A PROJECT REPORT

On

Natural Language Processing CPS 592-14

Submitted in fulfillment of the requirement of
University of Dayton for the Degree of
Master’s
In
Computer Science

Submitted By

Bhrugvish Vakil

Sravani Kanuri

Supervisor

Prof. Dr. Saeedeh Shekarpour

Department of Computer Science COLLEGE OF ARTS &SCIENCE Dayton -
45469 UNIVERSITY OFDAYTON Academic Year Spring 2020

2) Introduction (Motivation and Research Question):

We are surrounded by massive amounts of information in full-text documents i.e., web. Usually, we are interested in knowing answer to our question without having to read the entire document. QA systems are useful in retrieving useful information from the web and providing insights. The QA process can be broken into two parts:

INFORMATION RETRIEVAL:
Finding the document containing the answer to the question.

READING COMPREHENSION:
Given the document find the answer to the question.

To read and comprehend the human languages are challenging tasks for the machines, which requires that the understanding of natural languages and the ability to do reasoning over various clues. Machine reading comprehension which attempts to enable machines to answer questions after reading a passage or a set of passages, attracts great attentions from both research and industry communities in recent years. Below is an example of the machine reading comprehension task.

The key part in MRC task lies in how to incorporate questions information into passage, in which attention. mechanism is most widely used. Thus, massive improvement has been raised since the attention mechanism was introduced to MRC. They develop a class of attention based deep neural networks that learn to read real documents and answer complex questions with minimal prior knowledge of language structure. Pointer networks with one attention step to predict the blanking out entities. Iterative alternating attention mechanism to tackle machine comprehension tasks. Stochastic answer network (SAN) is a simple yet robust mechanism that simulates multi-step reasoning in machine reading comprehension.

3) Literature Review:

There is a long history of pre-training general language representations, and we briefly review the most widely used approaches in this section.

3.1 Unsupervised Feature-based Approaches

Learning widely applicable representations of words has been an active area of research for decades, including non-neural and neural methods. Pre-trained word embeddings are an integral part of modern NLP systems, offering significant improvements over embeddings learned from scratch. To pre- train word embedding vectors, left-to-right language modeling objectives have been used, as well as objectives to dis- criminate correct from incorrect words in left and right context. These approaches have been generalized to coarser granularities, such as sentence embed- dings or paragraph embeddings. To train sentence representations, prior work has used objectives to rank candidate next sentences, left-to-right generation of next sentence words given a representation of the previous sentence, or denoising auto- encoder derived objectives.

ELMo and it generalize traditional word embedding re- search along a different dimension. They extract *context-sensitive* features from a left-to-right and a right-to-left language model. The contextual representation of each token is the concatenation of the left-to-right and right-to-left representations. When integrating contextual word embeddings with existing task-specific architectures, ELMo advances the state of the art for several major NLP benchmarks, including question answering, sentiment analysis, and named entity proposed learning contextual representations through a task to predict a single word from both left and right context using LSTMs. Similar to ELMo, their model is feature-based and not deeply bidirectional shows that the cloze task can be used to improve the robustness of text generation models.

3.2 Unsupervised Fine-tuning Approaches

As with the feature-based approaches, the first works in this direction only pre-trained word embedding parameters from unlabeled text.

