Linguistic Question-Answering Reasoning Based on Intelligent Perception of Attribute Weight

Liang Xiao¹, Dan Wei²

¹Dalian Neusoft University of Information
²Dalian University of Science and Technology

Abstract: Intelligent Question Answering System aims to realize the communication between human and machine. Based on the real data of question-answer, of front end design, and of daily chat in Chinese, this paper proposes a design method of Intelligent Question-Answering System based on deep neural network and optimizes it. Through real data from authentic view and online daily chat, the paper constructs the property weight, laying foundation for the system to play its role in the real scene such as text cleaning and word segmentation, part of speech tagging, word vector representation, word weight adjustment, etc. By using the mixed model of BOW and Skip-gram to represent the word vector, the paper describes the correlation between words and makes for the shortcomings of BOW. Meanwhile, it retains Bow’s excellent ability of discrete-feature processing.

1. Introduction

The technology of Natural Language Processing (NLP) is the theoretical basis of natural language communication between human beings and devices and within devices themselves. Intelligent Question-Answering System based on natural language processing technology is the key element of human-computer interaction. Intelligent Question Answering System is a dialogue agent based on text, it can imitates social applications such as Wechat, providing information and complete transactions or related services by simulating human dialogue. Users can ask questions to the system in a way they do to each other. Nowadays, intelligent systems based on natural language processing can learn and update knowledge by reading all existing electronic articles [1].

The most obvious disadvantage of earlier Intelligent Question Answering Systems is that they often misunderstand or completely ignore what users type. It was not until the researchers abandoned the method based on language rules and switched to probabilistic thinking to design systems based on attribute weight or statistical theory that natural language processing began to develop rapidly [2]. With the improvement of computer functions and the increase volume of data storage, the system can complete the call of a larger number of language data. Combined with these language resources and good-performed language analyzer, the machine can extract useful language information and develop practical systems for specific areas [3].

Based on the intelligent perception of attribute weight, an Intelligent Question Answering System based on deep neural network is designed and optimized. The system has embedded relevant corpus information, and conducted a series of process for these information, so that the system can recognize sentences and respond to answer questions by itself. On the one hand, the response principle is to match the sentences input by the users. By scoring those sentences and the embedded corpus information, the system could obtain the similarity of them. The higher the score is, the more the
similarity is, making the sentences easier to be adopted. Another method is to use deep neural network for semantic representation to understand the meaning of the input sentences and generate the answers.

2. System Structure Design

The second horizontal lexical layer is a series of processing of corpus text, including Particle module, Part-of-speech tagging module, Word vector representation module, Word vector weighting and adjustment module, as shown in Figure 1.

![System framework](image)

The first phase is training, which could be divided into three layers horizontally including corpus level, lexical level and sentence level, as shown in Figure 2.

![Logical layers of training phase](image)

The first horizontal layer is Corpus, which could be regarded as interface between users and system, and which could be divided into two kinds. The first is the attribute weight of industry dialogue provided by users. Questions and corresponding answers are predefined in the attribute weight, and are input into system training in the way of one question and one answer. The resources are generated and
collected in the process of actual customer service and user dialogue. The second is online daily attribute weight, which is used to supplement the language environment that is not involved in offline attribute weight [4].

These four functional modules are the common modules of the two question answering models. The main function is to segment the Chinese sentences in the corpus level, to mark the parts of speech of the words, and then to carry out the semantic representation based on the word vector and adjust the weight [5].

The third horizontal layer is the semantic representation module of sentence level, including the question and answer vector representation model and the sentence vector representation model based on attention mechanism. The former one is mainly to find the closest word vector in the template for question answering matching after a sentence is represented by word vector; the latter is to organize the word vector sequence into a sentence vector representation by LSTM network based on attention mechanism after a sentence is represented by word vector [6].

Vertically, the system can also be divided into two relatively independent modules, including similarity based retrieval matching model, hereinafter referred to as retrieval model and attention based generative dialogue model, hereinafter referred to as generative model. The switching between the two models is realized by judging whether the similarity between the question and the attribute weight is higher than the threshold. That is, setting a threshold in advance, if the similarity is greater than the threshold, the results generated by the retrieval model will be called, otherwise the results generated by the generation model will be called [7]. The system uses a combination of retrieval model and generation model to make up for the shortcomings of each system. In addition to the training phase, the system also has a test phase which is relatively simple. It is determined by simulating the user's input of questions, observing whether the generated reply answers the user's questions, and whether it meets the user's expectations [8].

