

RESEARCH ARTICLE

Streamlining Industrial Robot Maintenance: An Intelligent Voice Query Approach for Enhanced Efficiency

ZECHENG REN^{1,4}, ZENGNAN YU², WENYI ZHANG³, AND QUJIANG LEI⁴¹London School of Economics and Political Science, WC2A 2AE London, U.K.²Sino-French Institute in Engineering, Shanghai University, Shanghai 201900, China³School of Automation, Nanjing University of Information Science and Technology, Nanjing 210044, China⁴Greater Bay Area Institute for Innovation, Hunan University, Changsha 410082, China

Corresponding author: Qujiang Lei (qj.lei@aaia-ai.org)

This work was supported by the Greater Bay Area Institute for Innovation, Hunan University under Grant ZF202400133.

ABSTRACT Industrial robots are indispensable in modern manufacturing across various sectors, including nuclear, chemical, and aerospace industries. They serve to replace humans in hazardous operations, ensuring worker safety, while also enhancing production efficiency and product quality, particularly in tasks such as welding and assembly. However, the maintenance and repair of these robots present significant challenges. When malfunctions occur, workers often face the daunting task of sifting through extensive industrial manuals to diagnose the root cause based on error codes, a process that is time-consuming and hampers productivity. To address this issue, this paper proposes an intelligent voice query method designed to facilitate access to industrial manual content. The goal is to empower workers to swiftly comprehend the causes of robot failures and retrieve corresponding solutions through interactive voice commands, thereby streamlining troubleshooting efforts. Leveraging voice recognition technology, workers can effortlessly issue queries without the need for manual search and interpretation. The effectiveness of this method was rigorously evaluated through experimental validation in real-world scenarios, demonstrating notable improvements in maintenance efficiency.

INDEX TERMS Industrial robots, maintenance, voice recognition, natural language processing, troubleshooting, production efficiency.

I. INTRODUCTION

In the era of globalization, the trends of technological innovation and product diversification are swiftly propelling the transformation and advancement of the manufacturing industry [1]. Challenges such as innovation dynamics, shortened product production cycles, and an expanding range of products confront the manufacturing sector. This necessitates factories not only to prioritize innovation but also to enhance production efficiency. The emergence of industrial robots in various technological scenarios has significantly addressed challenges related to labor shortages and high employment costs [2]. Compared to manual labor, industrial robots offer

superior precision, enhanced consistency, and the capacity for continuous operation. In addition, artificial intelligence (AI) plays a pivotal role in facilitating the transformation of factories towards resource-friendly operations, particularly in the context of implementing smart and low-carbon initiatives within traditional industries [3]. Through the integration of industrial robots, there arises the potential to mitigate waste generation during production processes, diminish production timelines, and curtail energy consumption. Such advancements contribute significantly to the promotion of sustainable development objectives [4], [5], [6].

In the context of human production, industrial robots have a longstanding history, contributing to the efficiency and competitiveness of manufacturing enterprises since the Third Industrial Revolution. However, the contemporary demand


The associate editor coordinating the review of this manuscript and approving it for publication was Yu Liu .

TABLE 1. Common robot malfunctions.

Fault Type	Fault Description	Possible Causes
Mechanical Faults	Joint sticking or vibration Abnormal noise	Lack of lubrication, part wear Bearing damage, loose connections
Electrical Faults	Motor unable to start Inaccurate motion	Insulation damage, bearing wear Encoder damage, circuit failure
Drive System Faults	Power transmission failure	Component wear, gear dislocation, parameter error
Control System Faults	Program execution error Communication interruption	Programming logic error, parameter setting error Network failure, protocol configuration error
Structural Faults	Arm failure	Joint bearing wear, telescopic axis damage, loose parts
Sensor Faults	Sensing abnormality	Position, torque sensor failure
Power Supply Faults	Power supply abnormality	Unstable or faulty power supply
Operation Faults	Abnormal operation	Incorrect parameter setting, etc
Environmental Impacts	Reliability degradation	Temperature, dust, etc

for industrial robots surpasses that witnessed during the Third Industrial Revolution. The emergence of the Industry 4.0 concept [7], which emphasizes intelligent manufacturing and digitization, is being implemented globally, with modern industrial robots playing an essential role as a driving force in the transition towards intelligent manufacturing [8]. AI can promote technological development to improve production efficiency [9], [10], [11]. The combination of AI and robotics has also brought new directions to industrial development. Studies indicate that robotic systems are increasingly integrating AI to enhance performance, adaptability, safety, and cost-effectiveness [12].

Industrial robots play an increasingly pivotal role in contemporary manufacturing, serving as vital components for automating processes that integrate mechanical, electronic, control, computer, sensor, and AI engineering innovations. For instance, in utilizing AI for fault detection, industrial robots can substitute human labor for repetitive and monotonous tasks, and are particularly adept at undertaking high-risk operations that surpass human capabilities. However, in the event of a robot system failure, extensive troubleshooting and repair are often necessitated, leading to significant production downtime and losses [13], [14], [15], [16]. Table 1 shows the common robot malfunctions.

The background of this study originates from a real-world industrial production setting. For instance, within the industrial robot industry, user manuals of ABB robot typically span hundreds of pages. When an industrial robot encounters a fault during operation, workers commonly refer to these manuals based on error codes to pinpoint the cause of the issue. However, this conventional diagnostic method presents several challenges:

- a) Low efficiency: Navigating through extensive manual documents consumes significant time, and crucial information may be overlooked, delaying fault identification and resolution opportunities.
- b) Lack of robust human-computer interaction: While ABB robots are capable of collaboration with humans, the current methods and technologies for human-computer interaction remain limited. The diagnostic process heavily relies on the subjective

experience of workers. Challenges persist in achieving effective communication and collaboration between robots and humans, thus restricting the application and effectiveness of robots in certain tasks.

- c) Information silos: Information from various system manuals remains relatively segregated, lacking effective knowledge integration and sharing mechanisms, hindering information interoperability.
- d) High costs: Failure to promptly resolve faults can lead to prolonged downtime, reducing productivity and resulting in economic losses.
- e) Knowledge inheritance gaps: Effectively accumulating and transferring the valuable experience of skilled workers presents challenges, and talent attrition can perpetuate a ‘skilled trap’ within enterprises.

Existing robot maintenance methods also exhibit shortcomings. Firstly, they heavily rely on external services. Many enterprises may lack sufficient in-house technical teams to address breakdowns in ABB robots, necessitating reliance on external services. This can lead to slower response times for repairs and may result in inconsistent service quality. Additionally, the maintenance technology threshold is relatively high. ABB robots utilize advanced control systems and complex mechanical structures, requiring maintenance technicians to possess professional knowledge and skills in areas such as electrical, mechanical, and control engineering. These skills are necessary to accurately diagnose the cause of failure and execute effective maintenance.

Hence, there exists an urgent necessity to develop a more intelligent fault diagnosis system, which is voice interaction technology, aiming to enhance diagnostic efficiency, minimize maintenance costs, and facilitate the digital transformation of the manufacturing industry. The origins of voice interaction technology can be traced back to the 1950s. During this period, Bell Labs pioneered the development of the Audrey system, which recognized Arabic numerals in speech. This achievement is generally considered an early exploration in the field of speech interaction technology. In 1962, IBM took substantial steps by introducing a device called Shoebox, which could perform simple mathematical calculations through speech. This innovation marked the

TABLE 2. The comparison between the diagnosis methods against industry 4.0 requirements [19].

	Easy to implement	Industrial Internet of Things	Flexible	Easy to reconfigure	Large data analysis	Minimizes human intervention	Online Diagnostic	Keep traceability in real time	Usable for the prognosis	Based on automatic and Intelligent algorithms
FMECA	×	×	✓	✓	×	×	×	×	×	×
Failure Tree	✓	×	✓	✓	×	×	×	×	×	×
Expert Systems	✓	✓	×	×	✓	✓	✓	✓	✓	✓
Fuzzy Logic	×	✓	×	×	×	✓	✓	✓	✓	✓
Neural Networks	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
Tropical Algebra	×	×	×	×	×	✓	×	×	✓	×
Finite Automata	×	✓	×	×	×	✓	✓	✓	✓	×
Petrie Net	✓	✓	×	×	×	✓	✓	✓	✓	×
Bond Graph	×	✓	×	×	×	✓	✓	✓	×	×

initial trial and verification of voice interaction technology in real-world applications. This series of progress laid a solid foundation for the subsequent development of voice interaction technology.

In recent years, research into the utilization of cutting-edge technologies such as AI and natural language processing(NLP) within the industrial domain has witnessed a surge in activity. Voice interaction technology has already demonstrated initial success in customer service, navigation, and other domains, offering new perspectives for the industrial sector. This study aims to devise a voice-interactive fault diagnosis system tailored for industrial robots. This system will enable workers to describe observed fault conditions using voice input, following which the system will analyze the input utilizing advanced AI technologies like NLP and knowledge-based reasoning. Subsequently, it will provide interactive guidance on potential causes and corrective measures. This innovative system has the potential to effectively address various deficiencies associated with traditional fault diagnosis methods, thereby assisting factories in enhancing efficiency, reducing costs, and achieving information integration and sharing.

Moreover, this system holds broad applicability across various industrial settings, encompassing factory workshops and remote maintenance, thus presenting significant commercial prospects and economic potential. Through proper deployment and staff training, enterprises can leverage this system to curtail costs and enhance efficiency, thereby capitalizing on opportunities within the intelligent manufacturing market.

In conclusion, the voice-interactive fault diagnosis system plays a crucial role in amalgamating AI with manufacturing development, infusing technological vitality into the vision of “Industry 4.0”. This paradigm of human-machine collaboration will propel industrial robots toward heightened levels of intelligence, furnishing manufacturing enterprises with unparalleled competitive advantages. Looking ahead, there is

ample reason to anticipate that voice interaction technology will unlock substantial value across a broader spectrum of industrial fields, including intelligent manufacturing, thereby aiding enterprises in navigating challenges and fostering sustainable development.

In contrast to methods prevalent in the industrial domain, we present a novel troubleshooting scheme grounded in speech interaction, text processing, LangChain, and large language model(LLM). This solution efficiently retrieves content from industrial manuals and furnishes workers with prompt fault diagnosis and maintenance guidance. Unlike prior fault detection methods, our proposed solution boasts simplicity in processing and user-friendliness for robot operators. Evaluation of model results further underscores the high accuracy and practicality of our approach.

Based on speech-to-text(STT), LangChain, LLM, and text-to-speech(TTS) technologies, this paper proposes an interactive question-and-answer system for diagnosing robot faults based on industrial manuals. Compared with alternative troubleshooting measures, this method offers simplicity and efficiency for workers. Experimental results further validate the effectiveness of the proposed approach, showcasing its advanced capabilities in addressing such challenges and its potential for application across diverse domains.

