Reading Comprehension on Wikipedia Articles   
CS 221 Project Proposal  
  
Nick Landy Anand Natu Megan Knight  
nlandy@stanford.edu anatu@stanford.edu mlknight@stanford.edu

# **introduction**

Reading comprehension and question answering (QA) has been a popular natural language processing (NLP) problem for artificial intelligence (AI) systems over the recent years. Various approaches to this problem have been made with the use of neural network models. The purpose of this project will be to implement an AI that can take a Wikipedia article and answer questions directly related to the given text.

# **FRAMEWORK**

## Preliminary Data

The dataset that we are using includes a collection of 60 Wikipedia articles each with around 40 sets of questions and the corresponding correct answer that was collected by a group of students from Carnegie Mellon University [9]. The types of questions and answers vary from yes and no to full sentence answers. This gives us varying levels of difficulty to work with for our dataset.  
 After evaluating our baseline and our oracle, we discovered that the dataset contained a large amount of NULL answers and duplicate questions. As a result, we are also looking into other datasets to use such as Stanford Question Answering Dataset (SQuAD) and Microsoft’s Machine Reading Comprehension Dataset (MARCO) [3],[11].

## Input and Output Examples

The following is a short example of the input-output behavior for our system. The input is a full Wikipedia article and a question on the article. The output is the answer to the question which can be found as either a substring of the article or yes, no, or NULL (where NULL means the answer is not in the article).

**Passage**: “Worker ants do not have wings and reproductive females lose their wings after their mating flights in order to begin their colonies. Therefore, unlike their wasp ancestors, most ants travel by walking.”

**Question**: “Do worker ants have wings?”

**Answer**: “No”

**Question**: “How do most ants travel?”

**Answer**: “most ants travel by walking”

## Evaluation Metric

The evaluation metric we plan to use will be the accuracy of the predicted answer output by our system to the actual answer. This will be done by dividing the number of words in the actual answer by the number of elements of the union of the word indices in the actual answer and the predicted answer.

## Baseline

A dictionary was created from all the unique words in the articles and questions of our dataset. The inputs to the baseline included the question and the full article that the question was based off of. The question was reformatted to a vector of numbers for each word corresponding to the index of that word in the dictionary. This was input into a recurrent neural network (RNN) with an embedding layer on the input and a softmax layer on the output implemented using Keras. The baseline was trained for only one epoch so that we could get a metric quickly. The accuracy that we received from this baseline was 23% due to the baseline always outputting NULL, which happened to be the answer for 23% of our test set. This shows that the baseline was grossly underfitting, thus we have plenty of room to improve.

## Oracle

One of us took a random set of 50 questions from different articles in the dataset and answered them. Our oracle is based on the accuracy of our answers to the correct answer using our evaluation metric. The accuracy we received was 84.32 %.

The gap between our accuracy from our oracle and our baseline is 61.32%. This gap makes sense since the baseline was essentially just outputting the same answer (the most common out of all the questions) for every question in the dataset which is not going to provide a high accuracy. The oracle obviously gives us a much higher accuracy since a human answered the set of questions by using logic and reasoning.

## Challenges

The challenges with this project mostly come from answers that deviate from yes / no and one-word answers. This is because multiple answers can have the same meaning, but have different sentence structure or number of words. This causes complications with our evaluation metric in giving us the desired accuracy. Another issue seen from our dataset are questions that don’t have answers and we need to make sure our model can properly handle these types of questions. Other challenges include finding ways to map the input vector through character or word embedding and whether we want to train our embedding or use pre-trained embeddings like word2vec or Global Vectors for Word Representation (GloVe).

# **IMPLEMENTATION**

1. Diagramming Representation of Proposed Model Design

## Character / Word Embedding

The first step of our implementation involves representing words and sentences as embedding vectors. Embedding can be done at both the character and word level [10]. Pre-trained embeddings can be downloaded and used directly, GloVe being one of the most common [8],[5].

However, embeddings can also be learned directly from text corpora using a model of choice including Convolutional Neural Networks (CNNs), word2vec, and other machine learning techniques [10],[7],[1]. Generally, the input for this step is context / query text data with vocabularies , and outputs embedding matrices of d-dimensional vectors for each word (i.e. .

