



## Ensemble Learning Approach to Chatbot Design Based on Paraphrase Detection

H. Toprak KESGİN<sup>1,\*</sup> , Onur ÖZTUNÇ<sup>2</sup> , Banu DİRİ<sup>3</sup> 

<sup>1</sup> Department of Computer Engineering, Yildiz Technical University, İstanbul, Turkey, **ORCID:** 0000-0001-7554-1387

<sup>2</sup> Department of Computer Engineering, Yildiz Technical University, İstanbul, Turkey, **ORCID:** 0000-0001-8597-2770

<sup>3</sup> Department of Computer Engineering, Yildiz Technical University, İstanbul, Turkey, **ORCID:** 0000-0002-6652-4339

### Article Info

#### Research paper

Received : November 2, 2022

Accepted : February 9, 2023

### Keywords

Chatbot  
Question Answering  
Ensemble Learning  
Paraphrase Detection  
Chatbot Design

### Abstract

In this paper, we present a design for an ensemble chatbot based on paraphrase detection. Our proposed chatbot is intended to assist companies in reducing the need for costly call center operations by providing a 24-hour service to customers seeking information about products or services. Our algorithm is designed to work effectively on small data sets, such as an existing FAQ, and does not require a large number of instances. We evaluated the performance of our chatbot using publicly available data from the websites of major telecommunication companies and found that the ensemble model improved success rates by 6% compared to the single best model, with a top 3 accuracy of 84.54% and a top 1 accuracy of 70.10%.

## 1. Introduction

In recent years, many companies have established call centers to improve customer satisfaction by providing 24-hour service. However, this can be a significant financial burden for the companies. To address this issue, companies have turned to chatbots as a means of reducing the need for call centers. Chatbots can increase self-service engagement and handle a significant number of customer requests, reducing the time that customers spend while waiting for the information. Domain-specific chatbots tend to be more successful than general-purpose chatbots.

In this study, we focus on the telecommunications field and develop a chatbot designed as a question-answering system. We create a dataset of question-and-answer pairs specific to the telecommunications field to do this. While the study is conducted within this particular domain, the chatbot design could be applied to any field with a similar dataset.

The chatbot operates by taking in a customer's question and measuring its similarity to the questions

stored in the system. The answer to the most similar question is then returned to the customer as a response to their inquiry. To measure the similarity of the questions, we employ four different methods: cosine similarity of average sentence vectors, cosine similarity of weighted average sentence vectors, n-gram similarity of sentences, and word mover's distance. By combining all of these methods, we create an ensemble similarity measurement that outperforms any individual method. The proposed method is not limited to use in chatbot applications but can be utilized for any system requiring paraphrase detection.

The main contributions of this article to the literature are; the development of a novel ensemble chatbot for the telecommunications field that effectively reduces the need for costly call center operations by providing a 24-hour service to customers seeking information about products or services, the introduction of an ensemble model approach for measuring the similarity between questions, which demonstrates superior performance compared to individual methods and has the potential to be applied to a wide range of fields and systems requiring paraphrase detection, the design and evaluation of a chatbot that is able to achieve high accuracy and effectiveness with small datasets,

\* Corresponding Author: [tkesgin@yildiz.edu.tr](mailto:tkesgin@yildiz.edu.tr)



without the need for a large number of instances. This approach has the potential to significantly reduce the burden of data collection and annotation in chatbot development.

## 2. Literature Review

In this section, common techniques used in the design of chatbot models are presented. Many different techniques can be used to create a chatbot.

Template-based chatbots use pattern matching to answer user questions [1]. These systems are relatively easy to build and are a popular choice for small-scale applications [2]. Eslam et al. used BERT [3] vectors for chatbot design [4], training and testing their system on the SQuAD dataset [5]. Their workflow achieved 98% training and 96% test set accuracy. In a separate study, BERT was also used to develop a chatbot text classification system that aimed to reduce memory and storage needs by using a single model for multiple chatbots with multiple languages [6]. The proposed model demonstrated high accuracy and reduced system resources compared to individually trained models.

