Comprehension based Question Answering

Ashish Kumar Singh
14142

Bhangale Pratik Anil
14173

Kunal Chaturvedi
14342

Rohith Mukku
14402

Shubham Agrawal
14671

Swati Gupta
14742

Group No. - 8

Instructor: Dr. Harish Karnick
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

April 2018

Abstract

The problem statement of this project is to build a learning algorithm that enables the machine to read a comprehension and subsequently answer questions based on it. The ability of machines to read a com- prehension and answer queries is one of the most important part of machine human interaction. This field is growing rapidly. It has an important role in generating replies for chatbots or smart answers of Google inbox.

Contents

1 Introduction

One of the main task of NLP is to understand comprehension. It can be done in many ways. However the most popular one is comprehension based question answering. In this, some questions are asked based on the comprehension which need to be answered. Questions have different types, MCQ based, word(or span of word) from comprehension or summarizing the comprehension. We have selected the questions which need to be answered from selecting the word(or span of the words) from the comprehension

2 Project Details

In our project, we have implemented two models on two different datasets. Our first model is based on Memory network framework to solve the specific question answering task. The dataset that we have used is bAbI, which has 20 different tasks based on type of questions and the passages. The next model that we have implemented is FastQA which aim is to simplify the task of neural QA by using simple heuristic based neural network framework.

2.1 Datasets

Due to development in large-scale comprehension based question answering datasets, research on this topic has largely increased. The two major datasets on which we are testing our algorithms are :-

- **SQuAD**
  Stanford Question Answering Dataset (SQuAD)[1] is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage. It has 100,000+ question-answer pairs on 500+ articles.

- **bAbI**
  Developed by facebook[2], this dataset provides a set of training and test data. The training set is given by the true answers to questions, and the set of relevant statements for answering a given question, which may or may not be used by the learner. The tasks are set up so that correct answers are limited to a single word or a list of words.

2.2 Problem statement

We have formulated the above problem as follows. We have a paragraph $p = p_0, p_1, ...p_m$ of m length and a question $q = q_0, q_1, ...q_n$ of n length related to the paragraph. The goal is to generate an answer span a with starting index a_i and ending index a_e of paragraph p. Machine learns a method to generate this span of words from paragraph, and here, it learns a predictor function, $f(q, p) \rightarrow a$ from a training dataset of (q, p, a) triplets.

3 Literature Review

The task of comprehension based question and answering can be tackled by using different combination of start-of-the-art algorithms.

- **Deep Neural Networks**: In this approach, we use deep neural networks that uses memory units like RNNs and LSTMs. These are powerful sequence predictors that can be efficiently trained to learn to do inference over long term dependencies in the text. The drawback of this approach is the deficiency of structured memory compnent.

- **Memory Networks**: In this approach, we aim to learn how to reason with inference components and a long-term memory component. Memory network[3] serves as a knowledge base to recall facts from the past. The model tries to learn a scoring function to rank relevant memories.

- **Pointer Net**: Pointer Networks[4] are a neural architecture which learn the conditional probability of an output sequence with elements that are discrete tokens corresponding to positions in an input sequence rather than a large fixed vocabulary. Pointer Networks solve the problem of variable size output dictionaries using a mechanism of neural attention.

- **RaSoR**: Recurrent span representation[5] is an end-to-end neural network architecture to identify answer spans. The objective of this approach is to incorporate question as it plays an important part in predicting the answer. This is incorporated by creating concatenation of three embeddings. First is the question independent passage embedding, second is the question based embedding and third is passage combined with question based embedding.

4 Approach

Two models that we have implemented is MemNN and FastQA. For MemNN we have used bAbI dataset to training and testing purposes. The dataset is divided into 20 different tasks and our model is trained and tested on every tasks. Other model is FastQA which is trained and tested on SquAD dataset.

4.1 MemNNs

Memory networks reason with inference components combined with a long-term memory component; they learn how to use these jointly. The long-term memory can be read and written to, with the goal of using it for prediction. The central idea is to combine the successful learning strategies developed in the machine learning literature for inference with a memory component that can be read and written to.

