Computer Engineering and Applications Vol. 11, No. 1, February 2022

Query Reformulation for Indonesian Question Answering System Using Word Embedding of Word2Vec

Alvi Syahrini Utami, Novi Yusliani, Mastura Diana Merieska, Abdiansah Abdiansah*

Artificial Intelligence Laboratory, Faculty of Computer Science, Universitas Sriwijaya
**abdiansah@unsri.ac.id*

ABSTRACT

Query reformulation is one of the tasks in Information Retrieval (IR), which automatically creates new queries based on previous queries. The main challenge of query reformulation is to create a new query whose meaning or context is similar to the old query. Query reformulation can improve the search for relevant documents for Open-domain Question Answering (OpenQA). The more queries are given to the search system, and the more documents will be generated. We propose a Word Predicted and Substituted (WPS) method for query reformulation using a word embedding word2vec. We tested this method on the Indonesian Question Answering System (IQAS). The test results obtained an E-1 value of 81% and an E-2 value of 274%. These results prove that the query reformulation method with WPS and word-embedding can improve the search for potential IQAS answers.

Keywords: Query Reformulation, WPS, Word2Vec, IQAS

1. INTRODUCTION

Open-domain Question Answering (OpenQA) is a system that can answer common questions in natural language [1, 2]. To find answers to these common questions, OpenQA has to process many documents. The Internet can be used as a source of document searches because it is vast and growing. Document searches on the Internet can use search engines based on queries relevant to the question [3, 4]. One technique to increase document search results is to use Query Reformulation [5, 6], which is a technique to create new queries based on the initial query given automatically.

Several approaches have been used to reformulate the query, ranging from simple rule-based [7] approaches to deep learning [5, 8, 9] approaches. In general, there are two knowledge bases used by the query reformulation system [10, 11]. First, using the Query Log, namely queries stored in the database of a system, such as Search Engines (Google, Yahoo, Bing), Community Question Answering System (Yahoo! Answers, StackOverflow, Query), and others. New queries are generated by processing existing queries in the Query Log. Second, using a semantic dictionary (WordNet) and corpus statistics. New queries are generated by analyzing meaning relationships between words or word counts based on statistics [12]. This approach is more complicated than using Query Log because of its high semantic gap. In addition, the processing level of the two is different, and Query Log can be used to process sentences, while this approach only applies at the word level [13].

This research uses a knowledge base with the second approach. A new query is generated based on calculating the similarity of meaning between words based on the context of a corpus. The method used is word2vec which is based on an artificial conditional network model called Continuous Bag-of-Words (CBOW) [14]. The word2vec training corpus from Wikipedia Dump is about 600MB in size. The focus of the research is how to generate a new query based on the previous query because the more queries generated, the greater the chance that an answer will be obtained [15]. For example, there is a query "siapa nama presiden indonesia pertama" (*who is first president of Indonesia*), word2vec can provide a similarity for every word in the query. For example, from the word "presiden" (*president*), it is possible to generate similar words such as "pemimpin" (*leader*) or "proklamator" (*proclaimer*). Suppose these two words are used to replace the word "presiden". In that case, it will become: "siapa nama pemimpin indonesia pertama" (*who is first Indonesian leader*) and "siapa nama proklamator indonesia pertama" (*who is the first proclaimer of Indonesia*). For high precision similarity of meanings of words can use WordNet. Unfortunately, there is no WordNet for Indonesian. Therefore, it is necessary to explore word2vec to determine whether this technique is effective for Indonesian query reformulation.

Based on the background, we propose two research questions: can word2vec with the CBOW model be used as a query reformulation method to improve the performance of the Question and Answer System? and can query reformulation perform better than using a single query? The two questions are the reasons for this research. The contribution of this research is to find out the effectiveness of the word2vec method for Indonesian query reformulation. The following parts of this paper are organized as follows: Section 2 describes the related work. Section 3 describes the research method's details, including dataset, proposed method, experimental design, and evaluation. In Section 4, we present the experiment results and analysis of the proposed approach. Conclusions are given in Section 5.

2. RELATED WORKS

Several researchers have studied query reformulation using a word embedding [16, 17]. Roy et al. [18] proposes a word embedding framework based on word2vec with a distributed neural language model. Based on the framework, they extracted terms similar to queries using the K-Nearest Neighbor approach. The experimental study was conducted on standard TREC data, and the test results showed a significant increase compared to the term overlapping-based approach. The word2vec-based query reformulation approach works more or less the same with or without feedback information.

Diaz et al. [19] presents a query reformulation technique based on locally trained word embedding (such as word2vec and GloVe) for the Information Retrieval (IR) field. They also use local embedding to capture the nuances of a topic-specific language better than global embedding. They suggested that embedding be studied in topically delimited corpora rather than in topically unconfined large corpora. In the case of queries, their experimental results suggest adopting local embedding over global embedding because of the potentially superior form of representation.

