information

MDPI

Article

Design and Implementation of an Intelligent Web Service Agent Based on Seq2Seq and Website Crawler

Mei-Hua Hsih [1], Jian-Xin Yang [2] and Chen-Chiung Hsieh [3,*]

[1] Department of Product Design, School of Arts and Design, Sanming University, Sanming 365004, China; 20220204@fjsmu.edu.cn
[2] QSAN Technology, Inc., Taipei City 114, Taiwan; axince01@gmail.com
[3] Department of Computer Science and Engineering, Tatung University, Taipei City 104, Taiwan
* Correspondence: cchsieh@gm.ttu.edu.tw; Tel.: +886-2-21822928 (ext. 6571)

Abstract: This paper proposes using a web crawler to organize website content as a dialogue tree in some domains. We build an intelligent customer service agent based on this dialogue tree for general usage. The encoder-decoder architecture Seq2Seq is used to understand natural language and then modified as a bi-directional LSTM to increase the accuracy of the polysemy cases. The attention mechanism is added in the decoder to improve the problem of accuracy decreasing as the sentence grows in length. We conducted four experiments. The first is an ablation experiment demonstrating that the Seq2Seq + Bi-directional LSTM + Attention mechanism is superior to LSTM, Seq2Seq, Seq2Seq + Attention mechanism in natural language processing. Using an open-source Chinese corpus for testing, the accuracy was 82.1%, 63.4%, 69.2%, and 76.1%, respectively. The second experiment uses knowledge of the target domain to ask questions. Five thousand data from Taiwan Water Supply Company were used as the target training data, and a thousand questions that differed from the training data but related to water were used for testing. The accuracy of RasaNLU and this study were 86.4% and 87.1%, respectively. The third experiment uses knowledge from non-target domains to ask questions and compares answers from RasaNLU with the proposed neural network model. Five thousand questions were extracted as the training data, including chat databases from eight public sources such as Weibo, Tieba, Douban, and other well-known social networking sites in mainland China and PTT in Taiwan. Then, 1000 questions from the same corpus that differed from the training data for testing were extracted. The accuracy of this study was 83.2%, which is far better than RasaNLU. It is confirmed that the proposed model is more accurate in the general field. The last experiment compares this study with voice assistants like Xiao Ai, Google Assistant, Siri, and Samsung Bixby. Although this study cannot answer vague questions accurately, it is more accurate in the trained application fields.

Keywords: natural language processing; intelligent customer service agent; website crawler; deep learning; LSTM; Seq2Seq

check for updates

Citation: Hsih, M.-H.; Yang, J.-X.; Hsieh, C.-C. Design and Implementation of an Intelligent Web Service Agent Based on Seq2Seq and Website Crawler. *Information* **2024**, *15*, 818. https://doi.org/10.3390/info15120818

Academic Editor: Anselmo Peñas

Received: 10 October 2024
Revised: 13 December 2024
Accepted: 16 December 2024
Published: 20 December 2024

1. Introduction

In our daily life, many questions always need to be answered. At this time, we usually ask others for advice, look up books, or search for answers on the Internet. In the 21st century, looking for answers online has become the most common way. With the changes of the times, in addition to basic food, clothing, housing, and transportation, human beings have become increasingly diverse and complex in medical care, entertainment, finance, and logistics. The resulting problems have also gradually increased. Traditional human customer service is slow or unable to answer these questions accurately. They are progressively unable to keep up with the fast pace of modern life regarding their responses' speed, quality, or professionalism. Coupled with the continuous improvement of computer hardware technology, artificial intelligence, which was initially impossible to achieve, began to flourish. Hence the emergence of intelligent customer service agents.

Intelligent customer service agent is a question-answering system based on much knowledge. Currently, most provide services in the target knowledge domain, using technologies such as natural language understanding (NLU), big data (Big Data), and knowledge management. And because the system is built on servers, it's not like people need breaks or shifts. It can provide 24-h professional answers and services.

At the end of the last century, many scholars have already been researching intelligent customer service development. In 1950, Turing [1] proposed the famous Turing test [2] in his paper "Computing Machinery and Intelligence" as in Figure 1. This standard is the ultimate goal of every natural language research. In 1966, Weizenbaum [3] published the world's first chatbot, ELIZA, to imitate a psychologist's interaction with a patient. Although it only used simple keyword matching and replying to rules, the bot still exceeded the development team's expectations. In 1988, the University of California, Berkeley, developed "UC" to help users learn to use the UNIX system. An intelligent customer service agent can now analyze the input language, understand the user's intention, and select the appropriate dialogue content to answer the user. In 1995, Wallace [4] developed the ALICE system, and the AIML language was released along with ALICE. With the high development of AI technology, intelligent customer service has flourished in recent years. Companies such as Google, Amazon, and Microsoft have successively invested in research in this field.

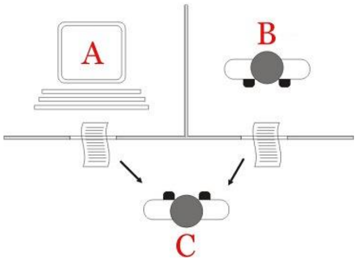


Figure 1. Turing test. C asks A and B whether they are human [2].

In addition to the rapid technological advancement, commercial demands have significantly increased the demand for intelligent customer service. With the development of the Internet, the way people communicate constantly evolves from traditional phone calls and faxes to e-mail, 3/4/5G communication, and current communication software such as Line and Skype. People's communication patterns are getting faster and faster. Therefore, traditional customer service has gradually failed to keep up with the times and has become a service that needs to be transformed.

Two main problems need to be solved while developing intelligent customer services. One is the enormous internet resources and almost all products or services have multiple websites. However, current usage of these websites mostly requires users to browse and click on their own. Therefore, when the content and structure of the website are slightly larger, users often need to spend a lot of time to find the answer. The other is to create a general natural language processing agent. Most current intelligent customer service agents are developed for general usage rather than specific fields. However, different fields may require corresponding training on brilliant customer service before being used. For example, the intelligent customer service agent of a bank needs to be trained using financial-related data, and the intelligent customer service agent of the Center for Disease Control and Prevention needs to be trained using disease-related data. Whenever a field is crossed, it must be retrained, and the service agent cannot be universal.

Therefore, our main contributions are to solve the above two problems: the first is to deploy a web crawler fetching web pages and using the knowledge on that website to build a knowledge base as a dialogue tree for intelligent customer service. The other is to adopt the Seq2seq model consists of Bi-LSTM layers with attention mechanism introduced to create an intelligent customer service agent. In the proposed model, the training mechanism is changed from the field's knowledge base to the grammar of that language.

This improves the versatility of the service. The ultimate goal is to target similar web services without training. As long as the relevant websites in this field are made into a dialogue tree (knowledge base), the knowledge of the intelligent customer service agent can be rapidly expanded.

This research is divided into five sections. The first is the introduction, which explains the motivation and goal of the study. The second section is divided into two parts. The first part introduces the current standard methods of natural language understanding, including rule-based, traditional machine learning, Recurrent Neural Networks (RNN), Long Short-Term Memory Networks (LSTM), and Encoder-Decoder framework-related papers. The second part introduces several practices of using crawlers to build knowledge bases. The third section discusses the details of the implementation of this research. Use the Seq2seq model and introduce the concept of the attention mechanism and bidirectional LSTM. Combined with a crawler and dialogue tree, the knowledge base can be rapidly expanded. Therefore, realize a cross-domain intelligent customer service. The fourth section discusses the system's functional testing and experimental results. The last section is the conclusion, which presents this study's contributions, achievements, and prospects.

2. Related Works

According to the assessment of the maturity of intelligent software web agents [3], there are four basic functions for an intelligent software agent: autonomy, reactivity, proactiveness, and social ability. Here, this study satisfies autonomy and reactivity by using a web crawler to adapt to environmental changes. The pursuit of proactive goal-directed behavior and social ability is fulfilled by natural language processing.

2.1. Natural Language Processing Techniques

The use of natural language processing (NLP) in customer service [4] is growing quickly. It allows users to communicate with models using different languages through text or speech, and the model will provide answers to the users. To improve the interaction between humans and machines, natural language processing is implemented [5] to handle the understanding and generation of the chatting language. The system consists of natural language processing, multi-agent, user interface, and knowledge base. However, Huang's system [5] could only deal with text documents.

2.1.1. Rule-Based

The commonly used natural language processing method in simpler environments is rule-based [6,7]. This method is based on many manual rules to establish a response mechanism quickly. However, when the system becomes more and more complex, this method will result in a tremendous amount of data that is difficult to maintain. Therefore, it is only suitable for use in small projects. From the 1960s to the 1980s, most of the more famous natural language processing systems used this method, such as the SHRDLU system [6] developed by Winograd and the ELIZA [7] designed by Weizenbaum. In the 1970s, researchers began to use real-world information for processing and make it into data that computers can understand. Examples [8] include MARGIE, SAM, PAM, TaleSpin, QUALM, Politics, and Plot Unit.