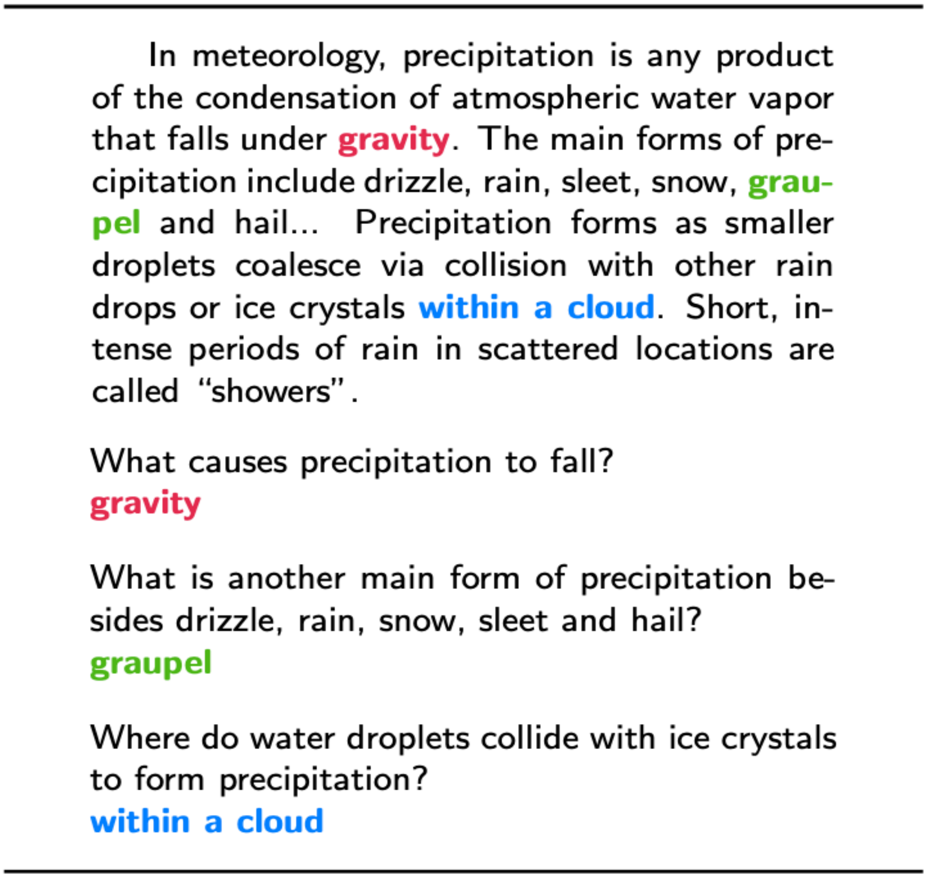
More recently, sentence or document encoders which produce contextual token representations have been pre-trained from unlabeled text and fine-tuned for a supervised downstream task. The advantage of these approaches is that few parameters need to be learned from scratch. At least partly due to this advantage, OpenAI GPT achieved previously state-of-the-art results on many sentence- level tasks from the GLUE benchmark. Left-to-right language modeling and autoencoder objectives have been used for pre-training such models.

3.3 Transfer Learning from Supervised Data

There has also been work showing effective transfer from supervised tasks with large datasets, such as natural language inference and machine translation. Computer vision research has also demonstrated the importance of transfer learning from large pre-trained models, where an effective recipe is to fine-tune models pre-trained with ImageNet.

4) Corpus Acquisition:

We have used the Stanford Question Answering Dataset (SQuAD), a new reading comprehension dataset consisting of 100,000+ questions posed by crowd workers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage. We analyze the dataset to understand the types of reasoning required to answer the questions, leaning heavily on dependency and constituency trees. The beauty of this dataset is that, the answer to the questions can be found somewhere in the refenced text. Every training and test sample in dataset consists of a reference passage and a Question.



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".

