Answering Single-Event Questions on Long-term News Article Archives

Wang Jiexin†, Adam JATOWT†, Michael Färber†, Masatoshi YOSHIKAWA†

†Graduate School of Informatics, Kyoto University

Yoshida-honmachi, Sakyo-ku, Kyoto, 606–8501, Japan

E-mail: †wangjiexin.scut@yahoo.com, ††adam@dl.kuis.kyoto-u.ac.jp, †††michael.faerber@cs.uni-freiburg.de, ††††yoshikawa@i.kyoto-u.ac.jp

Abstract With the development of digital preservation techniques, more and more news articles have been digitized. News archives contain immense value, helping people to know the detailed information of events that occurred at specific time points. Question Answering systems can help to utilize these documents effectively and satisfy users' information needs, which usually are represented as natural language questions that require returning of precise answers. This article presents a QAA (*Question Answering in Archives*) system, a question-answering system designed specifically for answering news events based questions like answering the number of participating nations in Olympic Games at a specific year and so on. In particular, the system uses New York Times news corpus composed of articles published between 1987 and 2007 as an underlying dataset and can both generate an answer sentence and provide its supporting sentences.

Keyword Document Archive, Question Answering System, Natural Language Processing, Deep Learning

1. Introduction

Nowadays, the amount and aggregated size of available digital news archives is increasingly growing as numerous news articles from newspapers or other media continue to be added. News articles belong to documents of strongly temporal character as they usually have publication date information which determines their timeliness, and on average constitute fairly reliable, accurate and time-aligned information sources. News digital archives offer immense value to our society, helping users to learn detailed information of an event that occurred at a specific time point in the past. Some professions, like historians, sociologists, or journalists need to deal with these temporal documents a lot.

To satisfy the information need of a user querying document archive such as news article archive, which is often represented as a natural language question, researchers typically use document retrieval systems like search engines that return top-n relevant documents, which might contain the answer of user's question. Yet, this style of output puts unnecessary burden on users who need to read or scan each retrieved documents to obtain the answer, which is extremely inconvenient.

Question-answering systems can help to solve this problem, as the objective of a QA system is to retrieve the most correct answer for the user question. The answer needs to be precise and concise, rather than a list of documents, so that the users don't need to locate the answer by themselves. For example, for a question: "How many countries attend the Summer Olympics in 1996?", the QA system is expected to return a number like "197".

There are usually three main modules in traditional QA systems, which are Query Analysis Module, Document Retrieval Module, and Answer Extraction Module. Firstly, the user's question will be analyzed to determine keywords, question type, syntactic information, etc. Secondly, the system will use keywords to search candidate documents. Finally, it will use the information from the query analysis part, and will process the documents to extract ranked candidate answers. The basic architecture of such a system is depicted in Figure 1. We also utilize these components in our proposal.

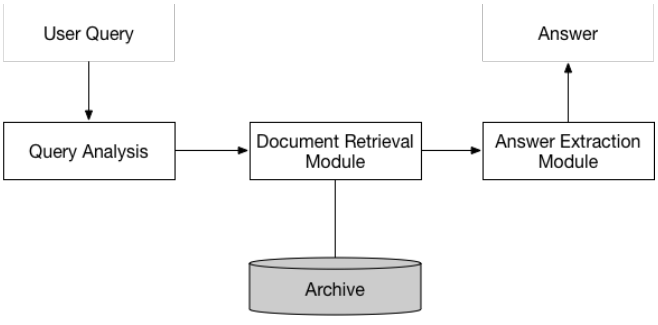

Figure 1. The architecture of traditional question answering

This paper presents QAA (Question Answering in Archives) system for news article archives. In the experiments we use New York Times Annotated News corpus which contains articles published between 1987 and 2007 as underlying dataset. Apache Solr engine is implemented in order to index documents. Our model also has the above three main modules but is implemented using diverse techniques. We improve some necessary functions in each module like question classification in Query Analysis Module by using Neural Networks on a large dataset that was generated by us, and add some functions like answer sentence generation function that can generate a natural language answer. We also add some functions based on the characteristics of news archives like time-scoping procedure that can analyze the time of news events (if any are mentioned) in user question, so the retrieved documents can be more accurate.

