PROJECT REPORT

IMPLEMENTATION OF SHALLOW AND DEEPER NLP PIPELINE

Submitted by
Nanditha Valsaraj(nxv160930)
Ramya Elangovan(rxe150030)
Shreya Vishwanath Rao(sxr169330)

Description

To implement a semantic search application using a keyword based strategy and an improved strategy using NLP feature and techniques. The main objective is to implement a shallow NLP pipeline and a deeper NLP pipeline to perform semantic search index creation.

Proposed Solution:

Task1: To download a corpus with 1000 articles and 100,000 words from the website http://www.nltk.org/nltk_data/.
Our corpus contains 1003 rural based news articles with 132032 words. It has been derived from the Australian Broadcasting Commission's corpora.

Task2: To implement a shallow NLP pipeline by performing below steps

Keyword search index creation:

- Split the input file into articles and segment them into sentences
- Tokenize each sentences into words to create a word vector
- Index Each word vector per sentence into search index in SOLR

Query parsing and search by:

- Segment user's input into sentences and tokenize each sentences into words
- Run the search query using word vector against the sentence word vector present in the SOLR search index created from the corpus
- Evaluate results using 10 search queries for the top 10 returned sentences

Task3: This phase included implementation of a deeper NLP pipeline. The main purpose of this pipeline is to do the following:

Keyword search index creation:

- Split the input file into articles and segment them into sentences
- The Tokenize each sentences into words
- Lemmatize the words to extract lemmas as features
- Stem the words to extract stemmed words as features
- POS tag the words to extract POS tag as features using pos_tag.
- Syntactically parse each sentence to extract dependency parse relations
- Extract Hypernyms, Hyponyms, Meronyms, Holonyms of each word as features using WordNet
- Index each of the above features as separate search fields in SOLR

Query parsing and search by:

- Run above deeper NLP on user's input to extract search query features
- Run Search against separate or combination of search index fields created from corpus

Task4: This phase included implementation of a shallow NLP pipeline to improve the results using combination of deeper NLP pipeline features. The main purpose of this pipeline is to do the following:

Keyword search index creation:

- Split the input file into articles and segment them into sentences
- The Tokenize each sentences into words
- Remove the stop words
- Lemmatize the words to extract lemmas as features
- POS tag the words to extract POS tag as features using pos_tag.
- Syntactically parse each sentence to extract dependency parse relations
- Extract Hypernyms of each word as features using WordNet
- Index each of the above features as separate search fields in SOLR

Query parsing and search by:

- Run above deeper NLP on user's input to extract search query features
- Run Search against separate or combination of search index fields created from corpus

Implementation Details:

Task1: This phase included creating a corpus with 1003 articles and 100,000 words from the website http://www.nltk.org/nltk_data/

Task2: This phase included implementation of a shallow NLP pipeline. The main purpose of this pipeline is to do the following:

Keyword search index creation:

- The input file is split into articles
- News articles were segmented into sentences using sent_tokenize of NLTK
- The sentences were tokenized into words using WordPunctTokenizer of NLTK to create a word vector
- Each word vector per sentence were indexed into search index in SOLR using **pysolr**

Query parsing and search by:

- Segmenting user's input into sentences using sent_tokenize of NLTK
- Tokenizing the sentences into words using WordPunctTokenizer of NLTK
- Running the search query word vector against the sentence word vector present in the SOLR search index created from the corpus
- The results were evaluated using 10 search queries for the top 10 returned sentences

Task3: This phase included implementation of a deeper NLP pipeline. The main purpose of this pipeline is to do the following:

Keyword search index creation:

- The input file is split into articles
- News articles were segemented into sentences using sent_tokenize of NLTK
- The sentences were tokenized into words using WordPunctTokenizer of NLTK
- The words were lemmatized to extract lemmas as features using WordNetLemmatizer of NLTK
- The words were stemmed to extract stemmed words as features using PorterStemmer of NLTK
- The words were POS tagged to extract POS tag as features using pos_tag.
- Each sentence were syntactically parsed to extract dependency parse relations using StanfordDependencyParser
- Hypernyms, Hyponyms, Meronyms, Holonyms of each words were extracted as features using WordNet
- Each of the above features extracted were indexed as separate search fields in SOLR using pysolr