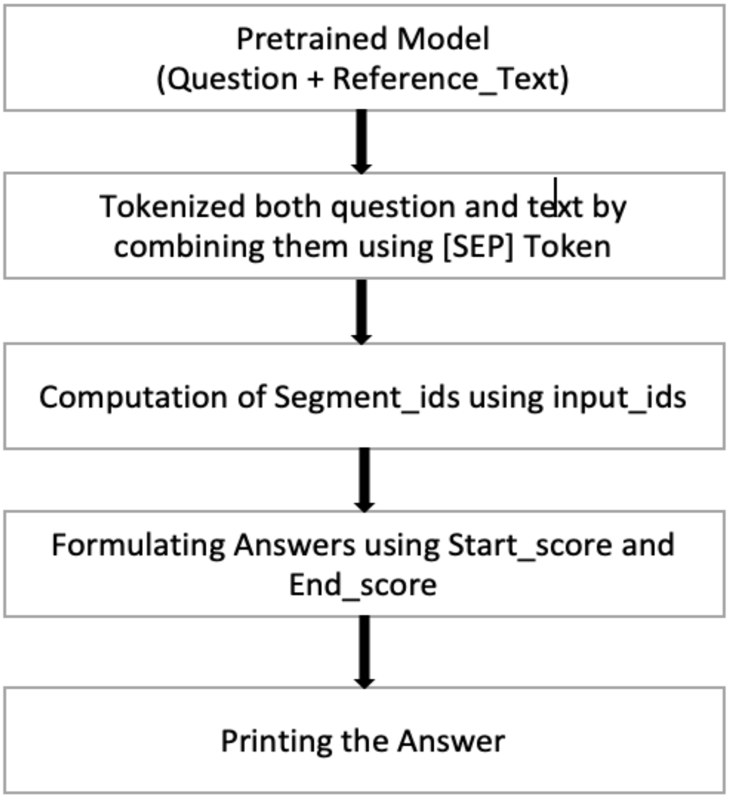
What causes precipitation to fall?
gravity

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.

5) Approach:


Pretrained Model
(Question + Reference_Text)
Tokenized both question and text by combining them using [SEP] Token
Computation of Segment_ids using input_ids
Formulating Answers using Start_score and End_score
Printing the Answer

First of all, we are going to be employing a pre trained model within which we are going to pass our passage together with Question. supported model for question will predict the referenced text and locate 100 tokens nearby. Once we have got the reference text and question, we pass that to Bert tokenizer within which both question and referenced text are going to be combined using SEP (Special token, Line Breaker) token. Then we've got to compute segment_ids using input_ids and SEP token location as a reference and convert a replacement list of 0's and 1's (0 for Questions and 1 for reference passage). Then the major task left is of formulating the answers and to calculate Start and End score. then print the solution.

6) Implementation details:

Software Required

- Python = 3.6
- Pandas
- Matplotlib
- Seaborn
- textwrap3
- pyttsx3
- pyTorch
- SpeechRecognition
- Time

Models:

- **Hugging face transformer Library**

  Transformers library provides general-purpose architectures (BERT) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32 plus pretrained models in 100 plus languages and deep interoperability between TensorFlow 2.0 and PyTorch.

  Features

  - As easy to use as pytorch-transformers
  - As powerful and concise as Keras
  - High performance on NLU and NLG tasks
  - Low barrier to entry for educators and practitioners

  State-of-the-art NLP for everyone:

  - Deep learning researchers
  - Hands-on practitioners
  - AI/ML/NLP teachers and educators

- **BRET Model (Pre-training of Deep Bidirectional Transformers for Language Understanding):**

  It’s a bidirectional transformer pre-trained employing a combination of masked language modeling objective and next sentence prediction on an oversized corpus comprising the Toronto Book Corpus and Wikipedia.

  The hugging face library defines this special class for doing question answering BertForQuestionAnswering. The model we used is “bert-Large-uncased-whole-word-masking-finetuned-squad”. we've got used whole-word because rather than masking out individual tokens which can be sub words it always masks entire word.

