

# Semantic Ambiguity Detection in Sentence Classification using Task-Specific Embeddings

Jong Myoung Kim<sup>1,2</sup> Young-Jun Lee<sup>2</sup> Sangkeun Jung<sup>3</sup> Ho-Jin Choi<sup>2</sup>

<sup>1</sup>SK-telecom <sup>2</sup>School of Computing, KAIST


<sup>3</sup>The Division of Computer Convergence, Chugnam National University

[jmkim71@sk.com](mailto:jmkim71@sk.com) {[yj2961](mailto:yj2961@kaist.ac.kr),[hojinc](mailto:hojinc@kaist.ac.kr)}@kaist.ac.kr [hugman@cnu.ac.kr](mailto:hugman@cnu.ac.kr)

## Abstract

Ambiguity is a major obstacle to providing services based on sentence classification. However, because of the structural limitations of the service, there may not be sufficient contextual information to resolve the ambiguity. In this situation, we focus on *ambiguity detection* so that service design considering ambiguity is possible. We utilize similarity in a semantic space to detect ambiguity in service *scenarios*<sup>1</sup> and *training data*. In addition, we apply task-specific embedding to improve performance. Our results demonstrate that ambiguities and resulting labeling errors in training data or scenarios can be detected. Additionally, we confirm that it can be used to debug services.

## 1 Introduction

The ability to accurately access the meanings of sentences is a key component of voice recognition-based agent services. This task is made difficult by the inherent ambiguity of some sentences, which can refer to different meanings in different contexts. **Adot**()<sup>2</sup> is a voice recognition-based service agent, akin to Amazon’s Alexa, for Korean users. This paper describes how we dealt with ambiguity, from a perspective of **Adot**’s developer.

For example, as illustrated in Figure 1, the Korean word "앞" has two different meanings depending on the context. As a result, when the media content is being played, the command "앞으로 가봐" may have two completely opposite meanings ("move forwards" vs. "move to previous"). In a test conducted within our company, approximately 61.7 percent of respondents interpreted it as “move forwards” and 38.3 percent as “move to previous”.

<sup>1</sup>A scenario refers to one expected input sentence and the analysis result of it predefined for service design.

<sup>2</sup>**Adot** is a virtual assistant service developed by SK Telecom, a telecom company in South Korea. Though officially denoted as **A.**, we will refer to it as **Adot** in this study to avoid misunderstanding.

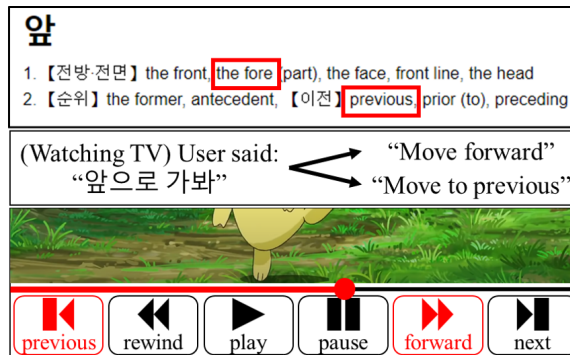


Figure 1: The various meanings of Korean word "앞" and the semantic ambiguity between user utterances.

Although there is no universally valid interpretation in this scenario, each respondent was certain that their own interpretation was the correct one. (Likert-scale score = 5.2/7). This semantic ambiguity leads to a labeling error of training data and scenarios in specification<sup>3</sup>. From the perspective of service providers who must provide specific services in response to user input, we contend that this is a challenging problem. In addition, ambiguity is very important in terms of cost. Manual inspection of the training data or specification is required for ambiguity handling, which is expensive in terms of cost and resources.

Recently, many researchers have attempted to handle ambiguity with various approaches such as semantic space (Rodd, 2020), entity linking (Yin et al., 2019), and attribute attention (Liu et al., 2019). In addition, the primary purpose of these studies is to resolve ambiguity using context. In spite of this, we do not have sufficient contextual information to apply these disambiguation methods because of the characteristics of a very short input sentence (which has 6.97 Korean letters and 2.05 words on average) and a one-turn based service.

Although contextual information required for disambiguation is not available within the scope

<sup>3</sup>We call the set of scenarios as the specification.

of the sentence classifier, we can still exploit non-linguistic traits, such as user habits and device information, which are accessible from the outside classifier to determine service. Therefore, we chose to approach this problem through detection rather than disambiguation. Detecting the presence of ambiguity in scenarios would allow for the consideration of external contextual information when providing the service.

This paper proposes a process of detecting ambiguity in scenarios or training data through similarity. To improve ambiguity detection performance, we apply task-specific embedding. We conduct an experiment detecting ambiguity and labeling error from ambiguity in **Adot**'s scenarios and training data. Additionally, we investigate Pearson correlation coefficients between similarity and the degree to which users expect two sentences to belong to the same class (this is referred to as user-aware class relevance). This study's findings affirm that similarity provides a means to identify scenarios and training data with potential ambiguity. In this process, we confirm that user-aware class relevance correlates with the similarity. Furthermore, it was confirmed that the training data that resulted in the misclassification could be specified.

## 2 Related Works

Ambiguity is a long-standing problem in natural language processing (NLP) tasks such as word sense disambiguation (Navigli, 2009), entity disambiguation (Barba et al., 2022), and database search result disambiguation (Qian et al., 2021) in the task-oriented dialogue systems. Ezzini et al. (2021) utilized domain-specific data to address the structural ambiguity of sentences. This ambiguity is largely divided into four categories (Berry et al., 2003).

