The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Assertion-based QA with Question-Aware Open Information Extraction

**Zhao Yan[§*], Duyu Tang[‡], Nan Duan[‡], Shujie Liu[‡], Wendi Wang[◇],
Daxin Jiang[◇], Ming Zhou[‡] and Zhoujun Li[§†]**

[§]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡]Microsoft Research, Beijing, China [◇]Microsoft, Beijing, China
{yanzhao, lizj}@buaa.edu.cn {dutang, nanduan, shujliu, wendw, djiang, mingzhou}@microsoft.com

Abstract

We present assertion based question answering (ABQA), an open domain question answering task that takes a question and a passage as inputs, and outputs a semi-structured assertion consisting of a subject, a predicate and a list of arguments. An assertion conveys more evidences than a short answer span in reading comprehension, and it is more concise than a tedious passage in passage-based QA. These advantages make ABQA more suitable for human-computer interaction scenarios such as voice-controlled speakers. Further progress towards improving ABQA requires richer supervised dataset and powerful models of text understanding. To remedy this, we introduce a new dataset called WebAssertions, which includes hand-annotated QA labels for 358,427 assertions in 55,960 web passages. To address ABQA, we develop both generative and extractive approaches. The backbone of our generative approach is sequence to sequence learning. In order to capture the structure of the output assertion, we introduce a hierarchical decoder that first generates the structure of the assertion and then generates the words of each field. The extractive approach is based on learning to rank. Features at different levels of granularity are designed to measure the semantic relevance between a question and an assertion. Experimental results show that our approaches have the ability to infer question-aware assertions from a passage. We further evaluate our approaches by incorporating the ABQA results as additional features in passage-based QA. Results on two datasets show that ABQA features significantly improve the accuracy on passage-based QA.

Introduction

Open-domain question answering (Open-QA) is a long-term goal in natural language processing area, which empowers the computer to answer questions for open domain. In this work, we present assertion-based question answering (ABQA), an open QA task that answers a question with semi-structured assertion instead of answer span in machine reading comprehension (Rajpurkar et al. 2016) or sentence/passage in answer selection (Yang, Yih, and Meek 2015). Here an assertion is a group of words with subject-predicate-object structure which is inferred from the passage guided by the content of the question. We believe that ABQA has many promising advantages. From an industry perspective, ABQA could improve smart speakers such as Amazon Echo, Google Home and Microsoft Invoke, where the scenario is to answer a user's question through reading out a concise and semantically adequate utterance. In this scenario, a short answer span does not convey enough supporting evidences, while a passage is too tedious for a speaker. From a research perspective, ABQA is a potential direction to drive explainable question answering. It explicitly reveals the knowledge embodied in the document that answers the question. Moreover, the results from ABQA could be used to improve other QA tasks such as answer sentence selection. An assertion graph could also be built on top of these assertions through aggregating the same nodes, which makes explicit reasoning practical (Khashabi et al. 2016; Khot, Sabharwal, and Clark 2017).

Question	who killed jfk
Method	**Answer**
PBQA	A ten-month investigation from November 1963 to September 1964 by the Warren Commission concluded that Kennedy was assassinated by Lee Harvey Oswald, acting alone, and that Jack Ruby also acted alone when he killed Oswald before he could stand trial.
MRC	Lee Harvey Oswald
ABQA	<Kennedy; was assassinated; by Lee Harvey Oswald>

Table 1: An example to illustrate the difference between three QA tasks, i.e. ABQA, MRC and PBQA.

The ABQA task is related to the answer sentence/passage selection (PBQA) task and the machine reading comprehension (MRC) task. The difference between ABQA and others is obvious although they all take question-passage pair as the input. As Table 1 shows, the assertion is organized into a structure with complete and concise information. The ABQA task differs from knowledge based QA (KBQA) in that the knowledge in KBQA is typically curated or extracted from large scale web documents. The goal of ABQA is deep document understanding and to answer question based on that. Various representations of a meaning makes directly linking the knowledge in KB to the document a challenging

[*]Contribution during internship at Microsoft Research Asia

[†]Corresponding author

problem. The ABQA task also relates to Open IE (OIE), the goal of which is to extract all the assertions involved in a document. The end goal of ABQA is not only to infer the assertions from both question and document, but also to correctly answer the question.

To study the ABQA task, we construct a human labeled dataset called WebAssertions. The questions and correspond passages are collected from the query log of a commercial search engine in order to reflect the real information need of users. For each question-passage pair, we generate assertion candidates by a state-of-the-art OIE algorithm (Del Corro and Gemulla 2013). Human annotators are asked to label whether or not an assertion is correct and concise and meantime can correctly answer the question. The WebAssertions dataset includes hand-annotated QA labels for 358,427 assertions in 55,960 web passages.