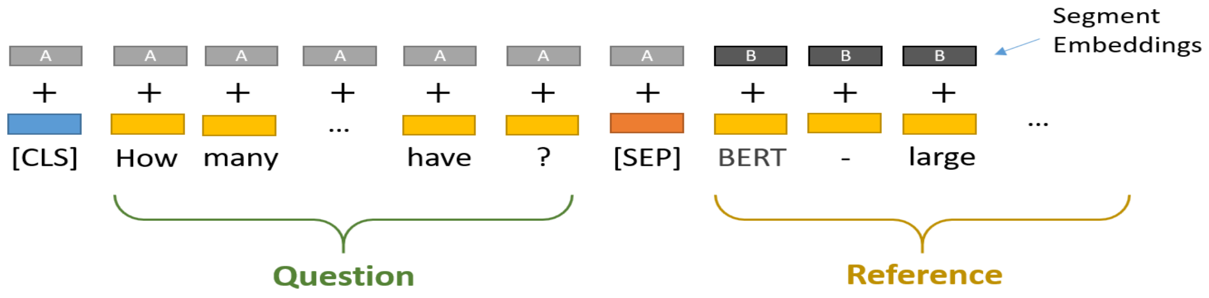
- **Bert Tokenizer**

  We have use Bert Tokenizer against both question and answer text (referenced text). We concatenate them together and place [SEP](Line breaker) special token between them to segment the data

- **SpeechRecognition:**

  Since our input will be either text or speech, we've used Speech Recognition library for speech as an input. this is often converting input speech into text and so will pass that text too bert tokenizer.

7) Experimental Setup:



Question: `How many parameters does BERT-large have?`

Reference Text: `BERT-large is really big... it has 24 layers and an embedding size of 1,024, for a total of 340M parameters! Altogether it is 1.34GB, so expect it to take a couple minutes to download to your Colab instance.`

Provided segment_ids to model take a look at those input_ids (input text which were given ids) and find the placement of SEP token and therefore the number of tokens before that are in segment_a and number of tokens at the moment are in segment_b.