Lastly, Kuzi et al. [20] proposes a query reformulation technique based on word embedding that uses the word2vec approach with the Continuous Bag-of-Words (CBOW) [21] model. CBOW represents the terms in the vector space based on their

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

co-occurrence in the text window. They also present a technique for integrating selected terms using a word embedding with an effective pseudo-relevance feedback method.

Question Answering System (QA System) is a system that accepts user questions in natural language and provides output in the form of answers to these questions [22]. In general, the QA system can be divided into two areas, namely: QA and OpenQA. In QA, the system can only answer questions related to a specific domain, while in OpenQA, the system must answer general questions.

To answer common questions, OpenQA must be able to access a significant source of information. One source of information that OpenQA often uses is the Internet [23-25]. Information retrieval applications on the Internet are called Search Engines, which receive queries (input) from users and then provide links to documents relevant to the query. OpenQA usually uses a search engine to retrieve documents that may contain answers to user questions.

Search engines accept queries as the primary input source for searching for information on the Internet. Document search results are highly dependent on the given query. Therefore, the query plays a vital role in determining the success of the search. One technique that can improve document search results by search engines is to use query reformulation to define a new query based on the previous query. The basic idea is that the more queries provided, the more documents will be generated.

3. RESEARCH METHODS

3.1 DATASET

3.1.1. WORD EMBEDDING TRAINING DATA

The word embedding training data (word2vec) used in this study was taken from the Indonesian Wikipedia data dump. The data is a compressed file type (bz2) with 654 MB (MegaByte). After being decompressed into a text file, the size was increased by 152 MB bringing the total to 806 MB. The compressed file is then extracted into a text file to be processed by the next stage. A total of 431,086 Indonesian Wikipedia articles were obtained. The data will be normalized first by making all letters lowercase. After that, it is ready to be used for training data for the word2vec CBOW model.

Before conducting the training, we first analyzed the dataset to determine its properties, such as the number of Wikipedia articles, the number of words, the number of tokens (unique words), and the number of words out of Vocabulary (OOV). OOV is words that are not in the Indonesian dictionary and words that are non-vocabulary. Next is the process of counting the number of words and tokens (unique words) from the text file extracted from Wikipedia data. Based on these calculations, the number of words is 120 million (120,210,328), and the number of tokens is 1.7 million (1,778,764). Then we identify the number of tokens that are OOV using the stemming technique. The basic idea is that the stemming results are then searched for in the essential word dictionary by stemming each token. If the token is not found, it will be considered as OOV and vice versa. After the stemming process, the number of tokens was reduced to around 15 million tokens (1,502,317) or reduced by 15.54% tokens (276,447).

TABLE 1.
Statistics of Indonesian Wikipedia Dump

Variables	Values
Number of Articles	120.210.328
Number of Tokens	1.778.764
Number of Stemming Tokens	1.502.317
Number of Out of Vocabulary	1.478.460
Number of Non Out of Vocabulary	23.857

We use common root words for lookup tables. To improve the prediction quality of OOV tokens, we can add a list of characters, places, events, and more. After the OOV calculation process, there were 1.4 million (1,478,460) OOV tokens and 23 thousand (23,857) tokens that were non-OOV. The results are pretty surprising, considering the massive difference between OOV and non-OOV tokens. Many OOVs can affect word embedding model training because user-supplied query tokens are rarely OOV unless typos, slang, or abbreviations are used. Overall, the statistics of the Wikipedia dump data can be seen in Table 1.

3.1.2 QUERY REFORMULATION TEST DATA

The query reformulation test data used a question-and-answer dataset containing 300 question-and-answer pairs data. The purpose of using the question and answer dataset is to see the effectiveness of query reformulation using word embedding. We wanted to find out if Word Embedding and Query Reformulation (WE+QF) could significantly increase the Indonesian Question Answering System (IQAS), albeit with some limitations of word embedding. The questions in the dataset will be converted into a query, and a query reformulation is performed. We only used 140 data with two types of question words, namely the question word "who," which amounted to 75 data, and the question word "when," which amounted to 65 data. The choice of question words "who" and "when" is to facilitate the evaluation of the WE+QF because the answers to these questions are easier to obtain and validate using a simple answer search system.

3.2 PROPOSED QUERY REFORMULATION METHOD

The query reformulation method that we use is the word substitution and word prediction method based on the word2vec model created in the previous stage. The word2vec model that has been trained previously has two main functions: looking for similarities in the meaning or context of a word and making word predictions based on the sentence context. In the substitution method, we use the similarity words in the query with the similarity of the word2vec model. While in the prediction method, we add the word results from the prediction of the word2vec model. Figure 1 shows the pipeline of our proposed query reformulation method. We call it the Word Predicted and Substitute or WPS method.

