natural language expressions into a machine-readable format. Figure 2 illustrates the architecture of NLU. The preprocessing stage of NLU is divided into two substages. The first substage is tokenization, in which a corpus is divided into tokens, which are the grammatically indivisible units of a language. The second substage is featurization, wherein the features of each token are extracted. After the preprocessing stage, intent is classified to ensure that the user's request message is sufficiently understood, and an entity is extracted to provide a suitable response. Through this process, a user-friendly chatbot system can be developed.

There are two types of NLU [10]: rule-based and deep-learning-based. Rule-based NLU depends on features extracted by humans, which means that it takes an extended period to manually extract features from sample user input. In addition, the accuracy of rule-based NLUs decreases as document volume increases. Deep-learning-based NLU automatically extracts and learns features from the data. Compared with the existing rule-based NLUs, it enables wider contextual processing and can also be used to build multimodal models by combining it with models that learn other types of data, such as images and voices. Accordingly, many recent studies have actively employed deep-learning-based NLU.

For example, Radford and Narasimhan [11] conducted research on improving natural language understanding using the generative pretraining model. Liu et al. [12] designed and developed multitask deep neural networks for natural language understanding. Jiao et al. [13] developed a method using the popular transformer-based machine learning technique BERT for natural language understanding. In the field of NLU research, studies on understanding natural language using transformer are becoming increasingly popular [14].

Among the different deep-learning-based NLU services, LUIS, Watson Conversation, API.ai, and RASA have been widely used in recent years [15–18]. Braun et al. [19] evaluated these NLU services for conversational question-answering systems (see Table 1). Such NLU services share the same basic concept: users can train intent classification and entity extraction based on example data. In the performance evaluation of the NLU services of chatbots, RASA and LUIS achieved an $F$-score of 0.94, whereas Watson Conversation and API.ai only achieved scores of 0.74 and 0.68, respectively. In particular, RASA, an open-source chatbot framework, is a strong framework as it enables users to easily

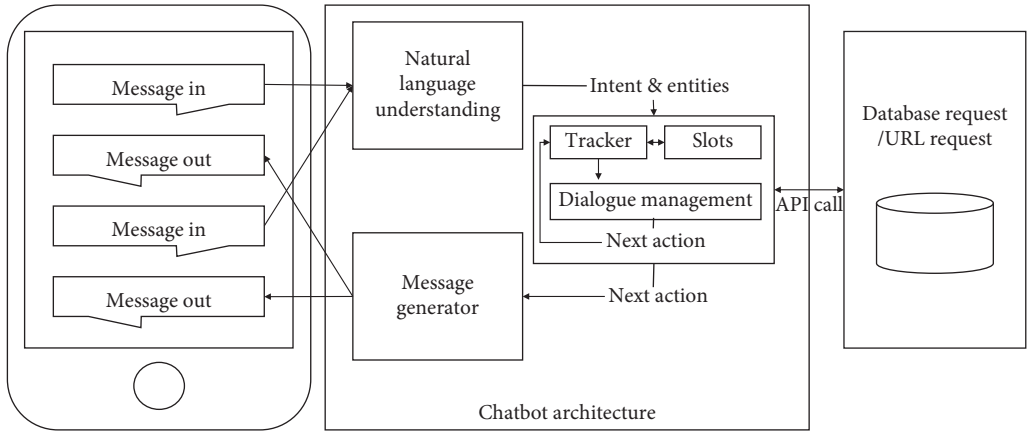


Figure 1: Chatbot architecture.

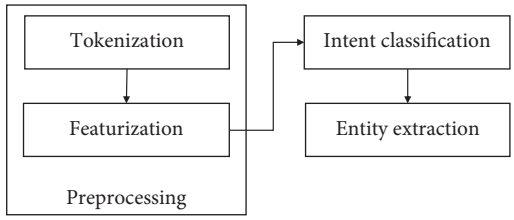


Figure 2: Architecture of natural language understanding (NLU).

develop the functions of machine-learning-based dialog management and natural language understanding. Therefore, in the present study, we adopted RASA as a deep learning-based open-source chatbot framework [18]. However, RASA is optimized for English; thus, to develop a chatbot for use in Korean industries, the framework must be optimized through performance experiments, examining the different stages in the pipeline, to enable it to understand the Korean language.