2. Related Work

Question answering has been studied for quite a long time, and many question-answering evaluation campaigns have been held up to date, such as TREC [1], CLEF [2], NTCIR [3] etc. Naturally, multiple question answering systems have been designed so far. One of the most famous question answering systems was called BASEBALL that can answer questions about baseball games played over a period of one year [4]. More specifically, Moriceau et al. [5] presented FIDJI, a question-answering (QA) system whose main goal is to validate answers by checking that all the information given in the question is retrieved in the supporting texts. FIDJI combines syntactic information with traditional QA techniques and does not require any pre-processing other than classical search engine indexing. Nanda et al. [6] built a WebBasedQA by parsing input, tokenizes and extracts features using previously calculated results and also using Naive Bayes as a classifier combined with known data store. Sun et al. [7] use syntactic dependency analysis for query expansion, and then use semantic relation analysis, based on the frame-based semantic representation generated by using a shallow semantic parser, for semantic answer extraction. Therefore, there are many different techniques of building traditional question answering systems, such as analyzing syntactic information like named entity recognition and semantic information [8,9], translating the natural language question to a structured query language like SQL, and also translating the of textual sentences into database that can be retrieved by the structured query [10], analyzing discourse relationships and textual [11]. In recent years, because the generation of large dataset, such as SQuAD [12], MS-MARCO [13], that provide the possibilities to build neural network models. Seo et al. [14] proposed BiDAF model (Bi-Directional Attention Flow) model, which represents the context at different levels of granularity and uses bidirectional attention flow mechanism to obtain a query-aware context representation, and finally get the answer by matching with the query. Microsoft Asia Natural Language Computing Group [15] proposed R-Net model, that get the representation by not only question passage matching but also self-passage matching to refine passage representation with information from the whole passage. Recently because of the strong power of Bert model, which is new language representation model that built by Google AI Language [16], most models proposed combine with it. Chen et al. [17] built a DrQA system, which adds a document retrieval module, and the performance drops dramatically when given Wikipedia to read, not a single paragraph. Therefore, the problem of neural network models is that the dataset they use provides both the answer and corresponding passage, rather than the whole corpus, so when given the whole corpus, the model cannot handle multiple documents effectively and the performance will drop dramatically. In contrast to the above-mentioned works, our QAA system is based on a traditional architecture of QA systems, and the answer extraction module of our system mainly uses traditional methods that can extract the evidence from each single document and rank the candidate answers in statistics.

3. QAA System

QAA system that we propose is a traditional question answering system though it operates on news article archive. It also has three main modules as the other traditional question answering systems, however we extend it with more functions and also we split the Answer Extraction Module into three sub-modules: *Document Processing*, *Sentence Selection* and *Answer Extraction*. The architecture of QAA system is shown in Figure 2. In the next parts of the paper, we will use a running example question "How many people were killed in Oklahoma City bombing in 1995?" to better demonstrate our QAA system.

3.1 Question Analysis

Question Analysis Module is the first module of QAA system, which is the basis of the latter modules. In QAA system, the Question Analysis Module helps to determine the expected answer type, generate declarative answer sentence with assumed answer content, identify named

entity information, extract predicate-argument structure of the declarative answer sentence, and detect event time scope in the question if mentioned in the question, and finally get keywords and their synonyms. Table 1 demonstrates the information that can be obtained by QAA system after question analysis step of the question "How many people were killed in Oklahoma City bombing in 1995?".