Transfer learning is widely used in the task of text classification, including in the context of chatbot intention classification. BERT and ERNIE (Enhanced Representation through Knowledge Integration) have been particularly effective in this regard, but their computational performance can be an issue in high QPS (queries per second) intelligent dialogue systems [7]. The authors propose using the knowledge of the ERNIE model to distill a FastText model, with the ERNIE model serving as a teacher model to predict massive online unlabeled data for data enhancement and guiding the training of the FastText student model. This approach not only maintains the FastText model's strong computational performance but also significantly improves its intention classification accuracy.

Recurrent Neural Networks (RNN) [8] have also been used in chatbot design, with one study proposing a system that actively interacts with the user to maximize their knowledge gain [9]. The proposed DeepProbe framework used a sequence-to-sequence (seq2seq) RNN and was evaluated using computer-based BLEU and AUC methods and human judge evaluation. Another study developed an ensemble-based dialogue system for the Amazon Alexa Prize competition, incorporating deep learning and reinforcement learning techniques, including seq2seq models and latent variable models for natural language processing [10].

Word embedding techniques, such as Word2vec and FastText, have also been applied to chatbot design.

Word2vec is an unsupervised and predictive method for representing words as vectors [11], with two deep learning algorithms: Continuous Bag of Words (CBOW) and Skip-gram. FastText, an improved version of Word2vec, also considers subword information while calculating word vectors [12].

Seq2seq is a technique that transforms an input sequence into an output sequence with a tag and attention value [13]. It consists of two RNNs: an encoder and a decoder. The encoder processes the input data and passes the last state of its recurrent layers to the decoder's first recurrent layers. The decoder then uses this initial state to generate the output sequence. Seq2seq has been used to analyze user input and generate a thought vector, which is then used to generate a response to the user. Martin Boyanov and Ivan Koychev proposed a system of answering questions in their work [14]. For this, they used the data collected from forums instead of dialogue question-and-answer datasets. The questions in the forums and the answers given to them were trained by using the seq2seq model. They used data collected from forums rather than dialogue question-and-answer datasets. The proposed system can answer correctly 49.5 % of the questions. In the questions asked in a completely new way, the accuracy rate decreases to 47.3%.

Chatbots can be used for many different applications. For example, a chatbot design consisting of 3 layers that analyzes and corrects Syntax errors in the Turkish language has been proposed [15]. In another study, a chatbot was proposed to be used in customer service management using the BERT model [16]. Another usage area of chatbots is recommendation systems. In another study, the seq2seq chatbot model was used in the Turkish venue recommendation system [17].

The reliance on large, labeled datasets and correspondingly large models is a common feature of effective chatbot systems. However, the acquisition and annotation of such data can be resource-intensive, and the cost of training large models can be high. In contrast, template-based or rule-based approaches that do not rely on extensive data may be less effective.

The use of chatbots in business marketing strategies has increased in recent years due to advances in artificial intelligence and the widespread acceptance of Internet and messaging platforms [18]. Also, it was found that the attitude of Turkish consumers toward chatbots has a positive effect on the intention to use chatbots [19]. In addition, the results of the research showed that the perceived usefulness and perceived ease of use of chatbots have a positive effect on the attitude toward chatbots.

We aim to advance the field of chatbot design and provide practical solutions for real-world applications, specifically addressing the need for cost-effective and

efficient customer service in the telecommunications industry. One of the key contributions of our research is the development of a novel ensemble model for measuring the similarity between questions, which not only outperforms individual methods for paraphrase detection in the context of our chatbot but has the potential to be widely applicable in any system that requires paraphrase detection. Additionally, we demonstrate the success of our chatbot using a small data set, which has the potential to significantly reduce the burden of data collection and annotation in chatbot development. Our research also differs from previous literature on chatbot design in that it focuses on a specific industry rather than general-purpose, and utilizes a domain-specific dataset of question-answer pairs. Overall, these contributions make our research a valuable addition to the field of chatbot design, particularly for applications that require paraphrase detection.

### **3. Materials and Methods**

To develop our chatbot, we used a dataset of question-answer pairs obtained from frequently asked question sections of major telecommunication companies. We preprocessed the collected dataset to improve the performance of machine learning algorithms and used the FastText method to create word vectors for the words in the dataset. We measured the similarities between the questions using various methods and combined the similarity ratios of these methods to form the final ensemble similarity measurement.