We introduce both generative and extractive approaches to address ABQA. Our generative approach which we call Seq2Ast is on the basis of sequence-to-sequence (Seq2Seq) learning. Seq2Ast extends Seq2Seq by incorporating a hierarchical decoder, which first generates the structure of an assertion through a tuple-level decoder, and then generates the words for each slot through another word-level decoder. The extractive method is based on learning to ranking, which ranks candidate assertions with well-designed matching features at different levels.

We conduct experiments on two settings. We first test the performances of our approaches on the ABQA task. Results show that Seq2Ast yields 35.76 in terms of BLEU-4 score, which is better that the Seq2Seq model. We further apply ABQA results as additional features to facilitate the passage-based QA task. Results on two datasets (Yang, Yih, and Meek 2015; Nguyen et al. 2016) show that incorporating ABQA features significantly improve the accuracy on passage-based QA.

In summary, we make the following contributions:

- We present the ABQA task, which answers a question with an assertion based on the content of a document. We create a manually labeled corpus for ABQA, which will be released to the community.
- We extend sequence-to-sequence learning approach by introducing a hierarchical decoder to generate the assertion. We also develop an extractive approach for ABQA.
- We conduct extensive experiments, and verify the effectiveness of our approach in both ABQA and PBQA tasks.

Task Definition and Dataset Construction

In this section, we formulate the task of assertion-based question answering (ABQA) and describe the construction of a new dataset tailored for ABQA.

Task Definition Given a natural language question (q) and a passage (p), the goal of ABQA is to output a semi-structured assertion (ast) that can answer the question q based on the content of the passage p. An assertion is represented as an n-tuple (n$\geq$3) which consists of a subject (sub), a predicate (pre), and one or more arguments (arg_i). Each field is a natural language sequence that includes one or more words.

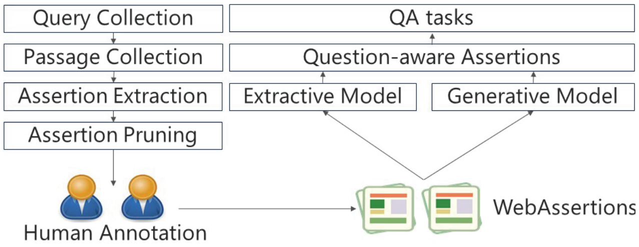

Figure 1: A brief illustration of the dataset construction, ABQA, and the application of ABQA in other QA tasks.

Steps	Details
Query Collection	Collect queries from the search log of a commercial search engine.
Passage Collection	Leverage a search engine, and collect pairs of query-passage if the passage is the direct answer of the query.
Assertion Extraction	Extract candidate assertions from passages based on an open IE algorithm.
Assertion Pruning	Prune assertions based on a combination rule in order to facilitate reasoning.
Human Annotation	Ask labelers to annotate if an assertion can correctly answer the question and meantime has a complete meaning.

Table 2: The details of the dataset construction process.

Dataset Construction Since there is no publicly available dataset for ABQA, we construct a dataset called WebAssertions through manual annotation. The construction of WebAssertions follows the steps described in Table 2.

Here we describe some important details during the data construction process. There exists several open IE algorithms in literature, including TextRunner (Yates et al. 2007), Reverb (Fader, Soderland, and Etzioni 2011), OLLIE (Schmitz et al. 2012). The result of an open IE algorithm has the same format with an assertion. We applied these open IE toolkits to a portion of randomly sampled passages from our corpus. We observe that the results extracted via ClausIE answer more questions than other algorithms. ClausIE is a rule based open IE algorithm which does not require any training data. The backbone of ClausIE is a set of predefined rules, which are based on the structures of sentences obtained via a dependency parsing tree. For more details about ClausIE, please refer to (Del Corro and Gemulla 2013).

We use a simple rule to enhance the assertions based on our consideration of facilitating reasoning. We believe that ABQA is a great way to drive explainable question answering and reasoning over documents. Different from the unexplainable deep neural network approaches in query-passage matching tasks, the structured assertions reveal which portion of knowledge embodied in the document answers the questions. Keeping these in mind, in this work we made a

Question		when will shanghai disney open
Passage		the Disney empire's latest outpost, Shanghai Disneyland, will open in late 2015, reports the associated press.
No.	Label	Assertion
1	0	<the Disney empire's latest outpost; is; Shanghai Disneyland >
2	0	<the Disney empire's latest outpost; will open; in late 2015>
3	0	<the associated press; reports; the Disney empire's latest outpost will open in late 2015>
4	1	<Shanghai Disneyland; will open; in late 2015 >

Table 3: A data sample from WebAssertions. The 4^{th} assertion is combined from 1^{st} and 2^{nd} assertion.

preliminary trial to compose new assertions based on the extracted assertions from a document. We consider the "is-a" relation and use that to do an extension. Supposing two assertions <A, is, B> and <A, pre, C> are extracted, we will generate an new assertion <B, pre, C>. An example is given in Table 3, in which the 4^{th} assertion is composed based on the 1^{st} and the 2^{nd} assertions. Table 3 gives an example of the human annotation result. Data statistics of WebAssertions are given in Table 4.