- Lexical ambiguity occurs when a word has several meanings
- Syntactic ambiguity occurs when a given sequence of words can be given more than one grammatical structure
- Semantic ambiguity occurs when a sentence has more than one way of reading it
- Pragmatic ambiguity occurs when a sentence has several meanings in the context in which it is uttered.

These studies are focused on resolving the ambiguity. However, we did not have contextual information to apply to these disambiguation methods.

In NLP, the sentence similarity task, which evaluates the similarity between pairs of sentences using several techniques, has been investigated extensively. Traditionally, edit distance has been used to quantify superficial similarities (Levenshtein et al., 1966). Despite being straightforward, it is noteworthy because of its similarity to manual identification of training data that induce misclassification. Recent studies have measured the similarity between pairs of sentences by projecting them into a meaningful semantic space using various neural networks, such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018), and BERT (Devlin et al., 2018), or using Siamese architectures, such as SentenceBERT (Reimers and Gurevych, 2019).

In this paper, inspired by (Ezzini et al., 2021), We utilize a task-specific embedding, extracted from a trained classification model, to identify semantic ambiguities.

## 3 Task Design

### 3.1 Ambiguity Detection in Scenarios

When ambiguity exists in scenarios, the accurate classification result is not aligned with the user's expectations. If the user expects a class (class- $B$ ) other than the class (class- $A$ ) defined in the specification to be correct for an input  $a$ , it implies that classification of the input under class- $B$  is sufficiently valid or that the user has experienced receiving a service corresponding to class- $B$  for at least one input  $a'$  similar to  $a$ . Let's suppose that the command "앞으로 가봐" we looked at in section 1 is defined as "move forward" in the specification. A significant proportion, 38.3%, interpret this command as "move to previous", and the similar phrase "앞으로 다시 가봐" is commonly used in Korea with the meaning "move to previous". (The interjected word '다시' usually means 'again'.) From the perspective of the user, we attempted to identify the ambiguity of the scenarios.

First, pairs of sentences are collected, and the correlation between similarity and user-aware class relevance is evaluated. If this correlation is sufficiently high, user-aware class relevance could be estimated based on similarity. In a given scenario, if specific training data  $y$  that has the highest similarity with a particular scenario sentence  $x$  is placed

in a different class than the class- $X$  defined for scenario sentence  $x$ , it implies that the user may not anticipate class- $X$  for sentence  $x$  and ambiguity may exist in the scenario. Subsequently, ambiguity or the mislabeled scenario is manually inspected in the specified scenarios to validate this process.

### 3.2 Ambiguity Detection in Training Data

The method of 3.1 could not be applied as it is to detect the ambiguity of the training data. In the process of generating training data, several similar data are generated all at once. Ambiguous training data is rarely observed in isolation; furthermore, the data demonstrate a marked degree of similarity to one another. Finding similar data with different classes for single data did not work well.

We noted that the ambiguity in training data manifests as mislabeled data that do not conform to the specification. Models trained using this mislabeled data yield outputs that deviate significantly from the designer’s expectation. In this paper, we begin with misclassified inputs and attempt to find the training data that caused the corresponding misclassification and mistraining.

Hence, input sentences misclassified by **Adot** are collected. Among all training data with misclassified labels, data exhibiting high similarity with misclassified inputs are assumed to be training data that induce misclassification. These include corresponding data with ambiguity or those mislabeled from ambiguity. We inspect the specified data manually to confirm any ambiguity or labeling errors. Additionally, the labels of the specified data are modified to the labels of the expected classes, and the model is retrained. It is verified whether the classification results of the model are corrected as expected. Via this, we confirm whether the specified data are the cause of the misclassification.

## 4 Methods - Task-Specific Embeddings

We note that each node of the trained classification model stores weights adjusted for the task, and the values output by each node in the inference process contain fragmentary information useful for inference (Wang et al., 2020). We create task-specific embedding vectors using the output values of each node in the classification model. Via this process, we expect to be able to create an embedding with better expressive power for ambiguity detection, albeit biased, compared to off-the-shelf embeddings, such as pre-trained LM or Word2Vec. Assuming a

classification model  $f_\theta$  with  $n$  layers on top of the embedding layer, we create a vector  $v_i$  representing the  $i$ -th layer as follows: ( $v_1$  = embedding vector,  $v_n$  = model output vector.)

**Whole Layer Vector (WLV)** WLV consists of output of all nodes from the model’s embedding layer to the layer below the final output layer. WLV contains the most amount of data and reflects all information flows in the model.

$$WLV = \text{concat}(v_1, v_2, \dots, v_{n-2}, v_{n-1})$$

**After Representation Layer Vector (ARLV)** WLV contains considerable information, but the embedding layer accounts for most of it, which is problematic (for instance, 25,640 nodes out of a total of 40,424 nodes are included in the embedding layer of **Adot**’s classification model). ARLV is defined using nodes from the layer following the embedding layer to the layer immediately preceding the output layer to avoid undue influence of the embedding layer.

$$ARLV = \text{concat}(v_2, v_3, \dots, v_{n-2}, v_{n-1})$$

**Conclusion Layer Vector (CLV)** Although WLV and ARLV contain a significant amount of data, they ignore discrete functions and consequently struggle with the corresponding differences. For example, when a layer uses an activation function such as RELU, the difference in the value input to the function might be negligible. However, the resultant value after processing the function might be completely different. CLV is only defined using the layer immediately preceding the output layer of the entire model to account for these characteristics. We anticipate that the overall model’s judgment process is structured in this layer.

$$CLV = v_{n-1}$$

## 5 Experiments

### 5.1 Datasets

#### 5.1.1 Scenario Ambiguity Test Set

The following experiment utilizes the task described in section 3.1 to detect scenario ambiguities from the user’s perspective.