2.1.2. Machine Learning

In the late 1980s, natural language processing entered the era of machine learning, and the most significant cause was the gradual increase in computer's computing power. The earliest used machine learning algorithms, such as decision trees, were systems composed of if-else, similar to the rule-based method mentioned earlier. Later, the function of part-of-speech tagging was introduced into NLP by adding the Hidden Markov Model (HMM). Research has also gradually developed into probability-based statistical models that align with the concept of human language.

After machine learning became a mainstream research field, machine translation was first developed, and more complex statistical models were gradually developed. However, as time progresses, machine learning has gradually encountered bottlenecks. With the increase in domains, data becomes too dispersed to be accurately classified, leading to a shift towards domain knowledge in application scenarios. The drawback is that it requires creating and training a custom corpus for different contexts. This process is time-consuming and is a major limiting factor for machine learning [9].

Rasa NLU [10] is an open-source natural language processing framework and one of the best machine learning solutions that can be achieved. Its backend supports many pipelines, such as spaCy, MITIE, tensor flow, etc. These pipelines have different characteristics. For example, Sklearn has a better intent classification, MITIE installation, faster training speed, and better feature vector and keyword recognition results. In this study, we use the Sklearn + Jieba + MITIE as the experimental framework for the control group. RasaNLU divides the user's input sentences into three parts: Intent, Entity, and Context. Intent represents the user's intention, Entity refers to the keywords in the sentence, and Context indicates the state, representing the current stage of the conversation, such as beginning, questioning, answering, or ending.

2.1.3. Recurrent Neural Network

Recurrent Neural Network (RNN) can be traced back to a paper by Jeffrey Elman in 1990 [11]. This paper mainly proposes how to find specific patterns and structures from time series, which is the original concept of RNN. Later, because of the characteristics of RNN taking time into account, it began to be used for research in fields such as machine translation and natural language processing. Language usually has contextual relevance, which makes general neural networks less suitable for language processing because of the problem of being out of context. That is to say, if the context can be considered, the accuracy rate will be significantly improved.

RNN and various models derived from it are the most commonly used neural network models in natural language processing research. RNN would consider the relationship between the current word and the previous text and make changes to the current state. Assuming X_t is the input at a particular time point t, the result (h_t) is given after passing through the RNN, and the result (h_t) is continuously passed to the next layer (h_t + X_{t+1}) as a subsequent judgment basis.

The shortcomings of RNN can also be seen in this model. Even as the sentence (time) gets longer, the influence of the words in front will be diluted or even forgotten during transmission. This phenomenon is known as the vanishing gradient. Some scholars have proposed the Long Short-Term Memory (LSTM) to solve this problem.

2.1.4. Long Short-Term Memory

Hochreiter and Schmidhuber proposed the LSTM [12] in 1997 to solve the problems RNNs face through the memory function. LSTM adds three new gates to implement the memory function and decide which memories need to be retained and which are not. The three valves that solve the memory problem are the forget, input, and output gates. The forget valve, denoted as ft, is usually a sigmoid function. A binary classification function determines whether past words should be retained. This valve will filter out the last sentence if the current sentence is a new topic or the opposite of the previous sentence. Otherwise, it will continue to be retained in memory. To judge whether the last word is related to the current word, all words will be converted into vectors first, and then the similarity of the two vectors will be calculated. If it is close to 1, the meanings of the two words are similar. On the other hand, if it is close to 0, the two words have opposite meanings.

The input valve, denoted by I_t, determines whether the current input word is added to the Long-Term Memory. The purpose is to filter out preliminary predictions. Ignore recent, less relevant results before they influence future forecasts. The output valve refers to the word reserved in the past and the current word to determine whether the current word is added to the output. This valve also uses the sigmoid function to indicate whether to join. Then, whether Long-Term memory is added to the output, the tanh function is usually used; the value will fall between [−1, 1], and −1 means to remove long-term memory.

2.1.5. Encoder-Decoder Framework

In the LSTM model, we deal with the problem of the input and output sequences being of equal length. However, many input and output sequences, such as machine translation, speech recognition, and question-answering systems, have different lengths. At this time, we need to design an RNN structure that can convert sentences of indefinite lengths to each other, which is the Encoder-Decoder framework. The Encoder-Decoder framework [13] is a product of the machine translation model, first proposed in 2014 in the Seq2Seq recurrent neural network.

In the translation model, the training steps can be divided into many steps, such as preprocessing, word alignment, phrase alignment, phrase feature extraction, language model training, and feature weight learning. The basic idea of the Seq2Seq model is straightforward. It is to use a recurrent neural network to read the input sentence and compress the information of the entire sentence into a fixed-dimensional code; then, use another recurrent neural network to read the code and decompress it into a sentence in the target language. These two recurrent neural networks are called encoder and decoder, respectively, which is the origin of the encoder-decoder framework.

2.1.6. Transformer

The Transformer model [14] has revolutionized natural language processing (NLP) by introducing a novel architecture that relies on self-attention mechanisms rather than recurrent or convolutional layers. This allows the model to process entire sentences simultaneously, capturing long-range dependencies more effectively. Transformers excel in tasks such as translation, summarization, and question answering due to their ability to handle large datasets and learn contextual relationships between words. The architecture's scalability and parallelization capabilities have led to the development of powerful models like BERT [15] and GPT [16], which have set new benchmarks in various NLP tasks. Overall, the Transformer model has significantly advanced the field of NLP, enabling more accurate and efficient language understanding and generation.

Transformers, while powerful, have several disadvantages [17]. The first is high computational demand. Training and running Transformer models require substantial computational resources, which can be costly and time-consuming. The second is data sensitivity. The performance of Transformers is highly dependent on the quality and quantity of training data. Limited or biased data can adversely affect model performance. The last is training time. The training process for Transformers can be lengthy, hindering quick experimentation and development. These challenges highlight the need for ongoing research to optimize and make Transformer models more accessible and efficient. For our application which applies natural language processing to get user intent and entities, the Transformer model is not deployed to save lots of data collection and training time. Seq2Seq models are indeed simpler and more interpretable compared to Transformers. They have been quite effective for various NLP tasks such as machine translation, text summarization, and more. One of their key advantages is that they require less computational power to train, making them accessible for a wider range of applications.

2.2. Create a Knowledge Base Using Website Crawlers

In the development of data science, because of the explosive growth of the Internet, there is a trend of collecting exciting data from the Internet, so there are web crawlers. Previous research has mainly focused on the algorithms of web crawlers to improve efficiency and collect as many relevant web pages as possible. Most of these studies are based on a single crawler. Therefore, Rungsawang and Angkawattanawit [18] proposed an algorithm to learn to gain experience in the process of multiple crawling, thereby increasing the accuracy of the knowledge base. Their method consists of the following three steps:

Step 1: Analyze the relevance of the hyperlinked content to the target topic. By hyperlinking the relevant HTML tags, the spatial vector of this information is analyzed to determine the target direction of the crawler after the second pass. Baeza-Yates and Ribeiro-Neto [19] used the spatial method to sort words into different vectors based on their relatedness. Assuming the two web pages are related, the vectors point in the same direction.

Step 2: Decide whether these pages are sufficiently professional to be relevant to the target topic. Kleinberg [20] proposed using a hyperlink-induced topic search (HITS) algorithm to rank these pages according to their authority.

Step 3: Use the above information to make the final web crawler direction. Many methods can be used, and the one used is the open directory project (ODP).

In building a knowledge base with crawlers, because the information on the Internet is too much and too complicated, Berners-Lee [21] proposed the semantic web concept. The idea is to establish a unified standard. According to this standard, computer-understandable semantics (tags) are added to HTML files on all websites. In this way, the vast amount of data on the Internet becomes a set of rules that can be followed. Various functions can also be derived faster.

The concept of semantic web has not been popularized on the internet. However, this concept can still be used in a personal knowledge base to understand better and apply the knowledge base of computers. Kim and Ha [22] refer to formal languages such as RDF [23] and OWL [24] to make the internet more accessible for computers to analyze. Their goal was to build a small business knowledge base. Offer a helpful marketing strategy for small businesses or provide a dynamic QA system. When crawling the data on the internet, they divided the data into static data and dynamic data. Static data is data on the general internet. After climbing down these data, analysis and semantic annotation actions are performed, and a knowledge base is built. Another data source is dynamic data, which is users' feedback. So, dynamic data changes over time. The service can be closer to the user's needs over time by building a knowledge base through real-time dynamic data acquisition.

Over the years, several chatbots have been created, but each has its limitations and difficulty. Choudhary and Chauhan [25] suggested a method to upgrade the performance of the chatbots so that they can respond to the user query with better accuracy. The proposed model is based on Long Short-Term Memory (LSTM) with attention mechanism, Bag of Words (BOW), and beam search decoding. The sequence-to-sequence (Seq2Seq) architecture with an LSTM encoder and decoder has been implemented. The Dialog Dataset is used to train and test the model, and the Bleu-N algorithm is used to evaluate the chatbot's accuracy. However, unidirectional LSTM may not be enough to deal with complex sentences.