# of question-passage	55,960
# of question-assertion	358,427
Avg. assertions / question	6.41
Avg. Words / question	6.00
Avg. Words / passage	39.33
Avg. Words / assertion	8.62

Table 4: Statistics of the WebAssertions.

Assertion based Question Answering (ABQA)

In this section, we describe a generative approach and an extractive approach for ABQA.

Seq2Ast: The Generative Approach for ABQA

We develop a sequence-to-assertion (Seq2Ast) approach to generate assertions for ABQA. The backbone of Seq2Ast is sequence-to-sequence (Seq2Seq) learning (Sutskever, Vinyals, and Le 2014; Cho et al. 2014), which has achieved promising performances in a variety of natural language generation tasks. The Seq2Seq approach includes an encoder and a decoder. The encoder takes a sequence as the input and maps the inputs to a list of hidden vectors. The decoder generates another sequence in a sequential way through outputting a word at one time step.

The main characteristic of ABQA task is that the output in ABQA is an assertion, which is composed of a list of fields and each field consists of a list of words. To address this, we present a hierarchical decoder which first generates each field of the assertion through a **tuple-level** decoder, and then generates the words for each field through a **word-level** decoder. In Seq2Ast, the tuple-level decoder memorizes the structures of the assertion and the word-level decoder learns the short dependencies in each field.

Specifically, we use GRU based RNN (Cho et al. 2014) as the tuple-level decoder to output the representation for each field of the assertion. On top of the tuple-based decoder, we use another GRU based RNN as word-level decoder to generate the words of each field. The architecture of Seq2Ast is given in Figure 2, which is inspired by the chunk-based NMT (Ishiwatari et al. 2017). To generate the representation of the k-th field $\boldsymbol{s}_k^{(t)}$, the tuple-level decoder takes the last state of the word-level decoder $\boldsymbol{s}_{k-1,J_{k-1}}^{(w)}$ and updates its hidden state $\boldsymbol{s}_k^{(t)}$ as follows:

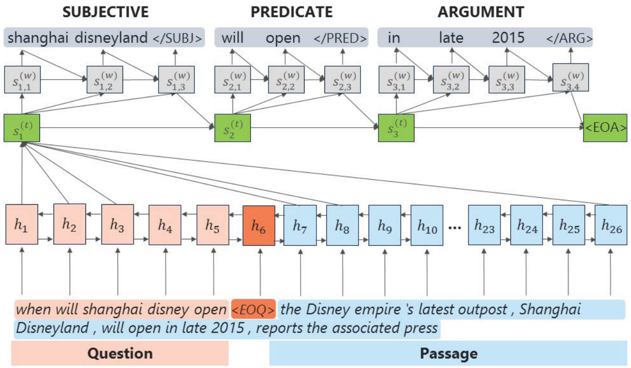
SUBJECTIVE
PREDICATE
ARGUMENT
shanghai disneyland </SUBJ>
will open </PRED>
in late 2015 </ARG>
$s_{1,1}^{(w)}$ $s_{1,2}^{(w)}$ $s_{1,3}^{(w)}$ $s_{2,1}^{(w)}$ $s_{2,2}^{(w)}$ $s_{2,3}^{(w)}$ $s_{3,1}^{(w)}$ $s_{3,2}^{(w)}$ $s_{3,3}^{(w)}$ $s_{3,4}^{(w)}$
$s_1^{(t)}$ $s_2^{(t)}$ $s_3^{(t)}$
<EOA>
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9 h_{10} ... h_{23} h_{24} h_{25} h_{26}
when will shanghai disney open <EOQ> the Disney empire 's latest outpost , Shanghai Disneyland , will open in late 2015 , reports the associated press
Question
Passage

Figure 2: The architecture of Seq2Ast with a hierarchical decoder.

$$\boldsymbol{s}_k^{(t)} = \mathrm{GRU}(\boldsymbol{s}_{k-1}^{(t)}, \boldsymbol{s}_{k-1,J_{k-1}}^{(w)}) \tag{1}$$

We consider the field representation as a global information to guide the prediction of each word, therefore the field representation $\boldsymbol{s}_k^{(t)}$ is also fed to the word-level decoder as additional input until it outputs all the words in the current field. Attention (Bahdanau, Cho, and Bengio 2014) is used in the word level decoder in order to selectively retrieve important content from the encoder part. To deal with the rare word problem, we use a simple yet effective copying mechanism which replaces the generated out-of-vocabulary word with a word of largest attention score from the source.

