Machine Comprehension using Match-LSTM and Answer Pointer

Anupam Gupta
Khoury College of Computer Sciences
Northeastern University
Boston, MA
gupta.anup@northeastern.edu

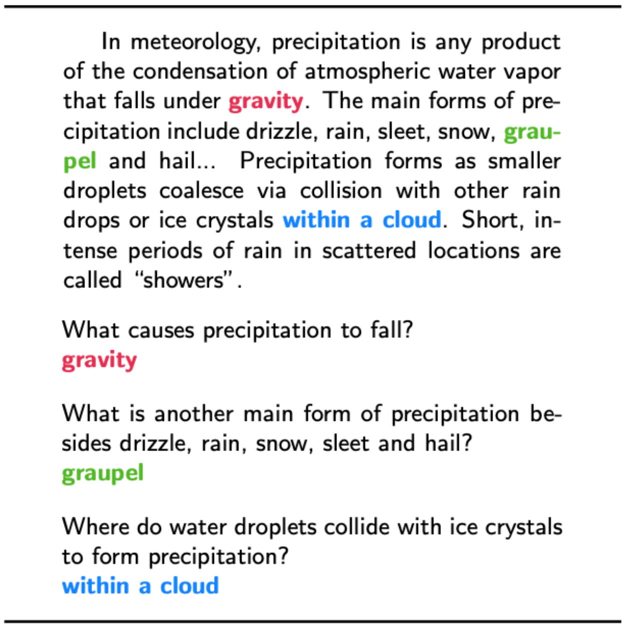
Sinjini Bose
Khoury College of Computer Sciences
Northeastern University
Boston, MA
bose.s@northeastern.edu

Manaswini Nagaraj
Khoury College of Computer Sciences
Northeastern University
Boston, MA
nagaraj.m@northeastern.edu

***Abstract*—Machine comprehension is a closed-form question and answering task, given a context/passage and questions where answers are spans extracted from the passage. Teaching machines to read, process, and comprehend natural language documents is a coveted goal for artificial intelligence and could serve as a benchmark to see how well machines understand the text. Several benchmark datasets have been created to focus on answering questions to evaluate machine comprehension and a machine's ability to understand the text. Here we explore LSTM based techniques to solve this problem.**

I. Introduction

We implement a deep learning model using Match-LSTM and Answer Pointer [2] on the Stanford Question and Answering Dataset [1]. The dataset comprises 100k+ questions in the training set and 10k+ questions in the test set, and a hidden dev set posed by crowdsourced on a set of Wikipedia articles where the answer to a question is a segment from the corresponding reading passage. Here is a sample passage :


In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?
gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

Figure 1: Question-answer pairs for a sample passage in the SQuAD dataset. Each of the answers is a segment of text from the passage.

The SQuAD does not provide a list of answer choices for each question. Instead, the system must select the answer from all possible spans in the passage, thus needing to cope with a relatively large number of candidates. Span based questions can be constrained from the open based and more interpretive questions, but we still find a rich diversity of question and answer types with lexical variation and syntactic divergence in SQuAD. Sample QA :

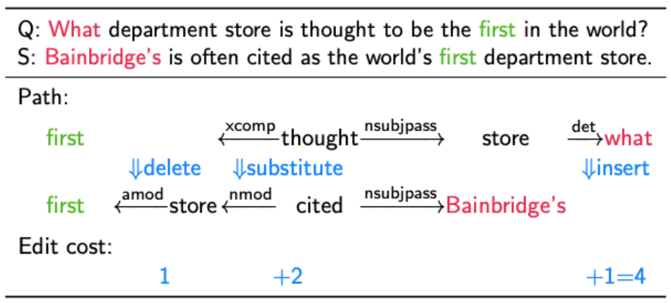

Q: What department store is thought to be the first in the world?
S: Bainbridge's is often cited as the world's first department store.

Figure 3: An example walking through the computation of the syntactic divergence between the question Q and answer sentence S.

II. Related Work

A. *Previous work done*

There has been a significant amount of work done in solving Machine comprehension tasks using deep neural networks. A common approach is to use recurrent neural networks(RNNs) to process the given text and the question to predict or generate the answers (Hermann 2015). Various models have also used Question-aware passage representation with Match-LSTM and Pointer Networks to generate answers that contain multiple tokens from the given passage (Wang and Jiang, 2016).

Hermann (2015) first introduced the attention mechanism into reading comprehension, which matches the question with the given passage [4]. The co-attention mechanism (Xiong, Zhong, Socher 2017; Seo 2017) is also a widely used model that is computed as an alignment matrix corresponding to all pairs of context words and query words, which can model complex interaction between the query and the context.

Multi-hop reasoning models have also been proposed for MC tasks (Shen, 2016; Xiong, Zhong, Socher 2017). These models typically maintain a memory state which incorporates the current information of reasoning with the previous information in the memory by following the framework of Memory Networks (Sukhbaatar 2015).