Budaev [26] proposed an intelligent analysis of customer feedback based on the use of a modified seq2seq deep learning model. Since the basic seq2seq model has a significant disadvantage "the inability to concentrate on the main parts of the input sequence, the results of machine learning may give an inadequate assessment of customer feedback". This disadvantage is eliminated using a model with an "attention mechanism". The model formed the basis for the development of a web application that solves the problem of flexible interaction with customers by parsing new reviews, analyzing them, and generating a response to a review using a neural network. However, the GRU-based neural network structure usually works no better than LSTM.

Jiang et al. [27] proposed four novel Automatic Text Summarization models with a Sequence-to-Sequence (Seq2Seq) structure, utilizing an attention-based bidirectional Long Short-Term Memory (LSTM), with added enhancements for increasing the correlation between the generated text summary and the source text, and solving the problem of out-of-vocabulary words, suppressing the repeated words, and preventing the spread of cumulative errors in generated text summaries. Though the neural network model is of the same structure as ours, the applications are different. Xie et al. [28] proposed using the same deep-learning model for image captioning endeavors.

From the above surveyed NLP techniques [18,21,22,25–28], we adopt the typical state-of-the-art Seq2seq as the model, introduce the concept of the attention mechanism, and use bi-directional LSTM as the layers. In Seq2Seq, after the user asks a question, the question is input to the first encoder model, which converts the input sentence into context vectors. These are then fed into the second decoder model to extract words most likely to be the intent and entities. In terms of natural language processing, the advantage of our study is that no special training is required for the target domain because the method adopted is to analyze the grammar to find the keywords. Combined with a crawler and dialogue tree, the knowledge base can be rapidly expanded. Ultimately, realize a cross-domain intelligent customer service. As we know, this study is the first proposed web service by integrating web crawler and Seq2seq. Overall, Seq2Seq models remain a valuable tool in the NLP toolkit, particularly for tasks where the simplicity and interpretability of RNN-based architectures are beneficial.

3. Proposed Methodology

3.1. System Architecture

This research is mainly divided into two parts: The first one is natural language processing. Using the Seq2Seq framework, adding attention mechanism and bi-directional LSTM to improve the existing shortcomings of Seq2Seq. The input is a question, and the output is the intent and entities of the question. The second one is the knowledge base, which uses a web crawler to extract the structure and content of the target web page. After adjustment, it is made into a dialogue tree. Import the intent and entities output from the first part of natural language processing and perform a tree search, as shown in Figure 2.

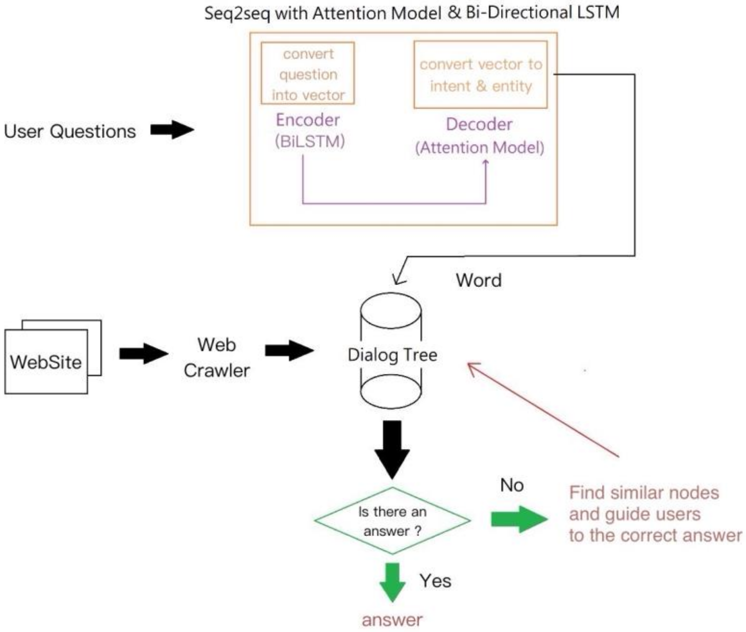

Seq2seq with Attention Model & Bi-Directional LSTM
convert question into vector
Encoder (BiLSTM)
convert vector to intent & entity
Decoder (Attention Model)
User Questions
Word
WebSite
Web Crawler
Dialog Tree
Is there an answer ?
No
Find similar nodes and guide users to the correct answer
Yes
answer

Figure 2. Proposed system architecture.

3.2. Nature Language Processing

Seq2seq, as in Figure 3a, consists of Encoder and Decoder. x_n is the input sentence sequence; the encoder will convert the input sentence (x_n) into a fixed unit vector (c), and then the decoder will convert it into the sentence or word (y_n) we want. This study will use a decoder to generate intent and entities.

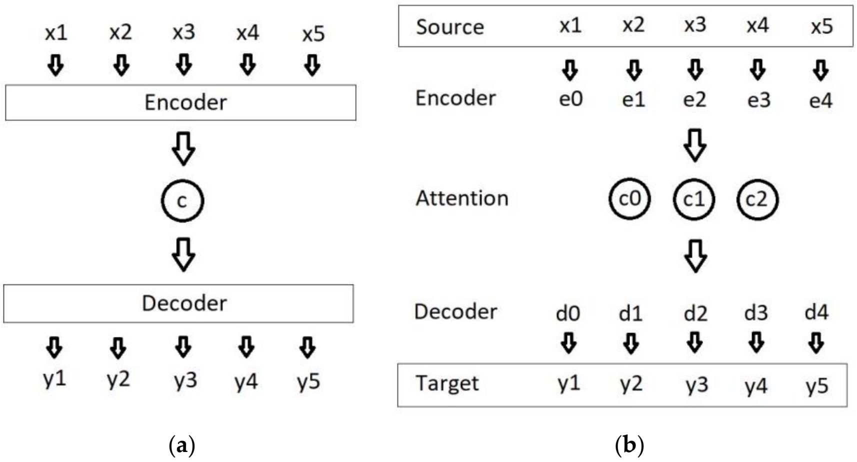


Figure 3. Seq2Seq framework (**a**) without attention mechanism. (**b**) with an attention mechanism.

3.2.1. Seq2Seq

In Seq2Seq, the decoder is usually a simpler RNN/LSTM [29], which is used to parse the context vector generated by the encoder, as shown in Figure 3a. A context vector contains the information input by the user, which is the last hidden state of the encoder. Seq2Seq solves the problem of LSTM input and output remaining at the same length. However, the encoder's mechanism of compressing the input sentence into a fixed-length context vector also creates another problem. The fixed-length context vector will not work well if the input sentence is very long. There is no way to express the meaning of each word in a sentence well. The attention mechanism [30] proposed by Luong et al. in 2015 solves this problem.

3.2.2. Attention Mechanism

The attention mechanism [30], as shown in Figure 3b, was initially proposed to address the problem of significant performance degradation in machine translation as sentence length increases. The paper's authors treat machine translation as an encoding-decoding problem, encoding sentences into vectors and decoding them into the content to be translated. However, in the Seq2Seq model, the encoder compresses the entire sentence into a fixed-length vector, which makes it challenging to save enough semantic information when the sentence is long. The role of the attention mechanism is to create a context vector for each word or word of the input sentence rather than just making a single context vector for the input sentence. The advantage of doing this is that the context vector generated by each word can be decoded more accurately.

There are similar properties in natural language processing. In early natural language processing, splitting sentences into many small words and processing them individually was common practice. Then, by building a large neural network model, it learns words in related fields and gives the results. The results from this approach can only be applied to that target domain. There is no such limitation when using the attention mechanism. This study significantly improves the accuracy of extracting intent and entity.

The encoder with the attention mechanism is no different from the encoder concept in Seq2Seq. The same is to generate [h_1, h_2, h_3 … h_n] from the input sentence [X_1, X_2, X_3 … X_n]. The difference is in how the context vector is calculated. Here, we first assume that the context vector is C_i.

The context vector (C_i) in Equation (1) is the sequential weighted sum of the input multiplied by the Attention score (α). Attention score is an essential concept proposed in the attention mechanism. It can measure the degree of importance each word in the input sentence brings to each word in the target sentence from Equation (2); the Attention score (α_{ij}) is calculated by score (e_{ij}). So, let's explain what score (e_{ij}) is.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{1}$$

$$Attention\ score\ \alpha_{ij} = {}^{e^{e_{ij}}} / \textstyle\sum_{k=1}^{T_x} e^{e_{ik}} \tag{2}$$

$$\text{score } e_{ij} = a\left(s_{i-1},\ h_j\right) \tag{3}$$

In Equation (3), a is an Alignment model that assigns a score e_{ij} to the pair of input at position j and output at position i, based on how well they fit. $e_{i,j}$ are weights defining how much of RNN/LSTM decoder hidden state $s_{i\text{-}1}$ and the j-th annotation h_j of the source sentence should be considered for each output. With score (e_{ij}), the Attention score can be calculated by softmax, and then the context vector C_i can be calculated. List the Attention score as a matrix, showing the correspondence between the input and output text.