Recently, deep learning has been actively applied in various industrial as well as NLU fields. O'Shea and Hoydis [20] developed a new fundamental method of considering communications system design as an end-to-end reconstruction task. They demonstrated the application of convolutional neural networks on raw IQ samples for modulation classification. Aceto et al. [21, 22] proposed a general framework for deep learning-based mobile and encrypted traffic classification based on a rigorous definition of its milestones. They overcame the design limitations of previous works by envisioning the simultaneous use of multimodal and multitask techniques. Hwang et al. [23] proposed Gen2Vec, a search engine-based framework for operating power plants using deep learning. Gen2Vec can be the core engine of the knowledge system for power plant operators and can be used in search and chatbot services for power plant operation programs and new employee training. Wang et al. [24] proposed a deep learning-based ensemble approach for probabilistic wind power forecasting. They could effectively extract the nonlinear and stochastic nature exhibited in each wind power frequency, and the latter could cancel out the diverse errors. Wang et al. [25] proposed a method of detecting potential adverse drug reactions (ADRs) using a deep neural network model. They could not only discover the potential ADRs of drugs but also predict the possible ADRs of new drugs. Roman et al. [26] proposed a machine learning pipeline for battery state-of-health estimation, providing insights into the design of scalable data-driven models for battery SOH estimation, emphasizing the value of confidence bounds around the prediction.

Table 1: *F*-scores for the different NLU services.

| Service | *F*-score |
|---|---|
| LUIS | 0.94 |
| Watson | 0.74 |
| API.ai | 0.68 |
| RASA | 0.94 |

## 3. KoRASA: Pipeline Optimization for Open-Source Korean Natural Language Understanding Based on Deep Learning

*3.1. Pipeline Architecture of KoRASA.* KoRASA is an optimization method that uses a deep learning-based open-source chatbot framework to understand the Korean language. KoRASA is developed based on the RASA framework [18], and it follows a typical NLU process, including tokenization, featurization, intent classification, and entity extraction, as shown in Figure 2. Figure 3 illustrates the pipeline architecture of KoRASA as proposed in this study. The tokenizer uses the Korean version of Mecab, and a count vector featurizer is adopted [27, 28]. Then, the Dual Intent Entity Transformer (DIET) is used for intent classification and entity extraction of KoRASA [29]. DIET is capable of fast learning using a common baseline model. However, as KoRASA is intended to be applied to the industrial field; the focus is not on the learning speed of the model but on its ability to perform intent classification and entity extraction. Accordingly, the architecture in this study was optimized by tuning its parameters. Moreover, to maximize the performance of entity extraction, a mapping function was added using the Entity Synonym Mapper, and the Transformer Embedding Dialogue (TED) policy was applied [18, 30].

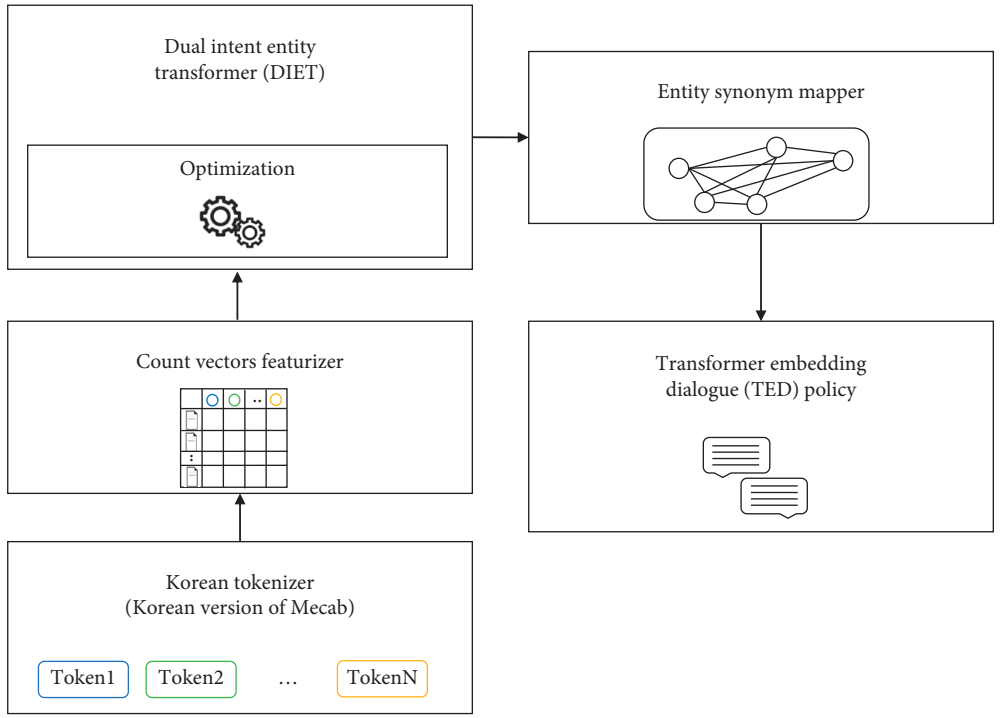


Figure 3: Pipeline architecture of KoRASA.