#### **3.1. Telecommunication FAQ Dataset**

To evaluate the performance of our chatbot, we constructed a dataset of question-answer pairs using publicly available data from the websites of major telecommunication companies. We collected frequently asked questions from the websites across all categories and organized them into a dataset containing 1887 question-answer pairs, each consisting of a question, an answer, and the company it pertains to. We did not include categories of questions in the dataset to allow users to communicate with the chatbot without selecting a specific category. To receive service, the user simply needs to select the company they wish to receive service from and write their question or problem. To evaluate the effectiveness of the system, test questions were created by independent users that are semantically equivalent to the existing questions but are worded differently and were not used in the training of the model.

#### **3.2. Preprocessing**

The dataset underwent preprocessing to improve the machine's performance. To achieve this, we converted all characters in the sentences to lowercase, removed URLs and numbers, punctuation marks, and special characters, stemmed all words in the sentences, removed stop words, and detected and merged multi-word expressions (phrases). These techniques were applied to the dataset in order to enhance its overall performance.

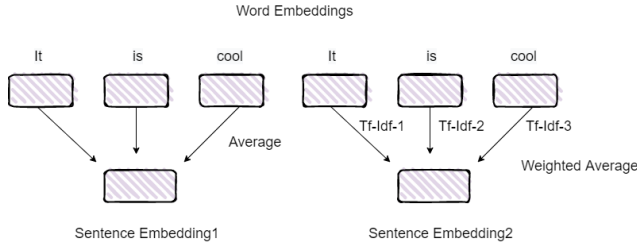
#### **3.3. Creating Word Embeddings**

After preprocessing, word embedding was used to represent each word as a vector in space. In this study, FastText was used to create word vectors using the questions-and-answers dataset as input. FastText does not require a labeled dataset and can be trained in an unsupervised manner. It has been shown to be particularly effective at capturing semantic similarity. The CBOW and Skip-gram algorithms, window size, and word embedding size for the FastText method were compared as hyperparameters.

#### **3.4. Creating Sentence Vector**

The dataset consists of questions and answers. In order to find the most similar question to the user's question in the dataset, the user's question is compared to all questions in the dataset. Sentence vectors are created to measure the similarity between sentences. FastText vectors were used to create the sentence vectors. The goal of the sentence vector is to have sentences that contain similar words close to each other in the vector space. Two different methods were used to construct the sentence vectors: the average method and the weight average method.

In the average method, the vectors of the words comprising a sentence are summed and divided by the number of words in the sentence. The weight average method, on the other hand, gives more weight to words that contribute more meaning to the sentence. To calculate this effect, the term frequency-inverse document frequency (TF-IDF) measure was used to determine the importance of a word in a collection or corpus [20, 21]. The TF-IDF values of the same word in each sentence were averaged to create a fixed TF-IDF value for each word. A summary of the process for creating sentence vectors is shown in Figure 1.



**Figure 1.** Creating Sentence Embedding

### 3.5. Sentence Similarity Methods

We calculated the similarity between the user-entered sentence and the questions in the dataset using the n-gram similarity method, the cosine similarity between sentence vectors for both average and weighted average sentence embeddings, and the reversed Word Mover's Distance method.

#### 3.5.1. Sentence Similarity Methods

The cosine similarity between the sentence vector of the user-entered question and the vector of each question in the dataset was calculated. The similarity is defined as the scalar product of two vectors divided by the product of the lengths of the two vectors, as shown in Equation 1, where  $t$  is the sentence vector of the user's question and  $e$  is the sentence vector of a question in the dataset.

$$\cos(t, e) = \frac{t \cdot e}{|t||e|} = \frac{\sum_{i=1}^n t_i e_i}{\sqrt{\sum_{i=1}^n (t_i)^2} \sqrt{\sum_{i=1}^n (e_i)^2}} \quad (1)$$

The similarity between the two sentences ranges from -1 to +1, with a value closer to +1 indicating a greater similarity.