As the name suggests, the decoder uses the attention mechanism to simulate human attention. There are two standard attention mechanisms: the soft attention mechanism and the hard attention mechanism. When the soft attention mechanism calculates the probability of attention, it will calculate its probability for any word input by the encoder. The advantage is that the weight of each word can be known more accurately, but it is also inefficient because of this. The hard attention mechanism matches the input word with the expected word. Words below the set threshold will directly set their probability to 0. This method has good effect and speed in image processing, but the subsequent accuracy would be significantly reduced in word processing.

Finally, this study uses the static attention mechanism, an extension of the soft attention mechanism. The difference from the soft attention mechanism is that the static attention mechanism will only calculate a single attention probability value for the entire sentence input by the encoder—no need to calculate every word like the soft attention mechanism. Although the accuracy rate will drop slightly, this study uses the static attention mechanism under the performance consideration. We adopted Luong Attention [30], as shown in Figure 4, to implement tensor flow. The calculation process is $h_t > a_t > c_t > h_t$. The calculation method of the attention score is as shown in Equation (4), where h_t is the state of the hidden layer of all target words, h_s is the hidden layer state of the source word, and t is the attention weight.

$$a_t(s) = align\left(h_t, \overline{h_s}\right) = \frac{exp\left(score\left(h_t,\ \overline{h_s}\right)\right)}{\sum_s exp\left(score\left(h_t, \overline{h_s}\right)\right)} \tag{4}$$

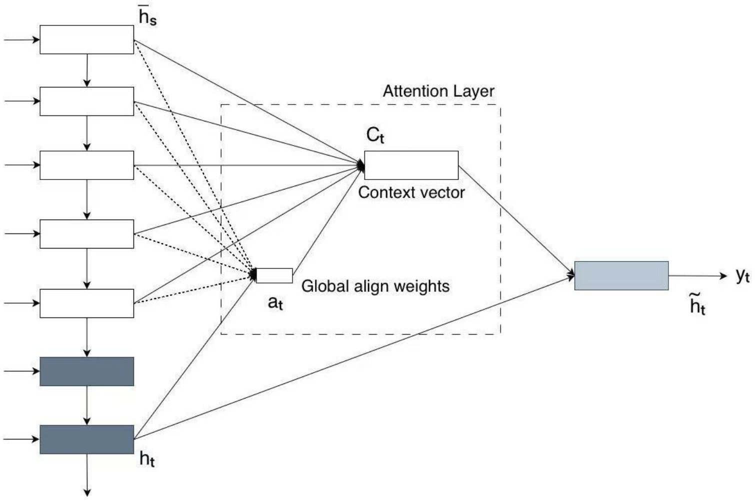
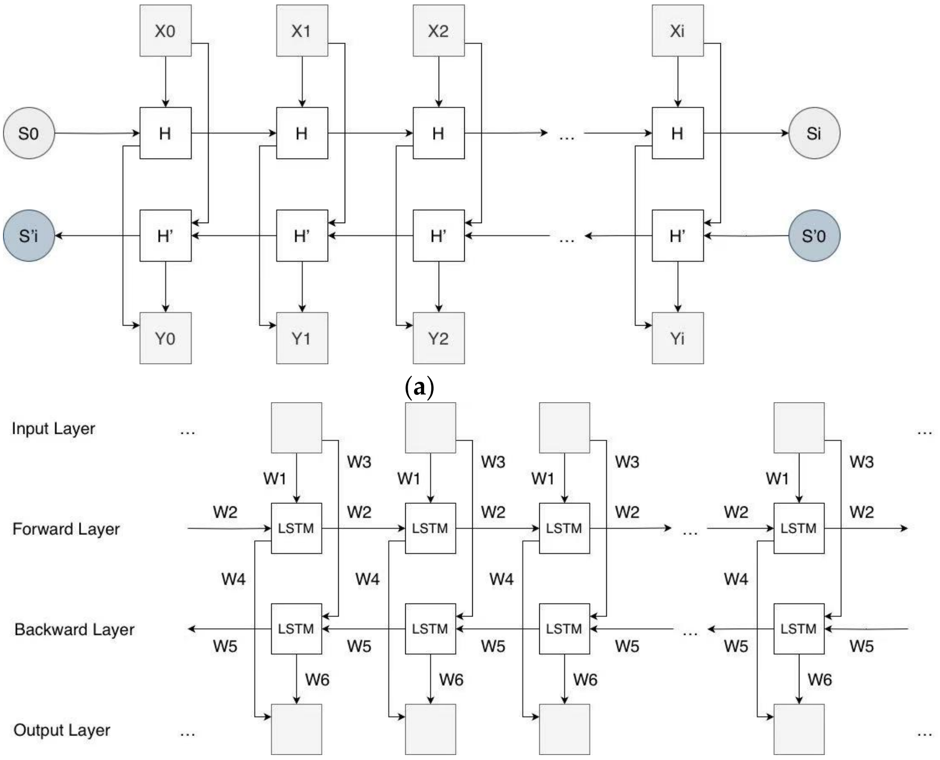


Figure 4. Luong Attention.

3.2.3. Bidirectional LSTM with Luong Attention Mechanism

The problem with unidirectional RNN/LSTMs is that they can only make predictions from information before the current time *t*. But in practice, a sentence sometimes needs to use future information to make predictions. Its mode of operation is that a hidden layer can be left to right or right to left. With a bidirectional RNN/LSTM, we can make better predictions about words. For example, "I want to buy a Mac" and "I love apples". In these two sentences, if you only read "I want to buy an Apple computer", you may not know whether Apple refers to fruit or mobile phone. But if you can get the message from the latter sentence, the answer is obvious. Figure 5a,b show the improved RNN and LSTM, respectively, bidirectional RNN [31] and LSTM [32].



(**a**)

(**b**)

Figure 5. Bi-directional (**a**) RNN and (**b**) LSTM framework.

As the name suggests, bidirectional LSTM splits the LSTM into two directions, as shown in Figure 5b. The one from the front to the back would use the said words as a reference to adjust the model. The back-to-front LSTM adjusts the model concerning future words. The bidirectional LSTM would generate three variables: Output, State_FW, and State_BW. Output represents the final output, and State_FW and State_BW calculate its value. The training process is divided into three parts: the first step is calculating State_FW from front to back, State_BW from back to front, and Output. The second step calculates the forward gradient from the back to the front and then calculates the reverse gradient from the front to the back. The last step updates the model parameters based on the gradient values from the previous step.

This study uses the encoder-decoder architecture (Seq2Seq) for natural language processing. Bi-directional LSTM is used in the encoder to increase the accuracy of sentences in the case of polysemy. The Luong Attention Mechanism [30] is added to the decoder, as shown in Figure 6 [32], which improves the accuracy problem decreasing as the sentence length increases.

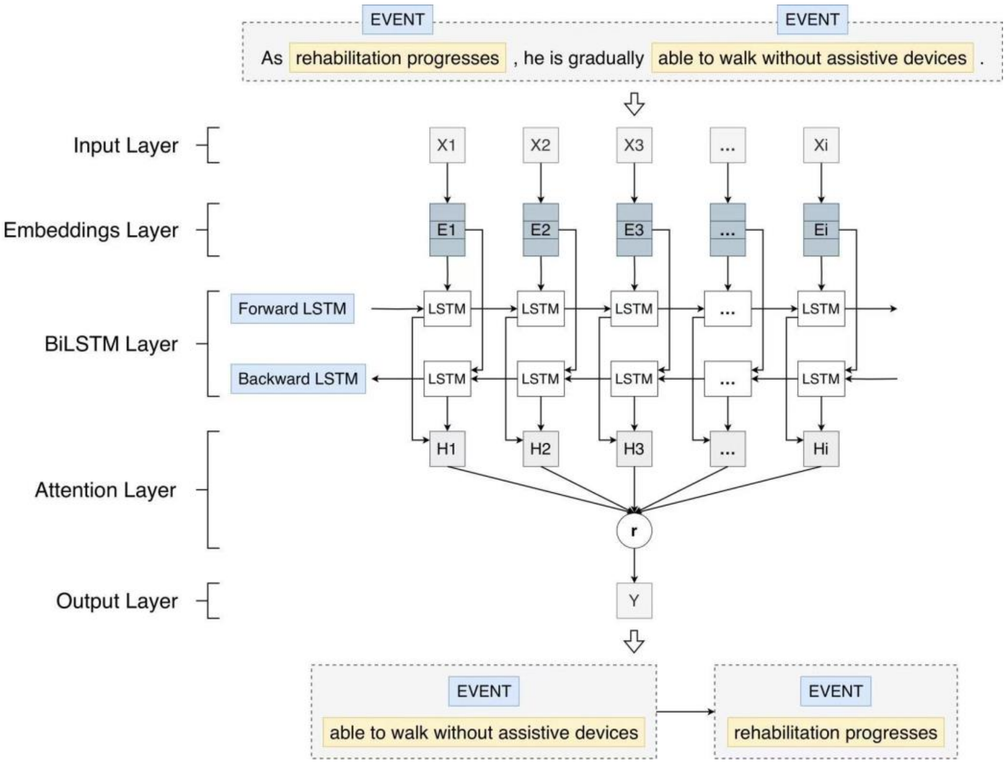

EVENT
EVENT
As rehabilitation progresses , he is gradually able to walk without assistive devices .
Input Layer
Embeddings Layer
BiLSTM Layer
Attention Layer
Output Layer
X1
X2
X3
...
Xi
E1
E2
E3
...
Ei
Forward LSTM
Backward LSTM
LSTM
H1
H2
H3
...
Hi
r
Y
EVENT
able to walk without assistive devices
EVENT
rehabilitation progresses

Figure 6. They used model architecture for natural language processing.