*3.2. Optimized Structure of DIET Model (DIET-Opt).* Figure 4 shows the optimization architecture of the DIET model of KoRASA (DIET-Opt). We refer to the Rasa blog for parameter optimization in KoRASA [31]. Tokens are extracted by the Korean tokenizer and are then input into the transformer layer through feed-forward layers. The sequence ($a$), which is the output from the transformer layer, becomes the sequence ($y_{\text{entity}}$) of entity labels and an input value for the conditional random field (CRF) algorithm. As shown in equation (1), the entity loss ($L_{\text{Entity}}$) can be calculated using the negative log-likelihood of CRF.

$$L_{\text{Entity}} = L_{\text{CRF}}\left(a,\ y_{\text{entity}}\right). \tag{1}$$

The CLS token ($a_{\text{CLS}}$) and intent labels ($y_{\text{intent}}$), which refer to the sentence encoding of an input sentence, are input into the respective embedding layers ($h_{\text{CLS}} = E(a_{\text{CLS}})$, $h_{\text{intent}} = E(y_{\text{intent}})$, where $h \in \mathbb{R}^{20}$). In addition, as expressed in equation (2), the dot-product loss used here not only maximizes the target label ($y^{+}_{\text{intent}}$) and the similarity measure ($S^{+}_{\text{Intent}} = h^{T}_{\text{CLS}} h^{+}_{\text{intent}}$) but also minimizes negative samples ($y^{-}_{\text{intent}}$) and similarities ($S^{-}_{\text{Intent}} = h^{T}_{\text{CLS}} h^{-}_{\text{intent}}$) [27].

$$L_{\text{Intent}} = -< S^{+}_{\text{Intent}} - \log\left( e^{S^{+}_{\text{Intent}}} + \sum_{\Omega^{-}_{\text{Intent}}} e^{S^{-}_{\text{Intent}}} \right) >. \tag{2}$$

Tokens include mask tokens, and values that are output from the transformer layer following the feed-forward layer are then input into the embedding layers. There are two embedding layers: an embedding layer for mask tokens and another for unmasked actual tokens. In practice, 15% of the input tokens are randomly selected in each sequence. A total of 70% of the tokens selected are then replaced by a special mask token MASK, and 10% of them are replaced by random tokens. For the remaining 20%, the original tokens are used directly as input. As shown in equation (3), the output of the transformer ($a_{\text{MASK}}$) for each selected token ($y_{\text{token}}$) is provided through a dot-product loss that is similar to the intent loss.

$$L_{\text{Mask}} = -< S^{+}_{\text{Mask}} - \log\left( e^{S^{+}_{\text{Mask}}} + \sum_{\Omega^{-}_{\text{Mask}}} e^{S^{-}_{\text{Mask}}} \right) >. \tag{3}$$

Finally, the model is trained to minimize the total loss ($L_{\text{Total}}$), which can be expressed by

$$L_{\text{Total}} = L_{\text{Intent}} + L_{\text{Entity}} + L_{\text{Mask}}. \tag{4}$$

Although the overall architecture of KoRASA is similar to the baseline model of DIET (DIET-Base), various experiments were performed to improve the performance of its intent classification and entity extraction processes. An experiment was conducted to determine the optimal number of epochs for best performance. Instead of the traditionally used 300 epochs, 500 epochs were adopted as the optimal number. The number of transformer layers was also increased from 2 to 4, while the weight sparsity was decreased from 0.8 to 0.7. The number of embedding dimensions was increased from 20 to 30, and the hidden layer size increased from 256 to 512, thus optimizing the parameters (see Table 2).