**Correlation Test Dataset** 913 pairs of sentences are collected from domain classification validation data for the **Adot** service (Domain information is used as a label in this experiment). Domains, in

the context of **Adot**, are defined service areas, and include categories consisting of 30 categories such as general, music, video, weather, and schedule. Sentences are randomly selected from the data of eight domains that frequently result in classification failure. The classes of each pair may be identical or different. We measure Pearson correlation between similarity and user-aware class relevance of sentence pair.

**Ambiguity Test Dataset** To evaluate ambiguity detection, 29,136 sentences with designer-defined labels are collected. These sentences are scenarios defined for the **Adot** service. For these scenarios of specification, the scenarios with possible ambiguity are detected using the process mentioned in section 3.1. Lastly, any actual ambiguity or labeling error corresponding to the specified scenarios is manually inspected.

### 5.1.2 Training Data Ambiguity Test Set

The following experiment based on the task designed in section 3.2 is performed to detect ambiguity in training data.

**Ambiguity Test Set** This test dataset consists of 2,300 sentences incorrectly classified by **Adot**. Based on the misclassified sentences, training data exhibiting high similarity with these sentences are assumed to be training data that caused the misclassification. We manually check for ambiguity or labeling errors in those specific sentences.

**Side-Effect Test Set** Identification of data that induces misclassification also requires that inputs analyzed successfully before correction must remain so after correction. To confirm this, 20,000 sentences classified correctly by **Adot** are collected, and their classification result is verified after the specified data are modified and retrained.

## 5.2 Baseline - Similarity Methods

**Edit Distance** A method for measuring the superficial similarity. It is very similar to the manual debugging process.

**Embedding Vector Similarity** A method for measuring the semantic similarity. Cosine similarity of embedding vectors for sentences is measured. Two types of representative off-the-shelf embeddings are prepared: Sent2Vec and KoGPT2. Sent2vec, was trained on both Korean Wikipedia and **Adot**'s training data. KoGPT2 is a pre-trained language model developed by SKT-AI, based on

GPT-2. These embeddings are also used to comprise an embedding layer for the classification models discussed below. We describe the embedding methods in detail in the Appendix A.

## 5.3 Classification models used in the Experiments

We employed classification models to generate task-specific embeddings and assess the detection performance of training data that cause misclassification.

**Adot Classifier** This is the model used for domain classification in **Adot**'s service. It is formed by concatenating a convolutional neural network (CNN) that uses a part of speech-tagged morpheme as a token with another CNN that uses character and dictionary information as a token followed by three fully connected layer on top.

**KoGPT2 Classifier** It is a classifier based on pretrained-LM. As in the basic classifiers of pre-LM packages, three fully connected layers are added on top of KoGPT2.

These models are trained using training data for domain classification of **Adot**' service. The data consists of 2.2 M sentences with 30 corresponding classes. We set the batch size to be 256 and fine-tune the model for 15 epochs using a learning rate of (1e-5). On the development dataset, the **Adot** and KoGPT classification models achieved accuracy rates of 98.4% and 94.2%, respectively. Because presenting a high-performance model is not within the scope of this study, a detailed description of the model is included in the Appendix A.2.

## 5.4 Human Evaluation

The task designed in section 3.1 includes the user test. The evaluator rates the degree to which two sentences belong to the same class on a scale of 1–7. An example of the test set given to the evaluator is included in the appendix B.2. For this experiment, two groups of evaluators are recruited.

**Ordinary Service User (User)** This group consists of six ordinary **Adot** users. Each user exhibits above-average linguistic proficiency and possesses a bachelor's degree or higher academic qualifications. Through correlation with this group, we checked whether user-aware class relevant could be estimated with similarity.

**Expert Evaluator (Expert)** This group consists of seven people with experience in generating training data or evaluating quality of service for **Adot**.

Each participant exhibits five years of annotation experience on average and demonstrates a thorough understanding of class boundaries.

## 5.5 Evaluation Metrics

Experimental performance is evaluated in terms of specific metrics. If the data requires correction but is not ambiguous, it is considered to be a simple mislabeling error. In our experiment, the terms commonly used to define the formulation for evaluation metrics are  $N_x$ ,  $s$ , and  $d$ , where  $N_x$ ,  $s$ , and  $d$  denote the number of  $x \in \{s, d\}$ , scenario or training data identified as having an ambiguity, and input data, respectively.

**Ambiguous Data Ratio (ADR)** measures the ratio of scenario or training data containing ambiguity in the identified scenarios or training data,  $N_{s_a}/N_s$ , where  $N_{s_a}$  and  $N_s$  denote the number of  $s_a$  and the number of  $s$ , respectively.  $s_a$  represents the scenario or training data with actual ambiguity among  $s$ . The existence of ambiguity was confirmed by a person directly checking if multiple interpretations were possible.

**Mislabeled Data from the ambiguity Ratio (MDR)** measures the ratio of scenario or data containing ambiguity and labeling error in the identified scenario or data,  $N_{s_m}/N_s$ .  $s_m$  denotes scenario or training data with actual ambiguity and labeling error among  $s$ . Mislabeled Data ( $s_m$ ) was verified by a person directly checking the need for label correction.