Section II provides an overview of research pertaining to fault maintenance and detection in industrial robots. Section III elaborates on the specific methodologies employed in the experiment. Section IV details the experimental environment configuration and the analysis of experimental findings. Lastly, Section V presents the conclusions of the experiment and outlines future research directions.

II. RELATED WORK

The onset of the Industry 4.0 era [17], [18] has precipitated significant transformations in corporate production models,

intensifying competition within industries. As production tools and technologies undergo continuous updates, factory machinery and equipment are increasingly susceptible to failures. Consequently, many enterprises are endeavoring to adopt intelligent fault diagnosis methods to address these challenges, ensuring production efficiency and effectively resolving equipment failures [19]. According to paper [19], Table 2 shows the comparison between the diagnosis methods against industry 4.0 requirements.

More specifically, numerous companies are attempting to tackle problems algorithmically. In a study by [20], an online passive fault diagnosis approach is introduced for systems modeled as finite-state automata. Within this framework, the system and diagnoser (fault detection system) do not need to be initialized simultaneously. Furthermore, no prior knowledge of the state of the system or fault condition is required before initiating diagnosis. Although the design demonstrates exponential complexity in the worst case, a polynomial-time model reduction scheme is proposed to alleviate the computational burden. The research delves into fault diagnosability, establishing necessary and sufficient conditions.

Furthermore, in [21], a qualitative simulation-based fault diagnosis technique for detecting broken rotor bars in AC induction motors is proposed. This method determines the qualitative values and changing directions of system variables through reasoning. Qualitative simulation can be implemented with minimal system parameter knowledge and imperfect models, which are common scenarios for most machine operators. Bond graph modeling, on the other hand, simulates dynamic systems via causal reasoning, leveraging energy flow for inference. Consequently, this paper plans to utilize bond graph modeling for fault detection without relying on specific machine parameter information.

Additionally, alongside relatively traditional intelligent fault diagnosis methods, machine learning offers novel approaches for factories. For instance, in [22], Deep Convolutional Transfer Learning Networks (DCTLN) are employed, consisting of two modules: condition assessment and domain adaptation. The condition assessment module utilizes a one-dimensional convolutional neural network (CNN) to automatically extract features and evaluate machine health conditions. The domain adaptation module enables the 1-D CNN to learn domain-invariant representations by maximizing domain recognition errors and minimizing probability distribution divergences. Furthermore, Pintoa and Cerquitellib [23] apply four different techniques (survival analysis, convolutional neural networks(CNNs), random forests, and k-NN) to predict robot failures and compare them in terms of accuracy. Compared with other algorithms, the accuracy, precision, and recall rate of image conversion in CNNs are significantly improved.

The study presented in [24] employs statistical process control based on principal component analysis and Nelson rules for online fault detection, utilizing rules of Nelson for sensitive detection. Upon detecting changes, the system

employs multi-class support vector machines for fault diagnosis operations. Additionally, to assess the health status of robots, a robot health index generator based on fuzzy logic is proposed. Figure 1 shows the whole process of this paper.

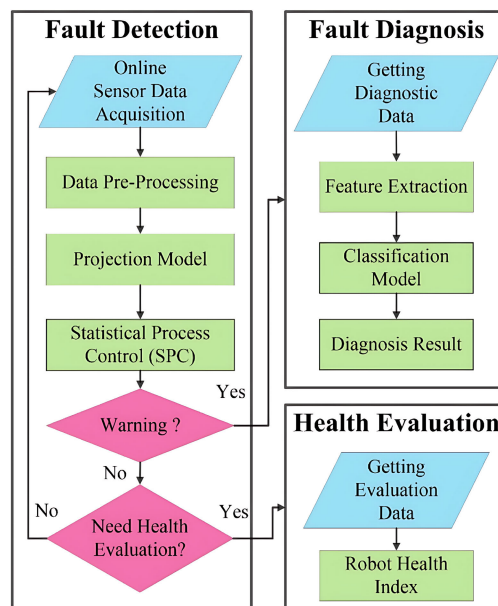


FIGURE 1. Intelligent online fault detection, diagnosis and health evaluation system [24].

In contrast, the work described in [25] introduces an unsupervised fault detection framework based on Gaussian mixture models to detect robot faults using current signals. The process involves two steps: first, cleaning and pre-processing the measured original current signal, and then employing unsupervised learning algorithms to detect faults that reflect the motion insensitivity of industrial robots by selecting features through physical models.

Moreover, the study outlined in [26] presents a measurement-based mathematical model termed Bode equations vector fitting to discern potential failures in robot joints. Simultaneously, they propose a table-based neural network classifier to identify faulty joints based on the time interval of task division. Faulty joints are diagnosed by detecting energy consumption rates and generating isolation metrics using an artificial neural network.

Lastly, the research discussed in [27] investigates the use of e-learning systems to enhance assembly work by replacing paper manuals with video manuals. While paper aids in fundamental understanding, videos optimize processes. The findings indicate that during the “improvement phase,” interactive video manuals are more effective than paper manuals, whereas during the “understanding phase,” the work is enhanced. In essence, interactive video manuals serve as guides that not only facilitate learning but also aid in improving work efficiency.

Currently, exploring the possibility of devising novel fault maintenance methodologies based on previous research

findings constitutes a significant research endeavor. In our pursuit, we endeavor to address this challenge by integrating user queries with intelligent query systems. Within the industrial domain, intelligent query systems facilitate the retrieval of various information, including production module information, historical data, and plant structure visualizations. For instance, the study in [28] introduces an intelligent search engine capable of retrieving vast amounts of data from diverse sources such as manuals, commissioning instructions, service notes, shift books, process data, repair instructions, data sheets, R/I flow charts, and CAD drawings. By combining user queries with intelligent query systems, users can simply voice their query demands, allowing the system to intelligently match and retrieve the required textual or multimedia information from multiple sources and respond vocally. This integrated intelligent query system offers more efficient and convenient information retrieval services for the industrial sector.

In fact, several studies have endeavored to amalgamate NLP with industrial robots. The paper [29] aims to integrate speech recognition and image processing techniques into an industrial robot control system to enhance operator safety. The robot can execute image recognition and perform corresponding operations through voice commands, enabling operators to avoid high-risk areas. Simulations and actual tests have demonstrated the efficacy of the system. Another paper [30] proposes an interactive product manual system based on a chatbot, allowing users to interact with the system through messaging applications to learn how to operate electrical appliances. Experiments have validated the practicality of the system. Additionally, the paper [31] introduces a method to enhance human-cobot interaction using natural language processing. By integrating a voice recognition system into collaborative robots, non-expert operators can program and control the robots through voice commands, promoting human-robot collaboration.

Building upon this premise, we propose an intelligent voice query method based on the content of industrial manuals. This method is tailored for industrial production settings to address the challenge faced by workers when operating robots, where they often need to consult extensive industrial manuals to identify the cause of errors indicated by error codes. Given the substantial thickness and volume of industrial manuals, manual consultation consumes significant time, leading to reduced work efficiency. Based on the accuracy of experimental results, our method demonstrates practicality and effectiveness.

III. PROPOSED APPROACH

A. OVERVIEW

This section presents the methodology for constructing an interactive voice question answering system, designed to address the realm of robot fault maintenance in the industrial sector, leveraging LLM and LangChain technologies. The system is composed of three main components: input

processing, building a local knowledge database, and output processing.

Initially, the input processing phase includes three steps. First, we collect the voice input of user, which serves as the input data for the entire system. Then, we perform a STT conversion to transform the speech data into text data. Finally, since we retrieve user queries from an industrial manual in vector form, we need to convert the text data into vector form at this stage.

Secondly, building a local knowledge database involves four sections. First, we obtain the industrial manual on robot fault maintenance, which serves as the foundation for problem queries. Next, due to the extensive content of the manual, we segment its body into manageable parts for subsequent processing. Following this, we vectorize these segmented paragraphs, just as we do with the query text data. Lastly, we build a vector database to cluster similar parts using a clustering algorithm. During querying, we can quickly find the cluster center containing the query vector and then conduct a detailed search within this cluster to enhance efficiency.

The final component is output processing, which encompasses three steps. The first step is to query the matched answers. Here, we calculate the cosine similarity [32] between the query vector and the paragraph vectors to obtain the top k most closely matched answers. Next, we collect these answers. Although we have the top k most matched answers, they might still be difficult for the user to comprehend. Therefore, we further process the output. This involves two sub-steps: First, we organize the output text into a format that combines the relevant paragraphs to answer the query coherently. Second, we input the reformatted answer into the ChatGLM3 model to compile the final output text. The last step is converting the final output text into speech using TTS technology, which then broadcasts the response to the user.

By following this structured approach, the system can efficiently handle user queries, process complex documents, and provide clear and actionable responses. Figure 2 illustrates the structure of the proposed approach.

B. INPUT PROCESSING

1) SPEECH-TO-TEXT

The core objective of the experiment is to develop and implement an interactive system. This system is designed to provide solutions for robot faults based on the voice description of user by referencing an industrial manual. Consequently, the initial step in this process is to convert the speech of user into text accurately.

STT functionality is a critical application in NLP, finding extensive usage across various domains like voice assistants, voice search, and automatic caption generation. Speech recognition systems are responsible for extracting, representing, and recognizing information from spoken language [33]. Many open-source frameworks are available to perform this task, including Google Cloud Speech-to-text API, Microsoft

