42nd Computers and Information in Engineering Conference
IDETC/CIE 2022
August 14 – 17, 2022, St. Louis, Missouri, USA

IDETC/CIE 2022-90768

DESIGN AND RESEARCH OF INTELLIGENT QA SYSTEM FOR FLIGHT CREW OPERATING MANUAL

XIN TAN[1], JINGSHU ZHONG[1], YU JIN[2], YAN LIANG[3], YU ZHENG[1], YING LIU[4]

[1]Shanghai Jiaotong University, Shanghai, China
[2] Shanghai Aviation Industry(group) Co., Ltd, Shanghai, China
[3]Expert IT Services, Hong Kong SAR, China
[4]School of Engineering in Cardiff University, Cardiff, UK

ABSTRACT

Aviation flight crews rely on a large number of complex standard documents and operation manuals when performing flight tasks. In order to relieve the pressure of manual retrieval of documents, intelligent question-answering technology based on reading comprehension is gradually applied. In this paper, the flight crew operation manual SQuAD dataset is studied and built, based on which the reader-retriever framework of text content-based reading question answering system (TCQA) is analyzed and established. Experiments are conducted to compare the relevant indexes of the QA system with different combinations of reader and retriever models under the open-source tool haystack. Based on the comparison of response speed and retrieval capability, the best model combination is obtained for the flight crew operation manual dataset, and suggestions are made for the model-related performance improvement.

Keywords: NLP, TCQA, SQuAD, reader-retriever framework

1. INTRODUCTION

The aerospace industry relies on a large number of normative standard documents in practice. Take the flight crew operating manual (FCOM) file as an example, this file contains the operating and safety instructions for a specific type of aircraft in various scenarios. The relevant FCOM files for each aircraft are usually thousands of pages, and most of them are in paper or electronic pdf format. Manually retrieving specific information from such massive documents takes a lot of time and effort. While in actual flight scenarios, the time is usually limited, which makes it difficult for the flight crew to find the required information within the specified time [1]. Therefore, it has gradually become a necessary trend to explore intelligent technologies for retrieving FCOM files to facilitate the use of relevant documents by enterprise personnel**.**

Intelligent question-answering technology combining natural language understanding and interactive document retrieval can be a measure to address the above needs. After the relevant files are converted into the corresponding format and imported into the background of the system, users can interact with the question answering system (QA system) by inputting questions organized in natural language. The system will process the questions and retrieve the background text. The intelligent QA system can effectively alleviate the need for users to understand the FCOM document structure and query syntax and significantly improves the users' retrieval speed, ensuring their work efficiency. Therefore, the related FCOM documents is planned to be processed, and an intelligent QA system for FCOM files is researched and designed. Its related performance is also be evaluated.

The rest of the paper is organized as follows. In section 2, the classification, development and composition of the intelligent QA system are introduced. In section 3, the intelligent QA system framework of FCOM files is established and the functions of the core components and various models are introduced in detail. Section 4 discusses experiments and the evaluation results of the QA system. Section 5 presents a summary of the main findings and future prospects.

2. BRIEF REVIEW OF LITERATURE

2.1 Classification of QA systems

Machine reading comprehension is an important task in QA system, and understanding natural language and completing basic reasoning are necessary in the task. The main purpose of the task is to extract the answer or generate the answer according to the given text and the question asked by the user. According to the answer type, the QA system can be divided into two types: extractive QA system and generative QA system [2].

In an extractive QA system, the machine is able to extract a span of text from the corresponding context as the answer to the question. In a few extractive QA systems, answers can also be multimedia forms. With the application of deep learning methods

and the emergence of datasets such as SQUAD [3], NewsQA [4], and TriviaQA [5], the accuracy of extractive QA systems has also been greatly improved, resulting in wider applications.

The generative QA system cannot directly extract the answer from the text, but needs to reason based on multiple texts and summarize the generated answer. The answer form of the generative QA system is not limited, which is more suitable for practical scenarios. Compared with the extractive QA system, the answer form of the generative QA system is more diverse and the answer is easier to understand. However, the method is more difficult, and there is a lack of means for evaluation [6]. The system meets the actual needs, which can track the relevant documents according to the problems, and obtain the original document information.

QA systems can also be divided into open domain QA systems and closed domain QA systems. Open-domain QA systems can answer almost all questions, and their texts come from websites such as Wikipedia. Closed-domain QA systems are able to answer domain-specific questions, and can be trained to build domain-specific QA systems to solve specific problems [7]. Therefore, this paper will introduce a closed-domain extractive QA system constructed using text and training datasets from the aerospace field to solve the application problem of documents.