#### 3.5.2. N-gram Similarity

The n-gram similarity method is a character-based string comparison technique. In this method, a string is divided into n-grams based on the specified value of n. The n-gram similarity is calculated by dividing the number of common n-grams in the two sequences by the total number of n-grams in both sequences. This is formally defined in Equation 2, where  $s1$  and  $s2$  are the n-gram sequences in sentence 1 and sentence 2, respectively.

$$\text{sim}(s1, s2) = \frac{2 * |\text{pairs}(s1) \cap \text{pairs}(s2)|}{|\text{pairs}(s1)| + |\text{pairs}(s2)|} \quad (2)$$

#### 3.5.3. Word Mover's Distance

The Word Mover's Distance method calculates the

distance between two sentences based on the word embeddings of the sentences. It has been shown to outperform many other methods in k-nearest neighbors' classification [22]. The method matches the two sentences using words or phrases that are closely related, even if they have no common words. The similarity is measured using word embeddings, and a word in one sentence may be expressed as a phrase in the other sentence. The Word Mover's Distance method handles this by merging words into phrases when necessary and calculating the distance by summing the distances of the nearest words.

For example, the distance between the sentences "Obama speaks to the media in Illinois" and "The President greets the press in Chicago" is 1.07, while the distance between "The band gave a concert in Japan" and the same first sentence is 1.63. This is because, after stop words are removed, the total distance of the matching closest words is higher in the second example.

### 3.6. Ensemble Method

Ensemble learning, which involves combining the predictions of multiple models, is a powerful technique for improving the performance of machine learning algorithms. Ensemble models are most effective when the individual models are both successful and diverse. In previous sections, we proposed four different methods for calculating sentence similarity: average sentence vector cosine similarity, TF-IDF weighted average sentence vector cosine similarity, n-gram similarity, and Word Mover's Distance. These methods approach the paraphrase detection problem in different ways.

The average sentence vector cosine similarity method is based on the assumption that the mean of the words in a sentence represents its meaning. The weighted average method takes a different approach by assigning higher weights to the most important words in the vector. The n-gram similarity method compares only n-grams and measures spelling differences between sentences, using a completely different approach from word embeddings. The Word Mover's Distance method, on the other hand, measures distance instead of similarity by matching the closest words or phrases in each sentence and adding the distances between them. This method is also distinct from the other approaches. The distance calculated by the Word Mover's Distance method is converted to similarity by taking its inverse.

Since our proposed method utilizes algorithms that make different assumptions and employ different techniques, combining the predictions of these algorithms significantly increases the success of the ensemble model.

## 4. Experiments

In this section, we describe our experiments, evaluation methodology, and results. We used the FAQ dataset described in Section 3.1 to evaluate the performance of our chatbot.

### 4.1. Measuring Performance

Measuring the model's performance requires questions that are not included in the training data. Therefore, we used questions that differ from the ones in the training dataset, but have the same meaning as one of them. The test questions were created by independent users. The accuracy of a question in a test dataset is calculated by computing similarities with each sentence in the training dataset. For the model to be successful, the sentence with the highest similarity value must be an equivalent sentence that has the same meaning as the sentence with the highest similarity value. After completion this process for all test sentences, the Top 1 accuracy of the model is determined.

In this measure, there is no difference between predicting the correct question in the second or third position and predicting the correct question in the last position. However, the Top n accuracy can also be important in chatbot-style applications. Therefore, we calculated the Top 1, Top 3, and weighted accuracy of the model in our experiments. Top 1 accuracy is whether the model answered the question correctly on its first prediction. The Top 3 accuracy is considered correct if the correct answer is among the first 3 answers of the model. In the weighted accuracy, if the model is correct on the first prediction, it scores 1; if it is correct on the second prediction, it scores 1/2, and so on until the 3rd prediction, that is, if it is correct on the third prediction, it scores 1/3. However, if the correct answer is not among the first 3 predictions, it is evaluated as 0. The mean of these scores is used as the weighted accuracy.

