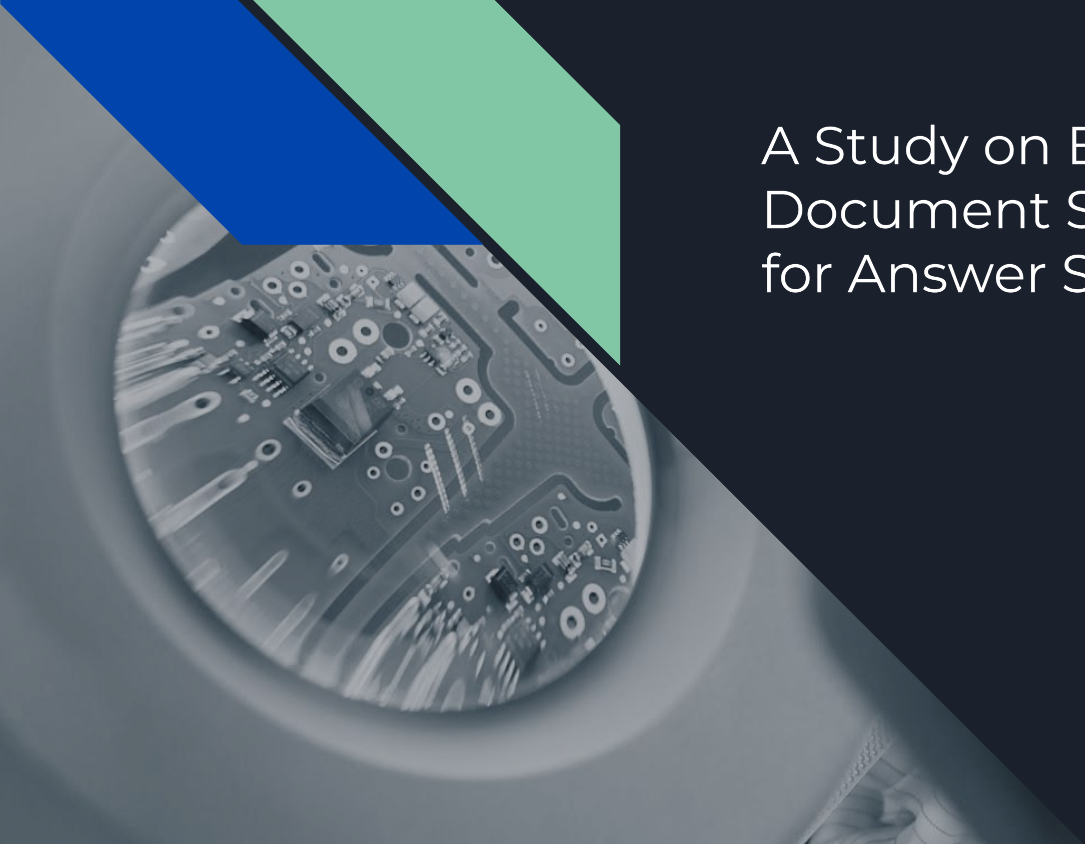

A Study on Efficiency, Accuracy and Document Structure for Answer Sentence Selection

Team 12:
Anantha Yadavalli
Aditya Kondai
Durga Koppisetti
Yashwanth Kottu

Introduction

- In this tech world we have a lot of models like BERT , ELMO, and GPT for "Question Answer Selection". The models like BERT take around 18 minutes for training. Another factor in using these compute heavy models for production is that we require powerful GPU's to achieve an acceptable service latency in the real time systems, especially if we are using any transformer based architecture.

- In this project, we are going to solve this time overhead, by using the simple approach described in the paper, while keeping the metric count much lesser than what is being provided by the standard baseline models like BERT.

Problem statement

The task of Answer Sentence Selection (AS2) can be formalized as follows: given a question ‘q’ and a set of answer sentence candidates, assign a score for each candidate such that the sentence receiving the highest score is the one that most likely contains the answer. These candidates are typically sentences either extracted from one or more documents preserving their natural order or retrieved by a search engine.

Dataset

We will stick with the wikiQA dataset as mentioned in the project doc as it is the dataset which most state of art models use for getting their metric.

Metrics

The most important metric that we are going to concentrate on this project is MRR (Mean Reciprocal Rank).

Mean Reciprocal Rank is a measure to evaluate systems that return a ranked list of answers to queries. For a single query, the reciprocal rank is ***1/rank*** where rank is the position of the highest-ranked answer (1,2,3,…,N answers returned in a query). If no correct answer was returned in the query, then the reciprocal rank is 0.

For multiple queries Q, the Mean Reciprocal Rank is the mean of the Q reciprocal ranks.

Problem with Current Approach

1. The primary source of inefficiency is, unfortunately, the contextual embedding, e.g., language models produced by Transformer networks or other methods such as ELMo.

These introduce at least one order of magnitude more parameters in the AS2 models than standard models based on CNNs or LSTMs.

2. The other significant source of inefficiency is the attention mechanism used in mainstream models.

Our Proposal

We propose to:

1. Build an encoder able to capture the relation between the question and each of its candidates, i.e, using an attention mechanism.

2. Preserve the structure of the sentences in the original rank, which is the original order of the sentences in the document or in the rank retrieved by the search engine.

Implementation

1. For the attention mechanism, we propose ***Cosinet***, a network that uses a sort of static attention, given by the cosine similarity between the embedding representation of the question and answer words.

2. We use an additional layer constituted by a bidirectional recurrent neural network (BiRNN), which is fed with the representation of question/answer candidate pairs. The latter are joint representations of the question and the answer obtained by the question-answer encoder.

Model

1. Cosinet :

It has two components:

★ Word-relatedness encoder: To encode the word-relatedness information, we first map the words in the question and the answer to their respective word embeddings. We then perform comparisons between all the embeddings in the question w_i^q and all the embedding of the answer w_j^c using the cosine similarity,

$$r_{i,j} \;=\; \left(w_i^q .\, w_j^c\right) \,/\, \left(||\, w_i^q||.\, ||\, w_j^c||\right)$$

We take the maximum relatedness score between its embedding and each word embedding of the candidate.

$$r_i \;=\; \max_j(r_{i,j})$$

The same process is performed for each word in the answer. This value represents how much a word is similar to the most similar word in the other text. The value is concatenated to the word embedding of the question, i.e.,

$$\hat{w}_i^q = [w_i^q; r_i]$$

A similar procedure is applied to the candidate, which is used to obtain the vector, i.e.,

$$\hat{w}_j^c = [w_j^c; r_j]$$

- Question-candidate encoder

  The question qe and candidate ce are then combined using their point-wise multiplication and their difference, i.e., $qc_e = [q_e . c_e \; ; \; q_e - c_e]$

2. We use a Bidirectional Recurrent Neural Network(Bi-RNN) applied on top of the qc_e representations for each c_i of a given question q, to leverage the global structure of the rank. The resulting contextual representations $q\hat{c}_e$ are passed to the feed-forward network.

Results

1. This models achieves better results than other efficient approaches, but
   a. The lack of contextual embeddings prevents the model from learning more complex functions, leading to lower results than the very expensive solutions.
   b. We partially solve this problem by using our joint model(***Bi-RNN + Cosinet***), which significantly improves the accuracy of efficient methods.

2. The word-relatedness encoder can replace the standard attention to enhance the speed of the fast attention-based approaches, resulting in a fast and accurate network in the class of fast methods.

Thank You!!!

- Team 12