3.3. *Knowledge Representation*

3.3.1. Implementation of the Web Crawler

Beautiful Soup is a Python library module whose functions include parsing HTML or XML files and repairing files with errors such as unclosed tags (often called tag soup). This software package can build a tree structure to parse the page to access the data. It allows developers to quickly and easily parse web pages, find out the information that users are interested in, reduce the development threshold of web crawler programs, and speed up program development.

The crawler is divided into three steps. The first step is to analyze the structure of the web page. Use the request.get() function to crawl the HTML tags of the target web page and use beautifulSoup to convert the removed HTML into text files, perform structural analysis, and find out the rules. The purpose of the second step is to extract the target text. According to the rules, use the find() function to query tags such as class, id, HTML tag, etc., extract the target text, and store it in the Dialogue Tree. The last step is to use the Depth-First Search algorithm to explore the web page deeply and store all the target texts in the Dialogue Tree.

3.3.2. Depth-First Search

There are two standard methods of web crawling: Breadth-First Search (BFD) and Depth-First Search (DFS) [33]. BFS will first crawl pages with a shallow directory structure and then enter a deeper level to crawl. This approach is suitable for crawling complete web content. More complete web page information can be saved at the same time. DFS algorithm traverses a tree or graph structure starting from the tree's root (or a point in the graph). First, explore an unvisited vertex or node on the edge and search as deep as possible. Until all the nodes on the edges of the nodes have been visited, backtrack to the previous node. Repeatedly explore unsearched nodes until the destination node is found or all nodes have been visited.

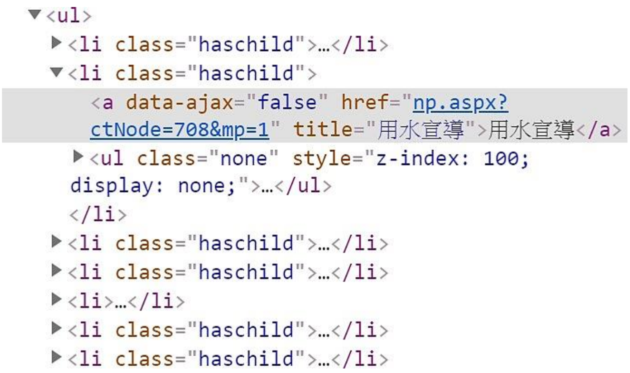
DFS accesses the web links of the next layer in descending order of depth until it can no longer go deeper. This method is more suitable for this study because the Dialogue Tree can be built only with nodes consisting of keywords. So, using DFS can make a complete Dialogue Tree faster. This study uses this algorithm to perform a comprehensive scan of the target web page, then convert it into a tree structure and store it according to the scanned structure.

3.3.3. Dialogue Tree

The concept of the Dialogue Tree [34] has flourished since the advent of computer games. But Dialogue Tree existed long before computer games. The earliest known Dialogue Tree appears in Jorge Luis Borges' 1941 short story "The Garden of Forking Paths", ref. [35] which allows branching routes from events to enter other branches or the main story via specified conditions. The story starts over as it progresses (since the possible outcomes will be close to n × m, where n is the number of options and m is the depth of the tree). Players advance the story by speaking to a non-player character and selecting a pre-written line from the menu. The non-player would respond accordingly to the player's choice and guide the player to the specified plot. This cycle continues until the goal is achieved. When the player chooses to leave the game, the dialogue ends, and the current state is remembered. Usually, there will not be only one set of tree diagrams in the game, and there will be a switch among different tree diagrams according to the player's choices. In addition to the essential Dialogue Tree, some games may be designed with a unique score system. Adjust the score according to different decisions, predict the player's likely thoughts, and change the plot direction to match the player's expectations.

The mechanics described above allow players to talk to each other with non-player characters. This study uses this dialogue mode to construct a knowledge-based Dialogue Tree. Replace players with users and replace non-player characters with a knowledge tree system based on a web crawler. Of course, there can be more than one knowledge tree, and they can be connected in series so that users can get a better experience. A set of statistical formulas can be further established to predict what users are interested in and push them to users.

Taking the Taiwan Water Company [36] as an example, the web page must be analyzed first. After analysis, it was found that most of the catalogs of the company are marked in the <ul> and <li> tags, as in Figure 7, and there are apparent tags such as class="child". So, find all the hyperlinks <a href> for these rules and use the DFS algorithm to crawl down layer by layer. In this way, a preliminary tree can be generated. After manual trimming, the final dialogue tree can be produced, as in Figure 8.



```
▼<ul>
  ▶<li class="haschild">…</li>
  ▼<li class="haschild">
      <a data-ajax="false" href="np.aspx?
      ctNode=708&mp=1" title="用水宣導">用水宣導</a>
    ▶<ul class="none" style="z-index: 100;
    display: none;">…</ul>
    </li>
  ▶<li class="haschild">…</li>
  ▶<li class="haschild">…</li>
  ▶<li>…</li>
  ▶<li class="haschild">…</li>
  ▶<li class="haschild">…</li>
```

Figure 7. A part of the web content of the Taiwan Water Company.

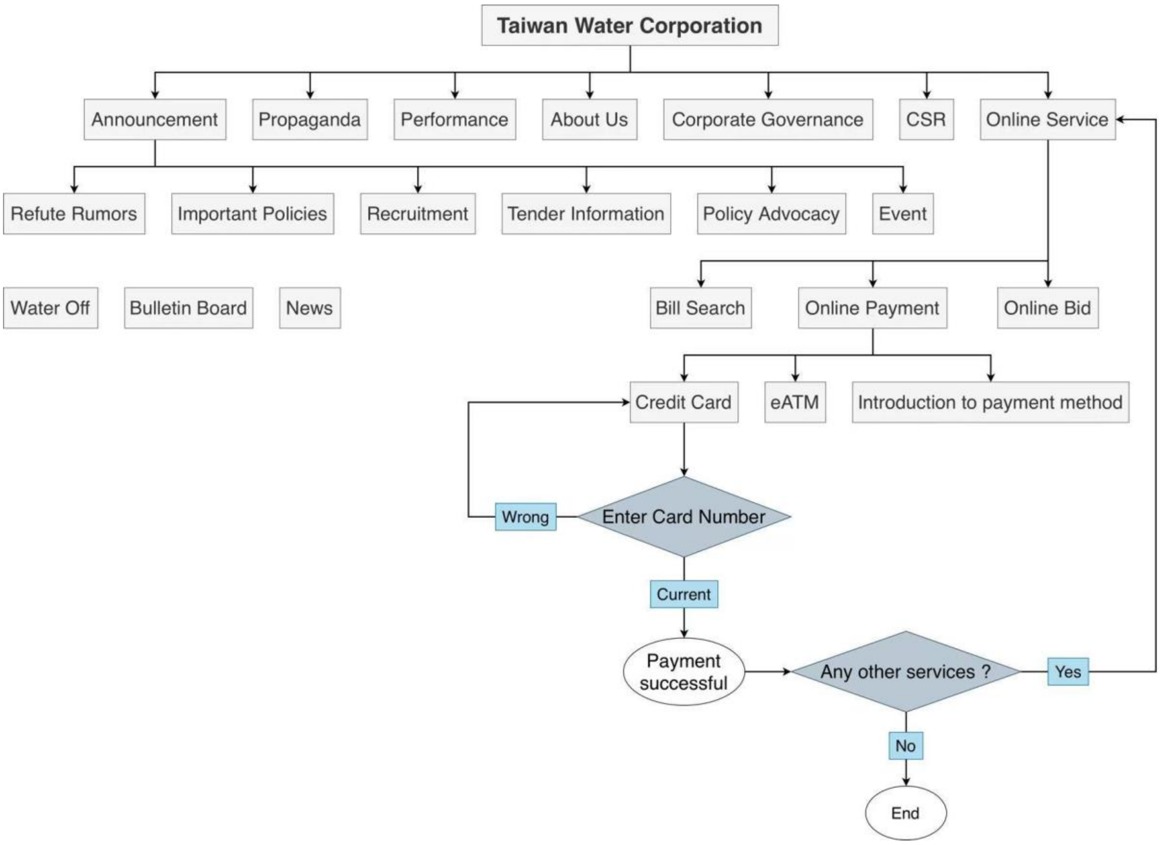


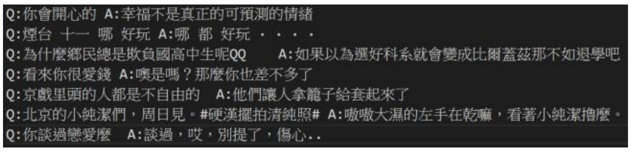
Figure 8. An example dialogue tree about the payment by credit card for the Taiwan Water Company.