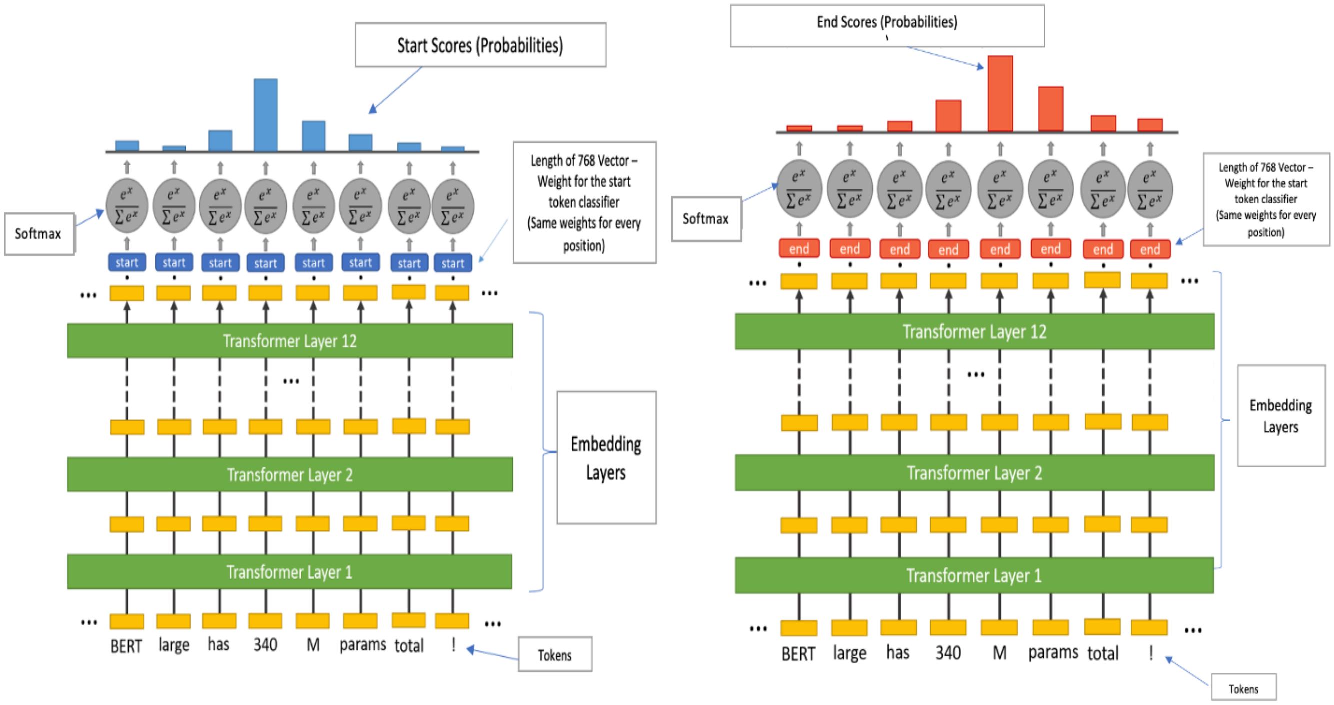
Segment_a = SEP [index]+1

Segment_b = input (length) – Segment_a

Then construct a listing of 0 and 1 (A and B) (as shown in Figure as Segment Embedding), 0(A) referred for Questions and 1(B) for referenced text. No padding is required because we aren't using batches as input instead, we are training entire referenced text.

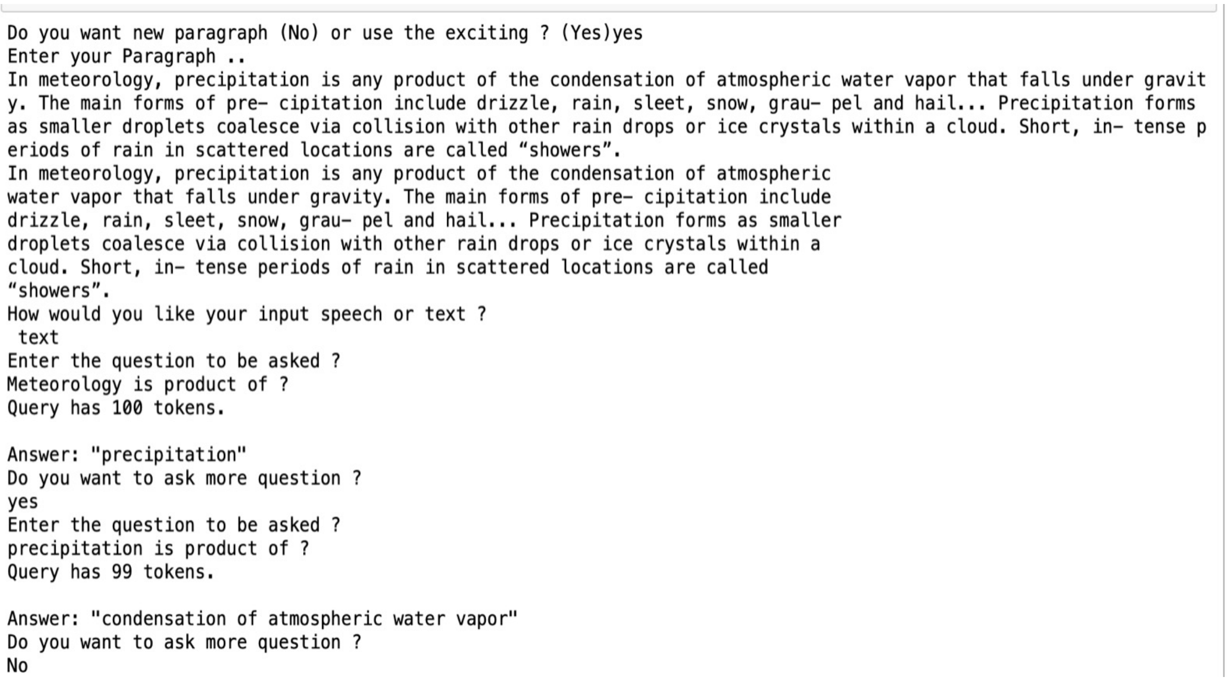
Feed input ids and segment ids into model for forwards process like calculation of start and end score. The token with highest start score would be start of answer and therefore the highest end score is going to be the tip of the solution.