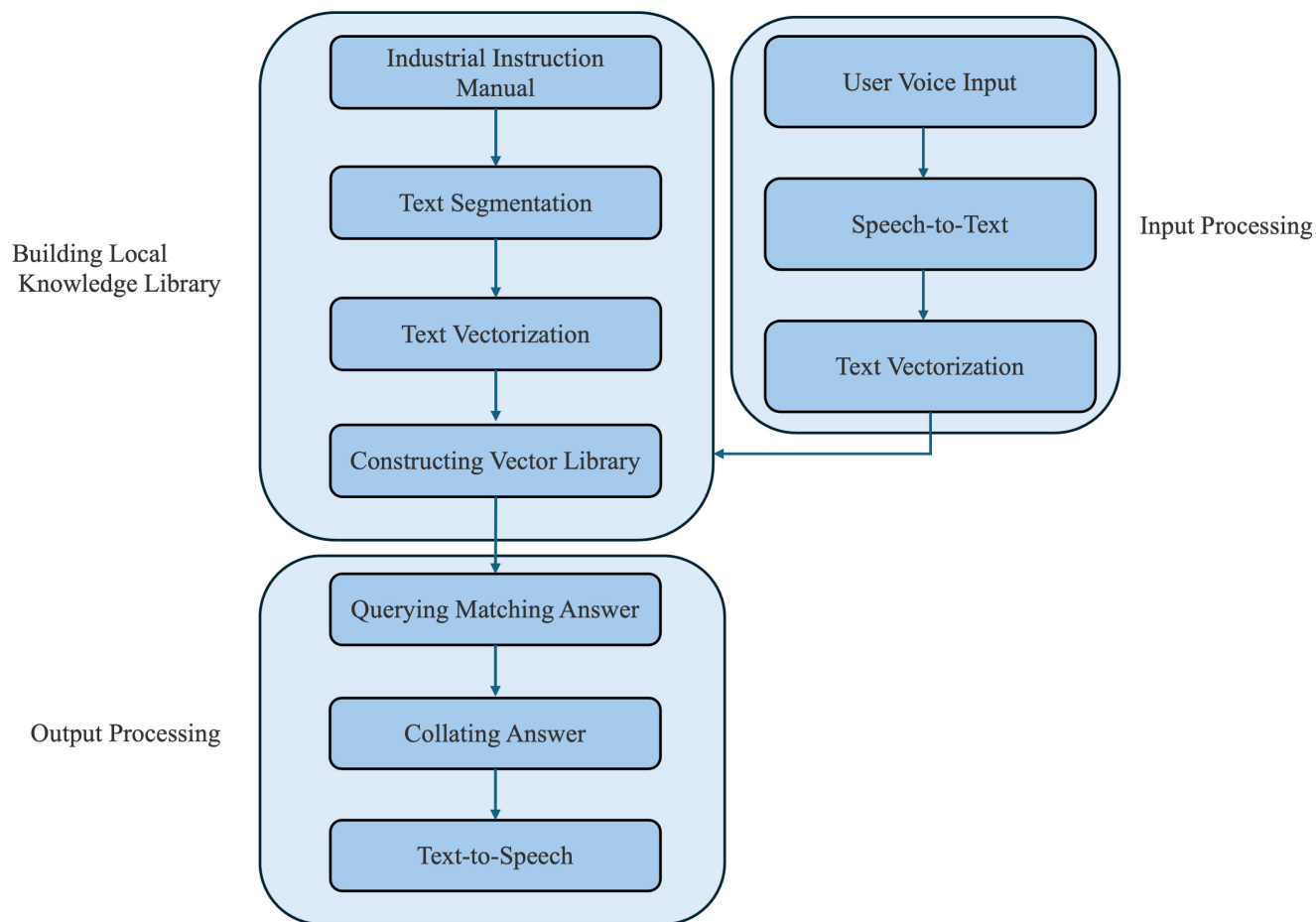


FIGURE 2. Task processing procedure.

Azure Speech Services, and Tencent AI Lab ASR (Automatic Speech Recognition) API. In this study, we utilize the Whisper model, an advanced ASR system. This model has been trained on 680,000 hours of multilingual and multitask supervised data collected from the web, boasting high accuracy rates [34].

The Whisper model from OpenAI leverages several advanced principles and mechanisms to achieve its high performance in ASR. These can be summarized as follows:

- a) Transformer architecture: The Whisper model is built upon the transformer [35] architecture, a deep learning framework extensively employed in voice and natural language processing tasks. This architecture excels at handling sequential data due to its self-attention mechanism, which adeptly captures long-distance dependencies within the data. This capability is crucial for understanding context over extended sequences, making it particularly effective for speech transcription where accurate context comprehension is essential for recognizing words and phrases accurately.
- b) Robustness to noise: Whisper’s training incorporates a diverse dataset that includes a range of noisy environments and complex acoustic conditions. This

extensive exposure equips the model with exceptional resilience to real-world audio challenges, such as background noise, varying speaker volumes, and overlapping speech. Whisper effectively filters out irrelevant noise and concentrates on the primary speech signals, thereby significantly enhancing transcription accuracy even in suboptimal conditions.

- c) Encoder and decoder modules: The architecture of Whisper model is composed of two main components: an encoder and a decoder. The encoder is responsible for processing the input audio features and capturing the acoustic properties of the speech. Meanwhile, the decoder generates the corresponding text by attending to the output of the encoder and previous tokens, producing coherent and contextually accurate transcriptions. This encoder-decoder configuration allows Whisper to efficiently address both the acoustic and linguistic dimensions of speech, ensuring precise and contextually relevant outputs.

Moreover, Whisper models come in various sizes: tiny, base, small, medium, and large, each corresponding to different levels of conversion accuracy. The higher the accuracy, the more parameters the model contains, resulting

in slower processing speeds. Following our tests, we observed that the results obtained from the tiny, base, and small models were not sufficiently accurate, despite their large parameter counts. Conversely, the medium-sized model, comprising 769M parameters, exhibited superior accuracy in conversion tasks without significantly compromising processing speed. Therefore, after careful consideration, we opt for the medium-sized model, achieving excellent results in the conversion task. Table 3 indicates the architecture details of the whisper model family.

TABLE 3. Architecture details of the Whisper model family [34].

Model	Layers	Width	Heads	Parameters
Tiny	4	384	6	39M
Base	6	512	8	74M
Small	12	768	12	244M
Medium	24	1024	16	769M
Large	32	1280	20	1550M

2) TEXT VECTORIZATION

After acquiring the text data from the user query, the next step involves mapping the text to a vector using word embedding techniques. Vectorizing text data to create structured data is a prerequisite for storing it in a network [36]. In NLP, embedding methods are commonly utilized to represent words or terms as continuous vectors. For instance, techniques like Word2Vec, GloVe, and FastText assign words to a continuous, low-dimensional vector space. This arrangement brings semantically similar words closer together in the vector space, facilitating easier processing of data in vector form by computers.

Notably, OpenAI, a leading entity in the industry, has released highly successful models such as GPT-3.5 and GPT-4, which have set benchmarks in the field. Given achievements of OpenAI in LLM and the evident link between LLM and text processing, we employ embedding method of OpenAI to perform vectorization operations on text. In this task, we utilize `text-embedding-ada-002` model for generating embeddings. This model is based on the transformer architecture and employs contrastive learning objectives. Contrastive learning is a supervised learning technique designed to train models to position similar data pairs closer together in vector space while pushing dissimilar pairs further apart. The goal of contrastive learning is to maximize the similarity between positive sample pairs and minimize the similarity between negative sample pairs.

Additionally, regarding initialization and training phases of the model, OpenAI typically initializes these models from a pre-trained generative model, such as those from the GPT family. These generative models have been trained on extensive datasets, capturing broad language representations. This pre-training process allows the model to adapt more swiftly to specific semantic tasks during the contrastive learning phase. In this paper, when building embeddings with

the OpenAIEmbeddings API, the process involves three main steps:

- Input text preprocessing:** The API first preprocesses the input text. This includes steps such as tokenization, normalization, and the addition of special tokens like [SOS] (Start of Sequence) and [EOS] (End of Sequence). These special tokens help delineate the beginning and end of the text, aiding the model in generating a more accurate and meaningful representation of the input sequence.
- Model calculation:** Once the text is preprocessed, it is fed into the encoder of the model. The encoder processes the text to produce a series of hidden states that capture the context and semantics of the input. Specifically, the hidden state corresponding to the [EOS] token is extracted as the embedding for the entire input text. This ensures that the embedding encapsulates the semantic information of the whole sequence in a single vector.
- The generated embedding is a fixed-dimensional vector,** typically of high dimensionality (e.g., 1536 dimensions in the case of the `text-embedding-ada-002` model). This vector represents the semantics of the input text in the embedding space. Such embeddings can be effectively utilized in various NLP tasks, including text similarity calculations and semantic searches.

C. BUILDING LOCAL KNOWLEDGE DATABASE

1) TEXT SEGMENTATION AND TEXT VECTORIZATION

The pivotal component of this experiment entails matching the query of user in the local knowledge database, which is constructed based on the industry manual. The objective is to identify the initial texts that closely correspond to the query, exhibiting the highest degree of similarity. Consequently, the successful establishment of the local knowledge library profoundly influences the outcomes of the experiment.

Given the extensive length of the industrial manual, processing the entire document as input is impractical. It becomes imperative to segment the text of the industrial manual and partition it into fixed-length paragraphs for subsequent processing. When working with text, it is important to recognize that it is not merely a sequence of words. Text is a coherent structure where the meaning of words unfolds and interrelates, forming a nuanced tapestry of ideas and contexts. Therefore, the way that the text is divided is particularly important [37]. In this study, we leverage LangChain to read the file and execute simple segmentation operations. Furthermore, we refine the segmentation outcomes to achieve the final partitioning of paragraphs.

In this study, text segmentation is performed based on specific character sets, including [`\n\n`, `\n`, `'o'`, `''`]. Within this set, `\n\n` denotes two consecutive line breaks, typically indicating paragraph breaks in multi-text documents. It serves to delineate distinct paragraphs in the original text. `\n`, on the

other hand, represents a single newline character, commonly denoting the end of a line in multi-text documents. Here, '\n' is utilized to separate each line within a paragraph. 'o' signifies the end of a sentence in Chinese, aiding in the segmentation of sentences within paragraphs. Lastly, '' represents an empty string, which is distinct from a blank space. Its special meaning can be summarized as follows:

- a) Preserving the last level of segmentation: In advanced text processing, the final segmentation often requires precise control to ensure that text blocks are neither too large nor too small. When '' is encountered as a delimiter in a sequence of splitters, it signals that further subdivision of the text is unnecessary. It actually represents the end point of the split operation. This is particularly useful in algorithms that recursively apply various delimiters to break down a text.
- b) Acting as a placeholder: As a placeholder, '' is used to maintain structural integrity in the definition of splitter sequences. It can be viewed as a "no operation" symbol. It can denote a point in the sequence where a splitting operation could occur, but is not required in the given context. This is especially useful in modular text processing systems where different configurations or stages might conditionally apply certain delimiters. In such systems, '' ensures that the logical flow of operations remains consistent, even when no specific action is needed at a particular step.

Differing from the previously employed character sets used for paragraph segmentation in English text, the inclusion of the Chinese character 'o' aims to better suit the Chinese context and align with the current paragraph segmentation task. This addition facilitates the division of paragraphs, especially in cases where the minimum paragraph length cannot be met with the existing paragraph and line separators. Additionally, 'o' aligns with Chinese writing conventions, contributing to improved coherence in sentence meaning during Chinese text segmentation.

In addition to defining the special character set, it is imperative to consider the length of the split paragraphs. In this study, we have chosen a length of 500 characters for each split paragraph. Furthermore, we must account for the overlap between paragraphs to maintain contextual coherence. Hence, when segmenting paragraphs, it is essential to retain a portion of the content from adjacent paragraphs. For this task, we opt to include 5% of the length of the following paragraphs, equivalent to 25 characters, as overlapping text between each paragraph. Additionally, we need to process the text of the divided paragraphs by removing any remaining special characters.

The detailed process flow for text segmentation in this task is as follows: Firstly, the contents of the industrial manual are read from the file, and the special character set is used to identify paragraph breaks, indicated by '\n\n'. Subsequently, the text is segmented accordingly. If the length of the segmented text still exceeds the maximum allowed paragraph length, further segmentation is performed by recursively

using the subsequent characters from the special character set, and so forth.