III. Model Description

We use an initial LSTM encoding layer to embed every token's contextual information in the passage and the question. Then the embeddings are passed to our Match-LSTM layer, which uses a standard word-by-word mechanism and produces an output of hidden states, which is then passed to the final Answer Pointer layer to predict two indices(start, end) and all the tokes inside these two tokens(combined) are taken as the final answer span. The answer pointer layer adopts the Ptr-Net model to predict answer spans from continuous text rather than a fixed vocabulary.

A. *Pre-processing Layer*

- Standard one directional LSTM layer to encode both the passage and the question separately.

$$H^p = \overrightarrow{LSTM}(P), H^q = \overrightarrow{LSTM}(Q)$$

- Objective is to encode contextual information into the representation of each token
- Since the model uses textual entailment, we consider the question to be the premise and the context to be the hypothesis.

$$H^p = (h^p_1, h^p_2, ...h^p_k), H^q = (h^p_1, h^p_2, ...h^p_j)$$

- Here each h represents the embedding for that particular token in the question or passage.

B. *Match-LSTM Layer*

- The objective of word-by-word attention is to introduce a series of attention-weighted combinations of the hidden states of the premise(in our case the question) where each combination for is for a particular word in the hypothesis(in our case for the passage). Let us use a_k to denote such a vector for each work h^p_k in the passage which is defined as follows :

$$a_k = \sum_{i=1}^{j} a_{kj} * h^j_s$$

- where a_{kj} is an attention weight which encodes the degree to which h^p_k is aligned with h^q_j
- then we combine this $a_k j$ with the corresponding h^p_k to generate the vector m_k

$$m_k = [a_k, h^p_k]$$

- this new vector is passed through another one-directional LSTM called to form the match-LSTM in the forward direction

$$H^m_{fw} = \overrightarrow{LSTM}(m)$$

- we build a similar LSTM in the reverse direction

$$H^m_{bw} = \overleftarrow{LSTM}(m)$$

- Final hidden states are calculated by concatenating the two forward and backward models

$$H^m = concat[H^m_{fw}, H^m_{bw}]$$

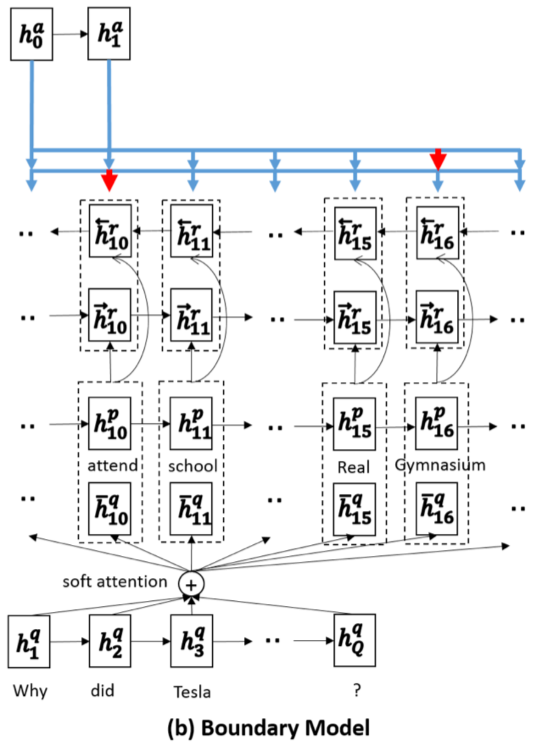
- This is the input for the final Answer Pointer layer

C. *Answer Pointer Layer*

- Uses the output of the Match-LSTM layer as the input to predict two indices from the passage
- All tokens inside including those two indices are considered as part of the answer span
- The probability of selecting the answer sequence is modelled as :

$$h(a|H^m) = p(a_s|H^m) * p(a_e|a_s, H^m)$$

Figure 4: Model Architecture



IV. Experiments

A. *Data Pre-processing*

We use the Stanford Question Answering Dataset (SQuAD) v1.1 to conduct our experiments. Passages in SQuAD come from 536 articles from Wikipedia covering a wide range of topics. Each passage is a single paragraph from a Wikipedia article, and each passage has around 5 questions associated with it. In total, there are 23,215 passages and 107,785 questions. The data has been split into a training set (with 87,599 question-answer pairs), a development set (with 10,570 question-answer pairs), and a hidden test set.

We used Stanford GloVe embeddings to obtain vector representations of words. GloVe performs training on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. We have used GloVe word embeddings of dimensionality d = 50, 100, 200, and 300, 6B tokens and vocab size of 400k pre-trained on Wikipedia 2014 and Gigaword 5. Words not found in GloVe have been initialized as zero vectors.

The data is in the form of triplets: context, question, and answer-index span (Answer and initial and final indices of the answer within the context passage). We transform the dataset to word index id representations using the dictionary from GloVe embeddings. These id representations are converted into NumPy arrays with their respective embedding and other related information before exploring them as a single hdf5 file. HDF5 is a compressed data format that can support large, heterogeneous, and complex datasets. PyTorch hdf5 enables lazy loading (upon request by the data loader) to work with datasets that do not fit in memory and allow custom transformations of the data.