## 4. Experiments and Results

*4.1. Dataset.* For the experiments performed in this study, datasets were built based on common tasks applicable to most industries. For the intent classification experiment, a

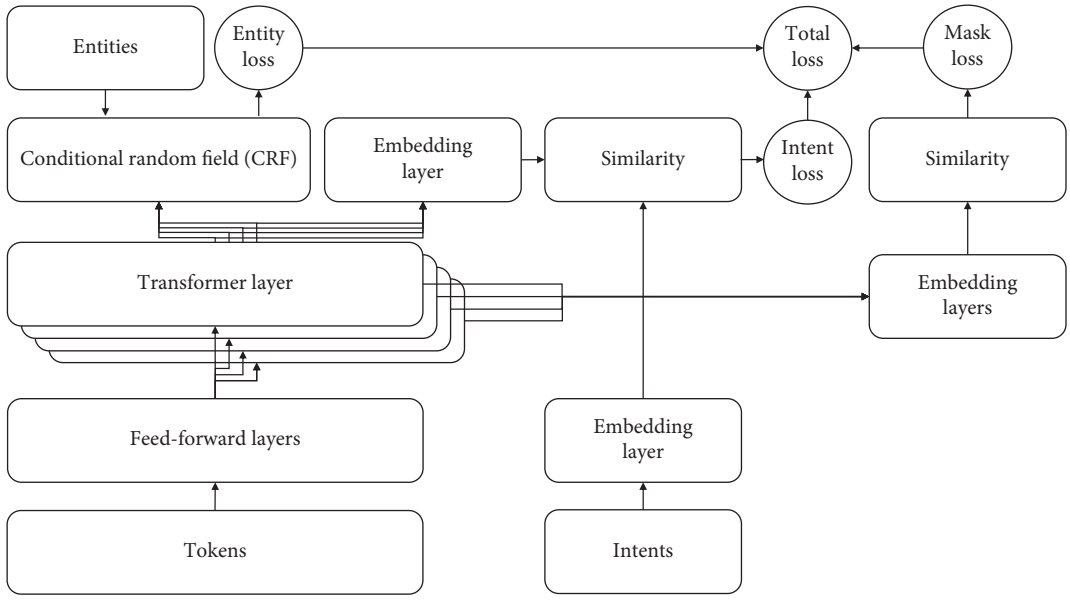


Figure 4: Architecture of DIET-Opt (KoRASA).

Table 2: Parameter comparison between DIET-Base and DIET-Opt (KoRASA).

| Parameter | DIET-Base | DIET-Opt (KoRASA) |
|---|---|---|
| Epoch | 300 | 500 |
| The number of transformer layers | 2 | 4 |
| Transformer size | 256 | 256 |
| Masked language model | True | True |
| Drop rate | 0.25 | 0.25 |
| Weight sparsity | 0.8 | 0.7 |
| Embedding dimension | 20 | 30 |
| Hidden layer size | (256, 128) | (512, 128) |

total of 487 datasets were generated with the following six components: greeting (Greet), closing (Goodbye), menu (Food_Menu), department contact (Dept_Contact), division of work (Pers_Work), and calculator (Calc) (see Table 3).

For the entity extraction experiment, a total of 3,354 datasets were generated with the following seven entities: date (Date), department (Dept), work (Work), name (Name), time (Time), number (Num), and absence of entity (No_Entity) (see Table 4).

*4.2. Preprocessing.* As shown in Figure 2, the preprocessing of the NLU can be divided into tokenization and featurization stages. In terms of tokenization, the performance of intent classification and entity extraction was measured by applying three tokenizers: the WhiteSpace Tokenizer, ConveRT Tokenizer, and Korean version of Mecab [18, 27, 32]. The count vector featurizer (CV), fallback classifier, and DIET-Base, which are the three baseline models recommended by RASA, were used as algorithms for featurization, intent classification, and entity extraction [18]. The tokenizers were compared with respect to the intent

Table 3: Intent dataset.

| Number | Intent | Meaning | Count |
|---|---|---|---|
| 1 | Greet | Greeting | 8 |
| 2 | Goodbye | Closing | 6 |
| 3 | Food_Menu | Menu | 53 |
| 4 | Dept_Num | Department contact | 60 |
| 5 | Pers_Work | Division of works | 355 |
| 6 | Calc | Calculator | 5 |

Table 4: Entity dataset.

| Number | Entity | Meaning | Count |
|---|---|---|---|
| 1 | Date | Date | 100 |
| 2 | Dept | Department | 246 |
| 3 | Work | Work | 640 |
| 4 | Name | Name | 54 |
| 5 | Time | Time | 13 |
| 6 | Num | Number | 20 |
| 7 | No_Entity | Absence of entity | 2,281 |

classification and entity extraction performance. The intent classification performance was similar to that of the tokenizer. With regards to entity extraction, however, the WhiteSpace Tokenizer and the ConveRT Tokenizer had an accuracy of 0.073 and $F1$-score of 0.050, whereas the Mecab tokenizer exhibited significantly better performance with an accuracy of 0.989 and an $F1$-score of 0.959 (see Table 5).