As the construction of a vector database for queries based on industry manuals is required, it is essential to vectorize the segmented paragraphs. In this study, we leverage `text-embedding-ada-002` model to perform vectorization on the segmented paragraph text.

2) CONSTRUCTING VECTOR DATABASE

a: BUILDING VECTOR DATABASE

To enable efficient retrieval, it is essential to construct a vector database based on vectorized paragraphs. We embed the vectorized text data into a spatial vector, with each basic unit representing a single vector. The calculation method is as follows: Suppose we define a text data set D , where the text d_i of each paragraph, after text segmentation, is embedded into a d -dimensional vector x_i :

$$D = \{x_1, x_2, x_3, \dots, x_n\} \quad (1)$$

b: BUILDING INDEX STRUCTURE

To expedite retrieval, we employ the K-means [38] clustering algorithm to construct an index structure for the vector library. The fundamental concept is to partition the vectors in the text vector database into K cluster centers, and subsequently utilize these cluster centers to build the index structure. We suppose there are n sample points, and each sample point is represented by x_i , where $i = 1, 2, \dots, n$. First, the cluster center with number of k is initialized by the optional method:

$$Centroids = [Centroid_1, Centroid_2, \dots, Centroids_k] \quad (2)$$

The following steps need to be repeated until convergence:

- a) For each x_i , calculate its distance to the center point of each cluster:

$$distance(i) = ||x_i - Centroid_j||^2 \quad (3)$$

where $j = 1, 2, \dots, k$.

- b) Each sample point x_i is assigned to the category j corresponding to its nearest cluster center point.
- c) Update the cluster center point so that it becomes the mean of all sample points in the current category:

$$Centroid_j = |S_j|^{-1} \sum_{x_i \in S_j} x_i \quad (4)$$

where S_j represents all samples in category j .

The establishment of cluster centers also signifies the creation of a vector index. During querying, it is only necessary to search for the nearest cluster center to the cluster containing the query vector, and then conduct more detailed queries within that cluster.

D. OUTPUT PROCESSING

1) QUERYING MATCHED ANSWER

In this paper, cosine similarity is employed to compute vector similarity. Cosine similarity serves as a measure to quantify the angle between two vectors. It is a widely utilized

and effective metric in constructing vector databases and identifying similar texts. Let the query vector be q , and for each x_i in the vector library there is:

$$\text{Similarity}(q, x_i) = \frac{\sum_{j=1}^n q_j x_{ij}}{\sqrt{\sum_{j=1}^n q_j^2} \sqrt{\sum_{j=1}^n x_{ij}^2}} \quad (5)$$

where n is the dimension of the vector. Finally, the results of the most matched Top K paragraphs are obtained through the results of similarity calculation.

2) COLLATING ANSWER

Although we have retrieved the top k paragraphs that best match the problem, there are redundancies in these paragraphs, and they are not easy to read. Therefore, we still need to further process these paragraphs. In this task, we utilize the ChatGLM3 language model for text output processing. ChatGLM3 is a dialogue pre-training model jointly developed by Zhipu AI and KEG Lab of Tsinghua University. The ChatGLM3 model represents a substantial advancement over its predecessors, offering significantly improved capabilities in generating natural and coherent conversational text. This model is specifically optimized for handling a broad spectrum of conversational tasks, making it ideal for various applications such as chatbots, automated customer service, and human-machine dialogue systems. ChatGLM3 model also excels in understanding and generating dialogue by maintaining contextual relevance and coherence over multiple turns. It adeptly grasps the flow of conversations, ensuring that responses are not only contextually appropriate but also engaging and natural. Furthermore, the model supports multilingual dialogue generation, with particular proficiency in both Chinese and English, allowing it to seamlessly produce fluid and contextually accurate conversations across diverse language environments.

ChatGLM3-6B is an open-source model in the ChatGLM3 family, and it performs exceptionally well in the field of question and answer (Q&A). Boasting 6 billion parameters, ChatGLM3-6B has the capacity to delve deeply into text, capturing subtle nuances and complex contextual information far more effectively than smaller-scale models. This extensive parameterization enhances its ability to understand and generate high-quality text, making it a powerful tool for various applications.

Furthermore, ChatGLM3-6B is particularly optimized for processing Chinese text, excelling in environments where handling the intricacies of the Chinese language is crucial. This optimization allows it to generate accurate and contextually appropriate responses in Chinese, making it especially valuable in applications tailored to Chinese-speaking users. In this experiment, output processing is divided into two steps: 1. Prompt engineering; 2. Output Generation.

a: PROMPT ENGINEERING

There are also challenges associated with using large models in downstream tasks: fine-tuning large models can

often be costly, and the output of language models is sometimes difficult to control. Prompt engineering offers a solution to efficiently utilize large models. A prompt is a set of instructions given to an LLM in order to enforce rules, automate processes, and ensure that particular output qualities (and quantities) are achieved [39]. Prompt engineering involves optimizing the structure, content, and other dimensions of prompts to better control output of the model by constraining the input to a specific range. By employing prompt engineering, the model can be directed to generate highly relevant and accurate text responses. In this task, we format the input to include the current question plus relevant paragraphs, enabling the language model to organize the response effectively. Figure 3 indicates the format of prompt.

b: OUTPUT GENERATION

After the normalized prompt is input into the large-scale ChatGLM3 language model, the model processes and interprets the text content of the prompt to extract and organize the most relevant answer that aligns with the question of user. ChatGLM3 employs advanced NLP techniques to understand the context and nuances of the query, ensuring that the generated response is both precise and contextually appropriate. This output represents the answer retrieved from the industrial manual in response to the query of user.

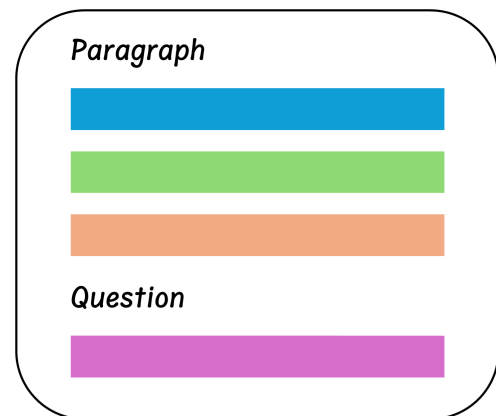


FIGURE 3. The format of prompt.

3) TEXT-TO-SPEECH

We utilize the Edge-TTS library in Python to perform TTS operations. Edge-TTS is a Python library that integrates with the Azure Cognitive Services of Microsoft to convert written text into natural, fluid speech. Leveraging advanced machine learning and deep learning algorithms, this library transforms text into speech with high clarity and natural intonation, making it suitable for a variety of applications such as screen reading tools, virtual assistants, and voice navigation systems.

The accuracy of TTS plays a pivotal role in determining the comprehensibility of fault solutions conveyed by robotic systems. The essence of TTS lies in effectively

mapping textual inputs to phonemes, thereby facilitating the construction of acoustic models and subsequent sound synthesis. This conversion process represents a critical phase in speech synthesis, where phonemes serve as the fundamental units of speech. By mapping textual inputs to phonemes, the system can seamlessly generate corresponding speech signals, enabling computers to emulate human speech and proficiently articulate textual content. The TTS process is typically implemented through the following steps: We represent the preprocessed text sequence as:

$$X = \{x_1, x_2, x_3, \dots, x_n\} \quad (6)$$

where x_i is the i th language unit. Here we directly reference the Transformer architecture to map the preprocessed text to phonemes:

$$Y = \text{Transformer}(X) \quad (7)$$

Through Transformer mapping:

$$Y = \{y_1, y_2, y_3, \dots, y_n\} \quad (8)$$

where y_j is the j th phoneme.

Additionally, TTS process leverages the transformer architecture for constructing acoustic models. In contrast to the process of mapping text to phonemes, wherein preprocessed textual data serves as the input and sound phonemes constitute the output, the acoustic model operates differently. Here, the input consists of phoneme sequences, while the output represents sequences encoding acoustic features. The formula can be expressed as follows:

$$W = \text{Transformer}(Y) \quad (9)$$

where W represents the acoustic features of the phonemic transition.

Subsequently, upon acquiring acoustic features, TTS process incorporate the WaveNet model for generating speech waveforms from these features. The WaveNet model stands as a prevalent vocoder model renowned for its capacity in high-fidelity speech waveform synthesis. At its core, the model comprises two crucial elements: Dilated Convolution and Gated Activation. Notably, WaveNet excels in capturing long-range temporal dependencies through expanded convolutional operations. We try to combine WaveNet with LSTM features. Specifically, we introduce the LSTM layer on top of WaveNet to process input features and provide an additional dimension of time context. Its output can be expressed as:

$$h_t = \text{LSTM}(x_t, h_{t-1}) \quad (10)$$

where x_t is the input feature of the current moment and h_{t-1} is the hidden state of the previous moment.

$$y_t = \tanh(W * x_t + V * h_t) \quad (11)$$

where $*$ represents the convolution operation, W and V are the parameters of the model, WaveNet controls the output of

the model through a gating mechanism. The output of a gated convolution can be expressed as:

$$z_t = \sigma(W' * x_t + V' * h_t) \quad (12)$$

where σ is the sigmoid activation function, and W' and V' are the model parameters. y_t and z_t are then combined via a gating mechanism:

$$s_t = (1 - z_t) * y_t + z_t * h_t \quad (13)$$

Finally, through the generation process of sample points one by one, the speech waveform is generated.

Edge-TTS offers a user-friendly API that supports numerous languages and voice options, making it versatile for global applications. After generating responses using the model, we save the output to a file, which is subsequently converted into the MP3 format for final use. This step ensures that the speech is compatible with a wide range of audio playback systems.

IV. EXPERIMENT SETUP

We implement all models in Python 3.10.12 within the Google Colab platform environment, utilizing a Tesla T4 graphics card and a T4 GPU. Additionally, we employ several libraries for various tasks, including but not limited to the medium size Whisper model for STT conversion, PyPDFloader for document loading, OpenAIEmbeddings for embedding based on `text-embedding-ada-002` model, ChatGLM3-6B for answer generation, and Edge-TTS for TTS conversion. When segmenting text paragraphs, we standardize the chunk size to 500 with an overlap of 25. The sample rate of audio data is 16kHz. We adhere to these processing procedures to achieve optimal performance.

The experimental data for our study are primarily categorized into speech data and text data, sourced from the EC66 user manual, version 6.0.134, of Suzhou Elytte Robot Co., LTD. For the text processing component, all test data were meticulously extracted by our team from the EC66 user manual. The extracted text data are structured in a question-and-answer format, aligning with the manual's content.