2.2 Development of QA system framework

The QA system can be regarded as an extension of the search engine. Compared with the search engine, the answer provided by the QA system is more accurate, rather than simply returning the results based on the keyword search. The development of the QA system originated in 1961. Green et al. proposed the BASEBALL system to transmit the information of the American baseball league. The answers of the system are mainly in the form of date and location [8]. In 1973, Woods et al. proposed the Lunar system, which provided relevant information about soil samples [9]. Although these early systems had relatively good performances, the lack of information in the repositories limited the application of these early systems [10].

With the continuous development of research, open-domain QA systems such as IBM Watson [11] are also emerging, and these systems are no longer limited to QA in a single domain. At the same time, the application range of QA system is also wider, and it can play a very important role in medical [12], agriculture [13], tourism [14] and other fields. There are also QA systems in the aviation field. Alexandre [15] et al. introduced a QA system that helps aircraft pilots obtain document information. Pilots can ask questions to the system in natural language and interact with the system to obtain answers, which effectively improves the application efficiency of aviation documents.

Early search engines were mainly based on information retrieval, but it was difficult for information retrieval to accurately locate the answer. Therefore, the improved QA systems use traditional information retrieval techniques combined with machine reading comprehension (MC) methods to complete paragraph retrieval and answer processing tasks respectively[16]. After optimization, the QA system generally adopts a pipeline structure, including three main modules: Question Analysis, Passage Retrieval, and Answer Extraction [17-19].

Chen [20] et al. continued to simplify the framework and proposed the Retriever-Reader two-stage framework. The framework is shown in Figure 1. Retriever is used to filter paragraphs containing correct answers, while Reader is used to extract correct answers in paragraphs. The two-stage framework is also used in QA system frameworks such as DrQA[20].

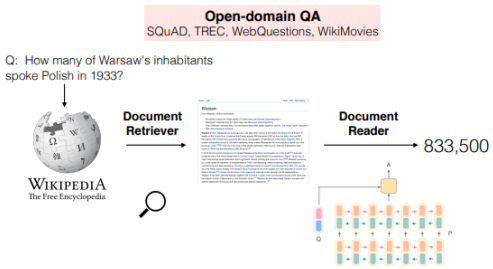

FIGURE 1: FRAME DIAGRAM OF TWO-STAGE QA SYSTEM [20]

2.3 Development of Retriever and Reader

Retriever in QA system often adopts traditional information retrieval methods such as TF-IDF or BM25 [21]. These traditional methods filter out candidate documents by matching keywords. However, these traditional methods cannot be trained, so it is difficult to perform complex retrieval tasks well. In recent years, models such as Bert [22] overcome the limitation with self-supervised learning. These models allow Retriever to be trained using specific datasets.

Retriever models are usually constructed with dual encoders, where one encoder is used to encode the question and the other is used to encode the text [23]. Both supervised and unsupervised learning methods can be used to train Retriever models. Lee [24] et al. used the reverse cloze task to train Retriever and evaluated it on open datasets, demonstrating the effectiveness of the unsupervised learning method. Karpukhin et al. [25] applied the supervised learning method of Dense Passage Retriever and demonstrated that the performance is better than traditional retrieval methods such as BM25. The improvement and optimization of the existing Dense Passage Retriever methods have also achieved good results [26-27].

The traditional Reader model uses pattern matching [28] or machine learning methods to extract answers from texts. With the emergence of large-scale training datasets deep learning methods after 2015, the Reader model has also grown rapidly. Both the ELMo model [29] and the Bert model [22] use pre-training to improve the performance of the Reader model. The former is an autoregressive model. The training direction of the model is from left to right, but it cannot use context information at the same time, and its ability to represent text is weak. The latter belongs to the self-encoding language model, and the text representation ability is greatly enhanced, and it can better represent the effective information and context information of the text.

As the Attention mechanism is widely used in the field of natural language processing, many Reader models have also added the Attention mechanism to optimize model performance. The DrQA model [20] simply uses a bilinear term to compute the attention weights, resulting in word-level question-merged paragraph representations. The Match-LSTM [30] model applies the LSTM model to extract features in the text and applies the attention mechanism from two directions. Wang [31] applied the self-attention mechanism to the reading comprehension model and proposed the gated self-matching

network R-Net. They utilize a gated attention-based recurrent network to generate question-aware paragraph representations capable of pinpointing important parts of text.

