Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 7453653, 9 pages
https://doi.org/10.1155/2022/7453653

Research Article

Design of a Higher Education Question and Answer System Based on Multimodal Adversarial Networks

Xinyi Fu [1,2]

[1] *School of Marxism, China University of Mining and Technology, Xuzhou, Jiangsu 221116, China*
[2] *Department of Chinese Language and Literature School of Foreign Languages, Taihu University of Wuxi, Wuxi, Jiangsu 214064, China*

Correspondence should be addressed to Xinyi Fu; lb20140015@cumt.edu.cn

Received 24 May 2022; Revised 28 June 2022; Accepted 4 July 2022; Published 5 September 2022

Academic Editor: Ning Cao

Higher education question and answer system is a current research hotspot in the field of natural language processing and artificial intelligence, which can return accurate answers directly and also allow users to input in natural language, avoiding the need to return large collections of valid and invalid data links for users to filter twice. In recent years, deep learning techniques have made great progress in the field of natural language processing, thus making it possible to apply it in the field of teaching and learning in junior high school. Most of the traditional quiz systems based on traditional retrieval techniques have problems such as insufficient semantic portrayal of text, inability to extract semantic features in context, and poor processing of complex utterances. In order to fill the gap and deficiencies of question and answer systems in junior high school teaching, this paper applies multimodal and adversarial network-related technologies to build a multimodal adversarial network-based question and answer system for higher education. The experimental results show that the principles of the traditional application system and the multimodal adversarial system designed in this paper are similar, but the current application system is relatively more effective.

1. Introduction

Nowadays, society's resources are becoming more networked, and the evolution of the Internet has eventually evolved into a database with vast amounts of information resources, and the Internet has become an essential source of data for living, working, and learning [1]. Today's Internet has a wide variety of search engines, some of the more successful being Google and Baidu. Users can simply type a relevant search phrase into their search engine interface and immediately get a link to the relevant web page. However, there are still many drawbacks to these forms of search [2].

A leading expert on search engines, Roperstarch, conducted a survey on search engines in 2001 and concluded statistically that 36% of respondents spend an average of more than two hours a week searching for information on the Internet [3]. 71% of respondents find using search engines a hassle, taking around 12 minutes to set up their search criteria, and 46% of these people are frustrated by incorrect links [4]. 46% of these people are dissatisfied with incorrect links; 85% of respondents agree that there should be a more accurate way to distinguish and locate information [5]. This shows that keyword-based search techniques have the following disadvantages [6].

1.1. Too Much Relevant Information. When a user types a keyword into a search engine, the search engine will give links to all the pages that are linked to the keyword [7], which can be very large. Generally speaking, search engines have separate algorithms for ranking links, but even so there are a lot of unrelated, identical, and useless data. Therefore, the user has to waste a lot of time to search the results of the second selection [8].

1.2. Search Is Specific. In terms of use, search engine users are becoming more familiar with the keyword approach of the

past [9]. Although this ability to abstract the desired problem into keywords is improving, in many cases, it is difficult to express the user's needs in a few simple words. Natural language is a natural way for people to communicate, and people can use it to express their needs [10].

1.3. Poor Retrieval Results. The traditional method of information retrieval is too simple; it only matches keywords and does not analyze the user's deeper needs [11]. Therefore, even an efficient and precise match is unlikely to properly return the real required information to the user [12].

After the above analysis, the traditional keyword matching search engine technology is far from satisfying the real needs of users [13], who want a more intimate and accurate search technology to locate the required information. Traditional keyword matching search methods result in a large amount of irrelevant data, spam, and duplicate data [14], and useful information is usually buried in these data [15]. It is therefore of great importance to explore more efficient and user-friendly ways of providing search services to users. There is no doubt that a multimodal adversarial network question and answer system is the way forward for question and answer systems, as opposed to keyword matching search methods [16]. The aim of this system is to build a multimodal question and answer interface that will ensure that users can perform question and answer functions quickly, easily, and accurately [17].

2. Introduction to the Technology

2.1. Deep Neural Networks. The basic deep neural network (DNN) is usually composed of three parts: input layer, hidden layer, and output layer. Its basic structure is shown in Figure 1 [18].

