COMP90042 Project 2018: Question Answering

Username: wuangs, Team name: Wuang Shen

1. Introduction:

In modern life, with the booming of technology, quests for knowledge and information through network and applications becomes an essential part of people's daily life. Technologies like information retrieval and search engine can help people find relevant documents or information to satisfy their needs. However, people, in most time, are only interested in getting a specific answer to their question instead of reading long pages of information for their quest. Question Answering system, as a computer science field intersecting with natural language processing, information retrieval, machine learning and other knowledge-based technology, extracts specific answers from relevant documents to users' quest. In this paper, the QA system is formed by four phases - question type identification, question classification, sentence retrieval and answer processing. Approaches like named entity recognition, sentence syntactic dependency parser, random forest classification and BM25 support different phases of the system, to search answers to factoid questions. The objective of this paper is to analyse and evaluate the performance of these techniques and QA systems.

2. Phases of QA System and Relevant Techniques

2.1 Question Type identification

To narrow down the scope of the quest, it is necessary to understand what type of information (date, location, person, etc.) matches user's needs before retrieving relevant information. For example, if the question is "What is the capital city of Australia?", then the expected answer type is a location. Therefore, the aim of the question type identification phase is to identify answer type for each question.

2.1.1 Methodology for Question Processing

2.1.1.1 Named Entity Recognition

Named Entity Recognition (NER) is a subtask of information extraction that usually uses sequence models like HMMs to classify entities into pre-defined categories, such as Person, Location, Date (Mansouri et al., 2008). The performance and categories of NER depend on applications. NER features from spaCy "en_core_web_sm" model are used in this paper, which provides the question processing with NER feature at 85.3 in F1-score (spaCy, 2018). NER classified each answer from training and development dataset to its pre-defined categories. Undetected answer type will be tagged as type "OTHER".

2.2 Question Classification

After the system classified answers into pre-defined categories, corresponding question type can be decided for training and development dataset. The classification model is required to classify answer type for the testing dataset. The task of question classification is to classify question's answer type based on features of the question sentence. The main approaches to implement question classification is through hand-writing rule or supervised machine learning. Question classification in this project is built on the random forest (a machine learning algorithm). Although hand-writing rule can give a fast classification, it tends to be overfitting, and can not classify questions out of rules.

2.2.1Methodology for Question Classification

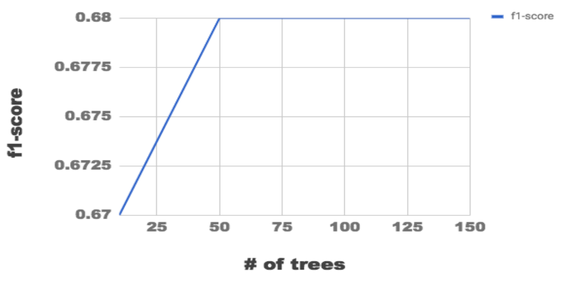
2.2.1.1Random Forest

A random forest is a group of classification trees constructed from the random selection of training data, whose prediction is based on the mode of multiple decision trees' classification results (Ali et al., 2012). Different from other machine learning algorithms like Naïve Bayes, features selection is included in the algorithm for the random forest. Scikit-learn (2011) package is used for constructing the random forest model for this project. Tokens in the question are extracted as model features.

2.2.2 Evaluation for Question Classification

The performance of question classification can be presented by the difference between pre-identified and predicted answer type for development dataset, which can be evaluated by F1-score. The accuracy of the random forest depends on the number of decision trees involved in the model.

As shown in the graph below, the performance of random forest tree can be optimal when the number of trees is around 50. Therefore, this parameter will be set at 50 to get higher accuracy and time efficiency.



Line Graph of F1-score for random forest with different number of trees

It is interesting to find that stopwords in a question have a significant impact on answer type. As shown in the table below, the model with stopwords features tends to have higher F1-score. This is mainly caused by some stopwords highly related to answer types such as “what”, “who”, “where” etc. Therefore, stopwords are included as features for the model, which is unusual practice in NLP tasks.

	Precision	Recall	F1-score
# of Trees = 10, with stopwords	0.66	0.72	0.67
# of Trees = 10, without stopwords	0.63	0.70	0.65

Table: Accuracy of two models

2.3 Sentence Retrieval

To extract answers from a sentence, the relevant sentence needs to be retrieved from a related document whose id is provided in testing dataset. First, a relevant paragraph needs to be extracted from a list of paragraphs. Then same information retrieval mechanism applies to paragraph level to retrieve a relevant sentence.