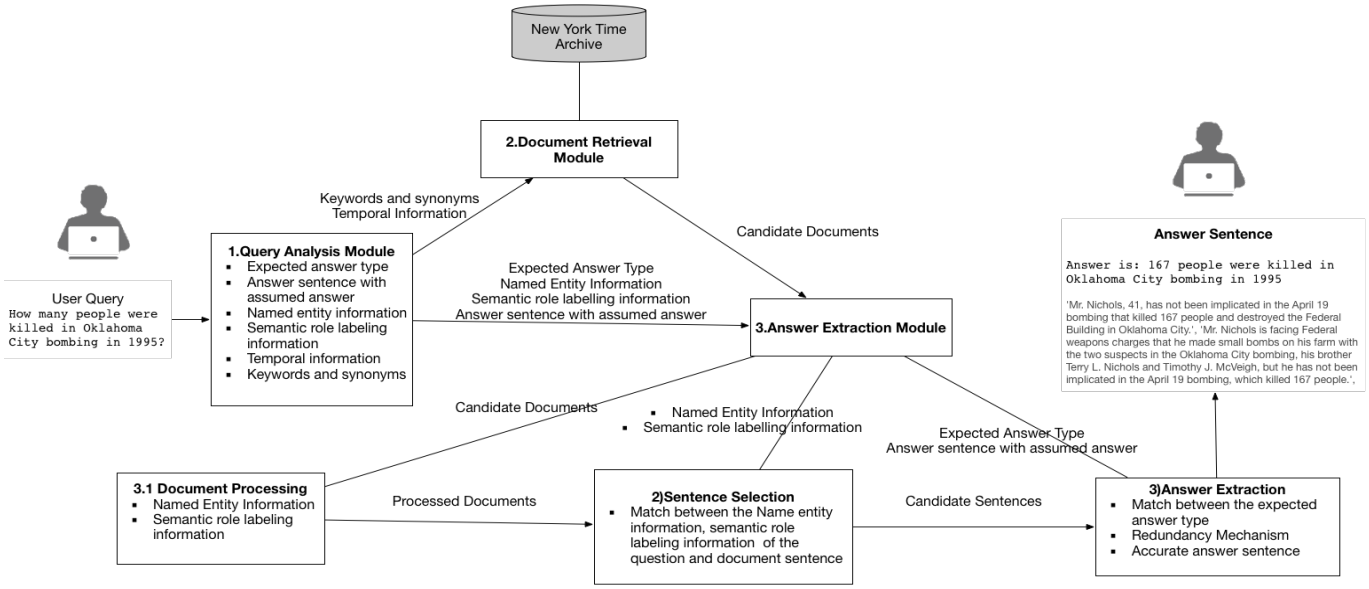

Figure 2. The architecture of QAA System

Question	How many people were killed in Oklahoma City bombing in 1995?
Expected answer type	CARDINAL
Declarative answer sentence with assumed answer content	50 people were killed in Oklahoma City bombing in 1995
Named entity information	{'DATE': [1995], 'GPE': [Oklahoma City]
Predicate-argument structure of declarative answer sentence	ARG0: 50 people Predicate: killed ARG1: in Oklahoma City bombing
Time scope of the event	{'begin': '1995-01-01', 'end': '1995-12-31'}
Keywords	['people', 'killed', 'Oklahoma City bombing']
Keywords synonyms	[['people', 'peoples', 'multitude', 'multitudes', 'masses', 'mass'], ['kill', 'kills', 'killing', 'killed', 'defeat', 'defeats', 'defeating', 'defeated'], ['Oklahoma City bombing']]

Table 1 Question Analysis of the question "How many people were killed in Oklahoma City bombing in 1995?"

3.1.1 Expected answer type identification

Firstly, the Question Analysis Module will determine expected answer type of the question. Expected answer type helps to determine the proper type of assumed answer content of the generated declarative answer sentence and validate the extracted answer in the Answer Extraction Module. However, current available expected answer type datasets are all too small or they do not contain enough expected answer types. The most popular one was released by Xin et al. [18] that contains nearly 6,000 labelled questions. This number however is still insufficient to train a good classifier. Therefore, we constructed our own dataset with common expected answer types by using the question-answer pairs from SQuAD [12] and MS-MARCO [13] datasets by applying the following steps: (1) transform the question into declarative answer sentence (the transformation method will be described later), (2) replace the interrogative part with the corresponding correct answer span, (3) use the named entity tool from spaCy library to detect the named entity type of the answer span in the answer sentence. The next step is to (4) correct some misclassified questions and remove records with uncommon or empty named entity types. Finally, (5) we can obtain 91,145 (question, expected answer type) pairs, that contain 10 common answer types (DATE, PERSON, CARDINAL, GPE, ORG, QUANTITY, TIME, MONEY, PERCENT, NORP, LOC). An example record is ("Which theater company was founded by several Northwestern alumni in 1988? ","GPE"). Then, we use this dataset to train a BiLSTM model, that can reach about 82% accuracy.

3.1.2 Declarative answer sentence with assumed answer content generation

Secondly, Question Analysis Module will generate the declarative answer sentence with assumed answer content,