Generally speaking, one layer is the input layer, one layer is the output layer, and the middle layer is the hidden layer. The layers are completely connected; in other words, at level I, any kind of neuron is connected to any kind of cell in layer I +1. There is a linear relationship between the inputs and outputs, resulting in an intermediate output result [19]:

$$z = \sum w_i x_{i_{i=1}} + b. \tag{1}$$

In the classification of linearly separable datasets, linear classifier is generally used. But the actual data often cannot be linearly segmented; for this problem, there are usually two solutions: introducing nonlinear function and linear transformation. In this context, nonlinear functions are called activation functions and generally include sigmoid, Tanh, ReLU, and so on. When training DNN, loss function and optimization algorithm are usually combined [20].

2.2. Deep Learning-Based End-to-End Question and Answer Systems. Before deep learning techniques became popular, most question and answer systems utilized more or less machine learning techniques, combining several subsystems such as question understanding, information retrieval, and answer extraction into a more complex question and answer system. Not only does this require a great deal of feature engineering for the question and answer dataset, but also each module of the system requires a large number of complex algorithms to implement, making it difficult to scale and expensive to maintain and therefore difficult to commercialize [21].

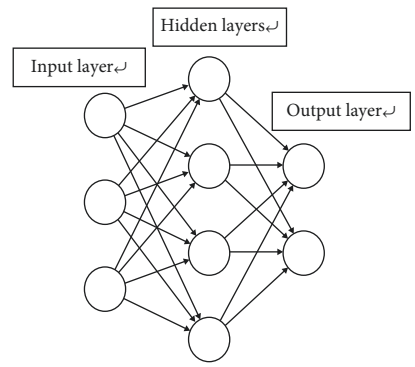

Figure 1: DNN structure.

In recent years, deep neural networks have achieved outstanding results in tasks such as image processing and speech recognition, demonstrating excellent representational learning capabilities. At the same time, there is an emerging trend to study the processing and representation of text through deep neural networks. In recent years, RNNs have shown good results in language representation and generation and have therefore become the preferred modelling solution for mainstream natural language processing.

2.3. Multimodal Techniques. Multimode machine learning (MMML) aims to process and understand multisource modal information by using machine learning techniques. The source and form of any kind of information is a mode. For example, humans touch, hear, see, and smell; information includes voice, video, text, and so on. Each of these can be called a mode. The diagram is shown in Figure 2.

Each source or form of information can be referred to as a modality. For example, people have the senses of touch, hearing, sight, and smell; people can also get information in the media of speech, video, and text and a variety of sensors such as radar, infrared, and accelerometers. Each of the above can be called a modality.

Modality can also be defined very broadly, for example, we can consider two different languages as two modalities or even two datasets collected in two different situations as two modalities.

Today, multimodal technology has a wide range of applications, such as Taobao image search, AI captioning, AI virtual digital human, humanoid interaction, intelligent assistants, product recommendations and infomercials, graph vector retrieval of face frames from video frames, voice interaction, and so on.

The technical points of multimodality are as follows:

(i) Multimodal representation learning (representation)

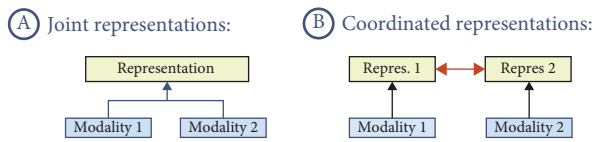

Figure 2: Architecture relationship diagram.

(ii) Representation of materialized information (text, images) by vectors, divided into joint representation and co-representation

(iii) Joint representation: mapping information from multiple modalities together into a unified multimodal vector space

(iv) Co-representation: each modality is mapped separately to its own representation space, but the mapped vectors satisfy certain correlation constraints (e.g., linear correlation)

Multimodal representation learning is essentially about whether the corresponding feature relationships in vector space are consistent with the original information.

2.3.1. Multimodal Fusion. This is one of the earliest research directions in MMML and is currently the most widely used. Other common aliases exist, such as Multisource Information Fusion (MIF) and Multisensor Fusion (MF) .

(i) Related tasks

(ii) Visual-audio recognition

(iii) Multimodal sentiment analysis

(iv) Mobile identity authentication

(v) Co-learning

This is the main technical point of multimodal research, with its multitasking approach to learning across different modalities. This is the main technical point of multimodal research, where knowledge from resource-rich (e.g., large data volumes) modalities is used to support the modalities with fewer resources (e.g., smaller data volumes) to build models. Collaborative learning essentially exploits the scarcity of resources in different modalities.