For featurization, a comparative experiment was conducted by applying the following three featurizers: Regex Featurizer, Lexical Syntactic Featurizer (LS), and Count Vector Featurizer (CV), while the Korean version of the Mecab tokenizer (Mecab (Ko)) was being applied [18]. Combinations of these featurizers were applied, and the top three performance results were charted. The entity extraction performance of the Count Vector Featurizer was 0.905 ($F1$-score), which was at least 0.3% higher than those of the

Table 5: Tokenization.

| Model | Intent classification | | | | Entity extraction | | | |
|---|---|---|---|---|---|---|---|---|
| | Train | | Test | | Train | | Test | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| WhiteSpace | 1.000 | 1.000 | 0.984 | 0.981 | 0.085 | 0.073 | 0.073 | 0.050 |
| ConveRT | 1.000 | 1.000 | 0.982 | 0.978 | 0.085 | 0.073 | 0.071 | 0.047 |
| Mecab (Ko) | 1.000 | 1.000 | 0.980 | 0.985 | 0.993 | 0.989 | 0.959 | 0.905 |

Table 6: Featurization.

| Model | Intent classification | | | | Entity extraction | | | |
|---|---|---|---|---|---|---|---|---|
| | Train | | Test | | Train | | Test | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| Regex + LS + CV | 1.000 | 1.000 | 0.980 | 0.980 | 0.988 | 0.089 | 0.949 | 0.882 |
| Regex | 1.000 | 1.000 | 0.980 | 0.978 | 0.993 | 0.989 | 0.959 | 0.902 |
| CV | 1.000 | 1.000 | 0.980 | 0.985 | 0.993 | 0.989 | 0.959 | 0.905 |

combined Regex + LS + CV Featurizer and the Regex Featurizer (see Table 6).

*4.3. Intent Classification and Entity Extraction.* A comparative experiment was performed to identify the optimal number of epochs for constructing the DIET-Opt model of KoRASA (see Figure 5). The accuracies and *F*1-scores for intent classification and entity extraction were analyzed when the model was trained with 100 to 900 epochs. In the experiment, when the epoch number was set to 500, the accuracy and *F*1-score of intent classification were found to be 0.982 and 0.984, respectively, and those of entity extraction were found to be 0.974 and 0.947, respectively.

The intent classifier of KoRASA proposed in this study was also compared with other intent classifiers on its intent classification and entity extraction performance. To this end, an experiment was conducted for the keyword classifier, fallback classifier, and DIET-Base classifier models (see Table 7) [18, 29]. The performance evaluation results revealed that the *F*1-score of KoRASA was 19.8%, 0.3%, and 0.4% higher in intent classification and 3.9%, 4.2%, and 3.7% higher in entity extraction than those of the keyword classifier, fallback classifier, and DIET-Base classifier, respectively.

An example of the results of intent classification conducted on the experimental data is shown in the following confusion matrix (see Table 8). After performing a total of six different types of intent classification, the greeting (Greet) and the closing (Goodbye) datasets were recognized as being composed of similar words, meaning that it was difficult to classify intent between those datasets. However, the intents of some of the other datasets, such as menu (Food_Menu), department contact (Dept_Num), and calculator (Calc), were completely classified.

In addition, the entity extractor of KoRASA was compared with other entity extractors. An experiment was conducted to compare the performance of KoRASA with CRF, a combination of DIET-Base and CRF (DIET-Base + CRF), and DIET-Base (see Table 9) [29, 33]. The

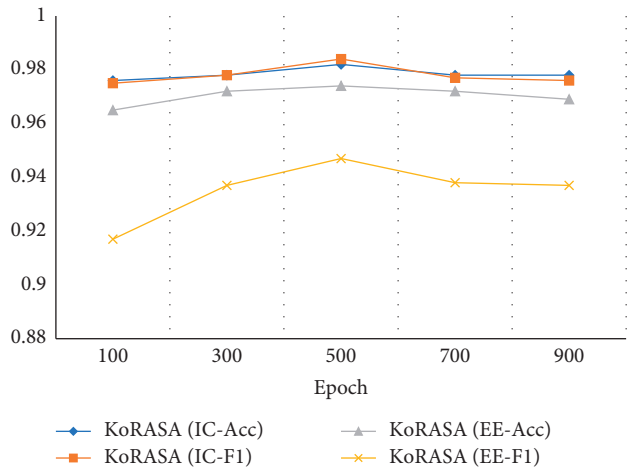


Figure 5: Performance comparison of epoch in KoRASA (IC: intent classification; EE : entity extraction).