**Correction Ratio (CR)** Measure the classification success rate for retrained model on previously misclassified sentences after modifying data and retraining,  $N_{d_f}/N_{d_{f_s}}$ .  $d_f$  represents input data misclassified by **Adot**, while  $d_{f_s}$  denotes data correctly classified after correction and retraining from within the  $d_f$ .

**Accuracy preservation Ratio (APR)** measures the classification success rate for previously successfully analyzed sentences after modifying the data and retraining,  $N_{d_{s_s}}/N_{d_s}$ .  $d_{s_s}$  denotes input data from the  $d_s$  that was correctly classified after correction and retraining, while  $d_s$  represents data successfully classified by **Adot**.

## 5.6 Results

**Pearson Correlation Analysis** As shown in Table 1, we show the result of pearson correlation analysis between user-aware class relevance and

the measured similarities. In general, various similarity methods achieve significant positive correlations<sup>4</sup>, which suggests that similarity methods can be used as a proxy estimator for user-aware class relevance. This result is regarded as an evidence of our hypothesis described in Section 3. This result reinforces our hypothesis that presenting data with high similarities but differing labels can aid in identifying ambiguous scenarios. Moreover, similarity methods more correlates with users than experts who know the service criteria accurately. Following (Pang et al., 2020), we also measure the mean inter-rater agreement. The study results indicate high agreement and yield two conclusions. Firstly, the methods utilized, particularly **Adot-CLV**, demonstrate a correlation almost equivalent to the average inter-rater correlation (0.589 vs. 0.633, respectively). Secondly, the manual evaluation explanation can be considered sufficient.

Similarity with ordinary users was significantly higher than with experts. To ascertain the underlying cause of this phenomenon, we computed the concordance of each sentence pair at three distinct points, namely the *service area*, *entity properties*, and *predicate*. We investigated the correlation between manual evaluation and concordances. Notably, the expert group demonstrated a markedly high correlation in the service area, whereas the ordinary user group exhibited comparable correlations across all three areas. Further elaboration on this matter is presented in the appendix C, as it is not directly pertinent to the primary focus of our paper, namely ambiguity detection.

**Specification Ambiguity Detection** As shown in Table 1, we presents the ambiguity and labeling errors resulting from ambiguity detected in the scenarios. Since the specification is the gold standard for services, it is mostly comprised of scenarios that have a clear meaning. Nonetheless, we can identify scenarios that contain ambiguity or require label modification. Interestingly, the number of data between containing actual ambiguity ( $(s_a) = (N_s \times ADR)$ , KoGPT2+ARLV- $(s_a) = 210$ , **Adot**+ARLV- $(s_a) = 152$ ) or requiring modification ( $(s_m) = (N_s \times MDR)$ , Sent2Vec- $(s_m) = 40$ , **Adot**+ARLV- $(s_m) = 36$ ) are not different significantly considering the number of scenar-

<sup>4</sup>In the case of Edit Distance, which takes the value 0 when completely identical and 1 when completely different, the sign is the opposite of that of the other similarity measurement methods.

Task →	Correlation			Ambiguity in specification			Ambiguity in training data			
	All ↑	Expert ↑	User ↑	$N_s$	ADR ↑	MDR ↑	ADR↑	MDR↑	CR↑	APR↑
Edit Distance	-0.243	-0.184	-0.286	755	0.170	0.033	0.660	0.373	0.559	0.990
Embedding										
+ Sent2Vec	0.217	0.154	0.275	4448	0.037	0.005	0.612	0.315	0.496	0.992
+ KoGPT2	0.268	0.155	0.394	1531	0.223	0.024	0.611	0.318	0.490	0.990
Task-Specific										
<b>Adot</b> +WLV	0.237	0.170	0.294	2968	0.056	0.007	0.610	0.316	0.474	0.992
<b>Adot</b> +ARLV	<u>0.565</u>	<u>0.468</u>	<u>0.604</u>	<u>362</u>	<u>0.419</u>	<u>0.102</u>	<b>0.712</b>	<u>0.458</u>	<b>0.689</b>	0.989
<b>Adot</b> +CLV	<b>0.589</b>	<b>0.489</b>	<b>0.628</b>	<b>243</b>	<b>0.469</b>	<b>0.132</b>	<u>0.711</u>	<b>0.463</b>	0.599	0.992
KoGPT2+WLV	0.336	0.227	0.440	1210	0.169	0.020	0.599	0.332	0.491	0.990
KoGPT2+ARLV	0.512	0.445	0.513	755	0.278	0.039	0.688	0.439	<u>0.601</u>	0.989
KoGPT2+CLV	0.512	0.445	0.513	507	0.391	0.047	0.694	0.448	0.600	0.990
Inter-Rater	0.633	0.752	0.520	-	-	-	-	-	-	-

Table 1: **Correlation**: Pearson’s correlation coefficient performance between similarity and user-aware class relevance. **Ambiguity in specification**: Identical performance of ambiguous scenario detection. **Ambiguity in training data**: Identical performance of ambiguous training data detection. The model with the best performance is indicated in **bold**, while the second best is underlined.

ios (29,136). During the scenario ambiguity detection experiment, we found a noticeable contrast in the number of identified scenarios, denoted by  $N_s$ . Task-specific embedding tended to have a lower  $N_s$  than off-the-shelf embedding. Furthermore, the upper layer-generated embedding displayed a lower  $N_s$ .  $N_s$  determined the performance difference in ADR and MDR.