The first step in the WPS method is to remove the question words in the query. Removing the question word does not change the context of the query too much. For example, the query "who is the name of the first president of Indonesia" would be

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

similar in context to the query "name of the first president of Indonesia." This statement is based on the tests we have done using the Google search engine. The test results show that there are only two different URLs out of 20 URLs (each query returns 10 URLs). These results prove that the elimination of question words does not significantly affect URL search results by search engines. Therefore, based on these findings, we eliminated question words to reduce the number of new query words and the word computation process.

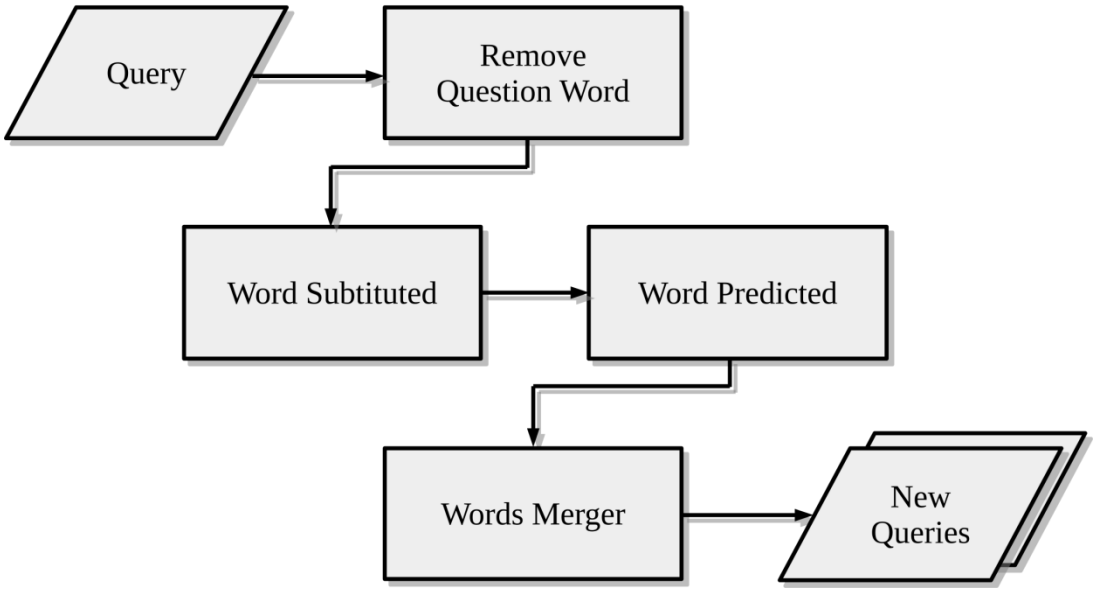

FIGURE 1. Word Predicted and Subtitute pipeline

The second stage is to substitute the word with the highest similarity based on the threshold value that we have determined, 0.7 (0.7162). The threshold value determines to anticipate situations if the value is too low, there is a possibility of obtaining inappropriate words. On the other hand, if the value is too high, there is a possibility that a substitute word will not be obtained. We determined the threshold value based on an experiment using a question-and-answer dataset containing 300 question-and-answer pairs. Each question is converted into a query, and then the tokenization process is carried out. The results of the tokenization process obtained as many as 569 words. Next, each word will look for similarity values using the word2vec model. Finally, the threshold value is obtained from the average value of the similarity of each word.

In simple terms, the process of word substitution can be illustrated by the following mathematical equation. For example, suppose there is a query Q which contains query words so that it becomes $Q = \{q_1, q_2, q_3, \ldots, q_n\}$. A similarity function accepts input Q_i and Q'_i (words resulting from the word2vec model). The function will replace (substitution) Q_i with Q'_i if the similarity value of the two words is above the threshold value. Next, the similarity function will be used by the substitution function to replace the query word. The equation of the substitution

function can be seen in Equation 1, while the pseudo-code of the word substitution function can be seen in Algorithm-1.

$$word\_subtituted(Q) = \begin{cases} Q'_i; \ similarity(Q_i, Q'_i) \geq 0.7 \\ Q_i; \ otherwise \end{cases} \quad (1)$$

The next step is to make word predictions based on the context of the input query. The Gensim library has a *predict_output_word* function, which gives the probability distribution value for words that are in the middle position of a word context. The function accepts two parameters: the context of the word (sentence) and topn (the number of words desired). The output of the function is the word and its probability value. The word that is in the first order has the highest probability value, and so on. The value of the topn parameter used in this study is 5 so that later five new queries will be generated with each query added with the predicted word2vec model.