3. Selection of Attribute Weight
For networks and models, there are great differences in performance when dealing with different attribute weights. So when we study the system, we usually follow two rules. First of all, when comparing the performance of the network or model, it is better to use the same data set, so that the external influence can be minimized [9]. Secondly, when the same network or model processes data sets with different features, it can only be be regarded as reaching real optimization effect if the performance is the same. Therefore, the top priority of network and model optimization is to select appropriate data sets with different features.

3.1 Pretreatment of Attribute Weight
Natural language texts cannot be understood by machines, and normally need a series of preprocessing to transform human language into a quantity that can be recognized by machines [10].

The purpose of word classification is to divide a sentence into the smallest semantic unit (word).

In the system, word segmentation is realized by reverse maximum matching method, which is based on probability and statistics. The specific process is shown in Figure 3.
From the perspective of statistics, assuming that the input string of the word segmentation is: \( C = [c_1, c_2, \cdots, c_n]^T \), and the output string is \( S = [w_1, w_2, \cdots, w_m]^T \), in both strings, \( m \leq n \). For a given string \( C \), it will correspond to a variety of segmentation methods \( S \). The purpose of word segmentation is to find the most possible segmentation in \( S \):

\[
\text{Seg}(C) \arg \max_{S \in S_c} P(S|C) = \arg \max_{S \in S_c} \frac{P(E)^{p(s)\times i}}{P(C)}
\]

For a given sentence, there are many segmentation schemes \( S_1, S_2, \cdots \). The conditional probabilities are calculated respectively \( P(S_1|C), P(S_2|C), \cdots \). Then the segmentation method with maximum likelihood is used. According to Bayes Formula:

\[
P(S|C) = \frac{P(S|C) \times P(S)}{P(C)}
\]

Within the formula, \( P(S|C) \) means that when the input string is \( C \), the conditional probability of \( P(C) \) in order to output string \( S \). \( P(S) \) is when participle string \( S \) appears, the probability of \( P(C) \) appears in the attribute weigh in order to output string \( C \), which is a fixed value for standardization. \( P(C|S) \) means when output string is \( S \), the conditional probability for input string being \( C \) there is only one way to return from the segmented string to the original string, so \( P(C|S) = 1 \). Therefore, the
comparison between \( P(S_1|C) \) and \( P(S_2|C) \) is equivalent to the comparison between \( P(S_1) \) and \( P(S_2) \) [11].

Assuming that the probability of each segmentation field is context independent, the probability of each segmentation scheme can be calculated by formula 3

\[
P(S) = P(w_1, w_2, \cdots, w_m) \approx \log P(w_1) + \log P(w_2) + \cdots + \log P(w_m)
\]

(3)

For different segmentation schemes \( S = [w_1, w_2, \cdots, w_m]^T \), \( m \) will have different values, and \( P(s) \) is inversely proportional to the value of \( M \). For example, under extreme conditions, each word is divided as a separate word, and the probability of this case is very small. The calculation result of formula 4 indicates the probability of proper division of each word

\[
P(w_i) = \frac{n_{wi}}{N}
\]

(4)

Within the formula: \( n_{wi} \) stands for the frequency of the word \( W_i \) in the attribute weight, and \( N \) stands for the total number of elements in the attribute weight.

The Decision Tree method is used for Part-of-Speech Tagging. In the decision tree, each node represents a part of speech, and the leaf node provides classification. The method of sample classification is: starting from the root, calculating the eigenvalue on each node, and then classifying to the appropriate branch, until providing the leaf node of classification. Its structure is shown in Figure 4.

The main problem in decision tree is to choose the best feature for classification. Normally, in the classification, the goal is to make the samples contained in the branch nodes belong to the same type with the maximum probability, so as to improve the "purity" of the nodes. The important index of branching is feature selection index, which is a standard for selection splitting to determine how to split data. There are three types of selection metrics: information gain, gain ratio and Gini coefficient. The calculation of these three indicators determines the priority of classification attributes.