Table 7: Performance comparison of intent classification.

| Model | Intent classification | | | | Entity extraction | | | |
|---|---|---|---|---|---|---|---|---|
| | Train | | Test | | Train | | Test | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| Keyword | 0.975 | 0.982 | 0.782 | 0.786 | 0.993 | 0.989 | 0.957 | 0.908 |
| Fallback | 1.000 | 1.000 | 0.980 | 0.981 | 0.993 | 0.989 | 0.959 | 0.905 |
| DIET-Base | 1.000 | 1.000 | 0.982 | 0.980 | 0.993 | 0.989 | 0.962 | 0.910 |
| KoRASA | 1.000 | 1.000 | 0.982 | 0.984 | 0.993 | 0.989 | 0.974 | 0.947 |

experimental results revealed that the *F*1-score of KoRASA was 1.5%, 1.2%, and 0.4% higher in intent classification and 4.2%, 4.3%, and 3.7% higher in entity extraction than those of CRF, DIET-Base + CRF, and DIET-Base, respectively.

Exemplary results from the experimental evaluation of entity extraction conducted are shown in the confusion matrix in Table 10. The results from seven different types of entity extraction show that our model demonstrated high accuracy for numerical datasets such as date (Date) and time

Table 8: Result example of confusion matrix for intent classification.

| True/predict | Greet | Goodbye | Food_Menu | Dept_Num | Pers_Work | Calc |
|---|---|---|---|---|---|---|
| Greet | 6 | 2 | 0 | 0 | 0 | 0 |
| Goodbye | 2 | 4 | 0 | 0 | 0 | 0 |
| Food_Menu | 0 | 0 | 53 | 0 | 0 | 0 |
| Dept_Num | 0 | 0 | 0 | 60 | 0 | 0 |
| Pers_Work | 1 | 0 | 0 | 1 | 353 | 0 |
| Calc | 0 | 0 | 0 | 0 | 0 | 5 |

Table 9: Performance comparison of entity extraction.

| Model | Intent classification | | | | Entity extraction | | | |
|---|---|---|---|---|---|---|---|---|
| | Train | | Test | | Train | | Test | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| CRF | 1.000 | 1.000 | 0.974 | 0.969 | 0.993 | 0.989 | 0.964 | 0.905 |
| DIET-Base + CRF | 1.000 | 1.000 | 0.976 | 0.972 | 0.993 | 0.989 | 0.964 | 0.904 |
| DIET-Base | 1.000 | 1.000 | 0.982 | 0.980 | 0.993 | 0.989 | 0.962 | 0.910 |
| KoRASA | 1.000 | 1.000 | 0.982 | 0.984 | 0.993 | 0.989 | 0.974 | 0.947 |

Table 10: Result example of confusion matrix for entity extraction.

| True/predict | Date | Dept | Work | Name | Time | Num | No_Entity |
|---|---|---|---|---|---|---|---|
| Date | 100 | 0 | 0 | 0 | 0 | 0 | 0 |
| Dept | 0 | 242 | 0 | 0 | 0 | 4 | 0 |
| Work | 4 | 4 | 567 | 2 | 0 | 0 | 63 |
| Name | 0 | 0 | 20 | 29 | 0 | 0 | 5 |
| Time | 0 | 0 | 0 | 0 | 13 | 0 | 0 |
| Num | 5 | 0 | 0 | 0 | 0 | 15 | 0 |
| No_Entity | 0 | 2 | 23 | 3 | 0 | 0 | 2253 |

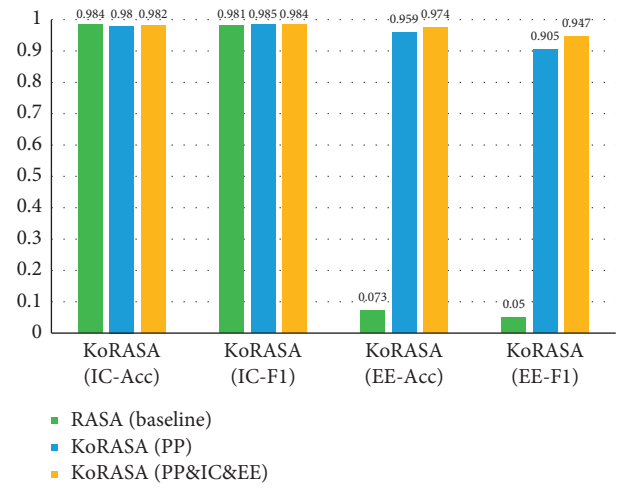


Figure 6: Performance comparison (PP: preprocessing; IC: intent classification; EE: entity extraction).