ALGORITHM 1. Word Subtitute Algorithm

```
1: procedure WORD_SUBTITUTE (model, query, th=0.7)
2:     new_query = query
3:     vocab = model.vocabulary()
4:     i=0
5:     while w in query do
6:         if w in vocab then
7:             if model.similarity(w, topn=1) ≥ th then
8:                 new_query[i] = w
9:     return new_query
```

ALGORITHM 2. Word Predict Algorithm

```
1: procedure WORD_PREDICT(model, query, k=5)
2:     words = model.predict(query, topn=k)
3:     new_query = query
4:     while w in words do
5:         if w not in query then
6:             new_query.append(w)
7:     return new_query
```

The following is an illustration of word prediction with mathematical equations. For example, suppose there is a query Q containing query words so that it becomes Q = {q_1, q_2, q_3, … q_n}. A predict function can give the word context (w_i) of a Q and k (the number of contexts desired). Every w_i generated by the prediction function will be added to Q so that it becomes a new query (Q'_i). The total number of new queries generated is Q_k. The equation of the word context prediction function by the model can be seen in Equation 2, while the equation for word prediction that generates a new query can be seen in Equation 3. The pseudo-code of the word prediction function can be seen in Algorithm-2.

$$predict(Q, k) = \{w_1, w_2, w_3, \ldots, w_n\} \quad (2)$$

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

$$word\_predict(Q,k) = \{Q + w_1, Q + w_2, \ldots, Q + w_n\} \quad (3)$$

The last stage of the Word Substitution and Prediction method is combining the query results from the word substitution and prediction process. The merging process is quite simple, namely by combining a substitution query and a prediction query. An example of query merge pseudo-code can be seen in Algorithm-3. The merger results will then be forwarded to the next stage for further processing by search engines.

ALGORITHM 3. New Queries

```
1: procedure NEW_QUERIES(Subtitute Query, Predict Queries)
2:     new_query = Subtitute Query + Predict Queries
3:     return new_query
```

3.3 EXPERIMENTAL DESIGN AND EVALUATION

3.3.1 EXPERIMENTAL-1

Experiment-1 aims to find out how much impact a new query has on the results of information search by search engines. In this experiment, there are three processes carried out, namely: (1) Finding the original query URL; (2) Searching URLs of all new queries; and (3) Finding the number of different URLs between the original query and the new queries. Evaluation of the experimental results is done by looking at the difference in the number of URLs. The result of calculating the number of differences is symbolized by the value E-1 (Evaluation for Experimental-1), whose value is between 0 to 1. If all URLs of the original query are different from all URLs generated by new queries, then the value of E-1 is 1. It means the new query has the potential to provide additional information to the original query. On the other hand, if all URLs of the original query differ from all URLs generated by the new queries, then the value of E-1 is equal to 0. It means that the new query can add no new information. Next, we calculate the value of E-1 in Equation 4. U'_i is the number of new i-th query URLs that differ from all original query URLs, and U'_n is the sum of all new query URLs. Here are the scenario steps for Experiment-1:

1. Load the QA dataset-140
2. Load word2vec model
3. Convert question to query
4. New Query = Word Subtituted + Word Predicted
5. Find URLs based on new queries (U 0)
6. Filter URLs based on the query words
7. Calculate the value of E-1
8. Calculate the average value of E-1

$$E_1 = \frac{\sum_{i=0}^{n} U'_i}{U'_n} \quad (4)$$

Experiment-1 uses a question-and-answer dataset containing 140 question-and-answer pair data. Experiment-1 only uses questioning data, while the answer data will be used in Experiment-2. The question data is then removed from the question words and converted into a query, and new queries are made using the WPS method. Then look for the URL of the query and new queries using the search engine. In this experiment, we use the Bing search engine because it is easier to implement. Search engines provide search pages based on queries in the form of HTML pages. Next, we carry out a URL extraction process that contains a URL filtering process so that only URLs relevant to the query will be retrieved. Finally, we calculated the value of E-1 and the average value.

3.3.2 EXPERIMENTAL-2

In Experiment-1, the evaluation of the query reformulation method was carried out by calculating the difference between the URLs generated by the query and the URLs generated by the new queries. The greater the number of different URLs, the better the results assuming that the different URLs can provide additional information relevant to the context of the query. However, not all URLs generated by the new query are relevant. It can be caused by various factors: the limitations of URL extraction techniques and the word2vec model itself. We examine the query reformulation to the Question Answering System (QAS) to determine the query reformulation's effectiveness. Especially in the Indonesian Question Answering System (IQAS). So the purpose of this experiment is to find out whether the results of query reformulation using the WPS method and word embedding can impact the IQAS field, especially in increasing the potential for IQAS answers. Here are the scenario steps for Experiment-2:

1. Load the QA dataset-140
2. Load word2vec model
3. Convert question to query
4. New Query = Word Subtituted + Word Predicted
5. Find URLs based on the query (U)
6. Find URLs based on new queries (U 0)
7. Filter URLs based on the query words
8. Extract text from (U)
9. Extract text from (U 0)
10. Find and count the number of answers from (U)
11. Find and count the number of answers from (U 0)
12. Calculate the value of E-2
13. Calculate the average value of E-2

The experimental results are evaluated by looking at the percentage of additional answers generated by the query (U_A) and its new queries (U'_A). The result of the percentage calculation is symbolized by the value of E-2 (Evaluation for Experimental-2). If the percentage value is high, then more potential answers can be given to IQAS. Conversely, the smaller the percentage value, the fewer potential answers that can be given. Reformulation of a query can be effective if the value of E-1 and E-2 are both high. Furthermore, Equation 5 is used to calculate the value of E-2.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

$$E_2 = \frac{U'_A x100}{U_A} \quad (5)$$

Experiment-2 uses a question-and-answer dataset containing 140 question-and-answer pair data. The question data is then removed from the question word and converted into a query, and new queries are made using the WPS method. Then look for the URL of the query and new queries using the search engine. The number of URLs generated is not fixed depending on the results of the extraction processing of search engine results pages. We use Regex to search for URLs and perform URL filtering to obtain URLs relevant to the query. Next, web scraping is done for each URL and continued with extracting web pages into text using the BeautifulSoup library. The last is to search for answers based on the answer data references in the dataset using the Regular Expression (Regex) method and calculate the value of E-2 and the average value.

4. RESULTS AND DISCUSSION

4.1 EXPERIMENTAL-1

Table 2 contains data from Experiment-1, which contains five tuples: the number of queries, the number of new queries, the number of query URLs, the number of new query URLs, and the average value of E1. From 140 queries, 594 new queries were obtained. If the word prediction method provides different words, 840 new queries (140*6) should be added. The maximum difference between the query and the experimental results is 29.29%. Based on these results, information is obtained that not all queries can be predicted maximally (topn) by the word2vec model. Furthermore, the difference between the total number of URLs generated by all queries with the total number of URLs generated by all-new queries is huge, namely 4,490 URLs or about 54.58%. The number of URLs generated by the new queries should be more than the queries. However, the experimental results give the opposite result. It happens because new queries are applied to URL filtering during the URL fetching process so that only URLs are retrieved that are relevant to the query. These results provide information that nearly half of the URL results of new queries are less relevant to the query. Finally, the average value of E-1 is 0.81 or 81%. These results indicate that query reformulation with word-embedding is sufficient to provide additional information.

Figure 2 shows the distribution of E-1 values. It can be seen that the E-1 value of a query is above 0.5. Even though there are some low values, such as queries number 70, 74, and 102. The queries are “siapa pendiri perusahaan apple” who is the founder of the apple company, “siapa pendiri google” (*who is the founder of google*), and “siapa aktor yang memerangkan lord voldemor di film harry potter” (*who is the actor plays lord voldemort in the harry potter film*). We suspect that queries with a low E-1 value contain less common words, but the experimental results show that these queries are quite common, especially for queries number 70 and 74. We then retest specifically for these three queries. From the results of the retest, the E-1 values for queries numbered 70, 74, and 102 were 0.77, 0.83, and 0.46, respectively. The results are pretty surprising for queries number 70 and 74, which contradict the previous results. Based on these results, we conclude that the

low E-1 value in the case of queries number 70 and 74 can be caused by an unstable internet connection, so the URL fetcher is not optimal.

TABLE 2.
Results of Experimental-1

Variables	Values
Number of queries	140
Number of new queries	594
Number of URL of queries	8.227
Number of URL of new queries	3.737
The average of E-1	0.81