3. Application Design

3.1. Design Overview. This higher education Q&A system is a part of the school website, which provides services for the whole website. Therefore, in this Q&A system, the interface designed is a service interface, and the interface is a public interface, you can input a question, and then the answer to a question is output in Xml format. The whole system consists of interface layer, logic layer, and data access layer. Here is a brief overview of each interface layer.

Interface Layer. The service provider implements external access through the interface, mainly the ontology construction of OntoBuild, and its filtering function is StockPick. External system requests are implemented by sending type HTTP.

Logical Level. The main role of this level is to deal with the main logic presented. Classes in the OntoBuild package are primarily used to build ontology modules, SentPro and SentAnalysis packages are used to analyze problems, the Search package is used to analyze patterns and queries, and the CallServer package has the ability to integrate external service interfaces and solutions.

Data Access Layer. This layer mainly accesses the database, and the classes in the DbOperate package access the database and perform the corresponding queries.

The Q&A system designed in this paper is divided into four parts: FAQ database, problem analysis, information retrieval, and solution. After the user enters a question, it can be queried against the MMAN database. If the same question exists, the corresponding answer is returned to the user from the MMAN library. This preprocessing method can make the system quickly get answers to common problems, omit the tedious semantic analysis and other processing processes, and greatly improve the speed of query.

3.2. Knowledge Base Construction Options

3.2.1. Building the Ontology Knowledge Base. This project uses the Resource Description Framework (RDF) format for data storage. The generation of this format is done through the Jena API, while the storage and querying of RDF is done using TDB and Joseki. The RDF format was chosen because the triad structure of RDF is largely similar to the subject-verb-object composition of natural statements, which makes it natural to store state; also, the natural language it corresponds to is somewhat similar to the SPARQL language, which makes it easy to convert queries to SPARQL queries.

3.2.2. Resource Description Framework. RDF (Resource Description Framework) is a data model for the Semantic Web that defines RDFS and OWL and consists of a collection of triples, which are presented as a series of three elements that form a triple, including Subject, Predict, and Object. If several triples have elements in common, then these triples can form directed graphs, with the subject-object of a triple forming the nodes of the graph and the predict of a triple forming the edges of the graph.

RDF identifies concepts by using a specific URI and characterizes this concept by an attribute, which is completed by a specific value; RDF has a variety of different attribute values, with numbers, times, dates, and strings being one attribute value. In theory, any kind of data that has a URI is a resource, and http://www.w3school.com.cn/rdf is a resource. An attribute is also a resource, but it has its own name, for example, “author” or “homepage.” An attribute value is defined as the value corresponding to the attribute, and “David” or “http://www.w3school.com.cn” are attribute values. As shown in Figures 2 and 3, it is clear how RDF describes data, where resources and values are represented by circles, and attributes are represented by edges.

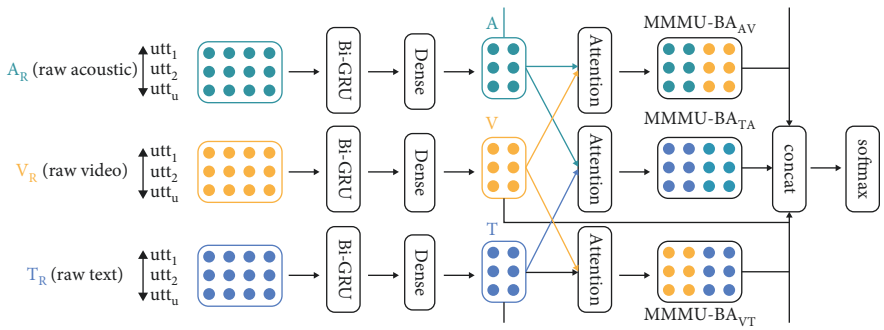

Figure 3: System flow diagram.

3.2.3. Intra-Domain Ontology Construction. The domain ontology library contains the classes (concepts), instances, relationships, and other elements of the ontology, which are constructed through an inductive process, with the intervention of domain experts. The process is top-down. There are three main steps in the process.