$$\boldsymbol{s}_{k,j}^{(w)} = \mathrm{GRU}(\boldsymbol{s}_{k,j-1}^{(w)}, [\boldsymbol{s}_k^{(t)}; \boldsymbol{y}_{k,j-1}]) \tag{2}$$

We use bidirectional GRU based RNN as the encoder. In this work, we concatenate the passage and the question, separated by a special tag.[1] The model is learned in an end-to-end way with back-propagation, the objective of which is to maximize the probability of the correct assertion given a question-passage pair. In the experiment, the parameters in Seq2Ast are randomly initialized, and updated with AdeDelta (Zeiler 2012).

[1] An alternative way is to regard the document as the memory (Sukhbaatar et al. 2015) and use the question to iteratively retrieve from and update the memory. In this paper, we favor to the simple concatenation strategy.

ExtAst: The Extractive method for ABQA

The extractive method is a learning to rank based method which selects top-ranked assertion from a candidate list based on features designed in different granularities. Our extractive method includes three steps: i) assertion candidate generation which has been described in the dataset construction process; ii) question-aware matching features; iii) assertion candidate ranking.

Question-aware Matching Features We design features at three different levels of granularities to measure the semantic relevance between a question (q) and an assertion (ast).

In **Word-Level**, we use a word matching feature f_{WM} and a word level translation feature f_{W2W}. The intuition of f_{WM} is that an assertion is relevant to a question if they have a large amount of word overlap. The f_{WM} are calculated based on number of words shared by question and assertion. The f_{W2W} denotes a word-to-word translation-based feature that calculates the relevance for a question and an assertion based on IBM model 1 (Brown et al. 1993). The probabilities of word alignments are trained on 11.6M "sentence-similar sentence" pairs by GIZA++ (Och and Ney 2003).

In **Phrase-Level**, we design a paraphrase-based feature f_{PP} and a phrase-to-phrase translation feature f_{P2P} to deal with the case that a question and an assertion use different expressions to describe the same meaning. Both f_{PP} and f_{P2P} are based on extracted phrase tables (PT) by existing statistical machine translation method (Koehn, Och, and Marcu 2003). The difference between f_{PP} and f_{P2P} is that the PT of f_{PP} is extracted from 0.5M "English-Chinese" bilingual pairs while the PT of f_{P2P} is extracted from 4M "question-answer" pairs.

In **Sentence-Level**, we use a CNN-based feature f_{CNN} and a RNN-based feature f_{RNN} to match a question to an assertion. The f_{CNN} features is based on the CDSSM model (Shen et al. 2014), a convolutional neural network approach which has been successfully applied in sentence matching tasks. The model composes question vector and assertion vector via two convolutional neural networks separately and calculate their relevance with cosine function.

$$f_{CNN}(que, ast) = cosine(cdssm_1(q), cdssm_2(ast)) \quad (3)$$

We also use a recurrent neural network based model to calculate f_{RNN}. We first use two RNNs to map a question and an assertion to fixed-length vectors separately. The same bi-directional GRU is used to get the question representation and assertion representation from both directions. Taking question representation as an example, it recursively transforms current word vector $e_{q,t}$ with the output vector of the previous step $h_{q,t-1}$. In the representation layer, we concatenate the four last hidden states and the element-wise multiplication of the vectors from both directions as the final representation. Afterwards, we feed the representation of question-assertion pair to a multilayer perceptron (MLP).

We train model parameters of f_{CNN} and f_{RNN} on 4M "question-answer" pairs with stochastic gradient descent. The pair-wise margin ranking loss for each training instance is calculated as:

$$\mathcal{L} = \max(0, m - f^+(q, ast) + f^-(q, ast)), \quad (4)$$

where $f^+(q, ast)$ and $f^-(q, ast)$ are the model scores for a relevance and irrelevance pair and m is the margin.

Assertion Candidate Ranking We use LambdaMART (Burges 2010), an algorithm for solving real world ranking problem, to learn the final ranking score of each question-assertion pair.[2] The basic idea of LambdaMART is that it constructs a forest of decision trees, and its output is a linear combination of the results of decision trees. Each branch in a decision tree specifies a threshold to apply to a single feature, and each leaf node is a real value. Specifically, for a forest of N trees, the relevance score of a question-assertion pair is calculated as

$$s(q, ast) = \sum_{i=1}^{N} w_i tr_i(q, ast), \quad (5)$$

where w_i is the weight associated with the i-th regression tree, and $tr_i(\cdot)$ is the value of a leaf node obtained by evaluating i-th tree with features $[f_1(q, ast), ..., f_K(q, ast)]$. The values of w_i and the parameters in $tr_i(\cdot)$ are learned with gradient descent during training.