which not only helps to output a clear result, but also increases the accuracy of extracting predicate-argument structure, since semantic parsers are mainly trained on declarative sentences. The main idea of transforming a question to a declarative sentence is to analyze the Penn Treebank structure of the question, which can be obtained by using Stanford Parser. Firstly, the whole question will be labeled as SBARQ in Penn Treebank structure and its two components: wh-phrase will be labeled as WHADJP, WHAVP, WHNP or WHPP, and main clause will be labeled as SQ. Secondly, the SQ clause will be readjusted using different rules that are based on internal Penn Treebank tags of the SQ clause to make each word locate in a correct order in the declarative sentence. For example, for the SQ clause "were killed in Oklahoma City bombing in 1995" in our example question, the SQ clause can be directly returned because the first part of SQ is a verb and the second part is a verb phrase. In addition, for the SQ clause "was Christopher Columbus born" in the question "Where was Christopher Columbus born ?", whose internal structure is comprised of verb, noun phrase and verb phrase, it needs to be readjusted to "Christopher Columbus was born" by changing the position of verb and noun phrase. Then the non-interrogative words of wh-phrase will be combined with a default text with the same expected answer type that detected previously to form a proper answer span. For example, "50" is the default text of CARDINAL answer type, and "New York" is the default text of GPE answer type, etc. Finally, the proper answer span will be inserted to the modified clause and form the declarative answer sentence with assumed answer content.

3.1.3 Named entity information and predicate-argument structure extraction

Thirdly, Question Analysis Module will identify the named entities of the question by named entity tool from spaCy library. Next, the predicate-argument structure of the declarative answer sentence with assumed answer content will be extracted, that is the task of semantic role labeling. We use a deep BiLSTM model built by He et al. [19] to do semantic role labeling, whose performance is close to the state of the art. As mentioned previously, the semantic parsing of declarative sentences can get better result than the semantic parsing of questions because the semantic parsers, like semantic role labeling models are mainly trained on declarative sentences rather than questions. The named entity information as well as predicate-argument information will be used to identify and rank the candidate sentences in Answer Extraction Module.

3.1.4 Time scope detection

Next, the Question Analysis Module will detect event time scope in the question to identify the possible temporal information, whose result contains a begin date and an end date, and we only use the news articles that were published within this period, since news generally contains the most detailed information about events happened near its publication date. Therefore, it can help to narrow and locate the relevant detailed news documents of a specific event. Moreover, the publication date of New York Times corpus that we used as knowledge source is between 1987 and 2007, so the easiest 'begin' is '1987-01-01' and latest 'end' is '2007-12-31'. For example, for a question like "What was the number of assassinations and attempts to assassinate in the U.S. since 1865? ", the result of event time scope detection is {'begin': '1865-01-01', 'end': '2007-12-31'}, and only the documents published within this particular period will be used. Specifically, the temporal expression phrase that is labeled as "DATE" or "TIME" in named entity recognition will be extracted firstly, then Stanford Temporal Tagger will be used to parse the phrase, and the rules based on preposition information of the phrase (e.g., "before 1999" has the latent 'begin' information: '1987-01-01') will also be combined to get the result. If the question has multiple temporal expression phrases, we will use the date that is the earliest as the 'begin' value, and the date that is the latest as the 'end' value. Moreover, for some expressions that cannot be parsed well by Stanford Temporal Tagger, like the 20th-century, between 1999 and 2000, etc., we obtain the event time scope by applying different processing ways for different expressions.

3.1.5 Keywords extraction and synonyms selection

Finally, Question Analysis Module will extract keywords, that are single-token nouns, compound nouns, verbs and adjectives describing the nouns in the question, which can be realized by analyzing the POS tags and dependency information of the words. Then, their synonyms that have most frequent senses will be selected by using WordNet, and by further filtering by choosing the synonym whose POS type match the original word in the sentence and the Glove word embedding vector similarity is over 0.5.

3.2 Document Retrieval

As mentioned before, the New York Times news corpus has been indexed by a search engine Solr, which can easily

index underlying archival documents. We use the keywords and their synonyms to generate the query and also use the time scope information of the event if mentioned in the question to return relevant document whose publication date is within this period, otherwise the period is unbounded. Moreover, we only use the top 100 relevant news documents for a question. Note that time scope information will not be used in the Answer Extraction Module to match the named entity information or predicate-argument structure because it has already been used here, corresponding to the publication time of event news.

3.3 Answer Extraction

Answer Extraction Module uses the information from the question analysis, processes the candidate documents and then matches the information between question and document sentences to extract candidate answers.