3. INTELLIGENT QA SYSTEM FRAMEWORK

Through the literature research in Section 2, it can be found that currently applied QA systems are usually built in a pipeline format, which mainly consists of different components. The functions of components are question processing, document retrieval, and answer processing, respectively. Based on the uniqueness of industry proper nouns in FCOM files, the text characteristics of some chapters with similar structure and the relevant requirements of QA system, this paper establishes a complete intelligent QA system framework for FCOM files shown in Figure 2.

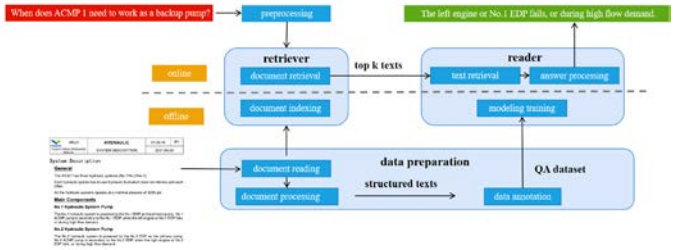

FIGURE 2: FRAMEWORK OF INTELLIGENT QA SYSTEM FOR FCOM FILES

The framework is composed of three parts, the data preparation part, the retriever part and the reader part. The data processing part mainly processes the reference texts, prepares for the initial offline process of the QA system, and supports staff to update through the background. The retriever and the reader part participate in the processing and interaction of the user's online query. The relevant content will be described in detail below.

3.1 Data preparation

The function of the document reading component is to extract valid text from the electronic document. After investigating the existing text extraction tools and algorithms, it is found that the PDFminer tool in python can extract text and recognize noisy texts such as headers and footers, which is consistent with the text structure of FCOM files and meets the extraction requirements of this paper. Therefore, the PDFminer is selected as the text extraction tool for FCOM files in pdf format.

While doing the document reading work, it is noticed that the FCOM file is a structured document. Its structure is usually a multi-level heading structure. The above titles have obvious difference in font and size, which can be recognized by PDFminer. The structure and title information of the document play an important role in understanding the content of the document. Document data conventionally stored by strings cannot reflect the structural characteristics of FCOM files. Therefore, it is necessary to convert the read text content into a specific data format. The data format not only needs to reflect the document structure characteristics, but also needs to support the storage of Q&A pairs corresponding to the text, so as to facilitate the subsequent training of the reader component and the performance evaluation of the QA system. Based on the above requirements, this paper chooses SQuAD as the data storage form of the QA system.

SQuAD is an open-domain reading comprehension dataset launched by Stanford University in 2016 [3]. Its original data includes 100,000 (question, original text, answer) triples, which is currently the most widely used NLP QA dataset. Since the SQuAD dataset is an extractive QA dataset, it is more in line with the data characteristics of FCOM files and the relevant assessment forms of flight crews.

Figure 3 shows the data structure of SQuAD. SQuAD assigns a specific number (id) to each specific Q&A pair, and stores the question text in the value of “question”, It also stores the textual fragment of the answer in the “text” value, and marks the position where the answer starts in the text as “answer_start”. At the same time, the entire text is stored in the “context” value, and the titles of all levels of this text is stored in the “title” value. The document processing stage mainly completes the SQuAD format conversion of text and title. Figure 4 is a schematic diagram of the related text extraction and structuring.