4. Experimental Results

The experiments in this study were divided into four parts. The first part aims to demonstrate that the neural network model adopted in this study outperforms traditional neural network models in natural language processing. The second part is questions and answers with knowledge of the target domain to test RasaNLU [37] and the proposed model in this study. The purpose is to verify whether the accuracy of the neural network model proposed in this study is better than the popular Rasa open-source conversational AI platform in the target domain. The third part of the experiments aims to question and answer knowledge in non-target domains and test RasaNLU and the proposed model in this study. The purpose is to verify whether the generality of the adopted neural network model is better than the RasaNLU machine learning method in cross-domain. The last part of the experiments compares this study with Xiao Ai, Google Assistant, Siri, and Samsung Bixby on the market to verify the gap between this study and some commercial products.

4.1. Public Dataset

The experiment uses the data compiled by codemayq [38] for the open-source Chinese chat data; some examples are shown in Figure 9, including from Chatterbot, Douban, PPT, Qingyun, TV drama dialogue, Tieba forum, Weibo, Xiaohuangji and other websites containing various dialogue materials. Many of these sentences have familiar Martian texts, such as ㄏㄏ, QQ, etc., that must be filtered and pre-processed.



```
Q:你會開心的  A:幸福不是真正的可預測的情緒
Q:煙台 十一 哪 好玩 A:哪 都 好玩 ・・・・
Q:為什麼鄉民總是欺負國高中生呢QQ     A:如果以為選好科系就會變成比爾蓋茲那不如退學吧
Q:看來你很愛錢  A:噢是嗎？那麼你也差不多了
Q:京戲里頭的人都是不自由的   A:他們讓人拿籠子給套起來了
Q:北京的小純潔們，周日見。#硬漢擺拍清純照# A:嗷嗷大濕的左手在乾嘛，看著小純潔擼麼。
Q:你談過戀愛麼   A:談過，哎，別提了，傷心..
```

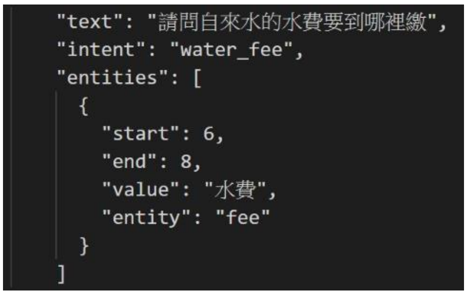
Figure 9. Chinese chat corpus.

4.2. *Experimental Environment and the Tested Models*

The same computer is used for training and testing, equipped with an Intel i5 8600K processor, Nvidia GeForce GTX 1080 Ti 11G, and 3584 CUDA cores. The batch size is set to 32, and each training group takes an average of 2 days.

4.2.1. RasaNLU

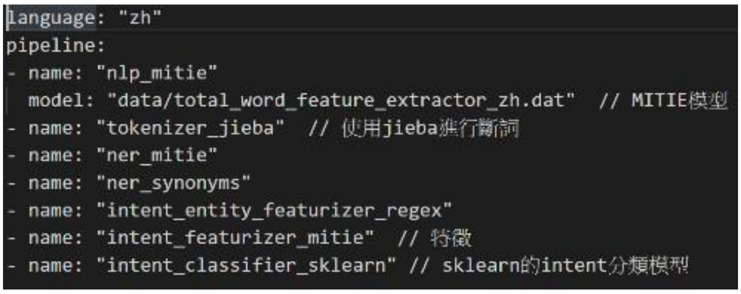
In this experiment, the Sklearn + Jieba + MITIE packages are used to train the RasaNLU, which has some advantages. First of all, the source data must be marked. This study takes the Taiwan Water Company as an example. Text is the input sentence, the intent is the purpose, and the entity is the keyword of the input sentence, as shown in Figure 10.



```
    "text": "請問自來水的水費要到哪裡繳",
    "intent": "water_fee",
    "entities": [
      {
        "start": 6,
        "end": 8,
        "value": "水費",
        "entity": "fee"
      }
    ]
```

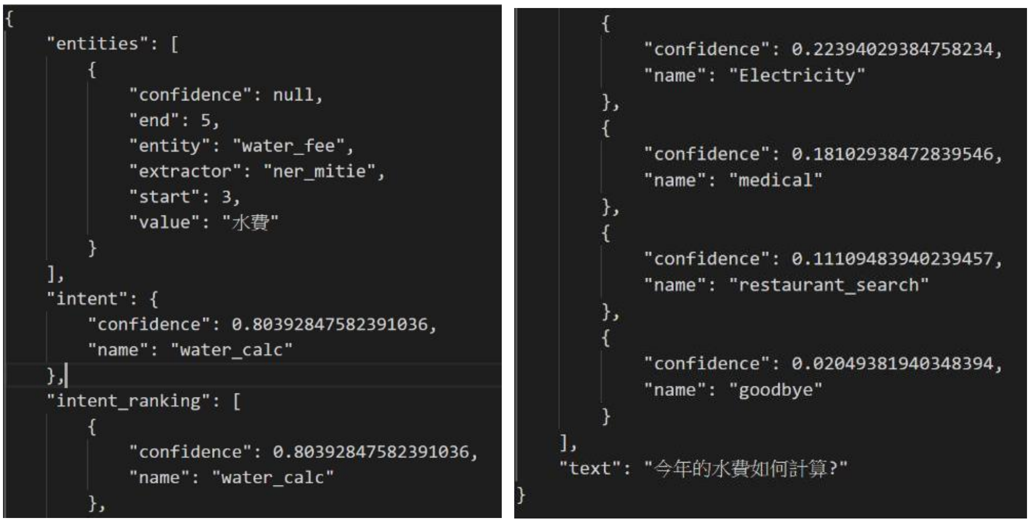
Figure 10. Annotation of the training data for RasaNLU.

We used MITIE, which is an unsupervised training model that requires a lot of Chinese data. We use the Chinese version of Wikipedia as the data source. The total_word_feature_extractor_zh.dat in Figure 11 is the file trained by MITIE. After the training, you can import the data and the Jieba tool into the RasaNLU model to start training. Test validation is required after training is completed. After enabling RasaNLU, curl will be used to verify the result. If you see that the intent and the entity can be correctly identified and the results of each direction would be scored as in Figure 12, the experiment is successful.



```
language: "zh"
pipeline:
- name: "nlp_mitie"
  model: "data/total_word_feature_extractor_zh.dat"  // MITIE模型
- name: "tokenizer_jieba"  // 使用jieba進行斷詞
- name: "ner_mitie"
- name: "ner_synonyms"
- name: "intent_entity_featurizer_regex"
- name: "intent_featurizer_mitie"  // 特徵
- name: "intent_classifier_sklearn" // sklearn的intent分類模型
```

Figure 11. RasaNLU pipeline.



```
{
    "entities": [
        {
            "confidence": null,
            "end": 5,
            "entity": "water_fee",
            "extractor": "ner_mitie",
            "start": 3,
            "value": "水費"
        }
    ],
    "intent": {
        "confidence": 0.80392847582391036,
        "name": "water_calc"
    },
    "intent_ranking": [
        {
            "confidence": 0.80392847582391036,
            "name": "water_calc"
        },
```

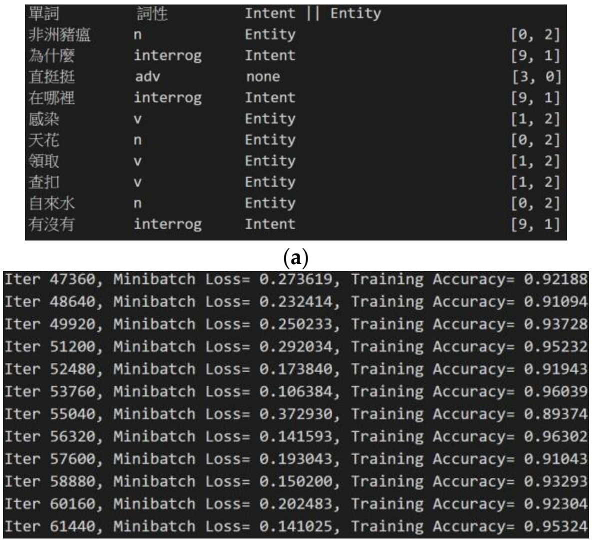
```
        {
            "confidence": 0.22394029384758234,
            "name": "Electricity"
        },
        {
            "confidence": 0.18102938472839546,
            "name": "medical"
        },
        {
            "confidence": 0.11109483940239457,
            "name": "restaurant_search"
        },
        {
            "confidence": 0.02049381940348394,
            "name": "goodbye"
        }
    ],
    "text": "今年的水費如何計算?"
}
```

Figure 12. A test example of the trained RasaNLU (read from left to right) for water fare query.