Start and End Token Classifier:



Let say these are all the tokens from references text that are rephrased to suit better. Every embedding we're visiting multiply the ultimate embedding by this vector that are labeled as start (as shown above). Weights of start token classifier so will take dot between embedding and begin token classifier weights then we'll apply the SoftMax function on output. Probability distribution that sums put to 1. it'll predict the best probability.

8) Results:



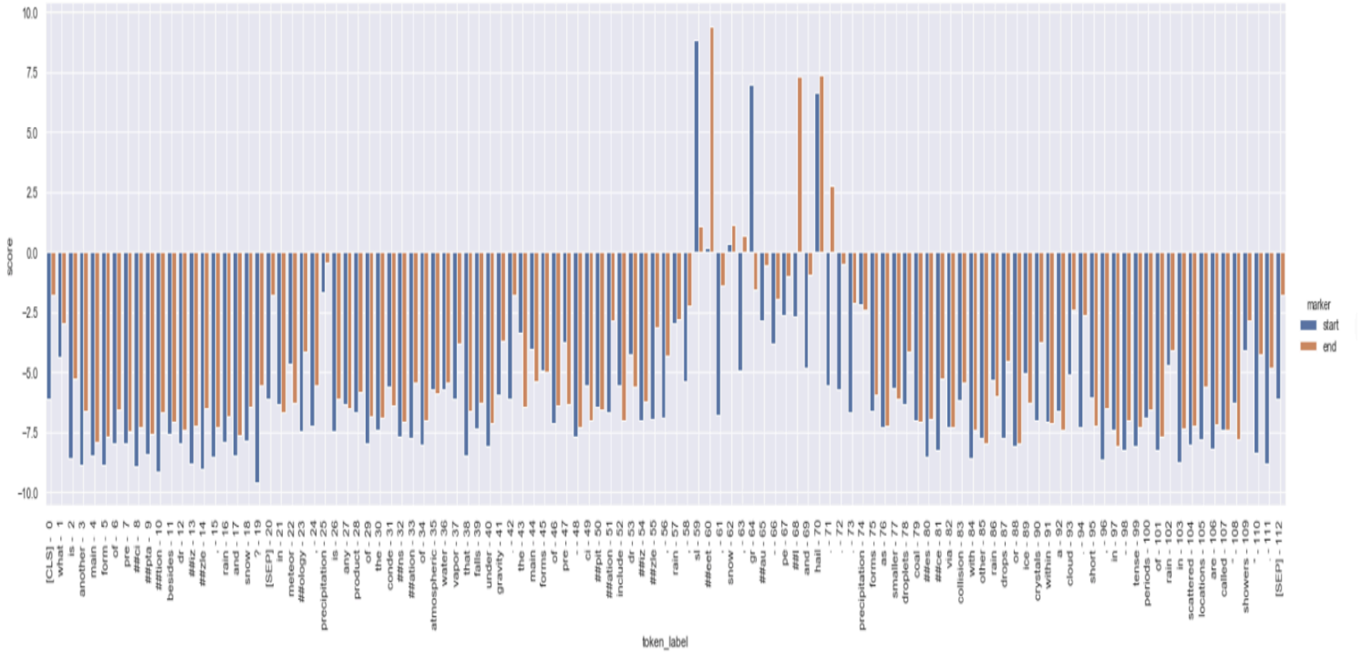
```
Do you want new paragraph (No) or use the exciting ? (Yes)yes
Enter your Paragraph ..
In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravit
y. The main forms of pre- cipitation include drizzle, rain, sleet, snow, grau- pel and hail... Precipitation forms
as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, in- tense p
eriods of rain in scattered locations are called “showers”.
In meteorology, precipitation is any product of the condensation of atmospheric
water vapor that falls under gravity. The main forms of pre- cipitation include
drizzle, rain, sleet, snow, grau- pel and hail... Precipitation forms as smaller
droplets coalesce via collision with other rain drops or ice crystals within a
cloud. Short, in- tense periods of rain in scattered locations are called
“showers”.
How would you like your input speech or text ?
 text
Enter the question to be asked ?
Meteorology is product of ?
Query has 100 tokens.

Answer: "precipitation"
Do you want to ask more question ?
yes
Enter the question to be asked ?
precipitation is product of ?
Query has 99 tokens.

Answer: "condensation of atmospheric water vapor"
Do you want to ask more question ?
No
```