Experiment

In this section, we describe experimental settings and report empirical results on ABQA and the application of ABQA in the answer sentence selection task.

Results on ABQA

We first test the generative and extractive approaches on the assertion-based question answering (ABQA) task. In this experiment, we randomly split the WebAssertions dataset into training, development, and test sets with a 80:10:10 split. Parameters are tuned on the development set and results are reported on the test set. The test set contains 36,165 question-passage-assertion triples from 5,575 question-passage pairs.

We first conduct evaluation from a text generation perspective. We use BLEU-4 score (Papineni et al. 2002) as the automatic evaluation metric, the goal of which is to measure the ngram match between the generated assertion and the referenced assertion. We compare to the standard Seq2Seq model w/ and w/o attention mechanism. Results are given in Table 6. We can see that Seq2Ast performs better than the standard Seq2Seq method, which verifies the effectiveness of the hierarchical decoder. As a reference, we can also report the BLEU-4 score of the extractive approach despite this is not a perfect to compare between the generative and extractive approaches. The BLUE-4 score of our extractive approach is 72.27, which is extremely high for a text generation task. But this is also reasonable because the extractive approach aims to select a most possible assertion from a candidate list which includes the referenced result. Therefore, the BLEU-4 score for a correct top-ranked result is 100.

[2] We also implemented a ranker with logistic regression, however, its performance was obviously worse than LambdaMART in our experiment.

Question		how much can your bladder hold
Passage		A healthy adult bladder can hold up to 16 ounces (2 cups) of urine comfortably, according to the national institutes of health. How frequently it fills depends on how much excess water your body is trying to get rid of.
Generative Result		<a healthy adult bladder; can hold; up to 16 ounces>
Extractive Result		
Rank	**Label**	**Assertion**
1	1	<a healthy adult bladder; can hold; up to 16 ounces; 2 cups of urine>
2	0	<a healthy adult bladder; can hold; up to 16 ounces; according to the national institutes of health>
3	0	<a healthy adult bladder; can hold; up to 16 ounces; comfortably>
4	0	<it; fills; how frequently>
5	0	<your body; is trying; to rid of; how much excess water>

Table 5: An example illustrating the results of the generative and the extractive approaches.

Further experiments that applying the results of both generative and extractive approaches in passage-level question answering task will be given in the following subsection.

Method	BLEU-4
Seq2Seq	22.01
Seq2Seq + attention	31.85
Seq2Ast	35.76

Table 6: Performance on generative based ABQA.

We evaluate our extractive method as a ranking problem, the goal of which is to rank the assertion candidates for a given question-passage pair and select the assertion that has the largest probability to correctly answer the question. Hence, we choose Precision@1 (P@1), Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) to evaluate the performance of our model (Manning et al. 2008).

We conduct an ablation test to study the effects of different features in the extractive approach. Results are given in Table 7. It is not surprising that sentence-level feature performs better than word-level and phrase-level features because of better modeling the global semantic relevance between a question and an assertion. Our system ExtAst that combines all the features obtains the best performance.

Methods	MAP	MRR	P@1
WordMatch	65.85%	66.67%	47.62%
Word-Level	71.13%	72.08%	55.47%
Phrase-Level	72.18%	72.86%	56.74%
Sentence-Level	76.49%	77.45%	63.34%
ExtAst	77.99%	78.90%	65.56%

Table 7: Performances on extractive based ABQA.

A sampled instance together with the results of our generative and extractive approaches are illustrated in Table 5. We can see that the generative model has the ability to produce the structure of an assertion, fluent expressions for each field of the assertion and a complete meaning to some extent. In this example, the generative result is even better than the extractive model in terms of concise. However, the generative model is far more perfect. After doing case studies, we find that fluency is not a big issue. The main issue of current approach includes generating duplicate content and generating an assertion that is irrelevant to the question. The first issue would be mitigated with coverage mechanism (Tu et al. 2016), which explicitly memorizes whether a source word has been replicated or not. Addressing the second issue might require exploring deep question understanding, and a decoder that is deeply driven by the question.

We also conduct error analysis on the extractive approach. We summarize the main errors into three categories. The first category is question type mismatch. For instance, the answer for "*When were the Mongols defeated by the Tran?*" is a reasonable assertion yet does not contain any time information. The second category is the mismatch between the entity in query and the different expressions in the passage. Co-reference resolution could be also grouped into this category. The third category is the require of reasoning. An exampled question is "*Which is the largest city not connected to an interstate highway?*". Our current model does not have the ability to handle the "not" type question.