Suppose there is a variable \( X \), which has \( n \) possible values, and the probability of each value is \( p_i \), then the entropy of \( X \) can be calculated by the formula:

\[
H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)
\]

(5)

The existing attribute weight is \( D \), and the variable \( X \) represents the type of sample. Suppose there are \( k \) kinds of samples, then each possibility is \( \frac{|C_k|}{|D|} \), with the formula, \( |C_k| \) represents the number of samples with \( k \) type in attribute weight \( D \), and \( |D| \) represents the total number of samples with attribute weight \( D \). The entropy of attribute weight \( D \) is:
Information gain is to describe the information changes of attribute weight before and after classification. The more information changes brought by a classification method, the more suitable it is for classification. When attribute weight \( D \) is classified by feature \( A \), information gain can be defined as:

\[
 g(D, A) = H(D) - H(D | A) 
\]

Within the formula, \( H(D) \) is the entropy of data set \( D \). \( H(D | A) \) represents the mutual information between data set \( D \) and feature \( A \). And \( g(D, A) \) is the information change before and after classification according to data set \( D \). However, many problems exist in information gain. When the number of samples of a feature is too large, it becomes easier to get a word subset with high purity through this feature classification, so that the entropy after the classification is low. Since the entropy before classification is a fixed value, it leads to a larger \( g(D, a) \).

The gain ratio could solve this issue, which is calculated by multiplying the penalty parameter and the information gain, as shown in formula 8

\[
 g_X(D, A) = \frac{s(0, 0)}{K(A)} 
\]

Within the formula, \( g(D, A) \) is the information gain. \( g_X(D, A) \) is the gain ratio. \( K(A) \) represents the empirical entropy obtained by using the current feature \( A \) as the random variable of the sample set \( D \). It is calculated by formula 9

\[
 H_A(D) = -\sum_{i=1}^{k} \frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|} |b| 
\]

Within the formula, \( D \) is the data set and \( D_i \) is the subset of the data set \( D \) divided according to the number \( I \) of feature \( A \). The penalty parameter is the reciprocal of empirical entropy calculated with attribute weight \( D \) and feature \( A \) as random variables. When the sample number of a feature is small, the value of field \( (D) \) is small and the reciprocal is large, resulting in a relatively large number of \( g_X(D, A) \). Therefore, the gain ratio tends to select the features with a small number of samples, which is normally achieved by finding \( g(D, A) \) larger than the average level within the undetermined features, and selecting the largest feature.

Gini coefficient shows the possibility of classification errors in attribute weights of randomly selected samples. The smaller the Gini coefficient is, the less likely the selected samples gone through a wrong classification. Gini coefficient is used to select classification features in decision tree, which is calculated by formula 10.

\[
 Gini(p) = \sum_k p_k (1 - p_k) = 1 - \sum_k p_k^2 
\]

Based on feature \( A \), the Gini coefficient of sample set \( D \) is calculated as formula 11:

\[
 Gini(D, A) = 1 - \sum_{i=1}^{k} \left( \frac{\log i}{|D|} \right)^2 
\]

Within the formula, \( D \) is the data set, \( D_i \) is the subset of the data set \( D \) classified according to the number \( I \) of feature \( A \). \( n \) is the total number of feature \( A \), and \( Gini(D, A) \) is the Gini index when the sample set \( D \) is divided based on feature \( A \).
3.2 Design and Optimization of Word Vector Representation Module

In early studies, natural language processing usually regards text as a set of words, focusing only on the number of words in sentences. By default, it is assumed that each word in the document is independent, namely, Bag-of-words Model (BOW), the idea of "BOW" is to put all words in a bag. In Figure 5, an example is shown. From the figure, it can be seen that only one position in each city vector is 1, and the rest is 0. The city vectors are independent of each other.

![Figure 5 Example of city names of bag of words model (BOW)](image)

In such a case, it is difficult for computer to understand each word, so it is not conducive to understand coherent statements.

In addition, the dimension of vector represented by word bag model is related to the number of words in the database. If a matrix is used to represent all cities in the world and each city corresponds to its own vector in the matrix, the matrix will be very sparse, leading to dimension disasters. This increases the cost of calculation and may result in overfitting problems that can reduce the performance of the model.

Through a hybrid word vector representation method combining Bag-of-words Model (BOW) and Word-Embedding-Model, the Word-Embedding-Model can transform the sparse vector generated by Bag-of-words Model (BOW) into low dimensional continuous value, which is called dense vector. In addition, for the problem that each word vector is independent of each other, the relationship between words is represented by distance in the Word-Embedding-Model. Synonyms will be mapped near the vector space, so the amount of information contained in the word vector increases, which is more helpful for the computer to understand continuous sentences. One of the most typical models of word embedding model is Skip-gram model.