## Contextual Embedding

The contextual embedding layer is an additional feature development layer which looks specifically at the temporal interactions between words (i.e. where words appear in relation to one another). This is usually accomplished using bi-directional Long Short-Term Memory (LSTMs) with established context on either side of a given word. This layer accepts the embedding matrices from the previous layer as input, and outputs two new context-aware embedding matrices for the query and context [10],[8].

## Context-Query Attention

This step models the interactions between the context and query words to determine which words the model should “attend’ to first. The attention matrix can be developed using a simple dot product of the context and query embedding matrices [8],[4]. The output varies depending on the attention modeling used, but typically can take a general form of a similarity matrix measuring similarity between context and query vocabulary words and .

## Prediction / Modelling / Output

The modeling layer is used to capture interactions between the context words conditioned upon a certain query. This can be accomplished using a bi-directional LSTM or variants of Recurrant Neural Networks (RNNs), or in simple cases just a linear layer [8],[7],[2]. A softmax activation function is typically used to evaluate the predicted answer.

# **RELATED WORK**

There have been numerous QA projects based on reading comprehension with various approaches. Much of this work has been done with the SQuAD, MARCO, CNN, and DailyMail datasets [3],[11],[6]. Some of the most noteworthy have been from Alibaba and Microsoft where they have developed AIs that outperform humans in reading comprehension. Other work has also been done by previous CS221 and CS224N students using the SQuAD dataset.

# **REFERENCES**

1. A. Bordes, J. Weston, and S. Chopra, “Question Answering with Subgraph Embeddings,” *Arxiv*, 04-Sep-2014. [Online]. Available: https://arxiv.org/pdf/1406.3676.pdf. [Accessed: 25-Oct-2018].
2. C. Xiong, S. Merity, and R. Socher , “Dynamic Memory Networks for Visual and Textual Question Answering,” 2016. [Online]. Available: http://proceedings.mlr.press/v48/xiong16.pdf. [Accessed: 25-Oct-2018].
3. D. Campos, “Microsoft MAchine Reading COmprehension Dataset,” *MS MARCO*. [Online]. Available: http://www.msmarco.org/. [Accessed: 25-Oct-2018].
4. G. Neubig, “Neural Networks for NLP Attention,” *Phontron*. [Online]. Available: http://phontron.com/class/nn4nlp2017/assets/slides/nn4nlp-09-attention.pdf. [Accessed: 25-Oct-2018].
5. J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” *Stanford NLP Group*, Aug-2014. [Online]. Available: https://nlp.stanford.edu/projects/glove/. [Accessed: 25-Oct-2018]
6. K. Cho, “DeepMind Q&A Dataset,” *DMQA*, 2015. [Online]. Available: https://cs.nyu.edu/~kcho/DMQA/. [Accessed: 26-Oct-2018].
7. M. Iyyer, J. Boyd-Graber, L. Claudino, R. Socher, and H. Daum´e, “A Neural Network for Factoid Question Answering over Paragraphs,” *ACL Web*, 29-Oct-2014. [Online]. Available: http://www.aclweb.org/anthology/D14-1070. [Accessed: 25-Oct-2018].
8. M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, “Bi-Directional Attention Flow for Machine Comprehension,” *Arxiv*, 21-Jun-2018. [Online]. Available: https://arxiv.org/pdf/1611.01603.pdf. [Accessed: 25-Oct-2018].
9. N. Smith, M. Heilman, R. Hwa, S. Cohen, and K. Gimpel, “Question-Answer Dataset,” *CMU School of Computer Science*, 23-Aug-2013. [Online]. Available: https://www.cs.cmu.edu/~ark/QA-data/?fbclid=IwAR2Zk2lomzyL0VcagXZyy101VLFEg1jyAB4ewCp6SvQRSaGopTRs\_f6gZ8A. [Accessed: 25-Oct-2018].
10. S. Kindu and H. T. Ng, “A Question-Focused Multi-Factor Attention Network for Question Answering,” *Arxiv*, 25-Jan-2018. [Online]. Available: https://arxiv.org/pdf/1801.08290.pdf. [Accessed: 25-Oct-2018].
11. SQuAD2.0,” *The Stanford Question Answering Dataset*. [Online]. Available: https://rajpurkar.github.io/SQuAD-explorer/. [Accessed: 25-Oct-2018].