CSE 576 Natural Language Processing

Project Phase 2 – Automated Data Creation

Jayasurya Sevalur Mahendran | ASU ID: 1217399443 | E-mail ID: jsevalur@asu.edu

Task Description

Since Semantic Information Availability (SIA) does not have a dedicated dataset for itself, the task is to use the publicly available dataset to create answer candidates and assign a SIA for each answer candidate and create a diverse dataset for SIA. For this purpose, I had chosen the **multi-hop question-answering QASC dataset**.

Structure and Details of the Dataset

Here is the link to the paper describing the Dataset https://arxiv.org/pdf/1910.11473.pdf and the dataset can be found at https://github.com/allenai/qasc.

QASC is a multi-hop question-answering dataset with a focus on sentence composition. It consists of 9,980 8-way multiple-choice questions about grade school science. It comes with a corpus of 17M sentences. It requires retrieving facts from a large corpus and composing them to answer a multiple-choice question.

It the dataset is split as:

Training: 8134 questions
Validation: 926 questions
Testing: 920 questions

Each **train, validation example** consists of a question with **8 answers candidates** as options, followed by **two supporting facts** named as f_S and f_L and **a composed fact** f_C , composed from f_S and f_L using the broad knowledge that is used to answer the question.

All questions in QASC are human-authored, obtained via a multi-step crowdsourcing process. To better enable development of both the reasoning and retrieval models, the pair of facts that were composed to create the question are also provided. These annotations are used to develop a novel two-step retrieval technique that uses question-relevant facts to guide a second retrieval step.

The corpus consists of 17 Million facts.

Pre-Processing the Dataset

Initially the training and the validation set, which are in 'jsonl' format are loaded into the script.

```
"id": "3UWN2HHPUY4HEFIDUE0DFN4T2J5SNS".
"question": {
  "stem": "What can trigger immune response?",
  "choices": [
    { "label": "A", "text": "harmful substances" },
    { "label": "B", "text": "Transplanted organs" },
    { "label": "C", "text": "desire" },
     "label": "D", "text": "an area swollen with pus" },
     "label": "E", "text": "death" },
     "label": "F", "text": "pain" },
    { "label": "G", "text": "colors of the spectrum" },
    { "label": "H", "text": "Contaminated wounds" }
},
"answerKev": "B".
"fact1": "Antigens are found on cancer cells and the cells of transplanted organs.",
"fact2": "Anything that can trigger an immune response is called an antigen.",
"combinedfact": "transplanted organs can trigger an immune response"
```

Figure 1. Structure of a Question in training and Validation Set

Each question, answer candidate, fact1, fact2 and combined fact are extracted and are appended to corresponding list. The answer candidate corresponding to the answer key is considered as the exact answer.

For every answer candidate a fact is retrieved from the corpus using the Anserini Information Retrieval toolkit built on Lucene. Also, the fact1, fact2 and the combined fact are pushed into answer candidate, hence every Question(query) has 11 corresponding answer candidates for training and validation set.

Pyserini

Pyserini is a python interface to the Anserini IR toolkit built on Lucene. **Okapi BM25** (best matching) is a ranking function used by search engines to estimate the relevance of documents to a given search query. It is based on the probabilistic retrieval framework.

BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document. It is a family of scoring functions with slightly different components and parameters. One of the most prominent instantiations of the function is as follows.

Given a query Q, containing keywords $q_{1},...,q_{n}$, the BM25 score of a document D is:

$$ext{score}(D,Q) = \sum_{i=1}^n ext{IDF}(q_i) \cdot rac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot \left(1 - b + b \cdot rac{|D|}{ ext{avgdl}}
ight)}$$

where $\mathbf{f}(\mathbf{q_i}, \mathbf{D})$ is $\mathbf{q_i}$'s term frequency in the document D, $|\mathbf{D}|$ is the length of the document D in words, and **avgdl** is the

average document length in the text collection from which documents are drawn. $\mathbf{k1}$ and \mathbf{b} are free parameters, usually chosen, in absence of an advanced optimization as K1belongs to [1.2,2.0] and b-0.75. is the **IDF** (inverse document frequency) weight of the query term. It is usually computed as:

$$ext{IDF}(q_i) = \ln(rac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1)$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i .

Processed Data frame for both train and validation set looks like this:

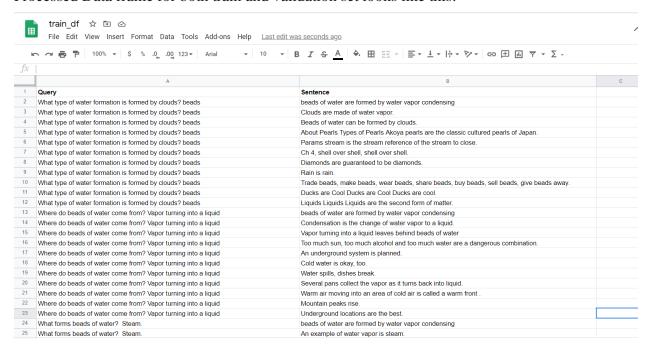


Figure 2. Data frame for training set after extracting facts from corpus

Where each query is a combination Question and Exact Answer <Q+E_A>, Sentence are nothing but the fact1, fact2, combined fact and the fact retrieved from the corpus corresponding to each answer candidate from 8 options. Hence every question has 11 answer candidates.