(Time), as well as for proper nouns such as department (Department) and work (Work). Alternatively, the performance with the name (Name) dataset was lower in terms of accuracy than with the other datasets.

To understand the optimization performance of each stage, we compared the experimental results between (1) the baseline model of RASA; (2) the optimized preprocessing model, which included tokenization and featurization stages (KoRASA(PP)); and (3) the model with optimized preprocessing, intent classification, and entity extraction stages, i.e., the model applying DIET-Opt (KoRASA(PP&IC&EE)) (see Figure 6). Compared with the baseline model of RASA, the results of the KoRASA (PP) model demonstrated comparable performance for intent classification. In contrast, entity extraction performance increased by 88.6% (accuracy) and 85.5% ($F1$-score). Finally, the model in which KoRASA (PP&IC&EE) was applied demonstrated similar results for intent classification. Contrarily, the entity extraction performance increased by 90.1% (accuracy) and 89.7% ($F1$-score).

## 5. Conclusion

This paper proposed KoRASA as a pipeline-optimization method for a deep learning-based open-source chatbot framework designed to understand the Korean language. KoRASA is a closed-domain chatbot applicable to most industries in Korea. It consists of four stages (tokenization, featurization, intent classification, and entity extraction) and is optimized to understand the Korean language. The tokenization stage uses the Korean version of the Mecab tokenizer, and the featurization stage adopts the Count Vector featurizer. For the intent classification and entity extraction stages, this study developed the DIET-Opt model by parameter tuning and optimizing the DIET-Base model. According to our experimental results, our model demonstrated an accuracy of 98.2% and an $F1$-score of 98.4% for intent classification. These scores were 19.8%, 0.3%, and 0.4% higher than those of the keyword classifier, fallback classifier, and DIET-Base classifier, respectively, which are popular existing intent classifier algorithms. Regarding entity extraction, the experimental results demonstrate that our model had an accuracy of 97.4% and an $F1$-score of 94.7%. Thus, the DIET-Opt model had 1.5%, 1.2%, and 0.4% better performance than those of the CRF, DIET-Base + CRF, and DIET-Base models, respectively, which are popular existing entity extraction algorithms. When the

results of intent classification and entity extraction are charted using the Confusion Matrix, the improved performance of our proposed method was confirmed across the different datasets, with the exception of the greeting (Greet), closing (Goodbye), and employee name (Name) datasets, which are recognized as being composed of similar words. Therefore, we confirmed the necessity of constructing an accurate dataset for effective chatbot training. The results of our experiment demonstrate that KoRASA is useful not only for developing a closed-domain chatbot for the Korean language, but also for various industrial areas such as mobile chatbot services, RPA interworking services, edge computing, Internet of Energy (IoE), and smart grids. We will apply KoRASA to RPA Chatbot and optimize its tokenization method using word embedding for Korean language. In addition, we will construct an open dataset and develop an automatic optimization function to adapt to additional datasets.

## Data Availability

No data were used to support this study.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this study.

## Acknowledgments

This work was funded by the Korea Electric Power Corporation (KEPCO).

## References

[1] K. Shinichiro, *CHATBOT: AI To Robot No Shinka Ga Henkakusuru Mirai*, Sotechsha Co.,, Tokyo, Japan, 2016.

[2] A. M. Rahman, A. A. Mamun, and A. Islam, "Programming challenges of chatbot: current and future prospective," in *Proceedings of the IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, pp. 75–78, Dhaka, Bangladesh, December 2017.

[3] H. J. Kang and S. I. Kim, "Evaluation on the usability of chatbot intelligent messenger mobile services -focusing on google (allo) and facebook (M messenger)," *Journal of the Korea Convergence Society*, vol. 8, no. 9, pp. 271–276, 2017.

[4] E. Adamopoulou and L. Moussiades, "Chatbots: history, technology, and applications," *Machine Learning with Applications*, vol. 2, 2020.