(1) Text collection and collation

(2) Extraction and classification of concepts

(3) Relation extraction

3.3. Structure of the MMAN Subsystem. The focus of the MMAN subsystem consists of three main parts: finding candidate problem sets, calculating sentence similarity, and updating the data in the MMAN library. The sentence analysis flowchart in the system is shown in Figure 3.

3.3.1. Searching for candidate sets of questions. In order to improve the efficiency of the MMAN library search, we would like to first perform a general filtering of the set of frequently asked questions, suggesting a part of it to form a candidate set and thus a small set, so that the subsequent steps of searching and calculating similarity can be controlled in this candidate set. In general, only 50% of the candidate set is selected for screening in the MMAN subsystem. Suppose we want to select one of the target questions, which contains a total of ^{n} words: $^{W}1$、$^{W}2$、…、^{W}n.

The total number of questions in the whole MMAN library is m, so for the $i(1 \le i \le m)$question there should be ^{n}i words $^{Q}1$、$^{Q}2$、…、^{Q}n. Then, the number of words that appear between the i question and the target question at the same time is denoted by ^{Num}i.

Then, $^{Num}{}_{i} \mid \{W_1, W_2, \ldots, W_n\} \cap \{Q_1, Q_2, \ldots, Q_n\} \mid$. Based on the Num_i value, the top half of the total number of questions with the larger value is filtered out, and a candidate set of questions is then successfully selected.

For the calculation of ^{Num}i values, it would be too much work to find the number of overlapping words by comparing the questions in MMAN with the target questions sentence by sentence. Therefore, in order to find the number of questions containing the same word as the target question in a quicker and easier way, we used this data structure for the filtering calculation, as shown in Figure 4.

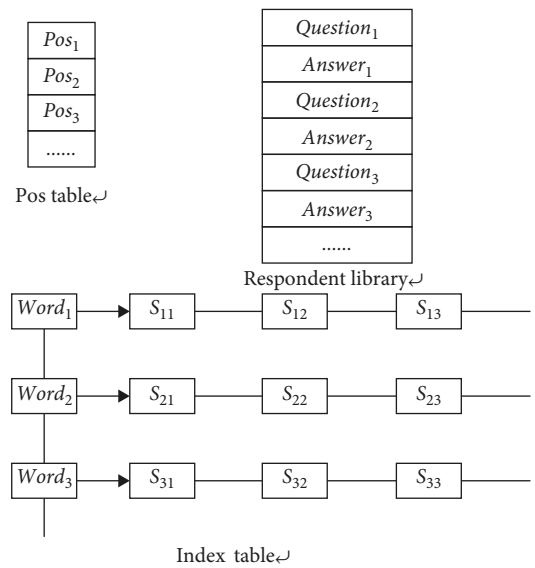

Figure 4: Data structure for finding candidate problem sets.

For Figure 4, the MMAN inventory contains all the sentences and their corresponding answers, while the POS table shows the location of each sentence. The index tables Word_1, Word_2, and …… are chained lists of the words of the sentences after sorting and ordering. Each Word_i corresponds to an ^{S} -link table, and each node in this ^{S} -link table represents the sentence number of a question.

The first step in a real search is to find the word W in the *Word* list. Then, based on the ordering of the Word table, we can easily locate the target node within the time complexity built by $O(\log n)$ by taking a half lookup. If we denote this target node as Word_k, we can find out which question contains this Word_k based on its corresponding S-link table. Repeating this process to find the corresponding question for each word in the target question will quickly give us the value of Num_i as described above.

Once the above steps are completed, we can obtain the sentence number of the first half of the candidate questions with a relatively large Num value and then locate the file location of the candidate questions in the MMAN library

```
(i) Num array is cleared to zero
(ii) while (the word in the question is not processed)
(iii) Find the next word in the question W
(iv) Find the same element in the Word list as W using the fold-and-half lookup method Wordk
(v) if(Wordk = W)
(vi) for(Wordk corresponds to each element of the S-linked table Si)
(vii) Num[Si] plus 1
(viii) store the subscript of the 50% largest element in the Num array in the Index array
(ix) for(each element of the Index array Indexi)
(x) Read the POS value corresponding to Indexi from the POS table, i.e. POS[Indexi]
    Read the question and answer in the MMAN file at POS[Indexi]
```

Pseudocode 1: The pseudocode of algorithm for finding the set of candidate questions.

according to the *POS* table and read and write them out, completing the whole search process.