The full name of Skip-gram Model is Continuous Skip-gram Model. Suppose the current word is given, it can predict the content of a given word context. That is, the format of the given sample is \((w, x_t(w))\), w means the label of the word. When processing the word W, the subset of negative sample is \(NEG^*(w)\), and u is any word in the text. The maximum value is obtained by formula 12

\[
g(w) = \int w \in Tiveconteart(w), E_{10}(w) p(u | w)\]

In which:

\[
p(u | w) = \begin{cases} a(vw)^{\theta u^{s}, u^{s}}(u) = 1 \\ 1 - a(vw)^{\theta u^{s}, u^{s}}(u), v^{(s)} = 0 \end{cases} \]

\[
l^{k}(u) = \log Tunceg(w) = \begin{cases} 1, u = w^2 \\ 0, u \neq w \end{cases} \]

Within the formula, \(l^{k}(u)\) is a logarithmic likelihood function, C is the attribute weight, and formula 13 defines the probability of w as the context.

In Figure 6, t represents the current word order, the input layer represents the word w as vector trial \(v(w)\), the projection layer is still \(v(w)\), but the projection layer contains a weight matrix, which
outputs an "embedded word vector" for each input word, and the hidden layer outputs the final "word
vector" of the input word. The output layer is a Huffman Tree, the nodes of which represent the words
existing in the database, and the weight is the word frequency:

![Figure 6. Neural network structure of skip-gram model](image)

The hidden layer has the function similar to the look-up table, and the structure is shown in Figure
7

![Figure 7. The function of hidden layers of skip-gram model](image)

3.3 Design and Optimization of Word Weight Adjustment Module

The importance of a word can be measured by its frequency. At first, the system weighted the word
vector in this way. According to the daily experience, the more frequent words are more likely to be
important. Term frequency (TF) refers to the number of words in the text. This value is often
processed through normalization

\[ tf_{ij} = \frac{m_i}{x_i} \]  

(15)

In formula 15, \( tf_{ij} \) is the frequency of the selected words in the document, and \( m_i \) is the number of
the word \( W \) existing in the text. \( M \) is the total number of words in the document.

If the importance of words is determined by word frequency alone, the common words could be
mistakenly used as keywords. Inverse Document Frequency (IDF) represents the number of texts
containing this word in all texts to be processed. If the number of texts with the word \( W \) is less, the
IDF value of the word \( W \) is greater, which indicates that the word \( W \) has excellent discrimination:

\[ idf_{w} = \log\left(\frac{w}{n+1}\right) \]  

(16)

\( idf_{w} \) is the reciprocal of the featured term calculated by formula 16. \( N \) is the total number of
documents in the database, and \( n \) is the number of documents with word item \( w \).

Term Frequency Inverse Document Frequency (TF-IDF) is a model to judge the importance of
words in attribute weight. If the word appears many times in the file and the number of documents
containing the word is very small in the whole data set, the weight will be very high. Therefore, the
function of TF-IDF is to retain key words and delete common words. The final weight of a word is
calculated by formula 17:
\[ w_i = tf_i \times idf_i = \frac{mi}{x} \times \log\left(\frac{3y}{m+1}\right) \]  

(17)

Normalization excludes the influence of document length on weight calculation, and obtains formula 18:

\[
\begin{align*}
  f(x)(x) \\
  f(x) = \frac{(x+1)^2(x-1)^2(x-1)^2\left(\frac{x}{x+1}\right)}{\sqrt[4]{2(x)^2\left(\frac{x}{\ln 14}\right)}}
\end{align*}
\]

(18)

4. Analysis of Experimental Results

4.1 Construction of Experimental Environment

Due to the large amount of training computation of deep neural network, it will waste a lot of time to train on the CPU of conventional computer. Therefore, the design and optimization of the Intelligent Question Answering System are based on the GPU provided by the laboratory. The experimental environment is shown in Table 1.

| surroundings   | Configuration information |
|----------------|---------------------------|
| hardwareinformation |                           |
| CPU            | i7-9700K                  |
| RAM            | 16GBDDR4-4000             |
| Harddisk       | 1TBSSD                    |
| Network        | 100Mbps                   |
| Operating system | Windows10                |
| Programming language | Python3.6.5             |
| Softwareinformation |                      |
| IDE            | PyCharm2017.1             |
| Database       | Mysql5.5.19               |

4.2 Chinese Text Cleaning and Word Segmentation Results

The main purpose of Chinese text cleaning is to remove punctuation and stop words to make the meaning of sentences more obvious. The Chinese word segmentation used in our system is based on probability and statistical model. The task of word segmentation is to find the most possible solutions among these schemes. This paper involves three different attribute weights, which are quite different in content and features, so this section will explain each attribute weight one by one. Figure 8 shows the frequency distribution of common words in question corpus and response corpus within the three attribute weights.
Among them, the language Q & A attribute weight and front end design attribute weight are more professional, and more repetitive words will be involved in the attribute weight, so the color segments in the dispersion diagram are more concentrated and easier to be observed. The topic of daily Chinese chat attribute weight is divergent, there is no clear scope, so it involves more words. Compared with the other two attribute weights, the words with higher frequency in the attribute weight are also sparse and scattered in the dispersion diagram, and there is no phenomenon of concentrated continuous segments.