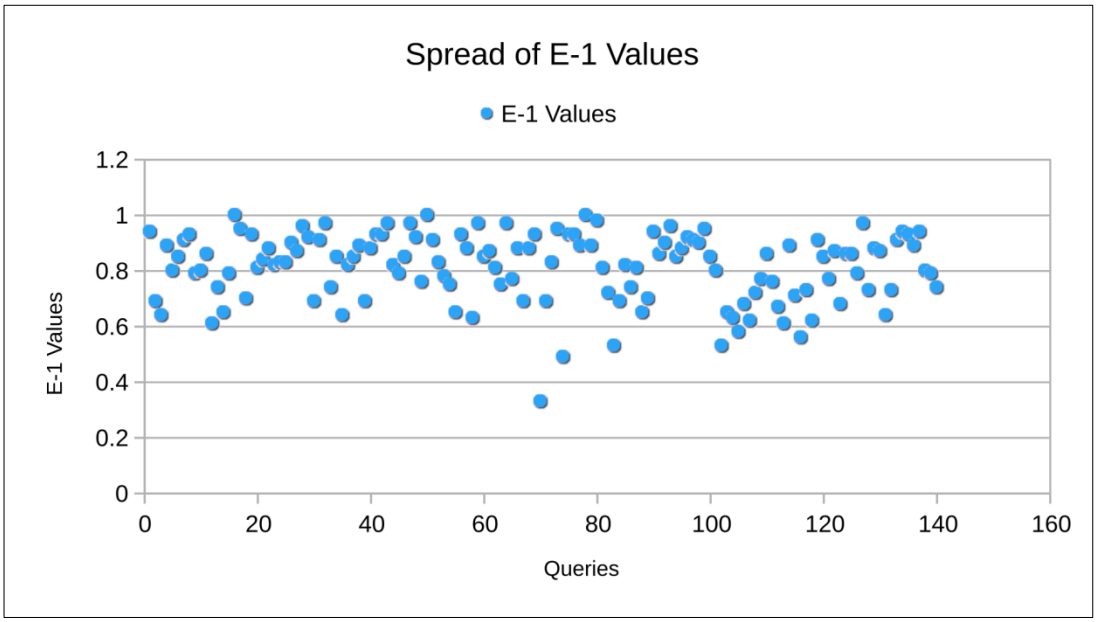

FIGURE 2. Distribution of E-1 values for the question-and-answer dataset

4.2 EXPERIMENTAL-2

Table 3 contains data from Experiment-2, which contains three tuples: a total of answers to queries, a total of answers to new queries, and the average value of E-2. Based on the experiment, the total of answers for 140 queries were 9,881 answers. Meanwhile, the answers to the new queries were 2,984, so an additional potential answer of 30.20% was obtained. This result is considered relatively low because it is still below 50%. The average value of the percentage addition per query (E-2) is 274.74%. Figure 3 shows the distribution of E-2 values for each query. In the picture, it can be seen that the average value of E-2 cannot represent the overall value of the data because there are some data whose E-2 values are high, so that they become data outliers. In addition, there are also some queries whose E-2 value is 0. It indicates that although the URLs generated by the new query are relevant to the query, they do not provide additional potential answers for IQAS. So, even though

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

the E-1 value is high, but the E-2 value is low, the query reformulation can be considered less effective for IQAS.

TABLE 3.
Results of Experimental-2

Variables	Values
Number of answer from the query	9.881
Number of answer from new queries	2.984
The average of E-2	274,74