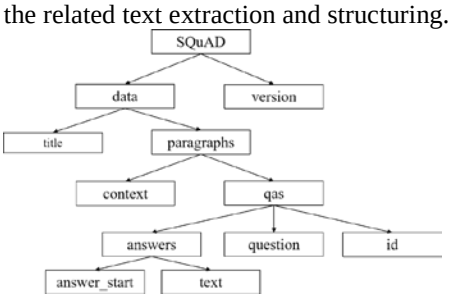

FIGURE 3: SCHEMATIC DIAGRAM OF SQUAD DATA STRUCTURE

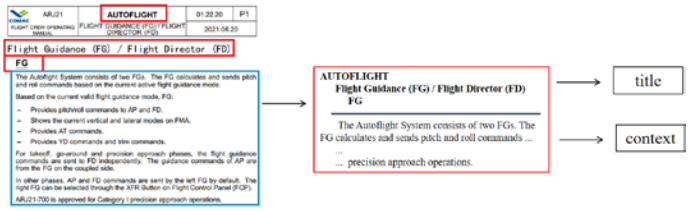

FIGURE 4: SCHEMATIC DIAGRAM OF TEXT EXTRACTION AND STRUCTURING

After completing the format conversion of the document, in order to meet the data training needs in the reader components, it is necessary to add Q&A pairs to the generated SQuAD format dataset, which is called the data annotation process. The traditional way to generate Q&A pairs in SQuAD is to hire crowdsourcers to manually annotate the text. To save time and reduce costs, questgen, an open-source NLP library for question generation algorithms, is used for automatic generation of Q&A pairs. Questgen mainly uses the T5 language model developed by Google to complete the Q&A pair generation for specified texts. The supported question generation features include multiple choice, Boolean questions, general FAQs, and paraphrase questions. The questgen online demo interface is shown in Figure 5. In order to ensure the quality of the Q&A pairs generated, questgen currently uses different T5 models for different types of questions and answers.

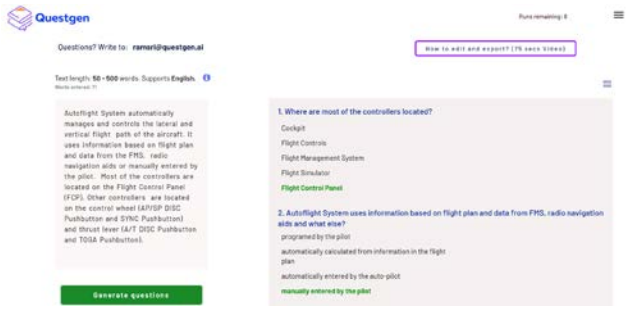

FIGURE 5: QUESTGEN ONLINE DEMO INTERFACE

After using the questgen to generate Q&A pairs, the current Q&A pair data is found to have the following problems:

Simple questions: As shown in Figure 5, the number of words in the answer is usually between 2-5 words, which can easily lead to ambiguous meaning of the answer and unclear directionality. It does not meet the requirements for accuracy in FCOM document retrieval.

Repeated questions in similar paragraphs: FCOM files generally have the characteristics of consistent chapter structure and similar content. Questgen tends to generate the same Q&A pairs in texts with similar content, which is not helpful for the quality of QA datasets.

Difficult to generate Q&A pairs about tables: Aviation standards documents often contain a large number of tables. The text extracted from the tables lacks semantic connection, making it difficult to generate Q&A pairs using questgen.

Therefore, the correct rate of Q&A pairs and the types of questions still need to be improved through manual annotation. In order to standardize the Q&A pair data format, cdQA-annotator in the closed-domain QA system cdQA is chosen as the manual annotation tool for the initial question of the text and the addition in the later maintenance process. The interface of the cdQA-annotator is shown in Figure 4. The supported input JSON file format is SQuAD 1.1. After manually entering the question, the staff can select the answer fragment in the original text to generate a Q&A pair. At the same time, the annotation tool also supports editing and deletion of existing Q&A pairs and deletion of original text fragments.

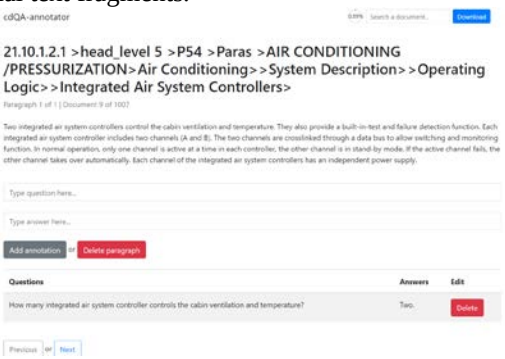

FIGURE 4: THE INTERFACE OF THE CDQA-ANNOTATOR

3.2 Retriever

For the retrieval function in the online part, the Reader component constructed by the latest transformer language system is the main model for document analysis and retrieval. However, this component has limited ability for responsiveness and processing when faced with a large number of documents. Therefore, a filter is needed to reduce the amount of text that the Reader component needs to process. Based on this concept, the Reader-Retriever framework of several QA systems such as cdQA and haystack is chosen, which are used to realize the functions of text retrieval and answer processing.

The function of the retriever component is to retrieve the most relevant paragraphs of text from a large number of stored texts and transmit them to the reader component according to the processed problems. Its essence is to solve the problem of semantic similarity ranking between the problem text and each candidate text. The main components are document storage and document retrieval.

The function of the document storage component is to store the processed text data in the system background, and establish a transmission interface of the document retrieval component. According to the text characteristics of FCOM files, this paper selects FAISSDocumentStore as the document storage tool. FAISSDocumentStore is often used for large-scale document storage. It uses SQL format to store the original document text, and can generate language vectors for the text and store them in the FAISS index. Therefore, it can support embedding-based intensive retrieval.