In the TTS and STT sections, the voice data were recorded by team members who read aloud the extracted textual content. This approach ensures that the spoken data accurately reflects the corresponding text from the manual. Our methodology involves creating a comprehensive dataset that bridges both text and audio forms, enabling robust testing of TTS and STT functionalities.

V. EVALUATION METRICS

A. UNIT TEST

1) SPEECH TO TEXT

In the evaluation of TTS modules, performance metrics are typically categorized into error rate indices and accuracy indices. The error rate indices predominantly include word error rate (WER), character error rate (CER). These metrics quantify the discrepancies between the output of system and the reference text by measuring various types of errors.

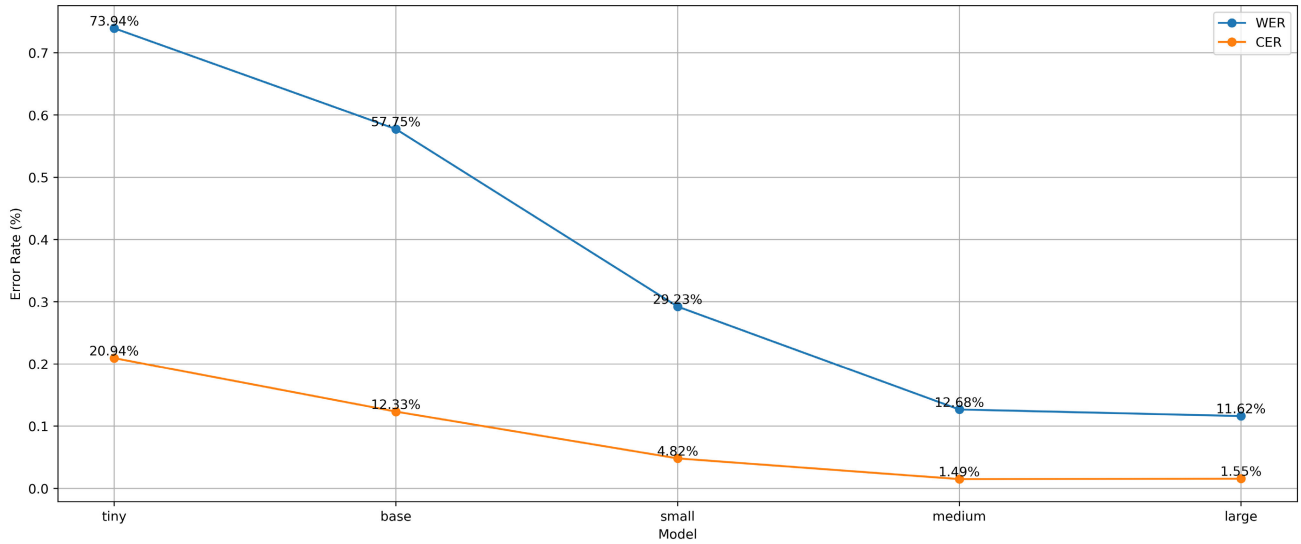


FIGURE 4. Error metrics for different models.

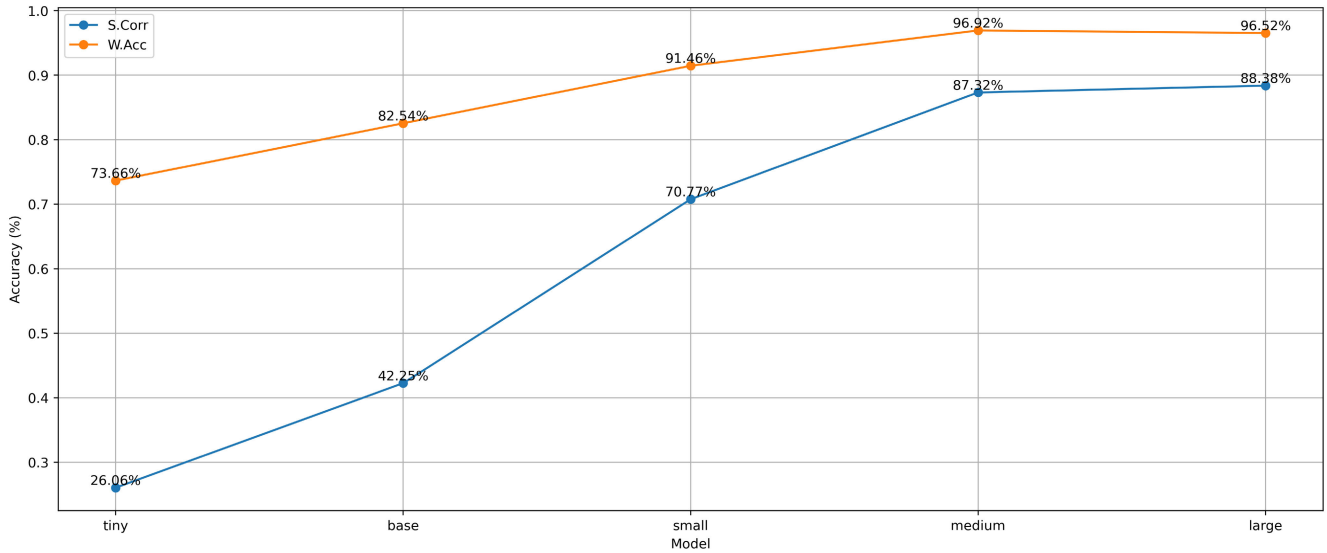


FIGURE 5. Accuracy metrics for different models.

- a) WER assesses the number of substitutions, deletions, and insertions at the word level, providing a comprehensive measure of the accuracy of word recognition. The formula of WER is as follows:

$$WER = \frac{S + D + I}{N} \tag{14}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions, N is the total number of words in the reference text.

- b) CER evaluates errors at the character level, which is particularly useful for languages or tasks where fine-grained textual precision is crucial. The CER formula is presented below:

$$CER = \frac{S + D + I}{N} \tag{15}$$

where S is the number of character substitutions, D is the number of character deletions, I is the number of character insertions, N is the total number of characters in the reference text.

On the other hand, the accuracy indices include sentence correctness (S.Corr) and word accuracy (W.Acc), which focus on the correctness of the recognized text.

- a) S.Corr measures the percentage of sentences that are completely correct, offering a holistic view of sentence-level accuracy. Below is the formula for calculating S.Corr:

$$S.Corr = \frac{N_{corr}}{N_{total}} \tag{16}$$

where N_{corr} is the number of completely correct sentences, N_{total} is the total number of sentences.

- b) W.Acc calculates the proportion of correctly recognized words, providing a detailed insight into the performance of system at the word level. The formula of W.Acc can be expressed using the WER:

$$\text{W.Acc} = 1 - \text{WER} \quad (17)$$

In this experiment, we evaluated Whisper models of different sizes based on our dataset against the above metrics. Figure 4 indicates the error metrics for different models.

According to the analysis presented in Figure 4, there is a significant decrease in WER as model size increases. Starting from a WER of 73.94% in the tiny model, this error rate drops dramatically to just 11.62% in the large model. This reduction underscores the enhanced capability of larger models to accurately recognize words, thereby minimizing word-level errors.

Similarly, CER shows a notable decline from 20.94% in the tiny model to 1.49% in the medium model, indicating a substantial improvement in character recognition. Interestingly, the large model has a slightly higher CER of 0.06%, which, despite being low, suggests that the medium model performs marginally better at the character level.

As illustrated in Figure 5, the accuracy metrics further validate the performance improvements with larger models. The tiny model shows the S.Corr at 26.06%, meaning that only about a quarter of its sentences are completely error-free. With each increase in model size, S.Corr improves significantly. The base model achieves a noticeable improvement at 42.25%, while the small model exhibits a substantial leap to 70.77%, demonstrating its superior capability in generating accurate sentences compared to smaller models. The medium and large models continue this trend, with S.Corr values of 87.32% and 88.38%, respectively. These figures indicate that nearly nine out of ten sentences produced by these models are entirely correct, showing a clear trend that larger models are more proficient in generating accurate and coherent sentences.

In terms of W.Acc, the tiny model achieves 73.66%, indicating that nearly three-quarters of the words it generates are correct. This metric improves to 82.54% in the base model, reflecting better word recognition accuracy. The small model significantly enhances this to 91.46%, showing that over 90% of its words are correct. The medium and large models exhibit very high word accuracy rates of 96.92% and 96.52%, respectively, indicating that these models are exceptionally proficient at producing correct words, with only about 3-4 words out of every 100 being incorrect.

In conclusion, the medium model emerges as the most balanced and effective choice. It offers an optimal combination of high accuracy and reasonable computational efficiency. With a S.Corr of 87.32% and a W.Acc of 96.92%, the medium model is highly suitable for a wide range of applications. While the large model provides slightly better S.Corr at 88.38%, the improvement is marginal compared to the medium model, and it does not sufficiently offset the potentially higher computational costs. Therefore, in this

task, we finally chose the medium model for the STT task, providing an excellent balance between performance and efficiency.

2) TEXT PROCESSING

In this paper, we utilize automatic evaluation metrics to assess the accuracy of knowledge base answers, serving as a means to evaluate the performance of the model. Specifically, we employ semantic similarity of text as a metric to gauge text processing accuracy. When computing text similarity, a single distance or similarity measure often falls short in providing a comprehensive and accurate evaluation of semantic similarity between texts. Therefore, we propose a novel approach that integrates multiple similarity calculation metrics to yield a more thorough and precise evaluation outcome. Specifically, we utilize cosine similarity and Pearson correlation coefficient [40] as primary indices, supplemented by mean square error (MSE) [41]. The performance of model is assessed by employing cosine similarity, Pearson correlation coefficient, and MSE to gauge the similarity between the vectors representing the model-generated answer and the correct answer from the industrial manual for a specific query.

The Pearson correlation coefficient is a statistical measure that quantifies the degree of linear correlation between two variables. It ranges in value from -1 to 1 , where a value of 1 signifies a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 denotes no correlation between the variables. The calculation method is as follows:

$$\rho_{X,Y} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (18)$$

where X and Y are variables and n is the number of samples. x_i and y_i are the samples of X and Y respectively. Also, \bar{x} and \bar{y} are the mean of X and Y respectively.

Figure 6 depicts the results wherein we present the cosine similarity and Pearson correlation coefficient in a line graph to observe their relationship.

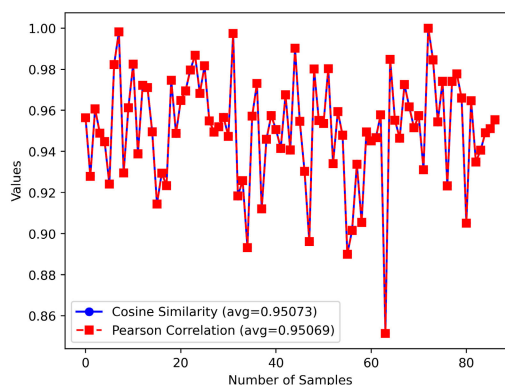


FIGURE 6. The relationship between cosine similarity and Pearson correlation coefficient.

