Question Answering System

Course:NLP

Team:Perseverance

Akash Bharadwaj Baraka Ramesh(axb172930)

Hasanth Sista(hxs171730)

**Abstract**:

Question answering (QA) is a well-researched problem in NLP. QA has application in a wide variety of tasks, such as information retrieval and entity extraction. Information retrieval is one of the NLP tasks of processing the corpus and answering fact-based questions. The fact-based questions range from “who”, “when”, “where”, “what” based on the given corpus. These systems are mainly divided into two domains.

1. Closed domain, where we are given a limited corpus and the questions are limited to the information given in that corpus.
2. Open domain, where the questions can be anything from the world knowledge.

**Objective:**

The objective of this project is to provide answers to fact-based questions that are given, based on the Wikipedia documents provided.

Input:

1. 30 Wikipedia articles:
   * 1. 10 Wikipedia articles related to Organizations
     2. 10 Wikipedia articles related to Persons
     3. 10 Wikipedia articles related to Locations
2. 20 questions based on the above Wikipedia files.

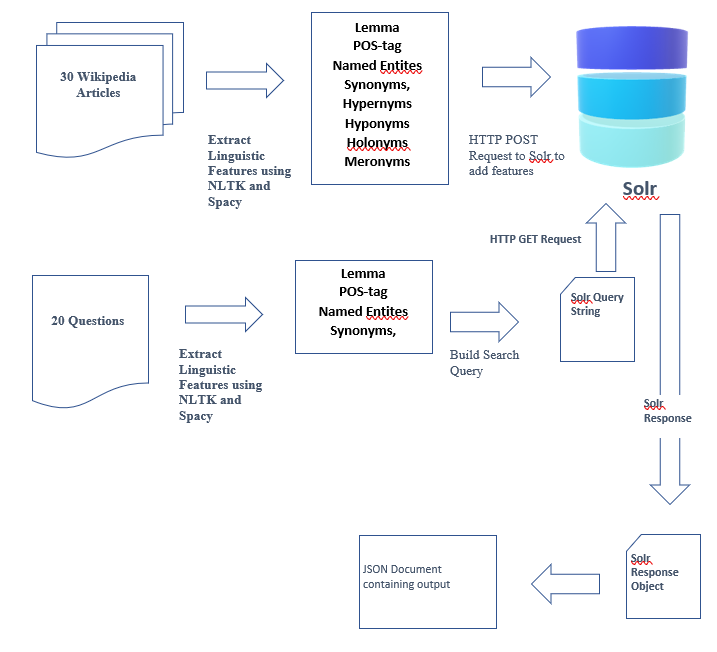
Output/Goal:

1. Exact answer phrase(s) to the question.
2. Supporting sentences in the Wikipedia documents.
3. Supporting Wikipedia document(s) names.

Technology and Packages:

1. Python 3.6 for executing the python programs used to realize the goal.
2. Spacy – to extract linguistic features.
3. NLTK – to extract linguistic features.
4. Solr – is highly reliable, scalable and fault tolerant, providing distributed indexing, replication and load-balanced querying

Architecture/Design-Flow:



**Approach**:

**Task 1**:

Parsing 30 Wikipedia document to generate linguistic features. The following steps are undertaken:

* First parse the document in to sentences using NLTK’s sent\_tokenize (tried spacy’s sents but dint work for some cases).
* Next parse the obtained sentences using spacy to generate linguistic features. The generated features are:
  + - * + Lemma – Base form of the token
        + POS­ – Part of speech tag
        + Named Entitles – Named entities for the tokens
        + Dependency – Syntactic dependency parse (relation)
        + Hypernyms
        + Hyponyms
        + Holonyms
        + Meronyms
* The output(Linguistic features) is stored in the Task1 folder for the respective Wikipedia files which have the same file name.

**Task 2:**

Used Solr elastic search index the linguistic features required to answer questions.

* Start the Solr server and create a core to index data.
* Storing the title of the documents, sentences, named entities, lemma, pos tags and dependency parse in Solr using pysolr to make HTTP POST Request.

**The image below depicts the stored data in Solr:**



**Task 3:**

Parse the given document of questions and output the result to a JSON document.

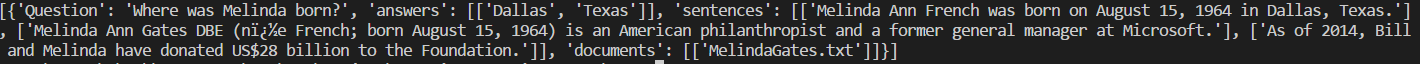
* Parse the questions using NLTK and spacy to generate a query string containing named entities, lemma and synonyms of the question and make a HTTP GET Request to Solr using pySolr.
* The obtained result object is used to create a JSON document containing answer phrase(s), relevant sentence(s), relevant document(s).

**Output**:

Sample Input Question:

**Where was Melinda born?**

Output for the sample input:



**Problems Encountered:**

* Trying to parse all Wikipedia files did not give expected results as the sentences were not parsed as expected. So, have used two encoding formats while parsing sentences.
* Initially tried to use spacy for parsing the sentences but it was not consistent in the parsing and tried NLTK’s sent\_tokenize which gave better results.
* To load spacy model takes a lot of time and were loading it each time to generate various linguistic features, therefore, the parsing of 30 Wikipedia files was taking more than 2 hours. Then initialized it once and made use of it across the application increasing the speed of execution.
* Figuring out Solr took a lot of searching on the web.

**Pending Issues:**

* Still the encoding formats (utf8, ISO-8859-1 and latin-1) does not solve all the problems of parsing.
* Spacy has issues with named entities.

**Potential Improvements:**

* Pronoun resolution can be done.