For document retrieval components, the widely used retrieval components are mainly divided into sparse retriever and dense retriever. The sparse retriever mainly uses the bm25 model to complete the document retrieval function, while the dense retriever uses the DPR model.

BM25 is a statistics-based retrieval model. Developed from the tf-idf algorithm, the algorithm needs to segment the query question, and calculate the similarity score of each word in the question with each piece of text stored. The final similarity between the question and a certain text is the sum of the similarity scores of each word. Assuming that the word segmentation result of the question is Q = (q_1, q_2, …, q_n), and a certain text is D_i, the BM25 calculation formula of Q and Di is as equation (1):

$$S_{BM25}(Q, D_i) = \sum_{k=1}^{n} \mathrm{idf}_{q_k} \times \mathrm{RD}(q_k, D_i) \times \mathrm{RQ}(q_k, Q) \quad (1)$$

Among them, idf_{q_k} is the inverse word frequency, and its calculation formula is as equation (2):

$$\mathrm{idf}_{q_k} = log \frac{N - df_k + 0.5}{df_k + 0.5} \quad (2)$$

N is the total number of texts in the index. df_k is the number of texts containing the word q_k. RD and RQ in Equation (1) represent the correlation between word q_k and text D_i and question Q, respectively. They both have a positive correlation with the word frequency of q_k in the text or question.

The DPR model is raised to solve the problem that words with the same semantics cannot be recognized in the BM25 algorithm. It passes a dual-encoder learned from Q&A pairs based on the training set to create embeddings for text and query questions. The model uses an encoder EP to map all stored texts to a D-dimensional real-valued vector, and index all M vectors used for retrieval. Similarly, during the query process, the DPR model uses another encoder EQ to map the queried question text to a D-dimensional real-valued vector, and uses the vector dot product form of Equation (3) to define the semantic similarity between the question q and the text p.

$$sim(q, p) = EQ(q)^T EP(p) \quad (3)$$

The encoder model used by the DPR model is an improved BERT model, which will be described in detail in Section 3.3.

The training process for the encoder is essentially a metric learning process targeting higher similarity. That is, by learning a better embedding function and creating a vector space, related Q&A pairs have smaller distances than unrelated Q&A pairs. Let $D = \{q_i, p_i^+, p_{i,1}^-, \dots, p_{i,n}^- \}_{i=1}^m$ be the training data consisting of m instances, where each instance contains a question q_i and a related paragraph p_i^+ and n unrelated paragraphs $p_{i,j}^-$. Then the loss function is optimized to the negative log-likelihood of the positive channel shown in Equation (4).

$$L(q_i, p_i^+, p_{i,1}^-, \dots, p_{i,n}^-) = -log \frac{\exp(sim(q_i, p_i^+))}{\exp(sim(q_i, p_i^+)) + \sum_{j=1}^n \exp(sim(q_i, p_{i,j}^-))} \quad (4)$$

For the use of the DPR model, it should be noted that the model will perform a process of creating embedding vectors for all stored texts every time the QA system starts and a query is entered.

3.3 Reader

Reader is a pre-trained deep learning model. Its main function is to further retrieve the text segment with the highest correlation with the queried question text in a small amount of text output by the retriever, and output this segment of text as the answer. At present, most of the reader models are fine-tuned models based on BERT developed by Google.

The BERT model is a pretrained language representation model. The original BERT model has been pre-trained with a large amount of text. For a closed-domain QA system or a specific text format, it only needs to add an additional output layer for fine-tune to be applied. In the cloed-domain QA system, no task-specific structural modification of BERT is required in the process. Common BERT fine-tuning models for the SQUAD1.1 dataset mainly include bert-large-uncased-whole-word-masking-finetuned-squad and RoBERTa-base-squad2 models. The reader model also supports pre-training on custom QA datasets. Figure 7 is a schematic diagram of the retrieval process of the BERT model based on aviation standard files established in this paper. After the model separates the input question and reference text into separate tokens, the final embedding is obtained after processing by different two-direction transformer layers. The functions implemented by the transformer layer mainly include the two-direction language representation of the generated text and the language representation at the sentence level, etc... Then the above final embedding is used as the input of the start/end marker classifier. The classifier contains the weight vector about the start/end words. After taking the dot product of the embedding and the weight vector, the probability distribution of all tokens can be obtained. The one with the highest probability is chosen as the start/end word. The text between the start and end words is the corresponding text span.