2.3.1 Methodology for Sentence Retrieval

2.3.1.1 BM25

Okapi BM25 is a ranking function used to rank documents based on their relevance to a given quest. Comparing with TF-IDF method, BM25 can directly rank the document based on the formula without using similarity methods like cosine similarity to incorporate. In this methods, relevance is presented by the probability a user will think the retrieval result is relevant. The score for each document is based on inverse document frequency, term frequency in the document, document length and query term frequency. Parameters k1, b and k3 will be set at default value -1.5, 0.5 and 0 respectively.

2.3.2Evaluation for Sentence Retrieval

In the evaluation process, a retrieved paragraph with the same paragraph id in the development dataset will be regarded as relevant paragraph retrieval. Similarly, a retrieved sentence with the answer provided in development dataset will be considered as relevant retrieval for each question.

	F1-score
Paragraph	0.7754
Sentence	0.3776

Table: F1-score for paragraph and sentence level retrieval

As shown in the table above, retrieval method worked well for a paragraph, but poorly on the sentence level. It can be explained by the fact that similar terms occurred in multiple sentences within the same paragraph. This makes multiple sentences with similar probability to be relevant retrieval.

2.4 Answer Processing.

In this phase, QA systems will extract a specific answer from a retrieved sentence. Basic QA is simply developed by answer type extraction, while enhanced QA is extended by sentence syntactic dependency parser.

2.4.1 Basic QA and Answer Type Extraction

With information about question’s answer type and relevant sentence, an answer can be easily extracted from a relevant sentence by searching entities with required answer type (Done by NER parser). However, the system can not extract right answer for questions with type “OTHER”, which is out of pre-defined categories.

2.4.2 Enhanced QA and Dependency Parser

Questions with answer type “OTHER” is handled by dependency parser in enhanced QA. First, each sentence is being divided into different noun chunks. How sentence tokens depend on each other can be parsed by spaCy’s dependency parser. If a token A from a sentence depends on a token B that is same as the one appeared in the question. Then the token A is highly likely to be the answer if it is not a token in the question. For example,

Question: “In addition to wool, what is uranium salt a mordant of?”

Sentence: “Uranium salts are mordants of silk or wool”

Actual Answer: “silk”

Available noun chunks for the sentence are “Uranium salts”, “mordants”, “silk” and “wool”. But only “silk” does not appear on the question. Besides, “silk” depends on the token “of”, which existed in the question. Therefore, the QA system will pick “silk” as the predicted answer, which is same as the actual answer.

2.4.3 Evaluation for Basic QA and Enhanced QA

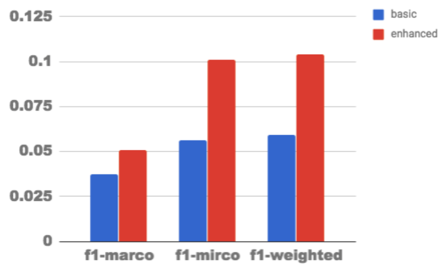


Chart graph: performance of basic QA and enhanced QA for development dataset

As shown in the chart, the accuracy of QA system improved a lot by extending the basic QA system with dependency parser. According to the training dataset, most answer type for questions is out of the pre-defined categories. How to improve the accuracy of “OTHER” type of question will be an important part to improve the overall QA system.

3. Conclusion and Further Improvements

Four phases of the QA system and techniques involved in them are explained in this paper. By reviewing the performance of each phase, sentence retrieval from a paragraph, and answer extraction for “OTHER” type question are weaknesses of the designed QA system. It might be better to extract answers from paragraph level rather than sentence level. Meanwhile, the deep learning algorithm can be applied to enable the system to learn how dependency pattern in the sentence match to the answers in the Answer processing phase.

Reference

Ali, J., Khan, R., Ahmad, N. and Maqsood, I. (2012). Random Forests and Decision Trees. *International Journal of Computer Science (IJCSI)*, 9(5), pp.272-278.

Anon, (2018). *spaCy Usage Documentation*. [online] Available at: https://spacy.io/usage/ [Accessed 21 May 2018].

Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc.

Mansouri, A., Affendey, L. and Mamat, A. (2008). Named Entity Recognition Approaches. *International Journal of Computer Science and Network Security (IJCSNS)*, 8(2), pp.339-344.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, pp.2825–2830.