4.1.1 Architecture

A memory network consists of two things. One is the memory which is an array of objects(which can be vectors or strings). Other one is the combination of models which are divided into 4 components. These models are I, G, O and R as described below :-

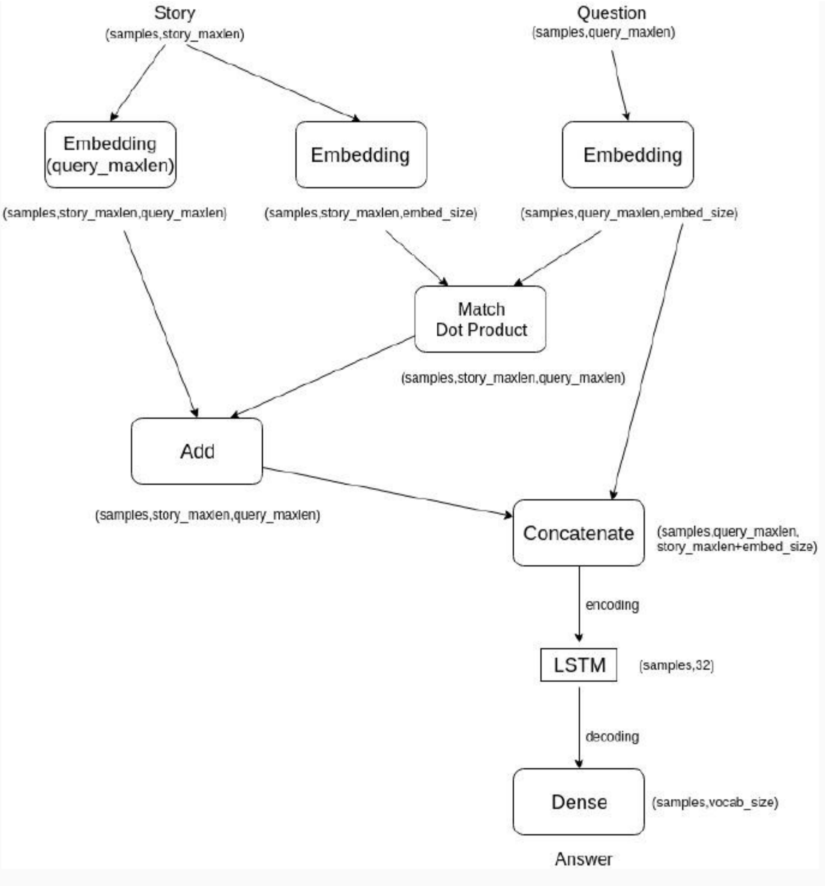

Story
(samples,story_maxlen)
Question
(samples,query_maxlen)
Embedding (query_maxlen)
Embedding
Embedding
(samples,story_maxlen,query_maxlen)
(samples,story_maxlen,embed_size)
(samples,query_maxlen,embed_size)
Match Dot Product
(samples,story_maxlen,query_maxlen)
Add
(samples,story_maxlen,query_maxlen)
Concatenate
(samples,query_maxlen, story_maxlen+embed_size)
encoding
LSTM
(samples,32)
decoding
Dense
(samples,vocab_size)
Answer

Figure 1: MemNN Architecture

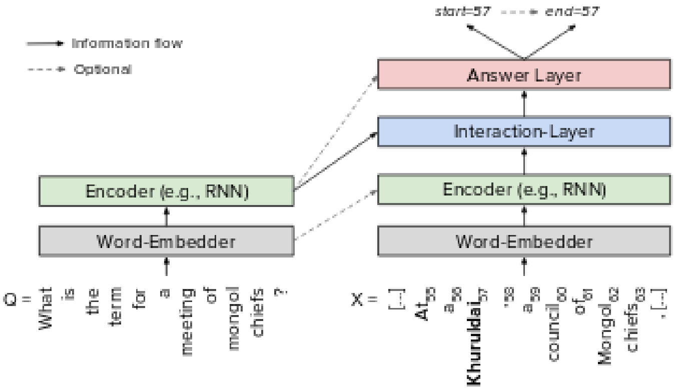
- Input Feature Map : This component is responsible for pre-processing of the input. It uses standard methods like Tokenisation, parsing, co reference, etc. Thus converting input into feature vector.
- Generalization : This component updates a slot of the memory based on current feature vector.
- Output Feature Map : This component is responsible for generating the new output which is based on input feature vector and the current memory slot.
- Response : This is the final component, which produces the output in the given format required. In our case the span of the words from the passage. We have used LSTM for this stage.