Improve PBQA with ABQA results

We further evaluate the performance of our ABQA algorithms by applying the results into the passage-based question answering task (PBQA), and use the end-to-end performance in PBQA to reflect the effects of our approaches. In this work, we use answer selection as the PBQA tasks, which takes a question and a passage as the input, and outputs a sentence which comes from the passage as the answer.

Given a question and a document, we first use our ABQA algorithms to output the top-ranked assertion through generative or extractive approaches. Afterwards, additional features for a question-assertion pair is appended to the original feature vector which is used for answer sentence selection. We use exactly the same feature set which we have used in the extractive ABQA approach. The basis features for answer sentence selection include a word-level feature based on the number of occurred words in both question and passage, and a sentence-level feature that encodes both question and passage as continuous vector with convolutional neural network. We also employ LambdaMART to train the ranking model for answer sentence selection. Feature weights in the ranking model are trained by SGD based on the training

Methods		WikiQA		MARCO	
		MAP	**MRR**	**MAP**	**MRR**
Published Models					
(1)	CNN+Cnt (Yang, Yih, and Meek 2015)	65.20%	66.52%	-	-
(2)	LSTM+Att+Cnt (Miao, Yu, and Blunsom 2015)	68.55%	70.41%	-	-
(3)	ABCNN (Yin et al. 2016)	69.21%	71.08%	46.91%	47.67%
(4)	Dual-QA (Tang et al. 2017)	68.44%	70.02%	48.36%	49.11%
(5)	IARNN-Occam (Wang, Liu, and Zhao 2016)	73.41%	74.18%	-	-
(6)	conv-RNN (Wang, Jiang, and Yang 2017)	**74.27%**	75.04%	-	-
(7)	CNN+CH (Tymoshenko, Bonadiman, and Moschitti 2016)	73.69%	**75.88%**	-	-
Our Models					
(8)	Baseline	69.89%	71.33%	45.97%	46.62%
(9)	Baseline+RndAst	69.17%	70.12%	46.62%	47.27%
(10)	Baseline+MaxAst	71.82%	72.81%	49.37%	50.05%
(11)	Baseline+ExtAst	72.33%	73.52%	**50.07%**	**50.76%**
(12)	Baseline+Seq2Ast	72.26%	73.35%	47.44%	48.10%

Table 8: Evaluation of answer selection task on WikiQA and MARCO datasets.

data that consists of a set of labeled < question, sentence, label> triples, where $label$ indicates whether the sentence is the correct answer for the question or not.

Results are reported on WikiQA and MARCO datasets, both of which are suitable to test our ABQA approach as the questions from these dataset are also real user queries from the search engine, which is consistent with the WebAssertions dataset. WikiQA is a benchmark dataset for answer sentence selection and precisely constructed based on natural language questions and Wikipedia documents. WikiQA dataset contains 20,360 instances in the training set, 2,733 instances in development set, and 6,165 instances in the test set. The MARCO dataset is originally constructed for the reading comprehension task, yet also includes manual annotation for passage ranking. In MARCO dataset, questions come from Bing search log and passage candidates come from search engine's results. Annotators will label a passage as 1 if the passage contains evidences to answer the given question. Since the ground truth of the MARCO's test set is invisible to public, we randomly split the original validation set into dev set and test set. In this paper we only use the information about passage selection to test our model. The MARCO dataset contains 676,193 instances in the training set, 39,510 instances in the development set, and 42,850 instances in the test set. In this experiment, we also use MAP and MRR as the evaluation metrics. Similar to other published works, the calculation of the evaluation metric does not include the instances whose candidate answers are all incorrect or all correct .

We compare to different algorithms for PBQA. Results are given in Table 8. The results of baseline approaches on these two datasets are reported in previous publications. **CNN+Cnt** (Yang, Yih, and Meek 2015) combines a bigram CNN model with word count by logistic regression. **LSTM+Att+Cnt** (Miao, Yu, and Blunsom 2015) combines an attention-based LSTM model with word count by logistic regression. **ABCNN** (Yin et al. 2016) uses an attention-based CNN model which has been proven very powerful in various sentence matching tasks. **Dual-QA** (Tang et al. 2017) take QA and question generation (QG) as dual task. The result of ABCNN model on MARCO dataset is reported in (Tang et al. 2017). **IARNN-Occam** (Wang, Liu, and Zhao 2016) is a RNN model with inner attention mechanism. **conv-RNN** (Wang, Jiang, and Yang 2017) is an hybrid model that combines both CNN and RNN. **CNN+CH** (Tymoshenko, Bonadiman, and Moschitti 2016) is an hybrid model combined convolution tree kernel features with CNN. As described before, our baseline system contains a word-level feature based on word overlap and a sentence-level feature based on CDSSM (Shen et al. 2014).