In this study, we propose a method for paraphrase detection that utilizes unsupervised techniques and does not require a labeled dataset containing examples of paraphrased and non-paraphrased sentences. Specifically, we use FastText and TF-IDF to generate sentence vectors, and then calculate the cosine similarity between these vectors and the distance of word carriers, and also use n-gram similarity to determine similarity between texts. All four of these methods are trained in an unsupervised manner, without the use of any label information. During testing, the most similar question to the query is identified and the answer to that question is provided to the user, eliminating the need for a dataset containing labeled

instances of paraphrased and non-paraphrased sentences. This approach allows for the identification of paraphrases without relying on supervised learning algorithms, which typically require a large number of labeled examples. Overall, our method offers a practical solution for paraphrase detection in real-world applications.

### 4.2. Experimental Results

In this section, we present an analysis of the chatbot's performance. The system has been tested using questions provided by independent users. Only one answer is allowed for each question in order to produce reliable test results. We searched only for the questions related to the company.

We optimized the parameters of the algorithms. The FastText algorithm used to generate word vectors has three hyperparameters. For the optimization of the FastText model parameters, we used vector size (50, 100, 200), training algorithm (Skip-gram, CBOW), and Window Size (2, 3, 4). In the n-gram similarity method, we tested different values of n (1, 2, 3, 4, 5).

#### 4.2.1. Stemming Parameter

Stemming is performed in the preprocessing step. Stemming is basically removing the suffix from a word and reducing it to its root word. In the preprocess stage, stemming is one of the parameters that most affect the success of the system. Tests were conducted to determine whether stemming was applied in parameter optimization.

In the optimization of the stemming parameter, skip-gram is used as the training algorithm with default parameters. Tests were performed for vector lengths 50, 100 and 200. The accuracy of the models created in cases where the stemming process is applied and not applied is shown in Table 1.

**Table 1.** Effect of Stemming

Stemming	Vector Size	Top 1 Acc	Top 3 Acc	Weighted Acc
Yes	50	62.89	80.41	72.44
Yes	100	<b>68.04</b>	83.51	<b>76.31</b>
Yes	200	67.01	<b>84.54</b>	76.07
No	50	59.79	79.38	70.23
No	100	60.82	78.35	71.76
No	200	61.86	80.41	72.72

Based on the accuracy values shown in Table 1, it appears that the stemming process improves performance. Therefore, stemming is applied in preprocessing step.

#### 4.2.2. Selection of Training Algorithm

FastText has 2 training algorithms, CBOW and Skip-gram. Choosing the training algorithm as CBOW or Skip-gram is a hyperparameter that directly affects the success of the training. In order to measure the performance of the CBOW and Skip-gram algorithms, experiments were performed with default hyperparameters and for vector sizes of 50, 100, 200. Stemming was also applied for training and test set. Table 2 shows the accuracy of the models created with the CBOW algorithm and the Skip-gram algorithm. Experiments show that Skip-gram algorithm performed better than CBOW algorithm. It is also shown that the Skip-gram algorithm gives better results when using a small dataset. For these reasons, we chose our algorithm as a Skip-gram.

**Table 2.** Selection of Training Algorithm

Training Algorithm	Vector Size	Top 1 Acc	Top 3 Acc	Weighted Acc
Skip-gram	50	62.89	80.41	72.44
Skip-gram	100	<b>68.04</b>	83.51	<b>76.31</b>
Skip-gram	200	67.01	<b>84.54</b>	76.07
CBOW	50	62.89	82.47	73.66
CBOW	100	63.92	81.44	73.61
CBOW	200	<b>68.04</b>	80.41	75.53

#### 4.2.3. Selection Window and Vector Size

The window size is another hyperparameter for FastText. One of the FastText assumptions is that the words around a word are the words associated with that word. This hyperparameter specifies the number of words to associate with the left and right sides of the word. For these experiments, the windows size selected as 2, 3, 4 tested for vector sizes of 50, 100, 200. The results are shown in Table 3. While window size 2 and vector size 100 gave the best results for top 1 accuracy, vector size 200 gave the best results for Top 3 and Weighted Accuracy.