**Training Data Ambiguity Detection** Table 1 shows the performance in training data ambiguity detection. We verify that the ambiguity and labeling error is viewed as the cause of misclassification. In practice, approximately 70% of the specified data contain ambiguity, and approximately 45% require label correction. CR and APR represent the performance of identifying training data that cause misclassification. The results also demonstrate that similarity can be used to identify training data that cause misclassification. Moreover, it is confirmed that ARLV and CLV of task-specific embeddings are more suitable for the detection task.

**Effect of Task-specific Embedding** As reported in Table 1, task-specific embeddings outperform other methods in both ARLV and CLV, while WLV does not achieve significant improvement. Possibly, this is because off-the-shelf embeddings occupy the majority of WLV, as discussed in the definition of ARLV (See in Section 4). To understand why this phenomenon happens, we visualize the

representation of training data samples obtained from each embedding method using T-SNE. As illustrated in Figure 2, the task-specific embeddings (i.e., ARLV and CLV) clearly represent the training data samples compared to general embeddings.

The task-specific embeddings based on the **Adot** model outperformed those based on KoGPT2. However, we do not consider the transformer-based embedding to be unsuitable for the task. The observed difference in performance appears to stem from **Adot**’s specialization in classifying this data, given that **Adot** has undergone extensive optimization as part of its use in the corresponding service. The differences in classification accuracy between the **Adot** and KoGPT2 models, as mentioned in the section 5.3, support this interpretation (98% vs. 94%). Observing Figure 2, one can discern a subtle yet distinct data segregation demonstrated by the **Adot**-based embeddings as compared to those based on KoGPT2. This differential expressive capacity seems to manifest as a performance discrepancy. In the case of general embeddings, KoGPT2 exhibits greater separation than Sent2Vec, which, in practice, is reflected by the overall superior performance of Embedding-KoGPT2 compared to Embedding-Sent2Vec.

## 6 Discussions

**Powerful Task-Specific Representation** This study demonstrates the feasibility of detecting am-

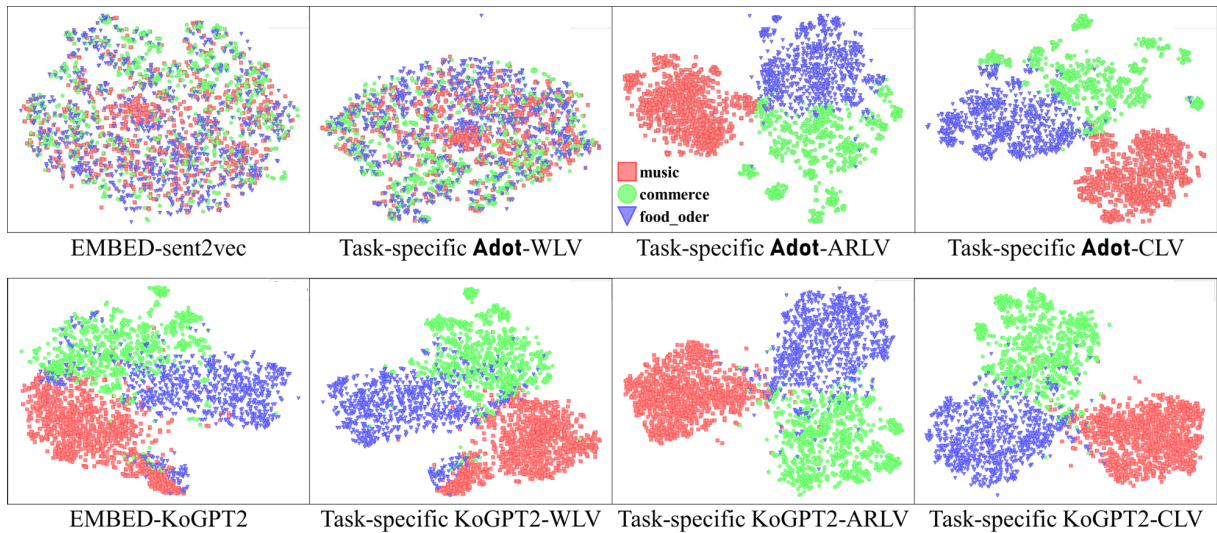


Figure 2: Data representation in the semantic space corresponding to three domains (red: music, green: commerce, blue: food\_order). We can observe that the boundaries of the represented data become clear depending on the location where the vector is generated.

biguity through similarity and highlights the potential benefits of utilizing task-specific embedding to enhance performance. The experimental results show a positive correlation between model size and detection performance according to the scaling law. Future research efforts for ambiguity detection will focus on generating task-specific embeddings using larger models for superior results.

**Multilingual Ambiguity Detection** This study was conducted for Korean-based services. Most users of **Adot** are Korean, and the inputs contain many Korean characteristics. And since the commands input to the smart speaker were the data of the study, fairly short and imperative sentences are targeted. Experimentation is required to determine if similar results can be obtained in other languages and other types of sentences.

## 7 Conclusion

We introduce two processes to detect training data and scenarios that induce ambiguity. Moreover, task-specific embedding is adopted to improve detection performance. Comprehensive analysis reveals that compared to off-the-shelf embeddings, Sent2Vec and KoGPT2, task-specific embedding is much more suitable for ambiguity detection. Moreover, the results demonstrate the viability of identifying training data that cause misclassification. In the future work, we aim to compare the performance of task-specific embedding using a wider range of models and node selection methods.

