Question Answering using Random Forests

Xie Chuan (Registration Number: XIECH18204)

Github: https://github.com/cx16528/ce888labs/tree/master/labs/assignment

Journal Standard: IEEE Transactions on Knowledge and Data Engineering

Abstract—Question answering (QA) problems can represent most tasks in natural language processing. In this paper we introduce a random forests question answering system to solve this kind of problems. We push dataset into this system as the knowledge memory and the system can reason and search the answer from the knowledge memories. In this experiment we used the Facebook question answering dataset (Facebook's bAbI dataset). Encoding the dataset at first and then giving the results from random forests classifier. This system can deal with the different kinds of dataset not only the natural language dataset but also other language dataset even if machine language dataset. The system tasks relies exclusively on the random forest having the memory about new inputs (the random forest have been trained by same or similar dataset).

Index Terms—Question Answering, bAbI, Random Forest

1. INTRODUCTION

Question answering (QA) system is very popular in recent years. And it is a very complex natural language processing task. It is a difficult business to let the computer understand the natural language, find the relationship of words, and catch the answer of natural language. Most of question answering system are about machine translation like what is the translation into Chinese. Sentiment analysis is also a popular problem to be researched [1].

Random forests is proposed to deal with this question answering problem. This system is based on random forests module and can solve the question answering task within natural language.

This system first translates the input data (natural language stories and questions) into numeric representation of words. It can let the computer understand the relationship of stories, questions, and words. And then put this numeric data into random forests searching and predicting the answering.

Fig. 1 offers the examples of stories, questions and answers for tasks that are trained and tested in this paper.

```
1  1 John travelled to the hallway.
2  2 Mary journeyed to the bathroom.
3  3 Where is John?    hallway 1
4  4 Daniel went back to the bathroom.
5  5 John moved to the bedroom.
6  6 Where is Mary?    bathroom    2
7  7 John went to the hallway.
8  8 Sandra journeyed to the kitchen.
9  9 Where is Sandra?  kitchen 8
10 10 Sandra travelled to the hallway.
11 11 John went to the garden.
12 12 Where is Sandra?     hallway 10
13 13 Sandra went back to the bathroom
14 14 Sandra moved to the kitchen.
15 15 Where is Sandra?     kitchen 14
16 1 Sandra travelled to the kitchen.
17 2 Sandra travelled to the hallway.
18 3 Where is Sandra?  hallway 2
19 4 Mary went to the bathroom.
20 5 Sandra moved to the garden.
21 6 Where is Sandra?  garden  5
22 7 Sandra travelled to the office.
23 8 Daniel journeyed to the hallway.
24 9 Where is Daniel?  hallway 8
25 10 Daniel journeyed to the office.
26 11 John moved to the hallway.
27 12 Where is Sandra?     office  7
28 13 John travelled to the bathroom.
29 14 John journeyed to the office.
30 15 Where is Daniel?     office  10
```

The purpose of this paper is finding the answers from learning the stories and questions.

Fig. 1 Example stories, questions and answers

2. BACKGROUND

There are many related research about question answering system and they gave me many shoulders on this paper standing.

2.1 Question answering system

There are three papers researching the question answering problems in different measures.

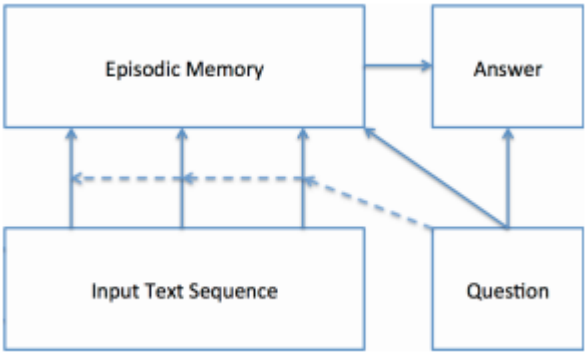

Fig. 2 Overview of DMN modules [2]

Kumar researched this problem by dynamic memory networks [2]. He proposed an improvement memory neural networks from memory neural networks that is dynamic memory networks [2]. He gave four modules in his dynamic memory networks and they are input module, question module, episodic memory module, and answer module [2]. The illustration of DMN is shown in Fig. 2. The input module and question module encoded the text words into numeric representation of words [2]. Episodic memory module was used to find and collection the input "memory" vector and find which previous vector is relevant with new information [2]. Answer module found the most relevant answer from the final memory vector of DMN system [2]. This is a neural networks system the memory vector was saved in the neuros and using the neural networks algorithm to find the final memory vector [2].

Jhenhao, Tingpo, and Yichuan generated a random coattention forest to solve the problem of question answering [3]. They used the glove word vector to describe the words in digital measure [3]. First they encoded the context and question word to find the context question interactions [3]. And then building the coattention random forest encoder to ensemble the several loosely correlated weak predictors. Final using the LSTM classification decoder found the final answer [3]. This system combines the neural networks and random forest method and generating a new algorithm to solve the situations of question answering [3].