4.3 Performance Comparison of Word-Vector-Representation Methods

In the word vector representation method, we compare the performance of the Bag-of-Words model and the hybrid model using BOW model and the Skip-gram model. As can be seen from Figure 9, when the learning rate is roughly the same, they could be compared with each other.

The hybrid model of Skip-gram model and BOW model uses multiple center words to generate the final word vector. Specifically, first a word selected as the first central word, and then another word closer to the first one is found as the second central word. In this way, if the most similar word of the first central one has appeared, the second similar word will be selected. If the second similar word also
exists, the third similar word will be selected, and so on. In this paper, the Weighted-Cross-Entropy-Loss function is used to describe the performance. The number of training steps is set to 4100 when dealing with the weight of linguistic Q & A attributes and the weight of front end design Q & A attributes. So far, the value of the loss function has been stable. When dealing with the attribute weight of daily Chinese chat, the number of training steps should be increased to 8000, and the result can be obviously stable. However, no matter what kind of content and feature attribute weight, the experimental results show that the hybrid model of BOW model and Skip-gram model can significantly improve the performance.

It can be seen that the hybrid model of BOW model and Skip-gram model has achieved good performance under three different attribute weights, so this optimization method of word vector representation is not limited by the scene, and has good generalization.

5. Conclusion
This paper studies a hybrid word vector representation method, which optimizes the word vector representation by combining Skip-gram model with Bag-of-Words model. It retains the advantages of BOW model in dealing with discrete vectors, and uses Euclidean distance to express the relationship between words, increasing the correlation between words. Also the method transforms sparse vectors into dense vectors. The "hybrid model" is composed of two parts. The Retrieval-Matching model is used to answer highly targeted questions, and the Generative-Diaglogue model based on attention mechanism can respond to a wider range of situations outside the corpus.

References
[1] Tajane K, Dave S, Jahagirdar P, et al. AI Based Chat-Bot Using Azure Cognitive Services[C]. 2018 4thInternational Conference on Computing, Communication Control and Automation, ICCUBEA 2018, 2018:1–4.
[2] Abdul-Kader S A, Woods J. Question Answer System for Online Feedable New Born Chatbot[C]. 2017 Intelligent Systems Conference (IntelliSys), 2018: 863–869.
[3] Bozzon A. Enterprise Crowd Computing for Human Aided Chatbots[C]. 2018 IEEE/ACM 1st International Workshop on Software Engineering for Cognitive Services (SE4COG), 2018: 29–30.
[4] P. Kucherbaev A B and G H. Human-Aided Bots[J]. IEEE Internet Computing, 2018, 22(6): 36–43.
[5] Reshmi S, Balakrishnan K. Implementation of an Inquisitive Chatbot for Database Supported KnowledgeBases[J]. Sadhana, 2016, 41(10): 1173–1178.
[6] Singh R, Paste M, Shinde N, et al. Chatbot using TensorFlow for small Businesses[C]. 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018:1614–1619.
[7] Hussain S, Athula G. Extending a Conventional Chatbot Knowledge Base to External Knowledge Sourceand Introducing User Based Sessions for Diabetes Education[C]. 32nd IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2018, 2018: 698–703.
[8] Ouyang J, Qi W, Wang Y, et al. SDA: Software-Defined Accelerator for General-purpose Big DataAnalysis System[C]. 2014 IEEE Hot Chips 26 Symposium (HCS), 2017: 1–23.
[9] Serban I V, Sordoni A, Charlin L, et al. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues Iulian[C]. The Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17),2017: 3295–3301.
[10] He R, Lee W S, Ng H T, et al. An Unsupervised Neural Attention Model for Aspect Extraction[C]. The 55th Annual Meeting of the Association for Computational Linguistics, 2017: 388–397.
[11] Baziotis C, Pelekis N, Doulkeridis C. DataStories at SemEval-2017 Task 4: Deep LSTM with Attentionfor Message-level and Topic-based Sentiment Analysis[C]. 11th International Workshop on Semantic Evaluations (SemEval-2017), 2017: 747–754.