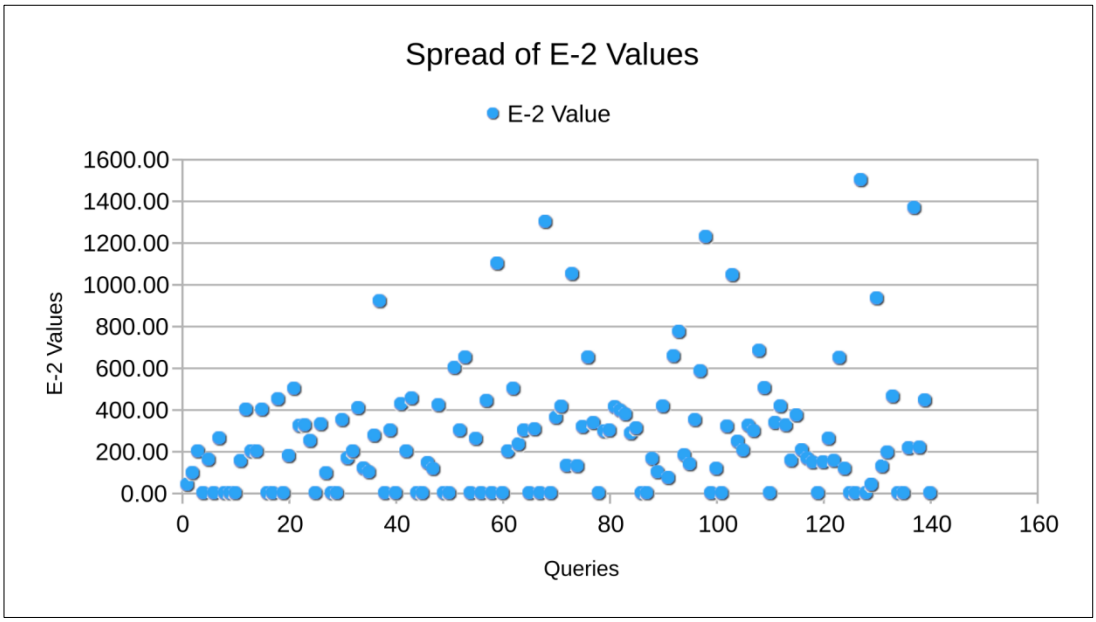

FIGURE 3. Distribution of E-2 values for the question-and-answer dataset

We traced the experimental data to investigate cases where a query had a high E-1 value but a low E-2 value. The search results show 37 queries out of 140 queries whose E-2 value is 0, or about 26.43%. Furthermore, from the 37 queries, we took the value of E-1 above the average value of E-1 or 0.81. The results obtained as many as 24 queries whose E-1 value is above the average value. Another interesting finding is that there are three queries whose E-1 value is one and the E-2 value is 0. The three queries are numbered 16, 50, and 78, with the queries consecutively being "kapan twitter dirilis" (*when is Twitter released*), "kapan manusia mengenal sandal" (*when a man knows sandals*), and "siapa yang menerima injil" (*who receives the gospel*). If the failure is seen from the search for answers, it cannot be accepted because the technique used is quite good. The answer search technique only looks for answers based on answer references already available in the dataset. This answer search technique is undoubtedly different from IQAS because, on IQAS, the answer is not yet known. We suspect that this failure is still related to other reasons, such as

text extraction techniques that have not been maximized, internet network failures, and others.

5. CONCLUSIONS

Open-domain Question Answering (OpenQA) is an automated question and answer system that can answer general questions in the form of natural language (human language). OpenQA is very complex because it processes large, unstructured text from multiple sources. One of the sub-components of OpenQA is query reformulation, which automatically creates new queries based on previous queries. Query reformulation can increase the value of search recall. A good query reformulation design can increase the number of potential answers of OpenQA.

This article contains the experimental results of query reformulation using word-embedding word2vec. The research focus is limited to the Indonesian QA System (IQAS), a QA System in Indonesia. Indonesian is one of the low-resource languages, so an in-depth investigation is needed. Two datasets are used in this research: Wikipedia Corpus for word2vec training data and 140 questions and answers data for testing the Word Predicted Substitute (WPS) query reformulation method. There are two experimental scenarios, namely: (1) Experimental-1 aims to find out how significant the impact of new queries generated by query reformulation is on the results of information search by search engines. The experimental evaluation was measured using the E-1 equation, and (2) Experimental-2 aims to determine the effectiveness of the new query generated by the query reformulation implemented in IQAS. The evaluation was measured using the E-2 equation.

In this study, reformulation of a query can be effective if the value of E-1 and the value of E-2 are both high. Experimental-1 results obtained an average value of E-1 of 81%. These results indicate that query reformulation with word-embedding is sufficient to provide additional information. While the results of Experimental-2 obtained an average value of E-2 of 274.74%. Although the average value of E-2 is high, there are some data outliers and data whose E-2 value is 0. Based on the values of E-1 and E-2, it is found that the query reformulation with word embedding word2vec using the WPS method can be used to increase the search for potential IQAS answers.

ACKNOWLEDGEMENTS

"The research/publication of this article was funded by DIPA of Public Service Agency of Universitas Sriwijaya 2021. SP DIPA-023.17.2.677515/2021, On November 23, 2020. In accordance with the Rector's Decree Number: 0007/UN9/SK.LP2M.PT/2021, On April 27, 2021".

REFERENCES

[1] Y. Zhu, L. Pang, Y. Lan, H. Shen, and X. Cheng, “Adaptive Information Seeking for Open-Domain Question Answering,” 2021, doi: 10.18653/v1/2021.emnlp-main.293.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

[2] F. Zhu, W. Lei, C. Wang, J. Zheng, S. Poria, and T.-S. Chua, "Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering," pp. 1–21, 2021, [Online]. Available: http://arxiv.org/abs/2101.00774.

[3] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading Wikipedia to answer open-domain questions," ACL 2017 - 55th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap., vol. 1, pp. 1870–1879, 2017, doi: 10.18653/v1/P17-1171.

[4] J. Chen, J. Mao, Y. Liu, F. Zhang, M. Zhang, and S. Ma, "Towards a better understanding of query reformulation behavior in web search," Web Conf. 2021 - Proc. World Wide Web Conf. WWW 2021, no. 61732008, pp. 743–755, 2021, doi: 10.1145/3442381.3450127.

[5] K. Cao, C. Chen, S. Baltes, C. Treude, and X. Chen, "Automated query reformulation for efficient search based on query logs from stack overflow," Proc. - Int. Conf. Softw. Eng., no. November, pp. 1273–1285, 2021, doi: 10.1109/ICSE43902.2021.00116.

[6] M. Breja and S. K. Jain, "Why-Type Question to Query Reformulation for Efficient Document Retrieval," Int. J. Inf. Retr. Res., vol. 12, no. 1, pp. 1–18, 2021, doi: 10.4018/ijirr.289948.