From the overall population, the lowest similarity between the two vectors is about 0.85, while the highest similarity

is close to 1, with the overall similarity remaining between 0.9 and 1. Based on the Pearson correlation coefficient in the experimental results, we can conclude that there exists a very strong linear positive correlation between the answer generated by the model and the correct answer. This implies that a change in one of the vectors can be almost entirely explained by a change in the other, and they tend to change in a very consistent linear direction.

The cosine similarity measure the cosine of the angle between two vectors and is utilized to assess their directional similarity in vector space. When the cosine similarity value is also between 0.9 and 1, it indicates that the angle between the two vectors is very close to 0 degrees, signifying that the directions between two vectors are very similar. The cosine similarity and Pearson correlation coefficient are both about 0.951. The nearly perfect alignment of cosine similarity and Pearson similarity suggests that the answer and correct answer generated by the model are highly similar in space and linear relationship. Therefore, it is reasonable to assume that the results provided by the model are exceedingly close to the correct results.

Furthermore, the MSE serves as a supplementary index in this experiment. MSE quantifies the distance between two vectors, aiding in measuring the degree of difference between them. By computing the mean of the squared variance of the distance between vectors, a more comprehensive evaluation of the similarity between two vectors can be achieved after considering the directional similarity of the vectors. Figure 7 shows the result of MSE:

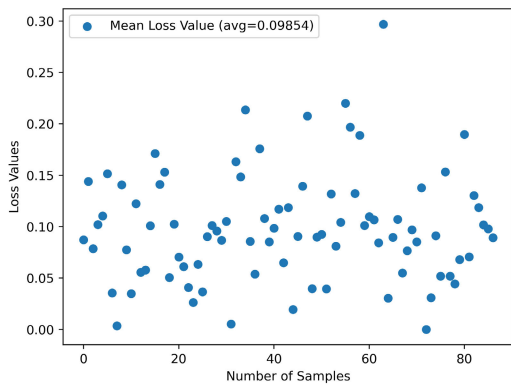


FIGURE 7. The result of MSE.

According to the figure above, except for some boundary points, the MSE value of the model generally remains between 0 and 0.2, and the average MSE is about 0.1. This indicates that the error is kept within an acceptable range. Compared with single character-level or word-level similarity calculation methods such as edit distance and Hamming distance, our comprehensive method not only considers surface similarity but also incorporates semantic information such as word vectors, thereby better capturing the actual semantics of the text. Additionally, our method outperforms traditional approaches that rely solely on single vector

space models or bag-of-words models because we integrate multiple types of similarity metrics, thereby avoiding the bias inherent in a single model.

3) TEXT-TO-SPEECH

In this section, we first evaluate the quality of synthetic speech using specific speech synthesis quality evaluation indicators. Next, we employ an ASR system to convert the synthetic speech back to text, and then compare the accuracy of the transcribed text with the original text used for speech generation. The indicators selected for evaluating the quality of speech synthesis are as follows:

- a) Average SNR (Signal-to-Noise Ratio): SNR represents the ratio of the signal to the noise, and the higher the value, the better the signal quality. When calculating SNR, it is necessary to calculate the signal-to-noise ratio of each signal sample:

$$SNR_i = 10 \log_{10} \left(\frac{P_{\text{signal},i}}{P_{\text{noise},i}} \right) \quad (19)$$

where $P_{\text{signal},i}$ is the signal power of the i -th signal, and $P_{\text{noise},i}$ is the noise power of the i -th signal. Then we need to calculate the average SNR for all signal samples:

$$\text{Average SNR} = \frac{1}{N} \sum_{i=1}^N SNR_i \quad (20)$$

- b) Average Pitch Variation: Pitch variation refers to the range of pitch variations in a signal. APV is calculated as follows:

$$APV = \frac{1}{N} \sum_{i=1}^N \text{Pitch Variation}_i \quad (21)$$

where Pitch Variation_i is the t reble variation of the i -th signal, and N is the total number of signals.

- c) Average Amplitude Consistency: Average amplitude consistency used to evaluate the consistency of the signal amplitude. A low amplitude consistency value means that the amplitude of the signal changes less and the signal is more stable. Below indicates the calculation of Average amplitude consistency:

$$AAC = \frac{1}{N} \sum_{i=1}^N \text{Amplitude Consistency}_i \quad (22)$$

where $\text{Amplitude Consistency}_i$ is the amplitude consistency of the i -th signal, and N is the total number of signals.

In this task, we utilize Edge-TTS for sound synthesis. The speech generated through this method is synthesized directly by the algorithm, typically without the addition of background noise during the synthesis process. Consequently, the experimental SNR tends towards infinity, indicating that the speech generated by the representation is exceptionally clear and free from noise interference.

Figure 8 illustrates the pitch variation for audio samples. In this task, the average pitch change is 1186.12 Hz. This substantial pitch variation indicates a wide range of pitch fluctuations in speech. Such significant pitch variations can enhance the naturalness and emotional expressiveness of speech, particularly in the context of generating dialogue or performative speech.

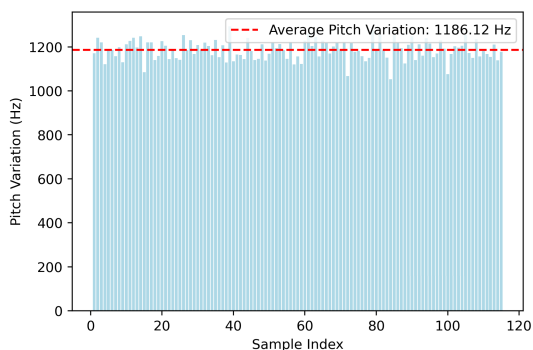


FIGURE 8. The pitch variation for audio samples.

Figure 9 displays the amplitude consistency for audio samples. In this task, the average amplitude consistency is 2.46, which falls within the low consistency range. This indicates that there are some variations in the volume of the audio signal. In speech synthesis, modest amplitude changes can contribute to a more natural-sounding speech, closely resembling human speech patterns.

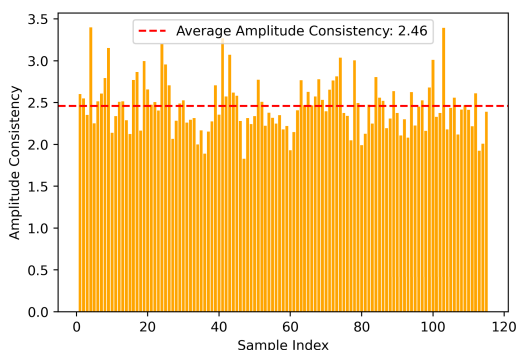


FIGURE 9. The amplitude consistency for audio samples.

When comparing the accuracy of raw text used for speech generation with that of speech converted back to text via ASR systems, we use the same metrics as for text processing: cosine similarity, Pearson correlation coefficient, and MSE. Figure 10 illustrates the relationship between cosine similarity and Pearson correlation coefficient for two texts.

According to Figure 10, we can observe that the cosine similarity and Pearson correlation coefficient values for most samples are 1. Except for a few outliers, the cosine similarity and Pearson correlation values for the remaining samples range between 0.9 and 1. The average cosine similarity and Pearson correlation values both are about 0.99. This indicates a high degree of similarity between the text used to generate speech and the text returned after speech synthesis,

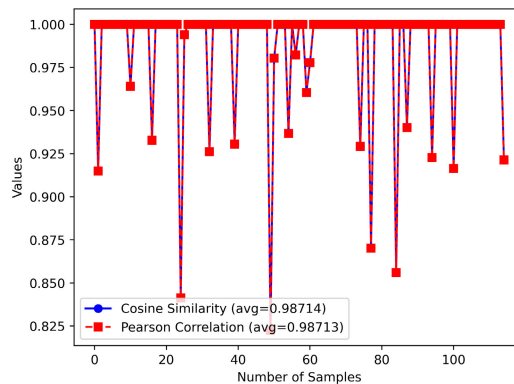


FIGURE 10. The relationship between cosine similarity and Pearson correlation coefficient for two texts.

demonstrating that the TTS results are highly accurate at the content level. Figure 11 shows the MSE of two texts.

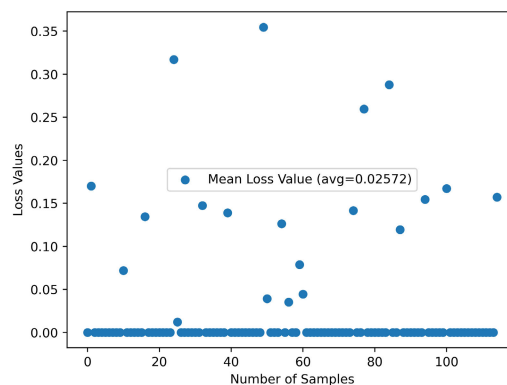


FIGURE 11. The MSE for two texts.

As seen in Figure 11, the MSE values for most samples are 0. Except for a few outliers, the MSE values for other samples range between 0 and 0.2. The average MSE is about 0.026. This result aligns with the observations in Figure 10, indicating that the error in the Chinese text-to-speech process in this task is minimal.

B. COMPREHENSIVE TEST

In this task, we primarily evaluate the performance of the entire system using two metrics. The first metric is the accuracy of generating answers to user questions. The second metric assesses the quality of the generated speech.

1) ACCURACY OF GENERATING ANSWERS

In the overall system evaluation, the system processes questions posed by the user via voice input, converting the speech to text. Using the industrial manual as the knowledge base, it generates answers through text processing. The primary metric we evaluate is the accuracy of the answers. For this, we adhere to the evaluation metrics used in the text processing section. Figure 12 illustrates the relationship between cosine similarity and Pearson correlation coefficient for generated responses. Overall, we observe that both cosine similarity and Pearson correlation coefficients are approximately 0.942.

This value is slightly lower than the results obtained from unit tests of Chinese text processing. Considering the potential errors introduced by the TTS conversion, the decline in accuracy remains within an acceptable range. Consequently, our output of the model maintains a high level of accuracy.

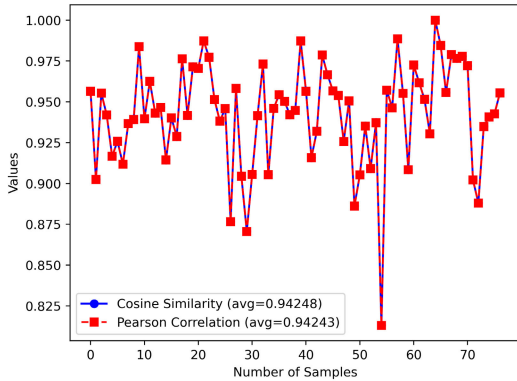


FIGURE 12. The relationship between cosine similarity and Pearson correlation coefficient for the generating answers.