**Table 3.** Selection Window and Vector Size

Window Size	Vector Size	Top 1 Acc	Top 3 Acc	Weighted Acc
2	50	62.89	77.32	72.19
2	100	<b>70.10</b>	82.47	77.01
2	200	69.07	<b>84.54</b>	<b>77.29</b>
3	50	62.89	80.41	72.44
3	100	68.04	83.51	76.31
3	200	67.01	<b>84.54</b>	76.07
4	50	63.92	80.41	72.83
4	100	62.89	81.44	73.41
4	200	64.95	<b>84.54</b>	75.05

#### 4.2.4. Selection Window and Vector Size

The results obtained from the experiments were used to determine the optimal parameters for the model. The model was retrained using the optimal parameters obtained from the experiments. The model was tested using the test set again. The results obtained from the tests are shown in Table 4.

**Table 4.** Test Results for Optimal Parameters

Model	Top 1 Acc.	Top 3 Acc.	Weighted Acc.
Ensemble	<b>70.10</b>	<b>84.54</b>	<b>77.29</b>
Cos Sim Av. SV	54.64	77.32	65.81
Cos Sim Wt. Av. SV	53.61	64.95	59.71
N-Gram Sim.	63.92	80.41	72.59
WMD	59.79	76.29	68.81

The abbreviations in Table 4 are explained as follows: "Cos Sim Av. SV" stands for "Cosine Similarity of Average Sentence Vectors," "Cos Sim Wt. Av. SV" stands for "Cosine Similarity of Average Weighted Sentence Vectors," "N-Gram Similarity" refers to the similarity between n-grams in the text, and "WMD" stands for "Word Movers Distance." "Ensemble" refers to the combination of four models and the final model. The performance of the ensemble model was compared to the performance of the individual models. The results of the tests show that the ensemble model has the best performance among the models. The Top 1 accuracy of the ensemble model is 70.10%, the Top 3 accuracy is 84.54%, and the weighted accuracy is 72.32%. The Top 1 accuracy of the individual models ranges from 53.61% to 63.92%. The Top 3 accuracy of the individual models ranges from 64.95% to 80.41%. The weighted accuracy of the individual models ranges from 59.71% to 72.59%.

As a result, it can be concluded that the ensemble model has a better performance compared to the individual models. The ensemble model has improved the success rate by 6% compared to the individual model with the highest accuracy.

## 5. Application

The user interface has been created for the chatbot model created and a demo of the chatbot is shown in this section. Chatbot application works with a web server. Question received from the user is transferred to the application via the web server. The answer that the application finds is transmitted to the user through the web interface. The overall architecture of the system can be seen in Figure 5.

In the Operators section, the user can select the company they want to ask the questions about. On the left side there are chatbot answers with a blue background, on the right side there are user questions with a green background. The user can ask the question they want to ask in the "message" section and click the button on the right to send his message. The web user interface of the chatbot application is shown in Figure 2.

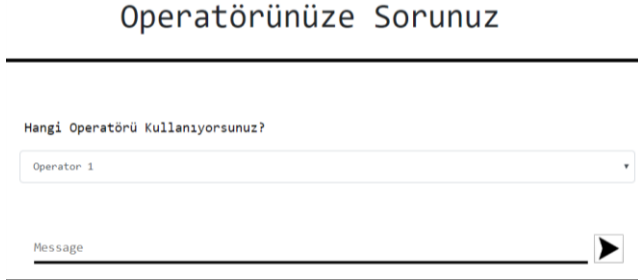


Figure 2. Web User Interface

The example in Figure 3 shows a successful example of how the system works. The user's question is shown on the right. Chatbot's answer is shown on the left. Likewise, if the user asks another similar question after the first, thinking that the user's problem cannot be solved, the system returns the second-best answer. An example of this is shown in Figure 4.