## Acknowledgements

This research was conducted based on the experience of developing and operating the **A.** (📍) service by SK Telecom in South Korea. This research was supported and funded by the Korean National Police Agency. [Project Name: XR Counter-Terrorism Education and Training Test Bed Establishment/Project Number: PR08-04-000-21]. This work was supported by Chungnam National University.

## References

- Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2022. Extend: extractive entity disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2478–2488.
- Daniel M Berry, Erik Kamsties, and Michael M Krieger. 2003. From contract drafting to software specification: Linguistic sources of ambiguity. *Univ. of Waterloo, Tech. Rep.*
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Saad Ezzini, Sallam Abualhajja, Chetan Arora, Mehrdad Sabetzadeh, and Lionel C Briand. 2021. Using domain-specific corpora for improved handling of ambiguity in requirements. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 1485–1497. IEEE.

- Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.
- Yang Liu, Jishun Guo, Deng Cai, and Xiaofei He. 2019. Attribute attention for semantic disambiguation in zero-shot learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6698–6707.
- Chris McCormick. 2016. Word2vec tutorial-the skip-gram model. *Apr-2016.[Online]. Available: <http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model>*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.
- Bo Pang, Erik Nijkamp, Wenjuan Han, Linqi Zhou, Yixian Liu, Kewei Tu, et al. 2020. Towards holistic and automatic evaluation of open-domain dialogue generation.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Kun Qian, Ahmad Beirami, Satwik Kottur, Shahin Shayandeh, Paul Crook, Alborz Geramifard, Zhou Yu, and Chinnadhurai Sankar. 2021. Database search results disambiguation for task-oriented dialog systems. *arXiv preprint arXiv:2112.08351*.
- Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.
- Jennifer M Rodd. 2020. Settling into semantic space: An ambiguity-focused account of word-meaning access. *Perspectives on Psychological Science*, 15(2):411–427.
- Zijie J Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Duen Horng Polo Chau. 2020. Cnn explainer: learning convolutional neural networks with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1396–1406.
- Xiaoyao Yin, Yangchen Huang, Bin Zhou, Aiping Li, Long Lan, and Yan Jia. 2019. Deep entity linking via eliminating semantic ambiguity with bert. *IEEE Access*, 7:169434–169445.



	<b>Adot</b>	KoGPT2 fine tuning
python Version	3.6.9	3.6.10
Tensorflow Ver.	1.15.2	-
Pytorch Ver.	-	1.6.0a0+9907a3e
epoch	15	15
batch size	256	256
learning rate	(1e-5)	(1e-5)

Table 2: Development environments

## A Implementation Details

### A.1 Development Environment

We experimented on NVIDIA A100 (40GB). The **Adot** model was implemented and trained within the Keras and Tensorflow environments, while fine-tuning of KoGPT was conducted in the PyTorch framework.

### A.2 Classification Models

The average **Adot** user only uses six Korean characters in an utterance, and there are typically 5 million calls per day. Owing to these characteristics, a lightweight and simple model capable of stable CPU prediction was required. Convolutional Neural network (CNN) based models are currently used in **Adot**. This domain classification model was created using Y. Kim’s CNN for sentence classification (Kim, 2014). It has a model concatenated with character-based CNN and CNN models based on part-of-speech (POS) tagged morphemes reflecting the Korean features. The character-based CNN model uses the dictionary-based embedding layer as a single channel to use the content domain’s dictionary information. The configuration of the fully connected layer on top of the convolutional layer is also slightly different from the model in our study. Figure 3 shows an outline of **Adot**’s domain classification model.

### A.3 Embeddings

We represented a sentence data in the semantic space through two methods.

#### A.3.1 Sentence Representation Using Word Embedding

Word embedding is a representative method of expressing meaning in a semantic space. We developed two kinds of embeddings, character-level embedding and POS tagged morpheme-level embedding, to express sentences. A sentence is repre-

sented by concatenating it through two embeddings. Our embedding developed with skip-gram and negative sampling based on McCormick (2016) tutorial. Table 4 shows the detailed parameters of the embeddings we used. These parameters were determined empirically to maximize the performance of the **Adot**’s domain classifier.

These embeddings were trained from Korean Wikipedia sentences. Korean Wikipedia is a set of documents written by many people and reflects the popular usage of the Korean language by Koreans. However, sentences on the Wikipedia are long, while the utterances coming into the **Adot** service are relatively short, colloquial, and that there are many content object names. To reflect this part, the embedding trained from the wiki were tuned with the training data of **Adot**.

#### A.3.2 Sentence Representation Using KoGPT2

GPT-2 is a natural language processing model that uses machine learning algorithms to generate input sample text into text with syntactic, grammatical, and informational consistency. KoGPT-2, an open source-based GPT-2 model trained in Korean. Character Byte Pair Encoding (KBPE) was used as tokenizer. Korean Wikipedia, Modu Corpus, and the Blue House National Petition and private data like news were used as training data. When a sentence passes through the KoGPT2, the output values from the last layer were used as sentence representation.

### A.4 Training Data

The training data of domain classifier of **Adot**’s was used as the training data of this experiment. **Adot** encompasses thirty service domains, including ‘general’, ‘video’, ‘music’, ‘schedule’, and ‘weather’. A person directly constructed about 2.2 million training data according to the design of the service. The data is primarily Korean sentences. Some foreign language sentences are also included. The table5 includes an extensive description of the model and training data. The data used in this experiment is from the December 2021 version, and it’s important to note that its structure and content differ from the current **Adot** service data.