[7] J. Z. Chen, S. Yu, and H. Wang, "Exploring Fluent Query Reformulations with Text-to-Text Transformers and Reinforcement Learning," vol. 2021, 2020, [Online]. Available: http://arxiv.org/abs/2012.10033.

[8] C. T. Lin, S. P. Ma, and Y. W. Huang, "MSABot: A Chatbot Framework for Assisting in the Development and Operation of Microservice-Based Systems," Proc. - 2020 IEEE/ACM 42nd Int. Conf. Softw. Eng. Work. ICSEW 2020, pp. 36–40, 2020, doi: 10.1145/3387940.3391501.

[9] X. Wang, C. Macdonald, and I. Ounis, "Deep Reinforced Query Reformulation for Information Retrieval," 2020, [Online]. Available: http://arxiv.org/abs/2007.07987.

[10] H. G. Cavalcante, J. N. Soares, and J. E. B. Maia, "Question Expansion in a Question-Answering System in a Closed-Domain System," Int. J. Comput. Appl., vol. 183, no. 23, pp. 1–5, 2021, doi: 10.5120/ijca2021921621.

[11] S. Hirsch, I. Guy, A. Nus, A. Dagan, and O. Kurland, "Query Reformulation in E-Commerce Search," SIGIR 2020 - Proc. 43rd Int. ACM SIGIR Conf. Res. Dev. Inf. Retr., pp. 1319–1328, 2020, doi: 10.1145/3397271.3401065.

[12] A. P. Bhopale and A. Tiwari, "Leveraging Neural Network Phrase Embedding Model for Query Reformulation in Ad-hoc Biomedical Information Retrieval," Malaysian J. Comput. Sci., vol. 34, no. 2, pp. 151–170, 2021, doi: 10.22452/mjcs.vol34no2.2.

[13] T. Lei, Z. Shi, D. Liu, L. Yang, and F. Zhu, "A novel CNN-based method for question classification in intelligent question answering," ACM Int. Conf. Proceeding Ser., 2018, doi: 10.1145/3302425.3302483.

[14] P. Panagiotou, G. Kalpakis, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "Query Reformulation Based on Word Embeddings: A Comparative Study," pp. 41–55, 2021, doi: 10.1007/978-3-030-69460-9_3.

[15] N. Stringham and M. Izbicki, "Evaluating Word Embeddings on Low-Resource Languages," pp. 176–186, 2020, doi: 10.18653/v1/2020.eval4nlp-1.17.

[16] H. K. Azad and A. Deepak, "Query expansion techniques for information retrieval: A survey," Inf. Process. Manag., vol. 56, no. 5, pp. 1698–1735, 2019, doi: 10.1016/j.ipm.2019.05.009.
[17] F. S. Khan, M. Al Mushabbir, M. S. Irbaz, and M. A. Al Nasim, "End-to-End Natural Language Understanding Pipeline for Bangla Conversational Agent," 2021, [Online]. Available: http://arxiv.org/abs/2107.05541.
[18] D. Roy, D. Paul, M. Mitra, and U. Garain, "Using Word Embeddings for Automatic Query Expansion," 2016, [Online]. Available: http://arxiv.org/abs/1606.07608.
[19] F. Diaz, B. Mitra, and N. Craswell, "Query expansion with locally-trained word embeddings," 54th Annu. Meet. Assoc. Comput. Linguist. ACL 2016 - Long Pap., vol. 1, pp. 367–377, 2016, doi: 10.18653/v1/p16-1035.
[20] S. Kuzi, A. Shtok, and O. Kurland, "Query expansion using word embeddings," Int. Conf. Inf. Knowl. Manag. Proc., vol. 24-28-October-2016, pp. 1929–1932, 2016, doi: 10.1145/2983323.2983876.
[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 1st Int. Conf. Learn. Represent. ICLR 2013 - Work. Track Proc., pp. 1–12, 2013.
[22] S. K. Dwivedi and V. Singh, "Research and Reviews in Question Answering System," Procedia Technol., vol. 10, pp. 417–424, 2013, doi: 10.1016/j.protcy.2013.12.378.
[23] W. Y. C. Meek, "W IKI QA : A Challenge Dataset for Open-Domain Question Answering," no. September 2015, pp. 2013–2018, 2018, [Online]. Available: http://www.aclweb.org/anthology/D15-1237.
[24] K. N. Lam, N. N. Le, and J. Kalita, "Building a Chatbot on a Closed Domain using RASA," PervasiveHealth Pervasive Comput. Technol. Healthc., pp. 144–148, 2020, doi: 10.1145/3443279.3443308.
[25] S. Vakulenko, S. Longpre, Z. Tu, and R. Anantha, "Question Rewriting for Conversational Question Answering," WSDM 2021 - Proc. 14th ACM Int. Conf. Web Search Data Min., pp. 355–363, 2021, doi: 10.1145/3437963.3441748.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)