Because LSTM is unlike the other three Encoder-Decoder models, it can convert the input sentence into a unit vector and then extract the Intent and Entities by the Decoder. Therefore, when training, the sentences are first segmented, and the training is performed according to each word. This study's LSTM and deep learning experiments deploy the TFLearn API. TFLearn is an API based on Tensor Flow, which can set each neural layer and activation function and filter more intuitively and quickly.

4.2.2. LSTM

The data needs to be pre-processed before training. After the sentences are segmented, the features, such as part of speech of each word, are converted into vectors that the model can understand. Taking Figure 13a as an example, the feature is a two-dimensional vector. The first-dimension vector is the part of speech, and each part of speech is numbered: noun is 0, verb is 1, and so on. The second-dimension vector divides each word into Intent and Entity according to its properties. After the data is pre-processed and basic actions such as batch data generation and parameter setting are processed, the LSTM model can be trained as in Figure 13b.



```
單詞        詞性        Intent || Entity
非洲豬瘟    n           Entity                    [0, 2]
為什麼      interrog    Intent                    [9, 1]
直挺挺      adv         none                      [3, 0]
在哪裡      interrog    Intent                    [9, 1]
感染        v           Entity                    [1, 2]
天花        n           Entity                    [0, 2]
領取        v           Entity                    [1, 2]
查扣        v           Entity                    [1, 2]
自來水      n           Entity                    [0, 2]
有沒有      interrog    Intent                    [9, 1]
```

(**a**)

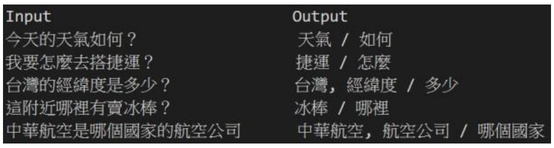
```
Iter 47360, Minibatch Loss= 0.273619, Training Accuracy= 0.92188
Iter 48640, Minibatch Loss= 0.232414, Training Accuracy= 0.91094
Iter 49920, Minibatch Loss= 0.250233, Training Accuracy= 0.93728
Iter 51200, Minibatch Loss= 0.292034, Training Accuracy= 0.95232
Iter 52480, Minibatch Loss= 0.173840, Training Accuracy= 0.91943
Iter 53760, Minibatch Loss= 0.106384, Training Accuracy= 0.96039
Iter 55040, Minibatch Loss= 0.372930, Training Accuracy= 0.89374
Iter 56320, Minibatch Loss= 0.141593, Training Accuracy= 0.96302
Iter 57600, Minibatch Loss= 0.193043, Training Accuracy= 0.91043
Iter 58880, Minibatch Loss= 0.150200, Training Accuracy= 0.93293
Iter 60160, Minibatch Loss= 0.202483, Training Accuracy= 0.92304
Iter 61440, Minibatch Loss= 0.141025, Training Accuracy= 0.95324
```

(**b**)

Figure 13. (**a**) Training sample data. (**b**) LSTM training process.

4.2.3. Seq2Seq

We adopt Google's open-source Seq2Seq model [39] as base. To make a correct comparison with other neural network models, some improved methods in the example are removed first, and only the part of Seq2Seq is retained. The training is divided into three parts: first, collecting the data, then setting the model parameters, and finally, training the model. The trained model has an accuracy of nearly 70%, as shown in Figure 14.



```
Input                          Output
今天的天氣如何？                 天氣 / 如何
我要怎麼去搭捷運？               捷運 / 怎麼
台灣的經緯度是多少？             台灣, 經緯度 / 多少
這附近哪裡有賣冰棒？             冰棒 / 哪裡
中華航空是哪個國家的航空公司     中華航空, 航空公司 / 哪個國家
```

Figure 14. Some test results of Seq2Seq.

4.3. Ablation Test of the Proposed Neural Network Model

The first experiment tested four groups of neural network models. The four groups of neural networks were LSTM, Seq2Seq, Seq2Seq + Attention, Seq2Seq + Bi-directional LSTM + Attention. Using the open-source Chinese corpus [38], 5000 questions were extracted from it as training data, and 1000 questions different from the dataset were extracted from the same corpus as test data. If the output is the expected Intent and Entities, it is judged to be the correct result. Otherwise, it is not. Finally, the number of correct questions/1000 × 100% becomes the accuracy rate. The results are in the order of LSTM, Seq2Seq, Seq2Seq + Attention, Seq2Seq + Bi-directional LSTM + Attention, and the accuracy rates are 63.4%, 69.2%, 76.1%, and 82.1%, respectively. It can be observed in Table 1 that using Seq2Seq and adding the Attention and bidirectional LSTM can indeed significantly improve the accuracy rate.

Table 1. Ablation test of the proposed neural network model.

Neural Network Models	Accuracy
LSTM	63.4%
Seq2Seq	69.2%
Seq2Seq + Attention	76.1%
Seq2Seq + Bi-LSTM + Attention	82.1%

4.4. Test in the Target Domain and Non-Target Domain

Using 5000 Taiwan Water Company data [37] as the training data and 1000 water-related questions that differ from the training data for testing, the Accuracy rates of RasaNLU and this study are 86.4% and 87.1%, respectively. It can be seen from Table 2 that the results of RasaNLU and this study in the target domain are not much different, and the accuracy rates are near.

Table 2. Compare RasaNLU and ours in general.

Accuracy	RasaNLU	Ours
Target Domain	86.4%	87.1%
Non-Target Domain	46.3%	83.2%

Five thousand questions from chat databases [38] from 8 public sources, including Weibo, Tieba, Douban, and other well-known social websites in mainland China and the PTT gossip version in Taiwan, were extracted as training data. And extract 1000 questions from the same corpus that differ from the training data for testing. The correct rates of

RasaNLU and this study are 46.3% and 83.2%, respectively. Because of the different training mechanisms, this study can still identify the correct Intent and Entities in areas the system is not targeted. It can be seen from Table 2 that RasaNLU and this study are in non-target areas, and RasaNLU would have misjudged.

4.5. Comparisons with Chatbots on the Market

Verify the difference between this research and commercial products in the professional field. Because the products on the market cannot extract intent and entities for experimentation, we use Trip.com, Taipei MRT, Taiwan Water Company, Wikipedia, and other websites with more standardized answers as test data sources. The dataset for this study is the Chinese corpus [38]. Five thousand questions were extracted from it as training data, and the test data were 500 groups of questions randomly selected from the above websites. Questions have only one answer like: Who founded Microsoft? Answer: Bill Gates. The accuracy rate is shown in Table 3. This experiment indicates that this commercial product has higher accuracy of comprehension for Wikipedia only, which hinted already optimized on it. However, our study has advantages over these products in other application fields.

Table 3. Accuracy comparison of this study with some commercial products in selected application fields.

Items	This Study	Xiao Ai Classmate	Google Assistant	Siri	Samsung Bixby y
Trip. co m	**73.4**	42.5	40.2	32.6	39.2
Taipei MRT	**82.1**	48.2	41.3	49.2	42.3
Taiwan Water Company	**78.3**	32.5	40.2	34.2	32.1
Wikipedia	81.2	82.3	83.4	81.7	**87.9**

5. Conclusions and Remarks

5.1. Conclusions

From previous surveyed related works, Refs. [25–28] are the most recently published works that are closely related to our adopted Bi-LSTM model with the Luong attention mechanism. However, the adopted dataset and target domains are different. So, we could not make a direct comparison. Here, the proposed innovative Bi-LSTM model works as an intelligent agent and is integrated with a web crawler for web service.

It can be seen from the above experimental results that there is much difference between this study and RasaNLU in the target field. Our advantages are that when applying the proposed Seq2seq based on the Bi-LSTM model to other similar websites, only the knowledge bases (Dialogue Tree) need to be expanded and don't need to re-train the proposed model. The difference is evident in the general field. RasaNLU, it is obvious that if there is no relevant information in the field during training, the trained model will be unable to find the correct answer. The system proposed in this study makes judgments based on grammar, such as part of speech and sentence structure, so the accuracy is much higher than that of RasaNLU. In the experiment, compared with the products on the market, this research can provide better answers than the products on the market if the dialogue tree is appropriately designed in the application field. However, when the answer to the question is more general, it is less likely to give an acceptable answer, as a human would.

As to the contribution, the first one is to demonstrate the function brought by the memory module of LSTM. It makes the computer read sentences no longer only to understand the words it has learned but to evolve to be more human-like. Grammatically determine which words in the sentence are necessary. This concept is similar to when humans read foreign languages; there may be some words in an article that we do not know. But with the context, we can still read this article. By adding the Attention Mechanism and bidirectional LSTM, some of the shortcomings of Seq2Seq are also improved and provide higher accuracy.