Firstly, the Answer Extraction Module will process the candidate documents. It firstly concatenates all the retrieved relevant documents to a single large document and then separates them into a list of sentences. Then it will extract predicate-argument structure of each sentence by performing semantic role labelling and remove the structures whose predicates are not contained in the synonym of the predicate of the declarative answer sentence with assumed answer content. For example, one sentence of a candidate sentence list is: "Mr. Nichols, 41, has not been implicated in the April 19 bombing that killed 167 people and destroyed the Federal Building in Oklahoma City. ". Although, this sentence has 5 predicates, we only extract the predicate "killed" and its ARG0 and ARG1, which are "the April 19 bombing" and "167 people". If there would not be any sentence that contain the predicate, we would extract predicate-argument structure of top 100 sentences that has most common words with the generated answer sentence. After obtaining the predicate-argument structure by using semantic role labelling, we then extract the named entity information of the sentence, and only store the named entity information whose types are also in the named entity information that we extract in Question Analysis Module. If the temporal information of the question, that is labeled as DATE or TIME in named entity recognition and also used as time scope information to locate relevant documents, this type of named entity information of the sentence will not be extracted. For example, in the above sentence, the extracted named entity information is "Oklahoma City", and "April 19" is not extracted.

Secondly, Answer Extraction Module will calculate the similarity of the semantic information and named entity information. We use a_s to denote the question answer sentence with assumed answer, and use c_s to denote the candidate sentence, and we use Pre to denote predicate, Arg to denote argument, and W to denote the word list of a sentence, ENT to denote the named entity information. To calculate the semantic similarity, firstly the predicate similarity is calculated by the vector similarity:

$$Sim_{Pre}(Pre_{a\_s}, Pre_{c\_s}) = Vec\_Sim(Pre_{a\_s}, Pre_{c\_s})$$

Secondly, we use jaccard similarity coefficient to calculate the similarities between two arguments:

$$Sim_{Arg}(ARGX, ARGY) = \frac{|W_{ARGX} \cap W_{ARGY}|}{|W_{ARGX} \cup W_{ARGY}|}$$

Because there are two pairs of ARG0, ARG1 in two sentences. So we define the argument similarities of a_s, c_s as following by using the above function:

$$Sim_{Arg}(Arg_{a_s}, Arg_{c_s}) = Sim_{Arg}(ARG0_{a\_s}, ARG0_{c\_s}) * Sim_{Arg}(ARG1_{a\_s}, ARG1_{c\_s})$$

Then, when calculating the similarity of the named entity information, we also use Jaccard similarity coefficient to calculate the similarity, that we denote as $Sim_{ENT}(ENT_{a\_s}, ENT_{c\_s})$. Next, we calculate the similarity of a_s, c_s as following:

$$Sim(a_s, c_s) = \alpha\left(Sim_{Pre}(Pre_{a_s}, Pre_{c_s}) * Sim_{Arg}(Arg_{a_s}, Arg_{c_s})\right)$$
$$+(1-\alpha)(Sim_{ENT}(ENT_{a\_s}, ENT_{c\_s}))$$

We set $\alpha = 0.7$, and we only extract the top 20 sentences.

Finally, we extract candidate answers by matching the argument that contains the assumed answer with the corresponding argument from these sentences. Then we validate the answer by expected answer type that is obtained in the first module, and rank the answers by their frequency and the similarity scores of their sentences. Finally, answer sentence can be generated by replacing the assumed answer with correct answer.

4. Experiments

This section first talks about the generation of the test set, and then presents evaluations of our QAA system.

To test the performance of our system, we built a test set with question-answer pairs that are selected from SQuAD [12]. We manually check the question-answer pairs and select the pairs whose questions can be answered by the New York Times. In addition, we divide the question-answer pairs into two categories: temporal question-answer pairs are the pairs whose questions do not contain temporal expressions, temporal question-answer pairs are the pairs whose questions contain temporal expressions.

Table 2 presents the evaluation results of our QAA system. Our system achieves 16.8% exact match and 19.1% F1 score on the non-temporal questions, and achieves 20.0% exact match and 22.3% F1 score on the temporal questions.

With the help of detecting time scope information of temporal questions, the document retrieval module can locate more relevant documents of the corresponding question. Therefore, the system works better on temporal questions. After analyzing the questions that cannot be answered correctly by our system, we found that most of them is due to the non-existence of the matching predicate-argument in the retrieved documents, which makes the system hard to find a good answer candidate. So it is necessary to use other patterns or other syntactic logic forms to compute the matching score when the system cannot determine good predicate-argument structures of the documents sentences. But unlike other end-to-end neural network models that do not have the ability to aggregate the answer from relevant documents, our system can compare the evidence from each relevant document and then return a final answer.