2.2 Random forest in sentiment analysis

Yun and Qigang researched the sentiment classification system of Twitter data [4]. They studied seven classifiers on sentiment classification including random forest classifier [4]. They used the Twitter search API to collect the data about airline services and divided the data into four sentiment class that are positive, negative, neutral, and irrelevant [4]. Next they loaded this data into seven classifiers and the seven classifiers are Lexicon Based, naïve Bayesian, Bayesian Network, SVM, C4.5, and Random Forest [4]. And the result of this experiment showed the Bayesian Network is the best measure and the Random forest is the second [4]. The random forest is also a good classifier in sentiment analysis.

3. METHODOLOGY

In this experiment two modules were built. One is translatetorBabiTask. It processed the bAbI dataset into digital representation of text. Another is RandomForestClassifier and it classified the digital dataset to predict the answers. And the illustration of random forests question answering system is showed by Fig. 3

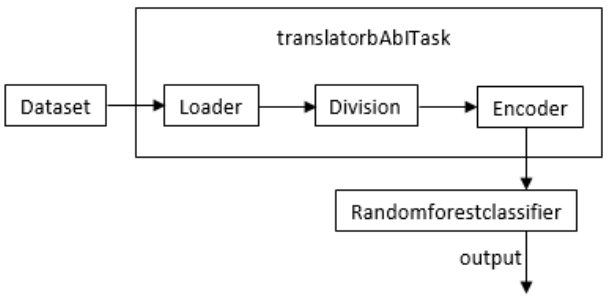

Fig. 3 Structure of random forest question answering system

3.1 translateorBabiTask

3.1.1 Loader

Using iteration to load the dataset one sentence by one sentence. And putting the sentence data into division to divide the data into stories and answers.

3.1.2 Division

In this experiment we needed the inputs and outputs for random forests to train and test. So we had to divide the data from loader. Because one story may have more than one question and one answer in this dataset, the story needed to be divided by the index of answers. If the

questions in a same story, we put the sentences before current question into current story. In this way can divide the questions in a same story and avoid the different answers mapping a same story data (different train_answers map same train_story). Example a story has 3 question and the first story data includes the first question and sentences before the first question. And the second story data should include the second question and sentences before the second question (including the first story). The third story data needs including the first and second stories. If there is a new story the new story data just including the data in the new story and the pre-story should not be included in the new story data.

3.1.3 Encoder

The purpose of encoder is encoding the text data into digital representation of words. In this experiment we used the dictionary function of Python to define the numeric representation of words. There we chose a sample measure that is regarding the words as the key of dictionary and when we found a new word we added it into dictionary and defined its value of dictionary as the current length of dictionary. It can make sure the different world mapping different number.

3.2 RandomForestClassifier

The RandomForestClassifier was used to classify the input dataset and predict the answers for the forest. The pre-task got the train_story, train_answers, test_story, and test_answers these four data. Loading these data we caught the accuracy scores of random forest question answering system.

The random forest algorithm is a measure depending on decision tree algorithm. There has many decision trees in the random forest and the decision trees are generated by the random data and random feature from original dataset. The most typical features of random forest are that the trees are random and the results of random forest is not same with same dataset. But the results is changed in a small range. The most advantage of random forest is that it reduce the computation complexity and keep the features form decision tree. It is a good measure to classify and regress the dataset.

3.3 bAbI

bAbI is the dataset which was using in this experiment [5]. This is a question answering dataset with natural language. It is generated form Facebook data. And Facebook provides the source code to generate the tasks dataset [5]. The dataset loaded in this experiment is download from the website of bAbI and that is http://fb.ai/babi [5].

4. EXPERIMENTS

There are two modules of this experiment. They are translatetorBabiTask and RandomForestClassifier. We load the train and test dataset into system. And then using the translatetorBabiTask digitize the words of dataset and divide the stories and answers. This digital dataset was loaded into the RandomForestClassifier and we can get the predictions. Final we computed the accuracy score of this system.

Table 1 accuracy score of random forests system

Task	Random Forests
1_single-supporting-fact	23.50%
2_two-supporting-facts	15.10%
3_three-supporting-facts	16.60%
4_two-arg-relations	50.90%
5_three-arg-relations	17.90%
6_yes-no-questions	49.60%
7_counting	60.60%
8_lists-sets	34.7%
9_simple-negation	61.20%
10_indefinite-knowledge	45.20%
11_basic-coreference	24.10%
12_conjunction	27.70%
13_compound-coreference	30.30%
14_time-reasoning	17.60%
15_basic-deduction	23.20%
16_basic-induction	30.10%
17_positional-reasoning	51.00%
18_size-reasoning	50.30%
19_path-finding	9.40%
20_agents-motivations	49.70%
Mean accuracy (%)	36.99%

Table 2 provides the result of random forests question answering system

5. DISCUSSION

The evaluation method is using RandomForestClassifier.score() function to catch the accuracy score of random forest classifier and compare the result with other results of question answering systems within same dataset that is bAbI.

According to the result of experiment we can find that the accuracy score is about 30%. It is not very high. This

system cannot find the relationship of every words and it is not enough for normal question answering situations.