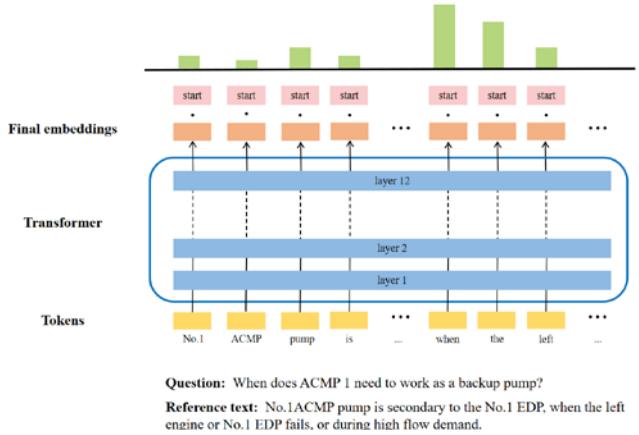

FIGURE 7: SCHEMATIC DIAGRAM OF THE PRINCIPLE OF THE BERT MODEL

4. EXPERIMENT

An experiment is to be performed to verify the feasibility of the QA system framework proposed in Section 3 for practical QA scenarios, and compare the performance of the QA system combined with different reader and retriever models. The performance test of the QA system is conducted by using the relevant flight crew operating manual of the ARJ21 aircraft. The data processing, Q&A pair generation, QA system construction, and performance evaluation of each model combination are completed, which will introduced in detail in the following.

4.1 Data source and preprocessing

The Airplane Flight Crew Operations Manual Volume 1 for ARJ21 is used as the text source. The full text of the document is in pdf format, with a total of 1164 pages. The relevant text features in the documents are utilized to implement rule matching-based text extraction and structuring and relevant text denoising using the pdfminer tool. Taking paragraphs as the separation standard, a total of 1007 paragraphs of text are obtained, with an average number of 202 words, and a JSON file in the corresponding square format is created.

According to the analysis of aviation industry forum websites and flight crew assessment questions, the flight crew manual is found to be basically a human-machine interaction manual. Therefore, there are fewer problems related to personnel, and more problems related to operation and principles. Also, the number of problems involved in the hatch, emergency and fuel is relatively large. Based on the above features, the Q&A pairs generated by the questgen tool described in Section 3.1 are manually corrected and supplemented. A total of 223 Q&A pairs are finally created. These Q&A pairs are randomly divided into 5 copies. One copy is used as the test set for evaluating the performance of the QA system, while the rest as the training set for the reader model.

4.2 QA system construction

In order to test the performance of the QA system under different combinations of reader and retriever models, the open-source tool haystack is chosen as the platform for building the retriever-reader framework. Haystack is a modular QA system framework developed by the deepset team capable of building pipelines for different search use cases. The function of building a QA system with different combinations of reader and retriever models can be achieved by modifying and combining

components in different modules. This paper formulates the model combination scheme shown in Table 1.

TABLE 1: COMBINATION SCHEME OF READER AND RETRIEVER MODEL

No.	Reader	Retriever
1	BERT	BM25
2	BERT	DPR
3	BERT+ Q&A pairs pretraining	BM25
4	BERT+ Q&A pairs pretraining	DPR

At the same time, this paper uses the tkinter tool in python to create the front-end interface of the QA system as shown in Figure 8.

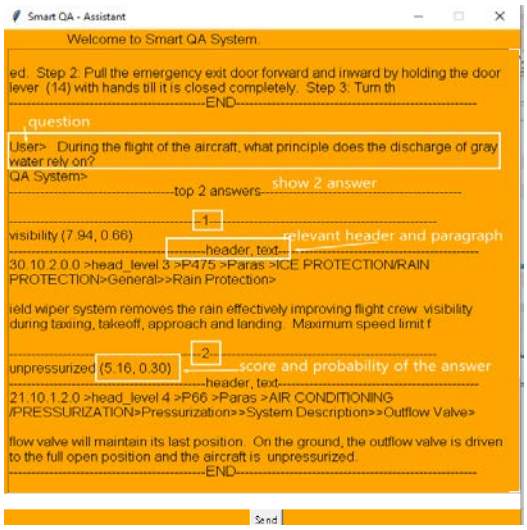

FIGURE 8: FRONT-END INTERFACE OF THE QA SYSTEM

4.3 Performance Evaluation and Analysis of QA System

To comprehensively compare the performance of the QA system under the combination of different reader and retriever models, EM, F1, SAS and average response time are chosen as the performance evaluation criteria of the QA system. The relevant indicators are explained as follows:

- **Exact Match (EM):** This metric represents the probability that the correct answer and the predicted answer exactly coincide. If the predicted answer exactly matches the correct answer, then EM = 1, otherwise EM = 0.
- **F1 Score:** It takes into account the precision and recall rate of the classification model at the same time, so it is widely used in the field of general NLP question answering, which mainly characterizes the word overlap ratio between the correct answer and the predicted answer.
- **Semantic Answer Similarity (SAS):** SAS uses a transformer-based cross-encoder architecture to evaluate the semantic similarity of two answers, rather than their lexical overlap. SAS is more helpful in finding answers that have no lexical overlap but are still semantically similar [32].
- **Average Response Time (ART):** It refers to the time it takes the QA system to respond to a question on average. It is mainly used to evaluate the efficiency of the QA system's implementation to answer the queried question.

The QA system performance test is also performed in the haystack framework, and get the results shown in the Table 2.

TABLE 2: QA SYSTEM PERFORMANCE TEST RESULTS

No.	EM	F1	SAS	ART
1	**top 1:** 0.510	**top 1:** 0.773	**top 1:** 0.850	12.2s
	top 5: 0.612	**top 5:** 0.894	**top 5:** 0.943	
2	**top 1:** 0.520	**top 1:** 0.792	**top 1:** 0.864	11.9s
	top 5: 0.602	**top 5:** 0.888	**top 5:** 0.939	
3	**top 1:** 0.444	**top 1:** 0.681	**top 1:** 0.782	5.31s
	top 5: 0.566	**top 5:** 0.845	**top 5:** 0.911	
4	**top 1:** 0.476	**top 1:** 0.727	**top 1:** 0.817	5.23s
	top 5: 0.578	**top 5:** 0.859	**top 5:** 0.921	

It can be seen that the data of the QA system after training is weaker than before training, which may be due to the larger amount of data in the default reader model of the system. Response time of the QA system after training is significantly faster than before training. There is no significant difference in performance between BM25 and DPR models. Combining various data, the BERT model pre-trained by Q&A pairs and DPR retriever model are the best performing reader and retriever model combination.

After further analysis, there are two possible reasons for the insignificant difference in performance between BM25 and DPR models. Firstly, both the closed-domain and the SQuAD data format have the characteristics that questions and answers have more text overlapping, so the performance of the BM25 algorithm based on word frequency is not much different from the DPR algorithm based on semantic recognition. Secondly, the SQuAD dataset is not enough to support the neural network training of DPR in terms of data volume, so it is difficult to take advantage of this algorithm.

5. Conclusion

This paper firstly proposes the framework of the intelligent QA system for FCOM files. In the data preparation process, based on the text features and document structure of FCOM files, the SQuAD format data set of the relevant flight crew operation manuals is established, and the automatic generation and manual annotation are used to form the QA dataset. In the document retrieval process, the "Reader-Retriever" framework is established, which is widely used in the QA system. The modular open-source tool haystack is used to build different readers and retrievers for the platform. The QA system under the model combination is evaluated from the aspects of answer accuracy, semantic similarity and average response time. In the end, the best combination is the BERT model pre-trained by Q&A pairs and DPR retriever model, and its top 5 F1 score is 84.5%.

In the future, several obvious improvements will be considered: the first one is to consider building a dual-channel weighted retriever model. It uses the BM25 and DPR models for document retrieval at the same time, and takes the weighted average of the dual-channel processing results as the final similarity score. In this way, the advantages of the above two retrieval models can be adopted at the same time. The second one is to improve the processing of tables and pictures in FCOM files. The tables and pictures in related FCOM files contain plenty of information. It is important to find a way to show the relationship between the context in tables and to develop the picture display function in the answer presentation module.

References

[1] Alexandre Arnold, Gerard Dupont, Catherine Kobus, and Franc¸ois Lancelot. 2019. Conversational agent for ´aerospace question answering: A position paper. *Proceedings of the 1st Workshop on Conversational Interaction Systems (WCIS at SIGIR).* Paris.

[2] Mishra A, Jain S K. A survey on question answering systems with classification[J]. Journal of King Saud University-Computer and Information Sciences, 2016, 28(3): 345-361.