Semantic Information Availability (SIA) Score Generation

Since there are not any existing Natural Language Processing models that can directly generate SIA scores, hence I had suggested to utilize existing State of the art (SOTA) Semantic Textual Similarity (STS) model to generate the STS scores.

Now as STS scores are generally in range [0, 5] however SIA scores fall in range [0, 4], the generated STS scores are converted to the range [0, 4] and are labelled as the SIA scores (gold Label) for the dataset.

Web STS Bert

Web STS Bert is an easy-to-use interface to fine-tune BERT models for computing semantic similarity, it contains and interface to fine-tuned BERT based semantic text similarity models. It modifies the pytorch-transformers by abstracting away all the research benchmarking code for ease of real-world applicability.

Web STS BERT was pretrained on the STS-B dataset and has a Pearson correlation of 0.893.

The generated scores for each Query, Sentence pair from the training and the validation set is in the range [0, 5] they converted to [0,4].

Experimenting with Commercially available STS Models

The following models were taken into consideration before sticking to Web Bert Model, to identify the best performing model for QASC Dataset.

- 1. ALBERT
- 2. RoBERTa
- Custom BERT model pre trained on STS-D dataset with a Linear output Layer to predict STS score.
- 4. Clinical STS BERT
- 5. Web STS BERT

Comparison of Scores Generated by various considered Model

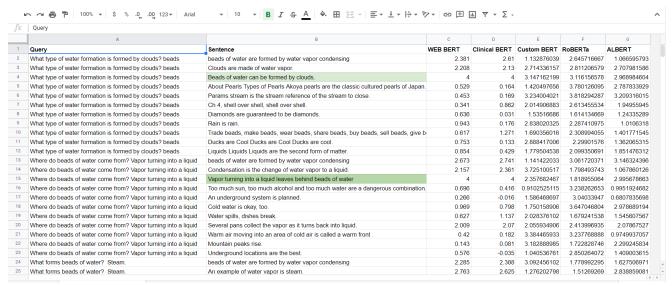


Figure 3. Comparison of scores predicted by different Models

Upon close observation I could refer that **Web BERT and Clinical BERT** were performing for more better than all other models, however there where instances where Clinical BERT

was calculating negative scores. Web BERT Model showed consistency over then entire Dataset.

Manual Evaluation of the Predicted Scores

20 randomly picked query and sentence pair from the train set, and were manually evaluated to calculate the STS score by extracting the key phrase in the question and corresponding terms were identified in the sentence (Answer Candidate). The predicted scores were similar to that of the scores predicted by the Web STS model.

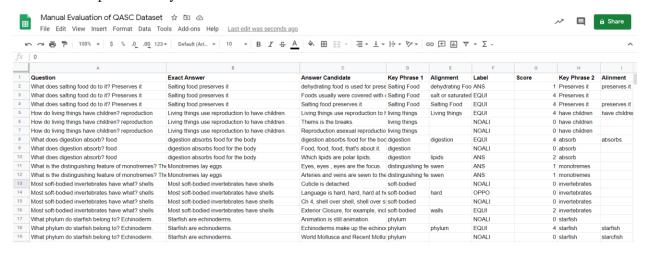


Figure 4. Manual Evaluation

Links:

Entire Team's submission can be found at: https://github.com/JainSahit/NLP576-SIA

All the files can be found at:

https://drive.google.com/drive/folders/1HCH6OYs6U56eNR5C03J_pOZW1TElDOYd?usp=sharing

Preprocessed Dataset and Results
 https://drive.google.com/drive/folders/1HCH6OYs6U56eNR5C03J_pOZW1TEIDOYd?u

Python notebook links

sp=sharing

- 1. Pyserini_and_Data_PreProcessing.ipynb https://colab.research.google.com/drive/1yKHTbOUMYdRdb0_N6fFlkoJU9h_hwPan?us p=sharing
- 2. SIA-Scores-Generation-Using-WEB-BERTandClinical-BERT.ipynb https://colab.research.google.com/drive/1ndFdUtDpT_Wh-H5kvhTiv0OoIh1MAp9A?usp=sharing
- 3. Generating-Dataset-Using-Preprocessed-Huggingface-Models.ipynb https://colab.research.google.com/drive/1117iKWm6Vju8yVPFHq1s_nxpkVCbS0Cq?usp=sharing