The algorithm for finding the set of candidate questions has the following pseudocode.

3.4. The Design of a Question Comprehension System. To answer a question correctly, it is necessary to clarify two issues: firstly, what knowledge is involved in the question; secondly, where the key to the question lies. When analyzing the interrogative, the following steps need to be gone through.

(1) Subprocessing of the interrogative and labelling of the relevant lexemes

(2) Preprocessing of the question and parsing of the relevant words

3.4.1. Question Analysis of the Question and Answer System. Problem analysis is to analyze the structure of the problem, grasp the key words, find out the key of the problem, and determine the type of the problem by understanding the grammar of the problem. In the problem analysis part, it mainly includes the following:

(1) Analyze the specific types of problems

(2) Find the key words contained in the problem

(3) Expand the keywords contained in the problem

From the perspective of natural language processing, word segmentation and POS tagging are also needed.

3.4.2. Question Classification. In general, questions can be classified into three categories: (1) definitions; (2) lists; and (3) statements. For the processing of different types of questions, a set of answer extraction rules is customized and applied to the question answer extraction stage. In general, most automated question and answer systems categorize questions in advance, but this categorization often has many loopholes, such as the fact that the questions are not sufficiently detailed to meet the actual requirements of the user due to many human factors. As a result, a number of researchers have proposed their own approach to categorization. First, a series of questions are collected and used as a training corpus; then, a program is used to count the question phrases that occur more frequently. For example, the cue words for asking why were "What is the difference?", "what is the difference", "what is the difference?" The key words used in this study were "What is the difference?"

3.4.3. Keyword Expansion. It is common for question and answer systems to use keyword expansion to achieve a significant increase in recall. However, given the large semantic gap between the document and the question space, querying the extracted keywords on the basis of question analysis gives less than satisfactory results. The main reason for this is that one or some of the words in the question are likely to be an extension of the keyword rather than the keyword itself.

Although the expansion of keywords is beneficial to the system's recall rate, the degree of expansion is not easy to grasp, and if it is excessive, it can affect the correctness of the retrieval rate. Therefore, question and answer systems are cautious when it comes to keyword expansion. They usually set a series of restrictions, such as limiting expansion to noun or verb keywords only. In addition, there are some question and answer systems that make full use of the relevant documents returned after the search when expanding keywords. The keywords are expanded, but their importance is greatly reduced. For this reason, many question and answer systems assign a certain weight to keywords as a way of clarifying their importance and thus achieving greater system accuracy.

Keyword expansion is part of the question analysis module, as keyword expansion aims to address which words are involved in the question, while search formula construction is part of the document retrieval module, as search formula construction aims to address possible correlations between these keywords and the target document.

3.4.4. The Overall Process of Parsing Question Sentences. Unlike English sentences, Chinese interrogative sentences cannot be divided into words using spaces but can only be preprocessed. The first task in the analysis of interrogative sentences is to divide them into words, mark their lexical

properties, and then analyze them in depth. Here we look at the preprocessing of interrogative sentences, which includes (1) concept recognition; (2) synonym conversion; (3) numerical conversion; and (4) logical word recognition. Finally, the contents of this question bank of sentences are parsed according to the parsing method of interrogative sentence analysis.

3.5. Analysis of the Answer Extraction Process. In the design section, the answer extraction process of the ontology library is discussed, while the text-based answer extraction will be discussed in the system implementation section. In general, questions are first uploaded by the user and then analyzed by the system before being passed on to the answer extraction module, which will use the Jena toolkit to complete the querying of answers in the ontology knowledge base. Through the analysis of commonly used question sentences, the system classifies them into six categories using the following structure.

3.5.1. Asking for the Value of an Attribute Relationship. Example + attribute relationship, e.g., how does the nursing profession conduct operational examinations?

3.5.2. Asking about Relationships. Example + example, e.g., how does a marketing major take the college entrance exam?

3.5.3. Asking for "Whether". Instance + attribute relationship + example, e.g., can your school recruit students for post-secondary education?

3.5.4. Asking for Examples. Instance + attribute relationship + class, e.g., what are the employment opportunities for accounting majors?