We further compare to different usages of assertions for PBQA. Without using our question-aware assertion generation/extraction approach, we could also use open IE approaches to extract all the assertions from the passage, and then aggregate these assertions as additional features for PBQA. We implement two strategies towards this goal. The **RndAst** means that we randomly select an assertion and use it to calculate the additional assertion-level feature vector. The **MaxAst** is similar to the max-pooling operation in convolutional neural network. We first get the feature vectors for all the extracted assertions from a passage, and then select the max value in each dimension from a list of feature vectors. From the results, we can see that our approaches (especially ExtAst) significantly improves our baseline system.

Related Work

Our work relates to the fields of open information extraction, open knowledge-based QA, passage-based QA and machine reading comprehension.

The ABQA task is related to the Machine Reading Comprehension (MRC) (Rajpurkar et al. 2016) task in that both take question-passage pair as the input. The difference between ABQA and MRC is that the output of ABQA is an assertion which organized as a semi-structure with complete and concise information, while the output of MRC is a short answer span. The ABQA task also differs from passage based QA (PBQA) where the answer is a long passage. Our extractive method is related to existing works for PBQA. LCLR (Yih et al. 2013) applied rich lexical semantic features obtained from a wide range of linguistic resources including WordNet, polarity-inducing latent semantic analysis (PILSA) model and different vector space models. Convolutional neural networks (Yu et al. 2014; Severyn and Moschitti 2015) and recurrent neural networks (Wang and Nyberg 2015) are used to encode questions and answer passages into a semantic vector space. ABCNN (Yin et al. 2016) is an attention based CNN which first calculates an similarity matrix and takes it as a new channel of the CNN model. Recent studies (Duan et al. 2017; Tang et al. 2017) also explore question generation to improve question answering system.

Open IE works extract triples of format <subject, predicate, arguments> from text in natural language, and does not presuppose a predefined set of predicates. TextRunner (Yates et al. 2007) is a pioneering Open IE work which aims at constructing a general model that expresses a relation based on Part-of-Speech and Chunking features. ReVerb (Fader, Soderland, and Etzioni 2011) restricts the predicates to verbal phrases and extracts them based on grammatical structures. ClausIE (Del Corro and Gemulla 2013) employs handcrafted grammatical patterns based on the result of dependency parsing trees to detect and extract clause-based assertions. This work differs from Open IE in that the end goal of our work is not only to infer the assertions from both question and document, but also to correctly answer the question. In addition, our generative method has the ability to generate words that do not occur in the source text.

There are two lines of studies in knowledge-based question answering (KBQA). One focuses on answering natural language question with curated KB (Berant et al. 2013; Bao et al. 2014; Yih et al. 2015), where the key problem is how to link questions in natural language to the structured knowledge in KB. Another line of research in KBQA focuses on large-scale open KB which is automatically extracted from web corpora by means of open IE techniques. To address KBQA, Fader *et al.* (Fader, Zettlemoyer, and Etzioni 2013) present the first open KBQA system which learns question paraphrases over a large corpus. OQA (Fader, Zettlemoyer, and Etzioni 2014) is a system that processes questions using a cascaded pipeline on both curated and open KBs. TAQA (Yin et al. 2015) is an open KBQA system, which operates on n-tuple assertions in order to answer questions with complex semantic constraints. TUPLEINF (Khot, Sabharwal, and Clark 2017) answers complex questions by reasoning over Open IE knowledge with an integer linear programming (ILP) optimization model, and searches for the best subset of assertions. ABQA differs from KBQA in that the assertions/knowledge are extracted from the document, and the focus of ABQA is document understanding and answering question based on that. Our method also differs from KBQA works in that the knowledge in KBQA is typically curated or extracted from large scale web documents beforehand, while our goal is to infer knowledge based on the question and the document.

Conclusion

In this paper, we introduce assertion-based question answering (ABQA), an open-domain QA task that answers a question with a semi-structured assertion which is inferred (generated or extracted) from the content of a document. We construct a dataset called WebAssertions tailored for ABQA and develop both generative and extractive approaches. We conduct extensive experiments in various settings. Results show that our ABQA approaches have the ability to infer question-aware assertions from the document. We also demonstrate that incorporating ABQA results as additional features significantly improves the accuracy of a baseline system on passage-based QA. We plan to improve the question understanding component and the reasoning ability of the approach so that assertions across different sentences could be used to infer the final answer.

Acknowledgments

We greatly thank the anonymous reviews for their valuable comments. This work is supported by the National Natural Science Foundation of China (Grand Nos. 61672081, U1636211, 61370126), Beijing Advanced Innovation Center for Imaging Technology (No.BAICIT-2016001), National High Technology Research and Development Program of China under grant (No.2015AA016004).

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bao, J.; Duan, N.; Zhou, M.; and Zhao, T. 2014. Knowledge-based question answering as machine translation. *Proceedings of ACL* 2:6.