Figure 13 shows the average MSE for generating answers. There is a slight increase in the average MSE value compared to text processing alone. The MSE values for most samples range from 0 to 0.25 and the average MSE is about 0.12. Overall, considering the errors introduced by the STT conversion, the slightly increased MSE values remain within an acceptable range.

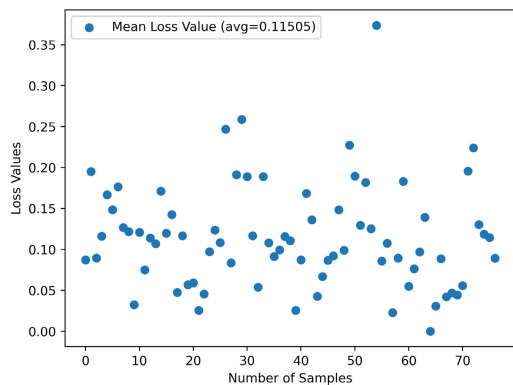


FIGURE 13. The MSE for generating answers.

In this experiment, the key factor that plays a decisive role in the experimental results is the accuracy of generating answers based on user questions. Firstly, we tested the BERT model and the ALBERT model based on the fine-tuning method (FiTT) [42]. Here, we use F1 scores to measure the average overlap between predictions and labeled answers. Table 4 shows the comparison results of F1 scores for different models.

The results show that the performance of the trained ALBERT model is significantly better than that of the BERT model. Compared to these two models, our model demonstrates a substantial improvement in accuracy. The F1 score of our model is 7.5% higher than that of the

TABLE 4. Comparison results of F1 scores for different models.

Model	F1 Score (%)
BERT (FiTT)	86.0
ALBERT (FiTT)	87.4
Our Model	93.5

trained BERT model and 6.1% higher than that of the trained ALBERT model.

Additionally, because the method used in this experiment is relatively novel in the industrial field, there has been limited similar work conducted. Therefore, at the level of model comparison, we perform a comparison for similar problems in different fields.

Using cosine similarity as an evaluation metric, we conducted a comparison between the question answering system in the petroleum industry and the question answering system in the electric power industry. The Q&A system in the petroleum industry primarily utilizes the Word2Vec model and a training model [43]. In contrast, the Q&A system in the electric power industry employs a method based on the combination of TF-IDF and cosine similarity to calculate similarity when determining accuracy [44]. Table 5 shows the results of this horizontal comparison:

TABLE 5. Comparison results of cosine similarity for different fields.

Model	Accuracy (%)
TF-IDF algorithm (single)	62.3
TF-IDF algorithm (cosine similarity)	87.4
Word2vec (Self-trained)	93.3
Our Model	94.2

According to the above results, we can gain a general understanding of the relative accuracy of question answering systems across different industries. It is evident that in the industrial field, our system shows a certain advantage in accuracy compared to other fields. However, the results of this comparison are only for reference, as different models emphasize different aspects within their respective fields. For instance, the medical field may involve more technical terms and complex sentence structures, while industrial manuals focus more on technical instructions and procedures.

Finally, compared to some other specific similarity calculation models, such as DSSM and SimCSE, our method computes the embedded vector through OpenAI and then computes the cosine similarity directly, which is more general and flexible. In terms of model training and complexity, both DSSM and SimCSE require training on large-scale datasets, which involves complex preprocessing, model architecture design, and parameter tuning. Conversely, the embedding method from OpenAI does not necessitate user-led model training. Instead, users can simply call the API to obtain high-quality embedding vectors, significantly simplifying the utilization process.

From the perspective of universality, DSSM is typically trained for specific tasks or domains, with model performance highly dependent on the data and features of the particular

task, leading to poor transferability. SimCSE generates sentence embeddings through contrastive learning, which can capture general semantic information to some extent. However, the performance of model remains contingent on the coverage of the training data. In contrast, embedding method of OpenAI, being based on a large-scale pre-trained model, adapts well to various types of text data, whether short or long, generating high-quality embeddings.

In terms of flexibility, DSSM is optimized for specific tasks, resulting in poor transferability across different tasks and limited flexibility. SimCSE, while generating embeddings through contrastive learning and offering some flexibility, may still require additional adjustments for certain tasks and necessitates significant computational resources for training. The OpenAI embedding method, leveraging cosine similarity calculation, facilitates easy comparison of text similarities. It is suitable for various application scenarios and offers strong flexibility. Moreover, it can be seamlessly combined with other methods, such as clustering and classification, to further extend the application range and meet diverse needs without requiring additional training or adjustments.

2) SPEECH QUALITY EVALUATION

After obtaining the answer to the question of user, we convert the text of the answer into speech as the final output. Although this part has been unit tested, it is necessary to re-evaluate the quality and accuracy of the output speech during the comprehensive testing phase. For the SNR metric, similar to the results in the TTS section, the value of the model output here is still very high. This indicates that the generated speech is clean and free of noise. Figure 14 shows the average pitch variation of the generated speech. The value of 1198.82 Hz represents a significant amplitude of pitch change, indicating substantial pitch variation in the audio sample. This is a slight improvement from the 1186.12 Hz observed in the unit tests. Such pitch variation often makes the synthesized speech sound more natural and expressive.

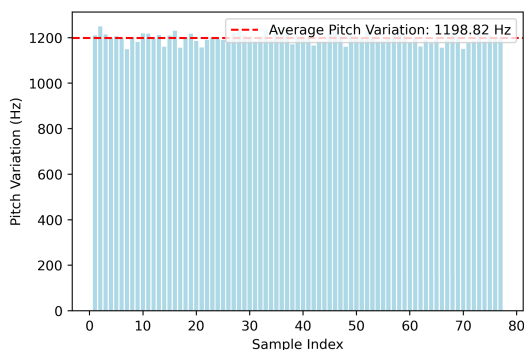


FIGURE 14. The pitch variation for generating answers.

Figure 15 shows the average amplitude consistency of the generated answer speech, with a value of 2.62, which is slightly higher than the 2.46 observed in the unit test. Moderate amplitude variation can make the speech sound

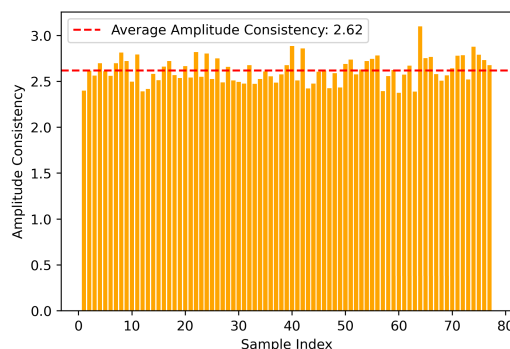


FIGURE 15. The amplitude consistency for generating answers.

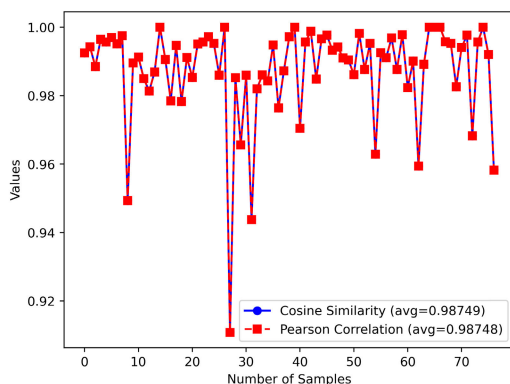


FIGURE 16. The cosine similarity and the Pearson correlation coefficient between the generated speech content and the transcribed speech text.

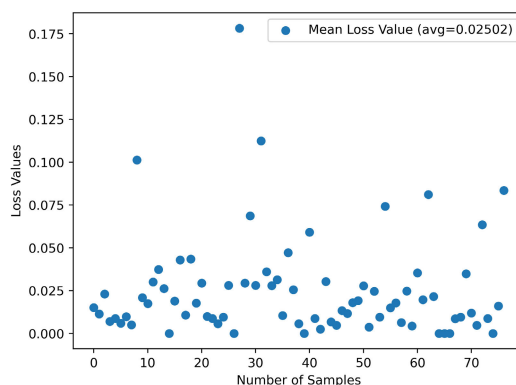


FIGURE 17. The MSE between the generated speech content and the transcribed speech text.

more natural and closer to human speech patterns, as the volume of human speech is not uniform. This variation in amplitude also enhances the intelligibility and audibility of the speech, preventing it from sounding monotonous or mechanical.

Finally, we need to evaluate the content accuracy of the generated speech. The evaluation metrics are the same as those used in the text processing section. Figure 16 shows the Pearson correlation coefficient and cosine similarity between the generated speech content and the transcribed speech text. Both the mean Pearson correlation coefficient and the cosine similarity value are approximately 0.99. This value is consistent with the results obtained in the unit tests, indicating

that the TTS content is of high quality and that the model performs reliably.

Figure 17 indicates the MSE between the generated speech content and the transcribed speech text. The average MSE value of nearly 0.025 is slightly lower than the 0.026 observed in the unit test, indicating that the model performed better in the comprehensive test.

VI. CONCLUSION AND FUTURE WORK

A. CONCLUSION

In this paper, we address the challenge of robot fault maintenance in industrial production by designing an interactive question and answer system. Workers can voice specific questions about robot failures to the system, which responds in voice format based on the industrial manual. Upon testing the system with the test set, we observed an average cosine similarity of 0.942, an average Pearson correlation coefficient of 0.942, and an average MSE of 0.115. This result is excellent both for other models in the field and for cross-domain question answering systems. It also has been found that the performance of the medium Whisper model in Chinese speech-to-text tasks is comparable to that of large model, while significantly reducing the number of parameters. Additionally, the Edge-TTS library excels in Chinese text-to-speech tasks, producing transcriptions that are clear, natural, and highly accurate.

The results of this experiment also have a significant impact on the industrial field. Firstly, it addresses the gap in industrial robot maintenance question-answering systems while ensuring high accuracy. Secondly, in practical applications, it reduces the time workers spend reading through extensive industrial manuals, thereby significantly enhancing their work efficiency. Finally, the system is highly interactive through voice interaction, it allows workers to easily obtain the information they need, thus improving the overall user experience. These results indicate strong performance of the model on the test set, affirming the practicality and effectiveness of our design.

B. FUTURE WORK

Currently, our model effectively processes plain text industrial manuals. However, it faces limitations when dealing with manuals that contain numerous images, where critical information may be embedded in the visual content. In future research, we aim to address these challenges by prioritizing two key areas:

- a) Enhancing textual content accuracy: We will focus on improving the accuracy of the existing model in handling textual content. This will involve refining our NLP techniques and optimizing the performance of model to ensure that it accurately interprets and responds to textual queries.
- b) Integrating image processing technologies: To overcome the limitations of processing visual content, we plan to incorporate advanced image processing

technologies into the model. Specifically, we will explore the application of optical character recognition (OCR) technology to extract text information from images. This extracted text will then be integrated with the existing textual content, allowing the model to comprehensively interpret and utilize both text and image-based information from industrial manuals.