3.5.5. Asking for the Class Name. Class + find subclass identifiers, for example, what kinds of employment?

3.5.6. Unordered Keywords. For a: we should complete the query operation of the corresponding instance attribute relationship value in the ontology knowledge base; if the corresponding attribute relationship value cannot be found, the parent class instance will be looked up and its attribute relationship value will be extracted as the answer; if the lookup of the parent class instance is fruitless, it should be traced back to the corresponding instance of the first level class. For b: the triples (connected by relationship identifiers) in the knowledge base are scanned and the corresponding relationship identifiers are identified and their attribute relationship values are then extracted.

For c: complete the search for two instances in the ontology knowledge base and clarify whether such an attribute relationship exists between them; if they do not have a direct triadic relationship, inference should be described following custom rules.

For d: find two triad instances in the ontology knowledge base.

(1) An instance of the attribute relationship matching class.

(2) A neutral instance to determine if it has a direct triadic relationship; if not, then inference should be made following custom rules.

For case e: in the ontology knowledge base, the subclasses of the class are looked up and analyzed and used as the answer.

For case f: as far as unordered keywords are concerned, the main components are

(1) The name of the attribute relationship.

(2) The name of the class.

(3) The instance name; the system will use the above five cases as a basis for arranging and combining the keywords and finding the relationships between them, which will then be made available to the user to facilitate further interaction between users.

Cases other than the six above should be dealt with by means of user interaction.

4. Application Implementation and Analysis of Results

4.1. Application Algorithm Implementation

4.1.1. Objective Function. We define the objective function of the model as follows:

$$L = \max\left(0, m - \cos\left(q, a^{+}\right) + \cos\left(q, a^{-}\right)\right), \tag{2}$$

where $a^{+}$is a vector of positive example answers, $a^{-}$is a vector of negative example answers, and m is a threshold parameter. The meaning of the objective function is to make the cos of the problem and the negative example answer smaller than the cos between the problem and the positive example answer. The SGD optimization function is used here.

4.1.2. Evaluation Metrics. In this paper, accuracy and MRR (mean reciprocal rank) are used to evaluate the final results achieved by the two models. The detailed definitions of these two metrics are as follows.

4.1.3. Accuracy. The accuracy rate is the ratio of the number of correct responses to the total number of samples tested.

Accuracy rate = number of correct responses/number of samples tested.

4.1.4. MRR. MRR is a common international metric used to measure the effectiveness of search algorithms, and the core idea is very simple: the stronger the returned result set, the better the results. The MRR is calculated as $1/n$ if the first correct answer is in the nth position for a query, or 0 if there is no correct answer.

$$MRR = \frac{1}{Q}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i}, \tag{3}$$

where Q is the set of sample queries, $|Q|$ denotes the number of queries in Q, and rank_i denotes the ranking of the first correct answer in the ith query of the first correct answer.

4.1.5. Training Process

Step 1. Define the problem.

Step 2. Define the architecture of the GAN.

Step 3. Train the discriminator with real data for N epochs and train the discriminator to correctly predict real data as true. Here N can be set to any natural number between 1 and infinity.

Step 4. Generate false input data with the generator and use it to train the discriminator. The discriminator is trained to correctly predict the false data as false.

Step 5. Train the generator with the discriminator's in and out. When the discriminator is trained, its predictions are used as markers to train the generator. The generator is trained to confuse the discriminator.

Step 6. Repeat step 3 to step 5 for multiple epochs.

Step 7. Manually check if the dummy data are reasonable. If it looks right, stop training; otherwise, go back to step 3. This is a manual task and manually evaluating the data is the best way to check how fake it is. When this step is finished, it is possible to assess whether the GAN is performing well.

```
import torch
from torch import nn.
from torchvision.utils import save_image
import os
import numpy as np
batch_size = 100
# learning_rate = 0.0002
# epochsize = 80
sample_dir = "test_images"
If not os.path.exists(sample_dir):
os.makedirs(sample_dir)
class Generator(nn.Module):
def __init__(self):
super(Generator, self).__init__().
Self.label_emb = nn.Embedding(10, 10)
self.model = nn.Sequential(
nn.Linear(110, 128),
nn.LeakyReLU(0.2, inplace = True),
nn.Linear(128, 256),
nn.BatchNorm1d(256, 0.8),
nn.LeakyReLU(0.2, inplace = True),
nn.Linear(256, 512),
nn.BatchNorm1d(512, 0.8),
nn.LeakyReLU(0.2, inplace = True),
nn.Linear(512, 1024),
nn.BatchNorm1d(1024, 0.8),
nn.LeakyReLU(0.2, inplace = True),
nn.Linear(1024, 784),
nn.Tanh()
)
def forward(self, noise, label):
out = torch.cat((noise, self.label_emb(label)), -1)
Img = self.model(out) # torch.Size([64, 784])
Img = img.view(img.size(0), 1, 28, 28) # torch.-
Size([64, 1, 32, 32])
return img
G = Generator()
G.load_state_dict(torch.load('G_plus.ckpt'))
z = torch.randn(batch_size, 100)
# label = torch.LongTensor(np.array([num for _
in range(10) for num in range(10)]))
label = torch.tensor([7,8,1,3,4,2,6,5,9,0] * 10)
# label = torch.full([100], 9)
# label = []
# for i in range(10):
# for j in range(10):
# label.append(i)
#
# label = torch.tensor(label)
print(label)
print("label.shape:", label.size())

save_image(G(z, label).data, os.path.join(sam-
ple_dir, 'images.png'), nrow = 10,
normalize = True)
```