The second is using web crawlers to build a knowledge base (Dialogue Tree). After experiments, most websites have not optimized this. Therefore, if some websites use a database-like table listing method or embed many pictures and PDF files in the website, it will increase the difficulty of Dialogue Tree formation. Instead, more staffing must be used to make corrections. Therefore, subsequent systems should make statistics on these exceptions and overcome them individually.

5.2. Future Works

At present, this system has two shortcomings. The first is that the speed is not fast enough. After a question is asked, it often takes about five or six seconds to respond. This part can be improved in the subsequent Dialogue Tree algorithm improvement and optimization of neural network parameters. Another disadvantage is that the accuracy rate in the general environment is about 80%. It should be optimized through more experimentation. In addition, other various Attention Mechanisms, like self-attention and multi-head attention [40], not only proposed a parallelization method but also improved some of the defects of the current Attention Mechanism. Transformer is such a model and is worth trying and studying.

Author Contributions: Methodology, C.-C.H.; software, J.-X.Y.; validation, C.-C.H.; data curation, J.-X.Y.; writing—original draft preparation, M.-H.H.; writing—review and editing, C.-C.H.; supervision, J.-X.Y. and M.-H.H.; project administration, M.-H.H.; funding acquisition, M.-H.H. All authors have read and agreed to the published version of the manuscript.

Funding: Sanming University. Research Foundation for Advanced Talents, Grant Number: 22YG11S.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available at [https://github.com/codemayq/chinese_chatbot_corpus (accessed on 18 June 2021)], Ref. [35].

Conflicts of Interest: Author Jian-Xin Yang was employed by the company QSAN Technology, Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Turing, A.M. Computing Machinery and Intelligence. *Mind* **1950**, *LIX*, 433–460. [CrossRef]
2. Wallace, R.S. *The Anatomy of A.L.I.C.E.*; Epstein, R., Roberts, G., Beber, G., Eds.; Parsing the Turing Test; Springer: Dordrecht, The Netherlands, 2009. [CrossRef]
3. Kirrane, S. Intelligent Software Web Agents: A Gap Analysis. *J. Web Semant.* **2021**, *71*, 100659. [CrossRef]
4. Mashaabi, M.; Alotaibi, A.; Qudaih, H.; Alnashwan, R.; Al-Khalifa, H. Natural Language Processing in Customer Service: A Systematic Review. *arXiv* **2022**, arXiv:2212.09523. [CrossRef]
5. Huang, C. The Intelligent Agent NLP-based Customer Service System. In Proceedings of the 2021 2nd International Conference on Artificial Intelligence in Electronics Engineering (AIEE '21), Phuket, Thailand, 15–17 January 2021; ACM: New York, NY, USA, 2021; pp. 41–50. [CrossRef]
6. Winograd, T. *SHRDLU*; MIT AI Technical Report 235; MIT: Cambridge, MA, USA, 1971.
7. Weizenbaum, J. ELIZA—A Computer Program for the Study of Natural Language Communication between Man and Machine. *Commun. ACM* **1966**, *9*, 35–36. [CrossRef]
8. Chen, G.; Liu, R.; Chen, R.; Fu, C.C. A Historical Review of the Key Technologies for Enterprise Brand Impact Assessment. In Proceedings of the 2024 International Conference on Applied Economics, Management Science and Social Development (AEMSS 2024), Luoyang, China, 22–24 March 2024; pp. 240–246.
9. LeCun, Y. Deep Learning Hardware: Past, Present, and Future. In Proceedings of the 2019 IEEE International Solid-State Circuits Conference—(ISSCC), San Francisco, CA, USA, 11–17 February 2019; pp. 12–19. [CrossRef]
10. Rasa: Open Source Conversational AI—Rasa. 2019. Available online: https://rasa.com (accessed on 18 June 2021).
11. Elman, J.L. Finding Structure in Time. *Cogn. Sci.* **1990**, *14*, 179–211. [CrossRef]
12. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
13. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. *arXiv* **2014**, arXiv:1409.3215.

14. Xiao, T.; Zhu, J. Introduction to Transformers: An NLP Perspective. *arXiv* **2023**, arXiv:2311.17633.
15. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
16. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20), Vancouver, BC, Canada, 6–12 December 2020; Article 159, pp. 1877–1901.
17. AIML.com. What Are the Limitations of Transformer Models? Available online: https://aiml.com/what-are-the-drawbacks-of-transformer-models/ (accessed on 18 June 2021).
18. Rungsawang, A.; Angkawattanawit, N. Learnable Topic-Specific Web Crawler. *J. Netw. Comput. Appl.* **2005**, *28*, 97–114. [CrossRef]
19. Baeza-Yates, R.; Ribeiro-Neto, B. *Modern Information Retrieval*; ACM Press: New York, NY, USA, 1999.
20. Kleinberg, J.M. Authoritative Sources in a Hyperlinked Environment. *J. ACM (JACM)* **1999**, *46*, 604–632. [CrossRef]
21. Lee, T.B. *Semantic Web*; World Wide Web Consortium: Cambridge, MA, USA, 1998.
22. Kim, S.M.; Ha, Y.G. Automated Discovery of Small Business Domain Knowledge Using Web Crawling and Data Mining. In Proceedings of the 2016 International Conference on Big Data and Smart Computing (BigComp), Hong Kong, China, 18–20 January 2016; pp. 481–484. [CrossRef]
23. W3C. RDF—Semantic Web Standards. Available online: http://www.w3.org/RDF/ (accessed on 18 June 2021).
24. W3C. OWL—Semantic Web Standards. Available online: http://www.w3.org/2001/sw/wiki/OWL (accessed on 18 June 2021).
25. Choudhary, P.; Chauhan, S. An Intelligent Chatbot Design and Implementation Model Using Long Short-Term Memory with Recurrent Neural Networks and Attention Mechanism. *Decis. Anal. J.* **2023**, *9*, 100359. [CrossRef]
26. Budaev, E.S. Development of a Web Application for Intelligent Analysis of Customer Reviews Using a Modified seq2seq Model with an Attention Mechanism. *Comput. Nanotechnol.* **2024**, *11*, 151–161. [CrossRef]
27. Jiang, J.W.; Zhang, H.; Dai, C.; Zhao, Q.; Feng, H.; Ji, Z.; Ganchev, I. Enhancements of Attention-Based Bidirectional LSTM for Hybrid Automatic Text Summarization. *IEEE Access* **2021**, *9*, 123660–123671. [CrossRef]
28. Xie, T.; Ding, W.; Zhang, J.; Wan, X.; Wang, J. Bi-LS-AttM: A Bidirectional LSTM and Attention Mechanism Model for Improving Image Captioning. *Appl. Sci.* **2023**, *13*, 7916. [CrossRef]
29. Su, G. Seq2seq Pay Attention to Self-Attention. 3 October 2018. Available online: https://bgg.medium.com/seq2seq-pay-attention-to-self-attention-part-1-%E4%B8%AD%E6%96%87%E7%89%88-2714bbd92727 (accessed on 18 June 2021).
30. Luong, M.T.; Pham, H.; Manning, C.D. Effective Approaches to Attention-based Neural Machine Translation. *arXiv* **2015**, arXiv:1508.04025.
31. Graves, A. *Supervised Sequence Labelling with Recurrent Neural Networks*; Computer Science; University of Toronto: Toronto, ON, Canada, 2012.
32. Alfattni, G.; Peek, N.; Nenadic, G. Attention-based Bidirectional Long Short-Term Memory Networks for Extracting Temporal Relationships from Clinical Discharge Summaries. *J. Biomed. Inform.* **2021**, *123*, 103915. [CrossRef] [PubMed]
33. Wikipedia. Depth-First-Search. 2019. Available online: https://zh.wikipedia.org/wiki/Depth-First-Search (accessed on 18 June 2021).
34. Wikipedia. Dialogue Tree. 2019. Available online: https://en.wikipedia.org/wiki/Dialogue_tree (accessed on 18 June 2021).
35. Borges, J.L. *Garden of Forking Paths*; Penguin Books: London, UK, 22 February 2018; ISBN -13 9780241339053.
36. Taiwan Water Company. 2016. Available online: https://www.water.gov.tw/mp.aspx?mp=1 (accessed on 18 June 2021).
37. Introduction to Rasa Open Source & Rasa Pro, RasaNLU. Available online: https://rasa.com/docs/rasa/nlu-training-data/ (accessed on 18 June 2021).
38. GitHub. Chinese_Chatbot_Corpus. 2018. Available online: https://github.com/codemayq/chinese_chatbot_corpus (accessed on 18 June 2021).
39. GitHub. Seq2Seq. 2017. Available online: https://github.com/google/seq2seq (accessed on 18 June 2021).
40. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. Cornell University, 16 December 2017. Available online: https://arxiv.org/pdf/1706.03762.pdf (accessed on 18 June 2021).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.