Table 2 provides the results of MemNN (Memory Neural Networks) and DMN (Dynamic Memory Networks) systems [2]. Comparing with them the random forests module is not good at question answering problems. This random forests question answering system has many problem need solved and improved.

Table 2 accuracy score of random forests system, MemNN, and DMN

Task	Random Forests	MemNN	DMN
1_single-supporting-fact	23.50%	100.00%	100.00 %
2_two-supporting-facts	15.10%	100.00%	98.20%
3_three-supporting-facts	16.60%	100.00%	95.20%
4_two-arg-relations	50.90%	100.00%	100.00 %
5_three-arg-relations	17.90%	98.00%	99.30%
6_yes-no-questions	49.60%	100.00%	100.00 %
7_counting	60.60%	85.00%	96.90%
8_lists-sets	34.7%	91.00%	96.50%
9_simple-negation	61.20%	100.00%	100.00 %
10_indefinite-knowledge	45.20%	98.00%	97.50%
11_basic-coreference	24.10%	100.00%	99.90%
12_conjunction	27.70%	100.00%	100.00 %
13_compound-coreference	30.30%	100.00%	99.80%
14_time-reasoning	17.60%	99.00%	100.00 %
15_basic-deduction	23.20%	100.00%	100.00 %
16_basic-induction	30.10%	100.00%	99.40%
17_positional-reasoning	51.00%	65.00%	59.60%
18_size-reasoning	50.30%	95.00%	95.30%
19_path-finding	9.40%	36.00%	34.50%
20_agents-motivations	49.70%	100.00%	100.00 %
Mean accuracy (%)	36.99%	93.30%	93.60%

Within further research and comparing with this experiment I find the most question answering systems are using neural network algorithm. And the DMN is the most common and best system. It uses the glove word vector and word dictionary of train set as the embedding layer of neural network. Because of glove word vector this system can find the semantic relationship of words. SVD and word2vevc are two other algorithms of word vector [6, 7]. SVD is good at high speed processing and word2vec is good at finding the relationship of words [6, 7]. Glove is a method combine this two methods. It can find the complex relationship of words like $vking - vman + vwoman \approx vqueen$ with high speed [7,8]. It is a good way to solve the complex relationship of words in this question answering problems and the random forests system can be improved in this way.

Another finding from further research is that random forests method is popular in sentiment analysis. Yun and Qigang was focus on ensemble sentiment classification system of Twitter data for airline services analysis [4]. The random forests algorithm is good way to solve the sentiment classification in text analysis.

Coattention forest network is also a good way to approach the answers [3]. This system includes the glove vector, neural network, and random forest method. It has this three algorithm features and solve the question answering problem in a best way which paper I have read.

6. CONCLUTION

From this experiment we can find that the random forests module is a good classifier but it is not enough for question answering system. The accuracy score is just about 30%. The random forests was built by train set. When we input a question this module just find the answer from this built forests. It cannot update with the user input new stories and questions. If we want to update the module, we need to rebuild the random forests. It is a huge task and it will spend a lot of time. It is not useful in normal situations.

The further work is focus on improving the accuracy score and solving the updating of system. There are two measures to improve this system. First is changing the method of classification. Using the neural network is a good way to improve. According my research of

word2vec and glove algorithms, this way is also difficult to update the word vector lib in an easy and fast way. QLMBP maybe a good measure to improve the speed of neural network and the high speed can let system update the module in a short time [9]. The second way is that adding a processing before random forest module. This processing is choosing the most useful data and after this processing we can get a small but high relevant dataset. The random forests can search the answer and update itself quickly. And combining the random forest and neural network method is a good way to solve the issues about random forests system.

REFERENCE

[1] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank" In EMNLP, 2013

[2] K. Ankit, I. Ozan, O. Peter, I. Mohit, B. James, G. Ishaan, Z. Victor, P. Romain, S. Richard, "Ask Me Anything: Dynamic Memory Networks for Natural Language Processing" arXiv:1506.07285v5 [cs.CL] 5 Mar 2016

[3] C. Jhenghao, L. Tingpo, C. Yichun, "Random Coattention Forest for Question Answering" CodaLab ID: tingpo

CodaLab Group: cs224n-tingpo_jhenghao_yichun

[4] W. Yun, G. Qigang, "An Ensemble Sentiment Classification System of Twitter Data for Airline Services Analysis" *Data Mining Workshop (ICDMW),* 2015 IEEE

[5] W. Jason, B. Antoine, C. Sumit, M. R. Alexander, M. Bart, J. Armand and M. Tomsa, "TOWARDS AI-COMPLETE QUESTION ANSWERING: A SET OF PREREQUISITE TOY TASKS" arXiv:1502.05698v10 [cs.AI] 31 Dec 2015

[6] "More Optimization (SGD) Review" [Online].Available: http://cs231n.github.io/optimization-1/

[7] "Word2Vec Tutorial - The Skip-Gram Model" [Online].Available: http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

[8] M. Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.

[9] X. Wenshang, Y. Qingming, S. Yaliang, "Computational Intelligence and Security, 2007 International Conference on" *IEEE* 15-19 Dec. 2007