4.2. Application Experiments. Before vocabulary learning, input data should be converted into a word vector first, so we must use Gensim to train the word vector model. Since the principle of sentence vector is similar to that of word vector, its training steps and parameters are exactly the same as the training method of sentence vector in the previous section.

The main training steps of this model are as follows:

(1) Data entry: this experiment is simulated by question and answer (Q and A), where Q is the question and the answer is A

Table 1: Main parameters of the hybrid neural network.

Parameters	Parameter values
Batch size	128
Learning rate	0.05
Number of LSTM layers	1
LSTM layer size	256
Convolutional kernel window size	1,2,3
Number of convolution kernel window size	500
Number of iterations	300
Dropout	0.5

Table 2: Experimental results of the basic neural network.

	Traditional applications	Multimodal adversarial network applications
Response rate	0.66	0.68
Accuracy rate	0.71	0.75
Recognition rate	0.70	0.73

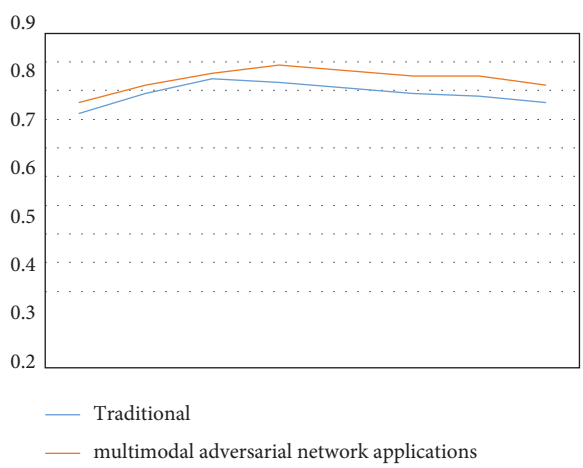

Figure 5: Overall efficiency comparison chart.

(2) Map the input questions and answers into vector expressions with embedding layer

(3) The same LSTM model is used to solve the problem and the solution

(4) MMAN algorithm is used to select the features of sequence LSTM, and it is next to the maximum pooling layer after the convolution layer

(5) Normalization uses Dropout and BatchNormalization to avoid over-fitting

Using the characteristics of the final problem and solution, the objective function is obtained. Table 1 shows the parameter values mainly used for training.

4.3. Analysis of the Application Results. The experimental results in Table 2 show that the principles of the conventional application system and the multimodal adversarial system designed here are similar, but the current application system is relatively more effective.

According to the table using Echarts for graphical visualization, its identification diagram is shown in Figure 5, where the yellow line represents the multimodal adversarial network application system and the blue line represents the traditional application system, and it can be seen that the overall results of the multimodal adversarial network application system are much better.

5. Conclusion

With the rapid development of the Internet, big data, artificial intelligence, and other technologies, new knowledge and technology emerge in an endless stream. How to apply these technologies to junior high school education is an urgent task to be solved. In this context, this paper first summarizes the research status of various chat robots and question answering systems at home and abroad and analyzes several existing question answering systems in detail. Combined with the proposed MMAN network model, it is applied to universities. Compared with the traditional question answering system, the question answering system based on deep learning technology can reduce the labor maintenance cost, can identify the user's questions more accurately, and can maintain the semantic context, so that the precision and scalability of the question answering system are greatly improved. Finally, the correctness of the model is verified by comparing the experimental results with the experimental results.