[3] Rajpurkar P, Zhang J, Lopyrev K, et al. Squad: 100,000+ questions for machine comprehension of text[J]. *arXiv preprint arXiv:1606.05250, 2016.*

[4] Trischler A, Wang T, Yuan X, et al. Newsqa: A machine comprehension dataset[J]. *arXiv preprint arXiv:1611.09830, 2016.*

[5] Joshi M, Choi E, Weld D S, et al. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension[J]. *arXiv preprint arXiv:1705.03551, 2017.*

[6] Liu S, Zhang X, Zhang S, et al. Neural machine reading comprehension: Methods and trends[J]. Applied Sciences, 2019, 9(18): 3698.

[7] Ramprasath M, Hariharan S. A survey on question answering system[J]. International Journal of Research and Reviews in Information Sciences, 2012, 2(1).

[8] Green Jr B F, Wolf A K, Chomsky C, et al. Baseball: an automatic question-answerer[C]//Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference. 1961: 219-224.

[9] Woods W A. Progress in natural language understanding: an application to lunar geology[C]//Proceedings of the June 4-8, 1973, national computer conference and exposition. 1973: 441-450.

[10] Ojokoh B, Adebisi E. A review of question answering systems[J]. Journal of Web Engineering, 2018, 17(8): 717-758.

[11] Ferrucci D, Brown E, Chu-Carroll J, et al. Building Watson: An overview of the DeepQA project[J]. AI magazine, 2010, 31(3): 59-79.

[12] Mutabazi E, Ni J, Tang G, et al. A review on medical textual question answering systems based on deep learning approaches[J]. Applied Sciences, 2021, 11(12): 5456.

[13] Devi M, Dua M. ADANS: An agriculture domain question answering system using ontologies[C]//2017 International Conference on Computing, Communication and Automation (ICCCA). IEEE, 2017: 122-127.

[14] Pathak S, Mishra N. Context aware restricted tourism domain question answering system[C]//2016 2nd International Conference on Next Generation Computing Technologies (NGCT). IEEE, 2016: 534-539.

[15] Arnold A, Dupont G, Furger F, et al. A question-answering system for aircraft pilots' documentation[J]. *arXiv preprint arXiv:2011.13284, 2020.*

[16] Dimitrakis E, Sgontzos K, Tzitzikas Y. A survey on question answering systems over linked data and documents[J]. Journal of intelligent information systems, 2020, 55(2): 233-259.

[17] Voorhees E M. The trec-8 question answering track report[C]//Trec. 1999, 99: 77-82.

[18] Lopez V, Uren V, Sabou M, et al. Is question answering fit for the semantic web: a survey[J]. Semantic web, 2011, 2(2): 125-155.

[19] Bouziane A, Bouchiha D, Doumi N, et al. Question answering systems: survey and trends[J]. Procedia Computer Science, 2015, 73: 366-375.

[20] Chen D, Fisch A, Weston J, et al. Reading wikipedia to answer open-domain questions[J]. *arXiv preprint arXiv:1704.00051, 2017.*

[21] Robertson S, Zaragoza H. The probabilistic relevance framework: BM25 and beyond[M]. Now Publishers Inc, 2009.

[22] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. *arXiv preprint arXiv:1810.04805, 2018.*

[23] Bromley J, Guyon I, LeCun Y, et al. Signature verification using a" siamese" time delay neural network[J]. Advances in neural information processing systems, 1993, 6.

[24] Lee K, Chang M W, Toutanova K. Latent retrieval for weakly supervised open domain question answering[J]. *arXiv preprint arXiv:1906.00300, 2019.*

[25] Karpukhin V, Oğuz B, Min S, et al. Dense passage retrieval for open-domain question answering[J]. *arXiv preprint arXiv:2004.04906, 2020.*

[26] Qu Y, Ding Y, Liu J, et al. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering[J]. *arXiv preprint arXiv:2010.08191, 2020.*

[27] Ren R, Lv S, Qu Y, et al. PAIR: Leveraging passage-centric similarity relation for improving dense passage retrieval[J]. *arXiv preprint arXiv:2108.06027, 2021.*

[28] Riloff E, Thelen M. A rule-based question answering system for reading comprehension tests[C]//ANLP-NAACL 2000 Workshop: Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems. 2000.

[29] Peters M , Neumann M , Iyyer M , et al. Deep Contextualized Word Representations[J]. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018.

[30] Wang S, Jiang J. Machine comprehension using match-lstm and answer pointer[J]. *arXiv preprint arXiv:1608.07905, 2016.*

[31] Wang W, Yang N, Wei F, et al. Gated self-matching networks for reading comprehension and question answering[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017: 189-198.

[32] Risch, Julian, et al. Semantic Answer Similarity for Evaluating Question Answering Models[J]. *arXiv preprint arXiv:2108.06130, 2021.*