By addressing these areas, we aim to create a more robust and versatile question-answering system that can effectively handle the diverse content found in industrial manuals, thereby further enhancing work efficiency and user experience.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to all those who have contributed to the completion of this research work. At the same time, they also extend their best regards to the anonymous reviewers and the associate editor. Thank you all for your invaluable contributions.

REFERENCES

- [1] Z. Pan, J. Polden, N. Larkin, S. V. Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," in *Proc. ISR 41st Int. Symp. Robotics ROBOTIK 6th German Conf. Robotics*, Jun. 2010, pp. 1–8.
- [2] W.-K. Chen, *Linear Networks and Systems*. Belmont, CA, USA: Wadsworth, 1993, pp. 123–135.
- [3] Y. Li, Y. Zhang, A. Pan, M. Han, and E. Veglianti, "Carbon emission reduction effects of industrial robot applications: Heterogeneity characteristics and influencing mechanisms," *Technol. Soc.*, vol. 70, Aug. 2022, Art. no. 102034.
- [4] A. N. Khan, X. En, M. Y. Raza, N. A. Khan, and A. Ali, "Sectorial study of technological progress and CO₂ emission: Insights from a developing economy," *Technol. Forecasting Social Change*, vol. 151, Feb. 2020, Art. no. 119862.
- [5] Q. Ma, M. Tariq, H. Mahmood, and Z. Khan, "The Nexus between digital economy and carbon dioxide emissions in China: The moderating role of investments in research and development," *Technol. Soc.*, vol. 68, Feb. 2022, Art. no. 101910.
- [6] C. Gao, S. Tao, Y. He, B. Su, M. Sun, and I. A. Mensah, "Effect of population migration on spatial carbon emission transfers in China," *Energy Policy*, vol. 156, Sep. 2021, Art. no. 112450.
- [7] R. Y. Zhong, X. Xu, E. Klotz, and S. T. Newman, "Intelligent manufacturing in the context of industry 4.0: A review," *Engineering*, vol. 3, no. 5, pp. 616–630, Oct. 2017.
- [8] J. Arents and M. Greitans, "Smart industrial robot control trends, challenges and opportunities within manufacturing," *Appl. Sci.*, vol. 12, no. 2, p. 937, Jan. 2022.
- [9] W. Leontief, "National perspective: The definition of problems and opportunities," in *The Long-Term Impact of Technology*. New York, NY, USA: Academic, 1983, pp. 3–7.
- [10] C. B. Frey and M. A. Osborne, "The future of employment: How susceptible are jobs to computerisation?" *Technol. Forecasting Social Change*, vol. 114, pp. 254–280, Jan. 2017.
- [11] G. Graetz and G. Michaels, "Is modern technology responsible for jobless recoveries?" *Amer. Econ. Rev.*, vol. 107, no. 5, pp. 168–173, May 2017.
- [12] S. Mittal, M. A. Khan, D. Romero, and T. Wuest, "Smart manufacturing: Characteristics, technologies and enabling factors," *Proc. Inst. Mech. Eng., Part B: J. Eng. Manuf.*, vol. 233, no. 5, pp. 1342–1361, Apr. 2019.
- [13] Y. Gan, J. Duan, and X. Dai, "A calibration method of robot kinematic parameters by drawstring displacement sensor," *Int. J. Adv. Robotic Syst.*, vol. 16, no. 5, Oct. 2019, Art. no. 172988141988307.
- [14] J.-Q. Xuan and S.-H. Xu, "Review on kinematics calibration technology of serial robots," *Int. J. Precis. Eng. Manuf.*, vol. 15, no. 8, pp. 1759–1774, Aug. 2014.
- [15] I.-W. Park, B.-J. Lee, S.-H. Cho, Y.-D. Hong, and J.-H. Kim, "Laser-based kinematic calibration of robot manipulator using differential kinematics," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 6, pp. 1059–1067, Dec. 2012.

- [16] H. Wang, S. Shen, and X. Lu, "A screw axis identification method for serial robot calibration based on the POE model," *Ind. Robot: Int. J.*, vol. 39, no. 2, pp. 146–153, Mar. 2012.
- [17] F. Yáñez, "The goal is Industry 4.0: Technologies and trends of the fourth industrial revolution," Independ. Published, U.K., 2017.
- [18] B. Chen, J. Wan, L. Shu, P. Li, M. Mukherjee, and B. Yin, "Smart factory of Industry 4.0: Key technologies, application case, and challenges," *IEEE Access*, vol. 6, pp. 6505–6519, 2018.
- [19] O. Amri, M. Fri, M. Msaaf, and F. Belmajdoub, "Industrial systems Fault's diagnosis approaches adapted to the context of the smart industry," in *Proc. 3rd Int. Conf. Innov. Res. Appl. Sci., Eng. Technol. (IRASET)*, May 2023, pp. 1–5.
- [20] S. H. Zad, R. H. Kwong, and W. M. Wonham, "Fault diagnosis in discrete-event systems: Framework and model reduction," *IEEE Trans. Autom. Control*, vol. 48, no. 7, pp. 1199–1212, Jul. 2003.
- [21] A. Alashter, F. Gu, A. D. Ball, and Y. Cao, "Fault diagnosis of broken rotor bar in AC induction motor based on a qualitative simulation approach," in *Proc. 25th Int. Conf. Autom. Comput. (ICAC)*, Sep. 2019, pp. 1–6.
- [22] L. Guo, Y. Lei, S. Xing, T. Yan, and N. Li, "Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabeled data," *IEEE Trans. Ind. Electron.*, vol. 66, no. 9, pp. 7316–7325, Sep. 2019.
- [23] R. Pinto and T. Cerquitelli, "Robot fault detection and remaining life estimation for predictive maintenance," *Proc. Comput. Sci.*, vol. 151, pp. 709–716, May 2019.
- [24] H.-K. Hsu, H.-Y. Ting, M.-B. Huang, and H.-P. Huang, "Intelligent fault detection, diagnosis and health evaluation for industrial robots," *Mechanics*, vol. 27, no. 1, pp. 70–79, Feb. 2021.
- [25] F. Cheng, A. Raghavan, D. Jung, Y. Sasaki, and Y. Tajika, "High-accuracy unsupervised fault detection of industrial robots using current signal analysis," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Jun. 2019, pp. 1–8.
- [26] A. H. Sabry, F. H. Nordin, A. H. Sabry, and M. Z. A. A. Kadir, "Fault detection and diagnosis of industrial robot based on power consumption modeling," *IEEE Trans. Ind. Electron.*, vol. 67, no. 9, pp. 7929–7940, Sep. 2020.
- [27] S. Umehara, T. Oshima, A. Ishigaki, and S. Yasui, "Effect of an instruction manual using an e-learning system on the improvement of assembly work," in *Proc. 9th Int. Congr. Adv. Appl. Informat. (IIAI-AAI)*, Sep. 2020, pp. 785–790.
- [28] S. Windmann and O. Niggemann, "Information retrieval in industrial production environments," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, vol. 1, Sep. 2018, pp. 1205–1208.
- [29] J.-X. Chen, G.-S. Huang, and M.-H. Yen, "A study on the application of industrial robot based on the integration of speech recognition and image processing," in *Proc. Int. Conf. Syst. Sci. Eng. (ICSSE)*, Aug. 2021, pp. 403–407.
- [30] H. Choi, T. Hamanaka, and K. Matsui, "Design and implementation of interactive product manual system using chatbot and sensed data," in *Proc. IEEE 6th Global Conf. Consum. Electron. (GCCE)*, Oct. 2017, pp. 1–5.
- [31] G. Siwach and C. Li, "Enhancing human cobot interaction using natural language processing," in *Proc. IEEE 4th Int. Multidisciplinary Conf. Eng. Technol. (IMCET)*, Dec. 2023, pp. 21–26.
- [32] J. Ye, "Cosine similarity measures for intuitionistic fuzzy sets and their applications," *Math. Comput. Model.*, vol. 53, nos. 1–2, pp. 91–97, Jan. 2011.
- [33] Y. H. Ghadage and S. D. Shelke, "Speech to text conversion for multilingual languages," in *Proc. Int. Conf. Commun. Signal Process. (ICCCSP)*, Apr. 2016, pp. 0236–0240.
- [34] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2023, pp. 28492–28518.
- [35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–22.
- [36] X. Yang, K. Yang, T. Cui, M. Chen, and L. He, "A study of text vectorization method combining topic model and transfer learning," *Processes*, vol. 10, no. 2, p. 350, Feb. 2022.
- [37] H. Kozima, "Text segmentation based on similarity between words," 1996, *arXiv preprint cmp-lg/9601005*.
- [38] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, 1967, pp. 281–297.
- [39] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with ChatGPT," 2023, *arXiv:2302.11382*.
- [40] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise Reduction Speech Process.* Springer, 2009, pp. 1–4.
- [41] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature," *Geoscientific Model Develop.*, vol. 7, no. 3, pp. 1247–1250, Jun. 2014.
- [42] Y. Chen and F. Zulkernine, "BIRD-QA: A BERT-based information retrieval approach to domain specific question answering," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2021, pp. 3503–3510.
- [43] L. Xia, W. Boyu, L. Lixia, and L. Ruidi, "Question-answering using keyword entries in the oil&gas domain," in *Proc. IEEE Int. Conf. Power, Intell. Comput. Syst. (ICPICS)*, Jul. 2020, pp. 282–286.
- [44] F. Meng, W. Wang, and J. Wang, "Research on short text similarity calculation method for power intelligent question answering," in *Proc. 13th Int. Conf. Comput. Intell. Commun. Netw. (CICN)*, Sep. 2021, pp. 91–95.



ZECHENG REN received the master's degree in health data science from London School of Economics and Political Science, in 2023. He is currently working as a Research Assistant with the Greater Bay Area Institute for Innovation, Hunan University, Guangzhou, China. His research interests include computer vision and natural language processing.



ZENGNAN YU is currently pursuing the degree with Shanghai University, China. He is also working as an Intern with Asia-Pacific Artificial Intelligence Association. His research interests include artificial intelligence, computer science, and software engineering.



WENYI ZHANG is currently pursuing the degree in robotics engineering with Nanjing University of Information Science and Technology. Her research interests include robot kinematics analysis and circuit system design.



QUIJIANG LEI received the Ph.D. degree from Delft University of Technology, Delft, The Netherlands, in 2018. He is currently working as a Professor with the Greater Bay Area Institute for Innovation, Hunan University. His research interests include intelligent robots, collaborative robots, robotic systems integration, robot vision, artificial intelligence, machine learning, and human–computer interaction.

• • •