Comprehension Based Question Answering

Group - 8

Project Members

1. Ashish Kumar Singh
2. Bhangale Pratik Anil
3. Kunal Chaturvedi
4. Rohith Mukku
5. Shubham Agrawal
6. Swati Gupta

IIT Kanpur

April 20 2018

Overview

1. Introduction
2. Datasets
    a. bAbI dataset
    b. SQuAD dataset
3. Existing Work
4. Our Approach
    a. memNN
    b. fastQA
5. Results

Introduction

- One of the main task of NLP is to understand comprehension
- The most popular way is comprehension based question answering in which questions are asked based on the comprehension
- Questions have different types, MCQ based, word(or span of word) from comprehension or summarizing the comprehension

Datasets

bAbI dataset

- Developed by facebook, this dataset provides a set of training and test data for 20 different types of tasks
- The training set has a set of relevant statements for answering a question, which may be used
- The tasks are set up so that correct answers are limited to a single word or a list of words

SQuAD dataset

- Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset
- 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

Relevant Work

Deep Neural Networks

- Uses memory units like RNNs and LSTMs
- Powerful sequence predictors that can be efficiently trained to learn to do inference over long term dependencies in the text
- **Drawback:** Deficiency of structured network

Relevant Work

Memory Networks

- Aims to learn how to reason with inference components and a long-term memory component
- Serves as a knowledge base to recall facts from the past
- Tries to learn a scoring function to rank relevant memories

Relevant Work

Pointer Networks

- 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
- Solve the problem of variable size output dictionaries using a mechanism of neural attention

Relevant Work

Match-LSTM and Answer Pointer

- Consists of four layers : LSTM preprocessing layers for query and paragraph, Match-LSTM layer and Answer Pointer Layer.
- LSTM preprocessing layers incorporate the contextual information into the representation of each token in the passage and the question.
- Match-LSTM is used to predict textual entailment between question and passage. This is a novel layer by (Wang & Jiang, 2016).
- Answer Pointer Layer predicts the sequence of answer tokens (sequence model) or start token and end token (boundary model)

Relevant Work

RaSoR

- An end-to-end neural network architecture that incorporates 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

Our Approach

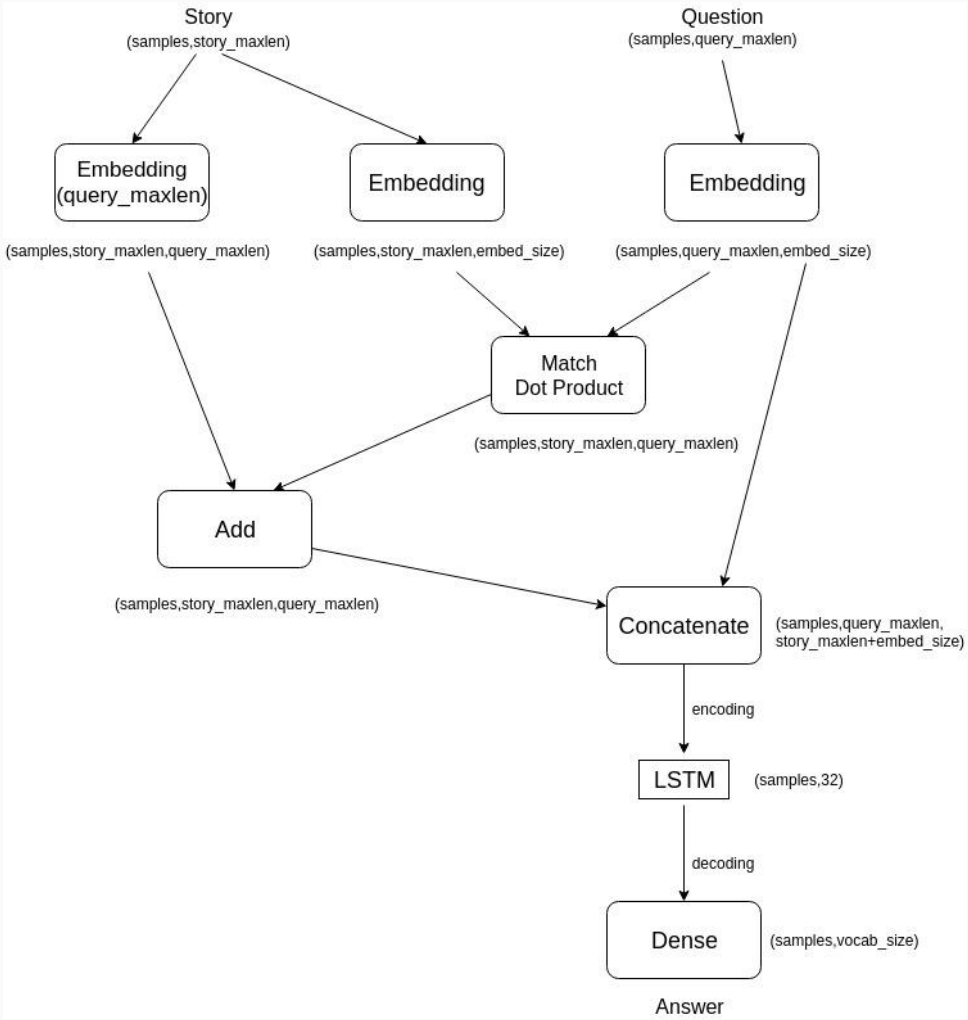
MemNNs

- 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 write

MemNN Preprocessing

- Each data point as tuple of (Context, Question, Answer)
- Tokenized our data
- Generated a vocabulary from the whole dataset
- Converted context and question into vector of size story_maxlen, query_maxlen
- Index of word in vocabulary was used for vectorizing context, question and answer
- Dataset has been provided with relevant sentences in context that are required for answer the question, but we haven’t used that information

MemNN Architecture



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

Our Approach

FastQA

FastQA is state-of-the-art QA system based on context/type matching heuristic. It has two components :-

- **Match 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 phrases in the question. For e.g. :- In “*When did building activity occur on St. Kazimierz Church?*”, "St. Kazimierz Church" is the important question word.

FastQA - Architecture

- **Embeddings:** Character and GLOVE embeddings
- **Encoder:** Embedded tokens are further encoded by LSTM
- **Interaction Layer:** Attention layer handles the word-by-word interaction between context and question
- **Answer Layer:** Predicts the start and the end of the span separately

SQuAD Preprocessing

- Each data point as tuple of (Context, Question, Answer Span)
- Tokenized our data and bring character span into token span. Max context size set to 500 and max question size to 50.
- 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 sum of all word vectors in which the character occurs.

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

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.

Predicting Answer Start

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

Features :

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

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

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 as a feature

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 been ended

Results

- Total 12,000 (Passage , Question) pairs used for training and 3,000 for testing
- After 20 epochs, Exact matching accuracy -
  - Answer start training accuracy : 23.42%
  - Answer start testing accuracy : 24.03%
  - Answer end training accuracy : 18.31%
  - Answer end training accuracy : 19.91%

Thank You!