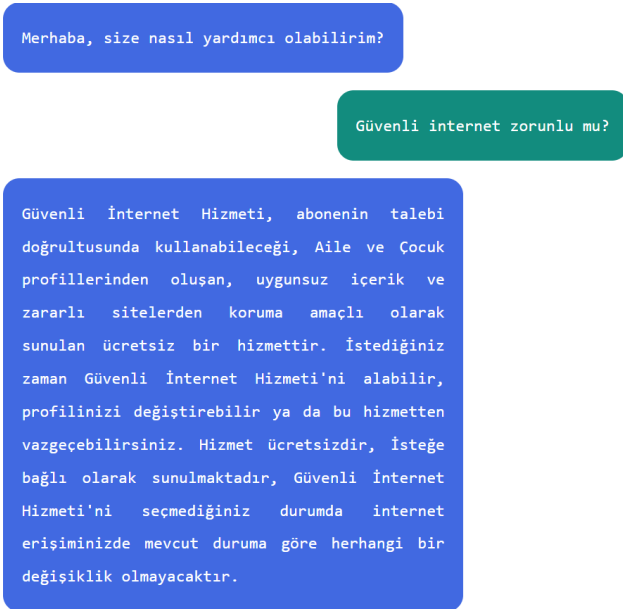


Figure 3. Web User Interface: Successful working example

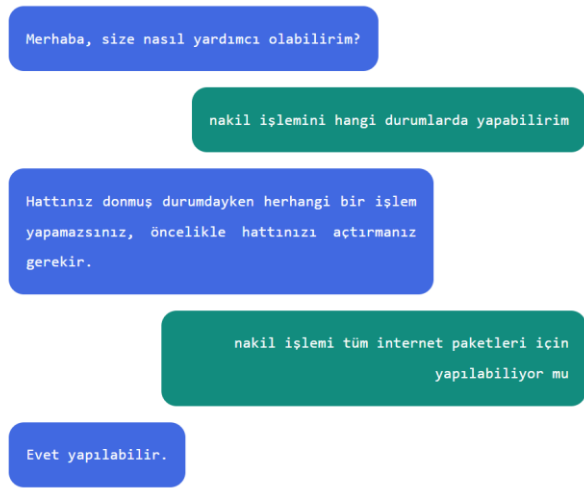


Figure 4. Web User Interface: Second best answer example

Furthermore, questions that are in the flow of conversation, such as greetings, are not in our database, these questions are kept separately and used as needed. Additionally, questions whose similarity to all sentences in the dataset is below a certain threshold will return an error message by the chatbot.

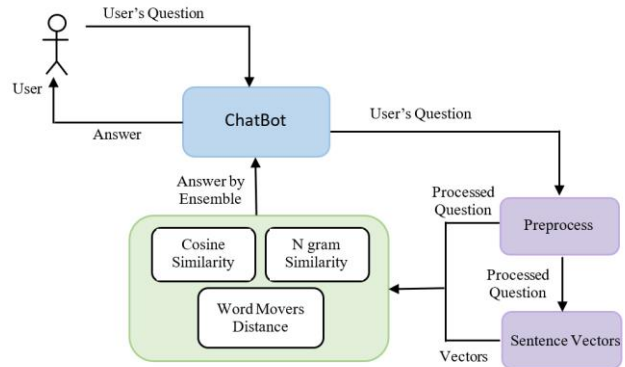


Figure 5. System Architecture

## 6. Conclusions

In this study, we aimed to show the answer to the user by measuring the similarity of the questions asked by the users to the questions in the system. To do this, we applied a detailed preprocessing on the dataset and the questions asked by the user.

We then used n-gram similarity and vectors created with the FastText model to measure these similarities. Sentence vectors were created using two different methods. With these sentence vectors, we calculated the cosine similarity of the user's question. In addition, we calculated the Word Mover's Distance using these vectors. Finally, we calculated n-gram similarities. The final similarity was measured using the results of these four methods together.



The proposed algorithm was tested with the prepared test questions for different parameters. According to two different accuracy measurement methods, the Top 3 accuracy was measured as 84.54% and the Top 1 accuracy was measured as 70.10% with the best parameters. These test results can be further improved by using a larger corpus with more texts and optimizing the parameters of the ensemble model.

It may take a long time to compare the user's question to all questions in a large dataset. In these cases, the category of the question can be determined with another model and the question can only be compared with questions within that category. The success of the ensemble model can be further improved by including the BERT vectors of the sentences in future studies.