4.2 FastQA

FastQA[6] is based on *context/type matching heuristic*. This heuristic aims to select the span of words from the comprehension based on two things :

- *Match the expected answer type* : The type of answer can be sensed from the question. For e.g. in case of ”when” question, the expected answer is time.
- *Proximity to important question word*: Important question words are some nouns in the questions. For e.g. :- When did building activity occur on St. Kazimierz Church? In this question ”St. Kazimierz Church” is the important question word.

4.2.1 Architecture :



start=57
end=57
Information flow
Optional
Answer Layer
Interaction-Layer
Encoder (e.g., RNN)
Encoder (e.g., RNN)
Word-Embedder
Word-Embedder
Q = What is the term for a meeting of mongol chiefs ?
X = [...] At$_{55}$ a$_{56}$ Khuruldai$_{57}$,$_{58}$ a$_{59}$ council$_{60}$ of$_{61}$ Mongol$_{62}$ chiefs$_{63}$, [...]

Figure 2: FastQA Architecture

- **Embedder**: Character and GLOVE embedding have been used.
- **Encoder**: Embedded tokens are further encoded by a composition function. A prominent type of encoder is the (bi-directional) LSTM.
- **Interaction layer**: Responsible for word-by-word interaction between context and question. An attention layer has been realized here.
- **Answer layer**: Predicts the start and the end of the span separately.

4.2.2 SQUAD Preprocessing

- Each data point is taken as a tuple of (Context, Question, Answer Span).
- It is then tokenized and character span is included into the token span. Max context size set to 500 and max question size to 50.
- We used pretrained GLoVE embeddings as 50 dim vector and average of character embeddings for any word that is absent in GLoVE.
- Character embeddings were calculated using GLoVE embeddings by averaging the sum of all word vectors in which the character occurs.

4.2.3 Context and Question Encoding

Every context and question is passed into Bidirectional LSTM of size equal to embeddings size to learn the effect of previous and future words on the current words. Output of Bi-LSTM is fed into fully connected dense layer to get final encoded context and question

4.2.4 Question Attention

This layer is used to find out important information in question, which later is multiplied with context to get relevant context to get answer from. The important information is calculated by applying softmax over question and calculate weighted sum over question encoding to get probability of importance of particular token in question.

4.2.5 Predicting Answer Start

Used fully connected neural network to predict the start of the answer with the following features-

- Question Attention Vector multiplied over passage embeddings (To get relevant components from passage)
- Question Attention Vector itself
- Passage Encoding

4.2.6 Predicting Answer End

Predicting answer end depends on the answer start and we tried to exploit this We extracted embedding of token from passage where answer started and used it along with the following features:

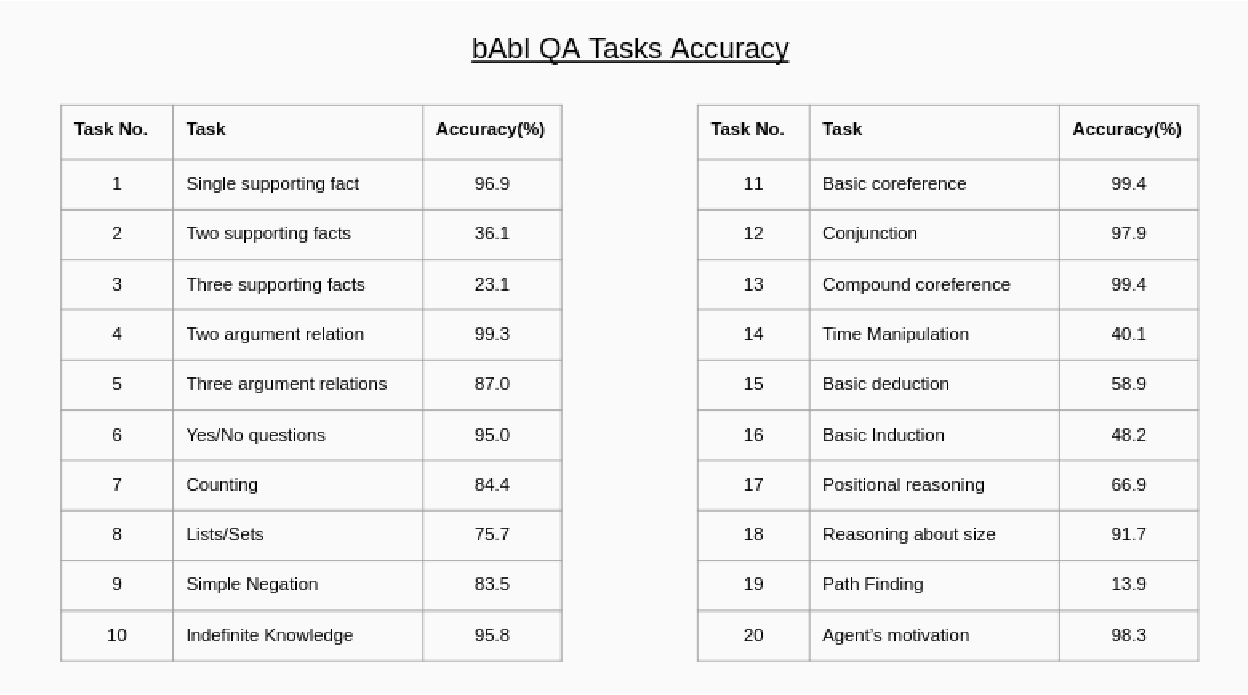
- Question Attention Vector multiplied over passage embeddings (To get relevant components from passage to question)
- Answer start vector multiplied over passage embeddings (To get relevant components in passage to answer start vector)
- Question Attention Vector itself
- Answer start vector Passage Encoding
- All these features are concatenated and using softmax over passage size, we try to predict single index where answer may have ended