First of all, a question will prompt to user “Do you want new paragraph (No) or use the exciting? (yes)”. If user says No as a solution, then the program will use inbuild comprehension else it will use user defined passage. Once the passage is defined then user would lean choice between text or speech for input. The question answering system continues until user answer the question does one want to ask more question? “as, “No”. This goes same with speech too, in speech machine will first attempt to hear question then print the question asked so user can keep a track weather the machine is ready to properly predict what he's trying to mention so will print the solution and speak out together with it.

Start and End Score combine:



The answer is formulated using the best probability of start and end score. the beginning of a solution is from the foremost start score and end if where the top score is maximum.

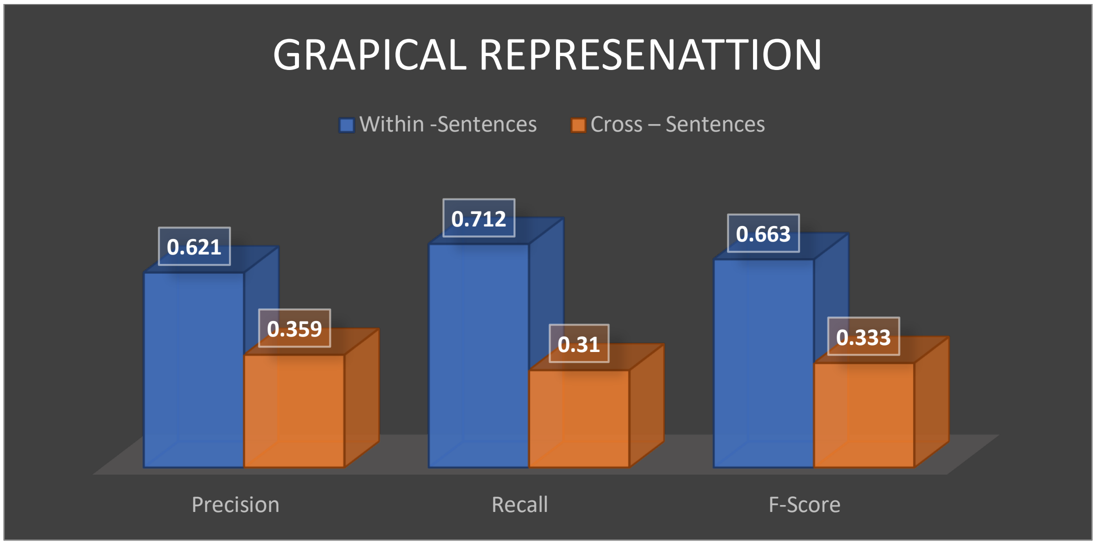
9) Discussion and Analysis of the Results:

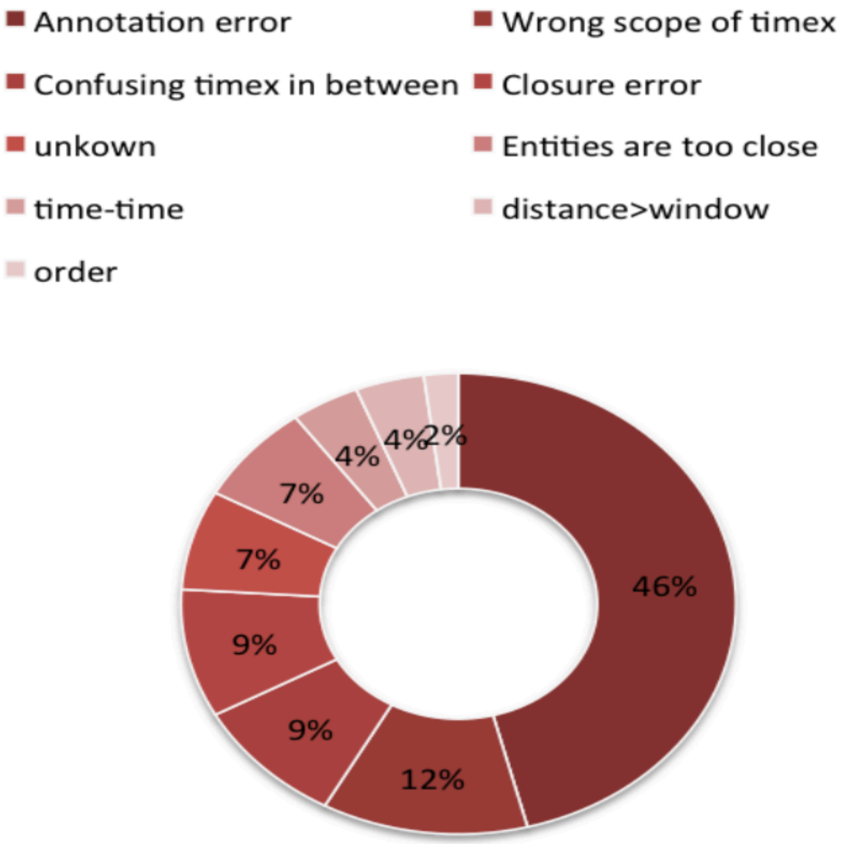
Different Models

Models	Precision	Recall	F1-Score
Lin et al	0.692	0.576	0.629
Galvan et al	0.983	0.463	0.629
Bi-LSTM	0.712	0.490	0.581
BERT	0.699	0.625	0.663

Error Analysis :

Categories	Precision	Recall	F-Score
Within -Sentences	0.621	0.712	0.663
Cross – Sentences	0.359	0.310	0.333





We sampled 100 errors evenly distributed over four categories: within-sentence false positives (FP), within-sentence false negatives (FN), cross- sentence FPs, and cross- sentence FNs. The sources of errors are summarized in fig.) "Annotation error" (46%) – errors in the gold annotations; 2) "Wrong scope of Timex" (12%) – the main reason for FP predictions, especially for cross-sentence ones (10%). The system fails to identify the subtle change of the Timex scope and incorrectly links an event to it; 3) "Confusing Timex in between" (9%) – there is another time expression occurring between the two arguments, thus the system incorrectly infers the scope of the time expression; 4) "Closure error" (9%); 5) "Unknown" (7%) – errors for which we could not provide a plausible explanation ; 6) "Entities are too close" (7%) – the two entities in question are too close to each other, thus limiting the context for correct reasoning. Prior knowledge would be helpful for these short-distance relations; 7) "Time-time" (4%) – the system generates time- time relations which are oftentimes FPs because gold time-time annotations are scarce ; 8) "Distance > window" (4%) – the distance between the two entities in question is bigger than the window size, resulting in cross-sentence FNs ; 9) "Order" (2%) – the system incorrectly extracts the order of the relation arguments.

10) Future Work:

- Formulation of nice response
- Answer questions where it has to combine from multiple sentence in order to answer it.
- Using lighter version of Bert. Like (ALBERT)

11) The contribution of team members:

Literature Review and Corpus Acquisition by Sravani Kanuri

Coding, Error Analysis and Results by Bhrugvish Vakil