Data Availability

The dataset used in this paper is available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] L. Chen, "Research on the evaluation of domestic Internet search engines [J]," *Statistics and Decision Making*, vol. 3, pp. 63–66, 2017.

[2] X. Mao and X. Li, "A review of question and answer system research [J]," *Computer Science and Exploration*, vol. 06, no. 3, pp. 193–207, 2012.

[3] T. Demeester, D. Trieschnigg, D. Nguyen, and H. Djoerd, "Overview of the TREC 2013 federated web search track[J]," *National Institute of Standards & Technology*, vol. 5, pp. 1–11, 2017.

[4] S. Tellex, B. Katz, J. Lin, and M. Gregory, "Quantitative evaluation of passage retrieval algorithms for question answering[C]," in *Proceedings of the International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pp. 41–47, ACM, Toronto, Canada, July 2003.

[5] J. Lin and B. Katz, "Question answering from the web using knowledge annotation and knowledge mining techniques," in *Proceedings of the 20th International Conference on*

Information & Knowledge Management, pp. 116–123, ACM, Queensland, Australia, November 2003.

[6] B. Katz, G. C. Borchardt, and S. Felshin, "natural language annotations for question answering[C]," in *Proceedings of the 19th International Florida Artificial Intelligence Research Society Conference, Melbourne Beach, Melbourne. Intelligence Research Society Conference*, pp. 303–306, DBLP, Melbourne Beach, FA, USA, May 2006.

[7] Z. Zheng, "AnswerBus question answering system [C]," in *Proceedings of the International Conference on Human Language Technology Research*, pp. 399–404, Morgan Kaufmann Publishers Inc, Alicante, Spain, June 2002.

[8] H. Xiong and L. Xia, "Application of natural language processing techniques in Chinese full-text retrieval," *Intelligence Theory Xiong*, vol. 7, pp. 201–210, 2021.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[10] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Netw*, vol. 61, pp. 85–117, 2014.

[11] Y. Jang, Y. Song, Y. Yu, and G. Kim, "TGIF-QA: toward spatio-temporal reasoning in visual question answering[C]," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1359–1367, IEEE Computer Society, Honolulu, HI, USA, June 2017.

[12] H. Jiang, "Information Retrieval and the semantic web[C]," in *Proceedings of the International Conference on Educational and Information Technology*, pp. 461–463, IEEE, Seville, Spain, November 2017.

[13] X. Zhou, M. S. Zhao, and M. Hu, "A study on feature selection in Chinese text classification. Research on feature selection in Chinese text classification[J]," *Journal of Chinese Information*, vol. 18, no. 3, pp. 17–23, 2004.

[14] C. Y. Shi, X. U. Chao-Jun, and X. J. Yang, "Study of TFIDF algorithm[J]," *Journal of Computer Applications*, vol. 5, pp. 135–142, 2009.

[15] P. Bille, "A survey on tree edit distance and related problems," *Theoretical Computer Science*, vol. 337, no. 1-3, pp. 217–239, 2005.

[16] P. Soucy and G. W. Mineau, "A simple KNN algorithm for text categorization[C]," in *Proceedings of the IEEE International Conference on Data Mining*, pp. 647-648, IEEE, San Jose, CA, USA, November 2001.

[17] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization[J]," *Advances in Neural Information Processing Systems*, vol. 3, pp. 2177–2185, 2014.

[18] M. Faruqui and C. Dyer, "Non-distributional Word Vector Representations[J]," 2015, https://arxiv.org/abs/1506.05230.

[19] Y. D. Li, Z. B. Hao, and H. Lei, "A review of convolutional neural network research[J]," *Computer Applications*, vol. 36, no. 9, pp. 2508–2515, 2016.

[20] Q. Zhu and F. Sun, "Research on the application of RNN neural networks[J]," *Journal of Beijing University of Chemical Technology (Natural Science Edition): Natural Science Edition*, vol. 3, no. 1, pp. 86–90, 1998.

[21] Y. Sun, F. Lin, and Q. Zhou, "Research progress of long and short term memory networks in mobile scenarios[J]," *Modern Computer*, vol. 2, no. 35, pp. 10–15, 2017.