## B Testset Details

### B.1 Designer-Annotator testset

To improve performance, we have collected utterance logs from **Adot** users and used them for accu-

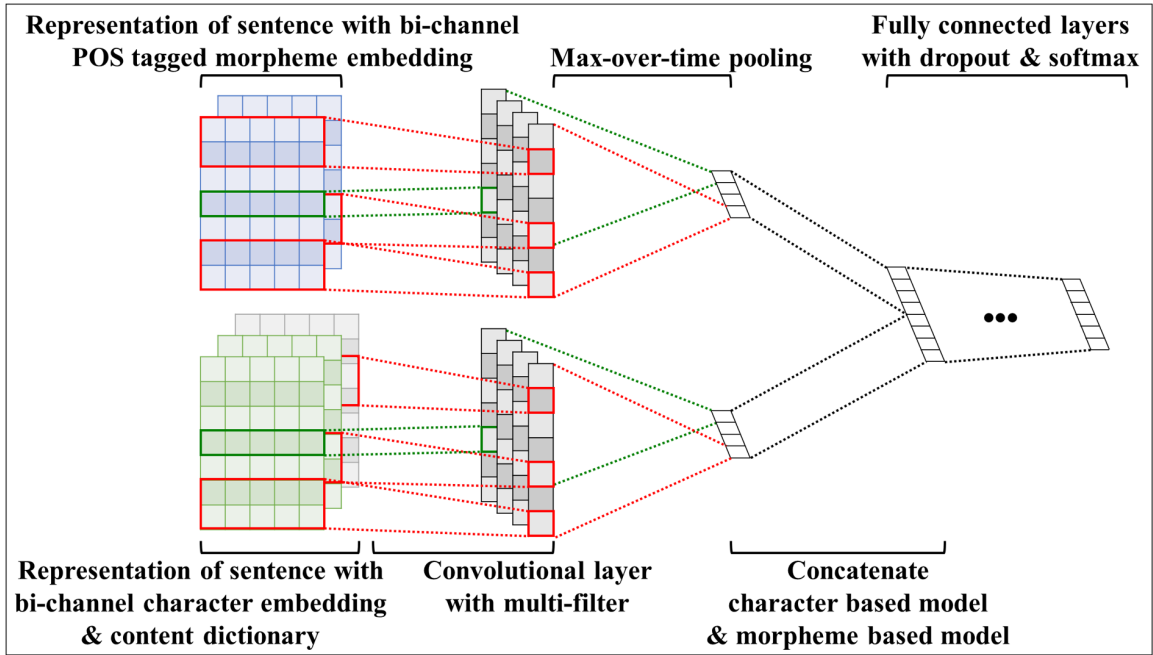


Figure 3: Overview of **Adot**'s domain classification model

	shape	objective
input 1	(None,25)	POS-tagged morpheme
input 2	(None,40)	Korean character
input 3	(None,40,107)	Named Entity & Content Entity Dict. Info
output	(None,30)	domain classification result

Input & Output Information of Classification Model

Total params	16,016,544
Trainable params	16,016,330
Non-trainable params	214
Total layer Numbers	41

Parameter & Layer Information of Classification Model

Table 3: **Adot**'s Domain Classification Model Training Information

racy evaluation. A test set was constructed using the utterances input in April 2022.

**Consensus Error& Ambiguity testset** Among those utterances, utterances that were classified incorrectly were targeted. To prevent data modification that occurred in both directions of the classes, only utterances with the correct class “general” were collected. The “general” class is for device manipulation such as volume control in **Adot**. We constructed 2,300 test cases. Because these data are misclassification data collected from the **Adot** classification model, only the **Adot** classification model was applied to this experiment.

**Side-effect test set** This set is a test set to evaluate the side effects caused by label modification. Through modification, even if it is possible to correctly classify misclassified utterances, the classification results of utterances that have been accurately analyzed should not be changed. We collected 20,000 utterances that the current classifier accurately analyzes and expect the data to be analyzed accurately even after modification.

## B.2 Scenario ambiguity testset

We built two test sets to detect consensus errors between designers and users.

	character-level embedding	morpheme-level embedding
token	Character (Korean, English, numbers)	POS tagged morpheme
Dictionary size	1638	9391
embedding space	256	256
skipgram window size	3	3
negative sampling rate	1	1

Table 4: Parameters of Two Kinds of Embeddings

There is sentence 1 and sentence 2.

Please rate how confident you are that the two sentences  
(should) belong to the same domain on a scale of 1-7.

A domain is a unit for grouping and processing similar services.

Score Criteria  
1: Completely different domains (cannot be viewed as the same domain)  
7: Exactly the same domain (must be the same domain)

Each sentence belongs to one of the eight domains below.  
general / schedule / commerce / food\_delivery / video / news / radio

No.	sentence 1	sentence 2	1	2	3	4	5	6	7
1	“Order an apple.”	“Order a banana.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
2	“Order an americano.”	“check payment method.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
3	“check payment method.”	“Add toppings to pizza.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
...	...	...	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
911	“Make a playlist.”	“Add toppings to pizza.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
912	“check payment method.”	“Pay for Youtube music.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
913	“Order a banana.”	“Make a playlist.”	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure 4: Human evaluation test set example. The evaluation was conducted in Korean; a more detailed explanation is attached.

training data size	about 2.2M
domain number	30
average character size	16.77
average word size	4.50
average morpheme size	8.56
average named entity numbers	1.1
average domain entity numbers	1.9

Table 5: Training Data Information of **Adot**'s Domain Classifier. Dec. 2021 version