Berant, J.; Chou, A.; Frostig, R.; and Liang, P. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of 2013 EMNLP*, volume 2, 6.

Brown, P. F.; Pietra, V. J. D.; Pietra, S. A. D.; and Mercer, R. L. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics* 19(2):263–311.

Burges, C. J. 2010. From ranknet to lambdarank to lambdamart: An overview. *Microsoft Research Technical Report MSR-TR-2010-82* 11(23-581):81.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the EMNLP*, 1724–1734.

Del Corro, L., and Gemulla, R. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on WWW*, 355–366.

Duan, N.; Tang, D.; Chen, P.; and Zhou, M. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 877–885. Association for Computational Linguistics.

Fader, A.; Soderland, S.; and Etzioni, O. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on EMNLP*, 1535–1545.

Fader, A.; Zettlemoyer, L. S.; and Etzioni, O. 2013. Paraphrase-driven learning for open question answering. In *ACL (1)*, 1608–1618.

Fader, A.; Zettlemoyer, L.; and Etzioni, O. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD*, 1156–1165. ACM.

Ishiwatari, S.; Yao, J.; Liu, S.; Li, M.; Zhou, M.; Yoshinaga, N.; Kitsuregawa, M.; and Jia, W. 2017. Chunk-based decoder for neural machine translation. In *Proceedings of the 55th ACL*, 1901–1912.

Khashabi, D.; Khot, T.; Sabharwal, A.; Clark, P.; Etzioni, O.; and Roth, D. 2016. Question answering via integer programming over semi-structured knowledge. *Proceedings of the IJCAI-16* 1145–1152.

Khot, T.; Sabharwal, A.; and Clark, P. 2017. Answering complex questions using open information extraction. In *Proceedings of the 55th ACL*, 311–316.

Koehn, P.; Och, F. J.; and Marcu, D. 2003. Statistical phrase-based translation. *Proceedings of Annual Conference of the (NAACL-HLT)* 1:48–54.

Manning, C. D.; Raghavan, P.; Schütze, H.; et al. 2008. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Miao, Y.; Yu, L.; and Blunsom, P. 2015. Neural variational inference for text processing. *arXiv preprint arXiv:1511.06038*.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Och, F. J., and Ney, H. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics* 29(1):19–51.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, 311–318.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Schmitz, M.; Bart, R.; Soderland, S.; Etzioni, O.; et al. 2012. Open language learning for information extraction. In *Proceedings of the EMNLP*, 523–534.

Severyn, A., and Moschitti, A. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of ACM SIGIR*, 373–382.

Shen, Y.; He, X.; Gao, J.; Deng, L.; and Mesnil, G. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the Conference on Information and Knowledge Management*, 101–110.

Sukhbaatar, S.; Szlam, A.; Weston, J.; and Fergus, R. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2431–2439.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Tang, D.; Duan, N.; Qin, T.; and Zhou, M. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; and Li, H. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th ACL*, 76–85.

Tymoshenko, K.; Bonadiman, D.; and Moschitti, A. 2016. Convolutional neural networks vs. convolution kernels: Feature engineering for answer sentence reranking. In *HLT-NAACL*, 1268–1278.

Wang, D., and Nyberg, E. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53nd ACL*, 707–712.

Wang, C.; Jiang, F.; and Yang, H. 2017. A hybrid framework for text modeling with convolutional rnn. In *Proceedings of the 23rd ACM SIGKDD*, 2061–2069. ACM.

Wang, B.; Liu, K.; and Zhao, J. 2016. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th ACL*.

Yang, Y.; Yih, W.-t.; and Meek, C. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the Conference on EMNLP*, 2013–2018.

Yates, A.; Cafarella, M.; Banko, M.; Etzioni, O.; Broadhead, M.; and Soderland, S. 2007. Textrunner: open information extraction on the web. In *The Annual Conference of the NAACL*, 25–26.

Yih, W.-t.; Chang, M.-W.; Meek, C.; and Pastusiak, A. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, 1744–1753.

Yih, W.-t.; Chang, M.-W.; He, X.; and Gao, J. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of ACL*.

Yin, P.; Duan, N.; Kao, B.; Bao, J.; and Zhou, M. 2015. Answering questions with complex semantic constraints on open knowledge bases. In *Proceedings of the 24th ACM International on CIKM*, 1301–1310. ACM.

Yin, W.; Schtze, H.; Xiang, B.; and Zhou, B. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *TACL* 4:259–272.

Yu, L.; Hermann, K. M.; Blunsom, P.; and Pulman, S. 2014. Deep learning for answer sentence selection. *NIPS Deep Learning and Representation Learning Workshop*.

Zeiler, M. D. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.