[5] J. Cahn, *CHATBOT: Architecture, Design, and Development*, 2017.

[6] D. Gwendal, C. Jordi, D. Laurent et al., "Xaikit: a multimodal low-code chatbot development framework," *IEEE Access*, vol. 8, pp. 15332–15346, 2020.

[7] A. M. Addi and L. Xjang, "KBot: a knowledge graph based ChatBot for natural language understanding over linked data," *IEEE Access*, vol. 8, pp. 149220–149230, 2020.

[8] Y. M. Park, S. W. Kang, and J. G. Seo, "An efficient framework for development of task-oriented dialog systems in a smart home environment," *Sensors*, vol. 181581 pages, 2018.

[9] S. Reshmi and K. Balakrishnan, "Empowering chatbots with business intelligence by big data integration," *International Journal of Advanced Research in Computer Science*, vol. 9, no. 1, 2018.

[10] L. Deng and Y. Liu, *Deep Learning in Natural Language Processing*, Springer, Berlin, Germany, 2018.

[11] A. Radford, K. Narasimhan, Tim Salimans et al., "Improving language understanding by generative pre-training," 2018, https://s3-us-west-2.amazonaws.com/openaiassets/research-covers/language-unsupervised/language_understanding_paper.pdf URL.

[12] X. Liu, P. He, W. Chen et al., "Multi-task deep neural networks for natural language understanding," 2019, http://arxiv.org/abs/1901.11504v2.

[13] X. Jiao, Y. Yin, L. Shang et al., "TinyBERT: distilling BERT for natural language understanding," 2020, http://arxiv.org/abs/1909.10351v5.

[14] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*, pp. 1–11, Long Beach, CA, USA, 2017.

[15] Microsoft, https://luis.ai, 2021.

[16] IBM, https://www.ibm.com/watson/developercloud/conversation.html, 2021.

[17] Google, https://www.api.ai, 2021.

[18] Rasa Technologies Inc., https://rasa.com, 2021.

[19] D. Braun, A. H. Mendez, F. Matthes et al., "Evaluating natural language understanding services for conversational question answering systems," in *Proceedings of the SIGDIAL 2017 Conference*, pp. 174–185, Saarbrücken, Germany, August 2017.

[20] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[21] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescape, "Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 445–458, 2019.

[22] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Toward effective mobile encrypted traffic classification through deep learning," *Neurocomputing*, vol. 409, pp. 306–315, 2020.

[23] M.-H. Hwang, I.-T. Lee, C.-H. Chae, and N.-J. Jung, "Gen2Vec: deep learning based distributed representation framework of words and documents for diagnostic services of power generation facility," *The Transactions of The Korean Institute of Electrical Engineers*, vol. 69, no. 12, pp. 1808–1815, 2020.

[24] H.-Z. Wang, G.-Q. Li, G.-B. Wang, J.-C. Peng, H. Jiang, and Y.-T. Liu, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Applied Energy*, vol. 188, pp. 56–70, 2017.

[25] C. Wang, P. Lin, C. Cheng et al., "Detecting potential adverse drug reactions using deep neural network model," *Journal of Medical Internet Research*, vol. 21, no. 2, pp. 1–12, 2019.

[26] D. Roman, S. Saxena, V. Robu, M. Pecht, and D. Flynn, "Machine learning pipeline for battery state-of-health estimation," *Nature Machine Intelligence*, vol. 3, no. 5, pp. 447–456, 2021.

[27] T. Kudo, *Mecab: Yet Another Part-Of-Speech and Morphological Analyzer*, https://sourceforge.net/projects/mecab/, 2012.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] B. Tanja, V. Daksh, V. Vladimir et al., "DIET: lightweight language understanding for dialogue systems," 2020, http://arxiv.org/abs/2004.09936v3.
[30] V. Vladimir, E. M. M. Johannes, and N. Alan, "Dialogue transformers," 2020, http://arxiv.org/abs/1910.00486v3.
[31] Rasa Blog., https://blog.rasa.com, 2021.
[32] M. Henderson, C. Inigo, M. Nikola et al., "ConveRT: efficient and accurate conversational representations from transformers," *Findings of the Association for Computational Linguistics: EMNLP*, vol. 2020, pp. 2161–2174, 2020.
[33] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: probabilistic models for segmenting and labelling sequence data," in *Proceedings of the 18th International Conference on Machine Learning(ICML)*, pp. 282–289, Williamstown, MA, USA, 2001.