	Exact Match	F1 Score
Non-Temporal Question	16.8	19.1
Temporal Question	20.0	22.3

Table 2. Evaluation Results of QAA system

5. Conclusions & Future Work

In this paper, we present QAA system that is based on a traditional architecture of QA systems yet it focuses on long-term document archives. Our system combines many techniques, like semantic role labelling, deep learning method and other NLP techniques. It can help certain users of professions, like historians, sociologists, and journalists to better analyze the news events recorded in news archives to to extract useful desired information. Still the proposed system does not work well in some cases, for example, the performance of semantic role labelling is not high when dealing with long or complex sentences.

In future work, we will build a larger evaluation dataset for more different kinds of events and also try to combine some advanced techniques like Bert model to do the comprehension on multiple documents.

References

[1] E.M. Vorhees, The TREC question answering track, Nat. Lang. Eng. 7 (2001) 361–378.

[2] A. Vallin, B. Magnini, D. Giampiccolo, L. Aunimo, C. Ayache, P. Osenova, A. Penas, M. de Rijke, B. Sacaleanu, D. Santos, R. Sutcliffe, Overview of the CLEF 2005 multilingual question answering track, in: Proceedings of the Third Cross Language Evaluation Forum (CLEF 2005), 2005.

[3] N. Kando, Overview of the fifth NTCIR workshop, in: Proceedings of the Fifth NTCIR Workshop Meeting Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-Lingual Information Access, 2005.

[4] Green BF, Wolf AK, Chomsky C, and Laughery K. Baseball: An automatic question answerer. In Proceedings of Western Computing Conference, Vol. 19, 1961, pp. 219–224.

[5] Veronique Moriceau and Xavier Tannier. 2010. FIDJI: Using syntax for validating answers in multiple ´ documents. Inf. Retr. 13, 5 (October 2010), 507–533. DOI:http://dx.doi.org/10.1007/s10791-010-9131-y

[6] Garima Nanda, Mohit Dua, Krishma Singla, "A Hindi Question Answering System using Machine Learning Approach", ICCTICT, 2016.

[7] R. X. Sun, J. J. Jiang, Y. F. Tan, H. Cui, T. S. Chua, M. Y. Kan. 2005. Using syntactic and semantic relation analysis in question answering. In Proceedings of the TREC.

[8] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Deepak Ravichandran, Chin-Yew Lin, Deepak Ravich, Toward semantics-based answer pinpointing, in:Proceedings of the First International Conference on Human Language Technology Research, ACL, Stroudsburg, PA, USA, 2001, pp. 1–7.9.

[9] Enrique Alfonseca, Marco De Boni, Jos-Luis Jara-Valencia, Suresh Manandhar, A prototype question answering system using syntactic and semantic information for answer retrieval, in: Proceedings of the 10th Text Retrieval Conference (TREC-10), 2001, pp. 680–686.

[10] Anette Frank, Hans-Ulrich Krieger, Feiyu Xu, Hans Uszkoreit, Berthold Crysmann, Brigitte Jörg, Ulrich Schäfer, Question answering from structured knowledge sources, Journal of Applied Logic 5 (1) (2007) 20–48.

[11] Sanda Harabagiu, Andrew Hickl, Methods for using textual entailment in open-domain question answering, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of

the Association for Computational Linguistics, ACL, Stroudsburg, PA, USA,2006, pp. 905–912.
[12] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. In Empirical Methods in Natural Language Processing (EMNLP), 2016.
[13] Nguyen, Tri, Rosenberg, Mir, Song, Xia, Gao, Jianfeng, Tiwary, Saurabh, Majumder, Rangan, and Deng, Li. Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268, 2016.
[14] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 , 2016.
[15] Microsoft Asia Natural Language Computing Group. R-net: Machine reading comprehension with self-matching networks. 2017.
[16] Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.
[17] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. arXiv preprint arXiv:1704.00051, 2017.
[18] Xin Li, Dan Roth,Learning Question Classifiers. COLING'02, Aug., 2002.
[19] He, L., Lee, K., Lewis, M., and Zettlemoyer, L. (2017). Deep semantic role labeling: What works and what's next. In the Association for Computational Linguistics annual meeting (ACL).
[20] Pavlos Fafalios, Vaibhav Kasturia, Wolfgang Nejdl, "Towards a Ranking Model for Semantic Layers over Digital Archives", Digital Libraries (JCDL) 2017 ACM/IEEE Joint Conference on, pp. 1-2, 2017.
[21] Helge Holzmann, Wolfgang Nejdl, Avishek Anand, Exploring Web Archives Through Temporal Anchor Texts, Proceedings of the 2017 ACM on Web Science Conference, June 25-28, 2017, Troy, New York, USA