B. Settings

The entire model was built and trained using PyTorch. We use a standard word-by-word neural attention model that uses the concept of textual entailment and tries to match the question (in our, case premise) against the passage (in our case the hypothesis). The dimensionality l of the hidden layers is set to be 150. We use ADAMAX optimizer (Kingma Ba, 2015) with the coefficients 1 = 0.9 and 2 = 0.999 to optimize the model and do not use L2 regularization. We train over a batch size of 32 with a total of 30 epochs.

C. Evaluation Metrics

We make use of two evaluation metrics over here namely Exact Match (EM) and F1 Score.

- Exact Match :
  - This metric measures the percentage of predictions that match the ground truth answers exactly
  - Strict 0 or 1 metric, meaning even if the model prediction is off by just 1 character, its EM Score will be 0
- F1-score :
  - This metric measures the average overlap between the prediction and ground truth answer. We treat the prediction and ground truth as bags of tokens, and compute their F1.
  - We calculate the Precision by dividing the number of shared tokens between the prediction and ground truth by the number of tokens in the model prediction. Recall does the same thing but divided using the ground truth. Finally we use the formula for F1 Score using this Precision and Recall.

D. Results

We trained the model using a batch size of 32 and for a total of 30 epochs. It took around 2.4 hours per epoch, which amounted to a total training time of 3 days. We achieved an Exact Match score of 45.2 and an F1 score of 53.

V. Future Work

The work done in this project serves as a benchmark for future improvements. For example, we used only 50 Glove vector embeddings. One of the enhancements could be to use the full 300-dimensional embeddings, which would give a nice jump to the EM and F1 scores. Although there were some successful models such as BiDAF, since the arrival of BERT [9], many models take advantage of it and have been able to beat human performance. BERT was initially created as a language model but later adapted for QA tasks. It is possible to use BERT alone or leverage pre-trained BERT embeddings generated using Masked Language Modeling and Next Sentence Prediction into existing models by replacing the first pre-processing layer. Since this is such an impressive feat, our future research will be focused on utilizing BERT based models to achieve a state of the art accuracy.

References

[1] Stanford Question and answering Dataset(SQuAD)

[2] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2016

[3] Shuohang Wang, Jing Jiang. Machine Comprehension Using MatchLSTM and Answer Pointer, 2016.

[4] Karl Moritz Hermann, Tom Koisk, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, Phil Blunsom. Teaching Machines to Read and Comprehend, 2015.

[5] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. CoRR, abs/1611.01603, 2016.

[6] Christopher Clark, Matt Gardner.Simple and Effective Multi-Paragraph Reading Comprehension, 2017

[7] SQuAD dataset Preprocessing starter code.

[8] Match-LSTM+ modules and starter code by laddie132

[9] How Does BERT Answer Questions? A Layer-Wise Analysis of Transformer Representations

[10] Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. WIKIREADING: A novel large-scale language understanding task over wikipedia. In Proceedings of the Conference on Association for Computational Linguistics, 2016

[11] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2014.

[12] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In Proceedings of the Conference on Advances in neural information processing systems, 2015.

Appendix

Here we would like to show some output of our model given a context and a questions, what the model predicts and the ground truth(Golden) answer :

Context: While the Commission has a monopoly on initiating legislation, the European Parliament and the Council of the European Union have powers of amendment and veto during the legislative process.
Question: Which two governing bodies have legislative veto power?
Gold Standard: European Parliament and the Council of the European Union
Model Prediction: European Parliament and the Council of the European Union

---

Context: ...Controlled, experimental studies exploring intrinsic motivation of college students has shown that nonverbal expressions of enthusiasm, such as demonstrative gesturing, dramatic movements which are varied, and emotional facial expressions, result in college students reporting higher levels of intrinsic motivation to learn...
Question: What is dramatic gesturing an example of?
Gold Standard: nonverbal expressions of enthusiasm
Model Prediction: emotional facial expressions

Context: In time, Kublai Khan 's successors lost all influence on other mongol lands across asia, while the mongols beyond the middle kingdom saw them as too chinese. Gradually, they lost influence in china as well... Uninterested in administration, they were separated from both the army and the populace, and china was torn by dissension and unrest.
Question: Why did Kublai 's successors lose control of the rest of the mongol empire?
Gold Standard: The mongols beyond the middle kingdom saw them as too chinese
Model Prediction: mongols beyond the middle kingdom saw them as too chinese

---

The above output shows the nuances of creating a model that can precisely select an answer span from the passage. The errors relate to Imprecise answer boundaries or dropping the word ”the” from the answer which is sometimes even ambiguous for a human to correctly identify. Fortunately these discrepancies do not have too much of a detrimental effect on the f1 score. Improving these can be experimented with hyperparameter tuning and optimizing other aspects of our model.