**Correlation testset** First, sentence pairs were collected to confirm the correlation between similarity and the degree to which the user thinks the classes are consistent. 913 pairs of sentences were collected from validation data for the **Adot** service. Among them, test sets were collected from eight domains in which domain classification frequently fails: ‘commerce,’ ‘food,’ ‘general,’ ‘news,’ ‘schedule,’ ‘video,’ ‘music,’ and ‘radio.’ Like a domain classifier, domain information was used as a label for classification. Each sentence pair is randomly collected within the verification data of 8 domains and may have the same class or a different class. Figure 4 and 5 shows example of testset.

similarity	scenario	entity	predicate
Expert	0.720	0.380	0.104
User	0.487	0.476	0.307
<b>Adot-CLV</b>	0.357	0.443	0.249

Table 6: Pearson correlation coefficients between manual evaluation results and the three features

**Ambiguity detection testset** To test consensus error detection between designers and users, 29,136 sentences in which designers defined labels were collected. These sentences are specifications defined for **Adot** service scenarios and defined by the designer of **Adot** service. For this Spec, the sentences in which Consensus error may occur were indicated through the method mentioned in A. It was confirmed whether there was an actual ambiguity or consensus error for the specified sentences.

### C Expert-ordinary user correlation comparison

With ordinary users, similarity was significantly higher than with experts. To determine the reason behind this phenomenon, concordance is estimated at three points for each sentence pair—service area, entity properties, and predicate. In addition, the correlation between manual evaluation and concordances is estimated. The results are presented in Table 6. Although the expert group exhibits a very high correlation in the service area, the ordinary user group exhibits similar correlations in all three areas. Let us consider the pair, “Play Netflix” and “Play Spotify” Ordinary users may believe that the classes are similar because of the similarity of the predicates. In contrast, experts believe that Netflix and Spotify provide different services (video vs. music)—so there is no agreement irrespective of the predicate. The results confirm that the similarity is more similar to the judgment of ordinary users than that of experts. Consequently, similarity is deemed to be useful in detecting in designer-user consensus errors.

문장 1과 문장 2가 있습니다.

두 문장이 같은 도메인에 속한다고 (혹은 속해야 한다고)  
 확신하는 정도를 1-7점으로 표현해 주십시오

도메인이란 서비스를 묶어서 처리하기 위한 서비스의 Topic 입니다.

각 문장들은 아래 8개의 도메인에 속해 있습니다.  
 general / schedule / commerce / food\_delivery / video / news / radio

점수의 기준은 다음과 같습니다.

1: 완전히 다른 도메인이다 (같은 도메인으로 볼 수 없다)  
 7: 완전히 같은 도메인이다 (같은 도메인임에 틀림 없다)

문장 1	문장 2	평가
내일이 9월 15일까지	21년 1월 23일에	
18번 주문 예약 캔슬해줘	5번째 메뉴 열판 스물 주문해	
볼륨 5로 라디오 틀어볼래	이전 회차 동영상 틀자	
중요하다고 했던 왕십리 오늘 일정 좀 실행해봐	이번주 화요일에서 금요일까지 설정된 알람 삭제 요청할래 다	
화요일 목요일 오후 10시에 반복 알람해달라고	월 새벽 4시에 알람 반복에 이동국	
알림 헬프	추천 어플	
다음 회차 틀어줘봐	알람 소리 틀어줘	
왼쪽 목록 좀 봐봐	타이머 버튼 실행해	
블루베리피자	블루베리 담아줘	
금 주 쇼핑하자	치킨 슈퍼파파스 시키자	
아래로 두번 이동	서치	
노래 중에서 죽겠다 유튜브에서 좀 검색해주겠니	노래 우울한 편지로 SETTING 부탁한다	
소녀시대 우기 목소리로 알람 설정 해봐봐	월 새벽 4시에 알람 반복에 이동국	
목록 전에 것으로 볼거야	지금 시간 볼 거야	
베이컨체더치즈 곡물 도우 피자	크레딧 카드로 결제	
다크 나이트 재생하렴	빠르게 재생하자	
그럼 낮	밤	
요즘 NEWS TOPIC 재생 GO GO	THE LAST WORD 재생해줘	
카툰릭 평화방송 틀어줘	가툰릭 평화방송 들려줘	
옥수수 KIDS 재생 고고	옥수수키즈 어떻게 써	
다음 항목으로 이동하기	너 사용법틀어줘	
내일 여덟시에 생생가스 오야고동 세 개배달 시켜줄래	이달 마지막 주 금요일에 주문했던 거 배달 현황 확인해	
대각선 오른쪽 아래쪽으로 이동해줘	타이머 벨소리모드 버튼 클릭	
뉴스 볼륨 10퍼센트 줄이자	티비 볼륨 좀 조용했으면	
또다른 상품 조회	BBQ 치킨 추천 상품 얼마예요	
99 9999 퍼센트	93 1 라디오로 깨워해볼래	
GUATEMALA CITY 시간	엘라스트 원준로 할게	
취침예약해놓은 것 좀 취소시켜줘	취침예약 하려면 어떻게 해	
배달 상태 체크하고 싶거든	배달 부탁할게	
2시간 2분 건너뛰고 들려줘	다음주 월요일 며칠인지 듣고싶어	
지금 이 채널 좋아요 목록에서 없애주라	화음에 사람들이 좋아하는 채널 커	
한국 총선때 스프스에서 한 뉴스 재생해봐	요즘 내가 봤던 뉴스 틀어봐줘요	

Figure 5: Human evaluation test set example. The evaluation was conducted in Korean; a more detailed explanation is attached.