All these features are concatenated and using softmax over passage size we try to predict single index where answer may have been started.

5 Results

5.1 MemNN on bAbI dataset

We implemented MemNN on bAbI dataset. The accuracy we get on this dataset is:-



bAbI QA Tasks Accuracy

Task No.	Task	Accuracy(%)
1	Single supporting fact	96.9
2	Two supporting facts	36.1
3	Three supporting facts	23.1
4	Two argument relation	99.3
5	Three argument relations	87.0
6	Yes/No questions	95.0
7	Counting	84.4
8	Lists/Sets	75.7
9	Simple Negation	83.5
10	Indefinite Knowledge	95.8

Task No.	Task	Accuracy(%)
11	Basic coreference	99.4
12	Conjunction	97.9
13	Compound coreference	99.4
14	Time Manipulation	40.1
15	Basic deduction	58.9
16	Basic Induction	48.2
17	Positional reasoning	66.9
18	Reasoning about size	91.7
19	Path Finding	13.9
20	Agent's motivation	98.3

5.2 FastQA on SQuAD dataset

We implemented FastQA on SQuAD dataset. The accuracy we got on the dataset is:-

Table 1: FastQA Result

Serial No.	Description	Accuracy
1	Answer start training accuracy	23.42%
2	Answer end training accuracy	24.03%
3	Answer start testing accuracy	18.31%
4	Answer end testing accuracy	19.91%

6 Scope for Improvement

For FastQA Model:

- Using 300 dimension GLoVE embedding instead of 50
- Increasing number of training Epochs, Context vector Length and Question vector Length
- Using a simple neural network for computing character embeddings of missing words in the vocabulary.
- Using better tokenization but at the same time calculating accurate spans

7 Contribution

Table 2: Percentage Contributions

SNo	Name	Percentage
1	Ashish Kumar Singh	25%
2	Bhangale Pratik Anil	25%
3	Kunal Chaturvedi	10%
4	Rohith Mukku	20%
5	Shubham Agrawal	10%
6	Swati Gupta	10%

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

[2] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698, 2015.

[3] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015.

[4] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc., 2015.

[5] Kenton Lee, Tom Kwiatkowski, Ankur P. Parikh, and Dipanjan Das. Learning recurrent span representations for extractive question answering. *CoRR*, abs/1611.01436, 2016.

[6] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Fastqa: A simple and efficient neural architecture for question answering. *CoRR*, abs/1703.04816, 2017.

Related links

- *https://keras.io/getting-started/sequential-model-guide/*
- *http://www.thespermwhale.com/jaseweston/icml2016/*
- *https://yerevann.github.io/2016/02/05/implementing-dynamic-memory-networks/*
- *https://cs224d.stanford.edu/reports/KapashiDarshan.pdf*
- *https://github.com/AnatoliiPotapov/squad*
- *http://minimaxir.com/2017/04/char-embeddings/*