### Declaration of Ethical Standards

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

### Conflict of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### References

- [1] AbuShawar B., Atwell E., 2015. ALICE chatbot: Trials and outputs. *Computación y Sistemas*, 19(4), pp. 625–632. Instituto Politécnico Nacional, Centro de Investigación en Computación.
- [2] Luo B., Lau R.Y.K., Li C., Si Y.W., 2022. A critical review of state-of-the-art chatbot designs and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12(1), pp. e1434. Wiley Online Library.
- [3] Devlin J., Chang M.W., Lee K., Toutanova K., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Paper presented at the North American Chapter of the Association for Computational Linguistics.
- [4] Amer E., Hazem A., Farouk O., Louca A., Mohamed Y., Ashraf M., 2021. A proposed chatbot framework for COVID-19. Paper presented at the 2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), IEEE, pp. 263-268.
- [5] Rajpurkar P., Zhang J., Lopyrev K., Liang P., 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. Paper presented at the Conference on Empirical Methods in Natural Language Processing.
- [6] Taşar D.E., Şükrü O., Kutsal S., Ölmez O., Gülüm S., Akca F., Belhan C., 2021. Performance Trade-Off for Bert Based Multi-Domain Multilingual Chatbot Architectures. *Journal of Artificial Intelligence and Data Science*, 1(2), pp. 144–149. Izmir Katip Celebi University.
- [7] Sak H., Senior A.W., Beaufays F., 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Paper presented at Interspeech.
- [8] Yin Z., Chang K.H., Zhang R., 2017. DeepProbe: Information Directed Sequence Understanding and Chatbot Design via Recurrent Neural Networks. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2131–2139. ACM.
- [9] Serban I., Sankar C., Germain M., Zhang S., Lin Z., Subramanian S., Kim T., Pieper M., Chandar A.P.S., Ke N.R., Mudumba S., de Brébisson A., Sotelo J.M.R., Suhubdy D., Michalski V., Nguyen A., Pineau J., Bengio Y., 2017. A Deep Reinforcement Learning Chatbot. *ArXiv*, vol. abs/1709.02349.
- [10] Mikolov T., Sutskever I., Chen K., Corrado G.S., Dean J., 2013. Distributed Representations of Words and Phrases and their Compositionality. Paper presented at NIPS.
- [11] Bojanowski P., Grave E., Joulin A., Mikolov T., 2016. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5, pp. 135–146.
- [12] Sutskever I., Vinyals O., Le Q.V., 2014. Sequence to Sequence Learning with Neural Networks. *ArXiv*, vol. abs/1409.3215.
- [13] Boyanov M., Nakov P., Moschitti A., Da San Martino G., Koychev I., 2017. Building Chatbots from Forum Data: Model Selection Using Question Answering Metrics. *ArXiv*, vol. abs/1710.00689.
- [14] Bilgin T.T., Yavuz E., 2021. Conceptual design of python ide with embedded turkish spoken chatbot that analyzes and corrects the syntax errors. *Avrupa Bilim ve Teknoloji Dergisi*, (29), pp. 415–424.
- [15] İçseri İ., Aydın Ö., Tutuk K., 2021. Müşteri Hizmetleri Yönetiminde Yapay Zeka Temelli Chatbot Geliştirilmesi. *Avrupa Bilim ve Teknoloji Dergisi*, (29), pp. 358–365.



- [16] Toprak G., Rasheed J., 2022. Machine Learning based Natural Language Processing for Turkish Venue Recommendation Chatbot Application. *Avrupa Bilim ve Teknoloji Dergisi*, (38), pp. 501–506.
- [17] Barış A., 2020. A new business marketing tool: chatbot. *GSI Journals Serie B: Advancements in Business and Economics*, 3(1), pp. 31–46.
- [18] Eroglu-Hall E., Sevim N., Bulut A., 2022. Çevrimiçi tüketici tutumları chatbotlara yönelik. *EKEV Akademi Dergisi*, (91), pp. 33–53.
- [19] Luhn H.P., 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4), pp. 309–317.
- [20] Sparck Jones K., 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1), pp. 11–21.
- [21] Kusner M., Sun Y., Kolkin N., Weinberger K., 2015. From word embeddings to document distances. Paper presented at International Conference on Machine Learning, pp. 957–966
- [22] Guo S., Wang Q., 2022. Application of knowledge distillation based on transfer learning of ERNIE model in intelligent dialogue intention recognition. *Sensors*, 22(3), pp. 1270.