Query parsing and search by:

- Above deeper NLP were run on user's input to extract search query features
- Search was run against separate or combination of search index fields created from corpus

Task4: This phase included implementation of a shallow NLP pipeline to improve the results using combination of deeper NLP pipeline features. The main purpose of this pipeline is to do the following:

Keyword search index creation:

- The input file is split into articles
- News articles were segemented into sentences using sent_tokenize of NLTK
- The sentences were tokenized into words using WordPunctTokenizer of NLTK
- Stop words were removed using stop words provided by NLTK
- The words were lemmatized to extract lemmas as features using WordNetLemmatizer of NLTK
- The words were stemmed to extract stemmed words as features using PorterStemmer of NLTK
- The words were POS tagged to extract POS tag as features using pos_tag.
- Each sentence were syntactically parsed to extract dependency parse relations using StanfordDependencyParser
- Hypernyms, Hyponyms, Meronyms, Holonyms of each words were extracted as features using WordNet
- Each of the above features extracted were indexed as separate search fields in SOLR

Query parsing and search by:

- Above deeper NLP were run on user's input to extract search query features
- Search was run against separate or combination of search index fields created from corpus

Programming tools:

Python(V 3.6): All of the tasks are implemented using python.

Pysolr (V 3.5): A Lightweight Python wrapper for Apache Solr that provides an interface that queries the server and returns result based on query.

SOLR(V 7.1.0): An open source enterprise search platform, that includes features like full-text search, hit highlighting, faceted search, real-time indexing, dynamic clustering, database integration, NoSQL features and rich document (e.g., Word, PDF) handling.

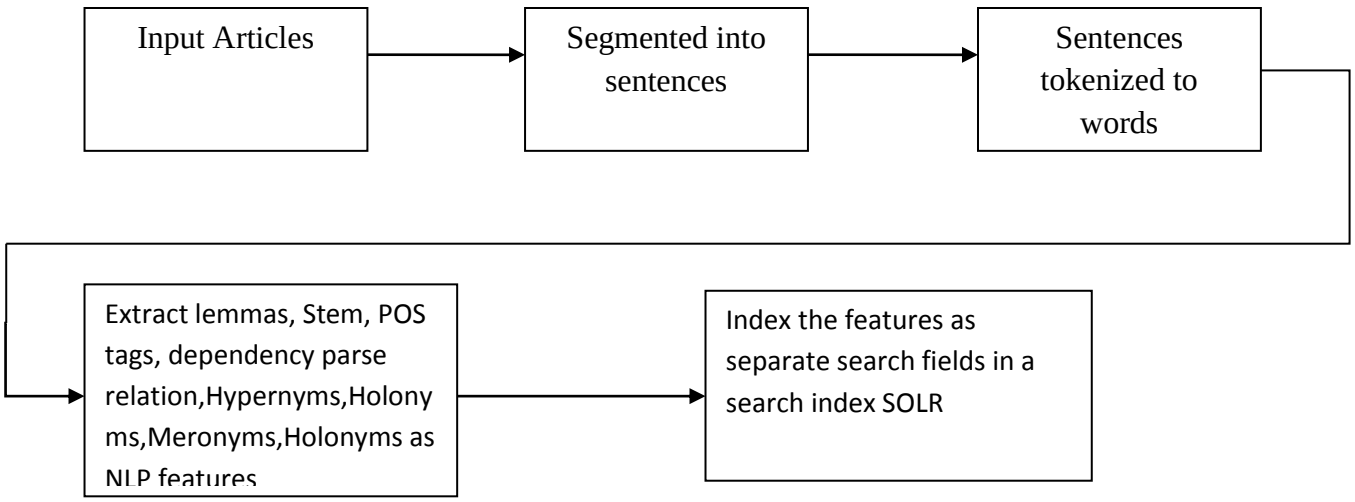
NLTK (V 3.2.5): Processing libraries used

nltk.corpus, nltk.tokenize, nltk.stem.wordnet, nltk.stem, nltk.wsd, nltk.corpus

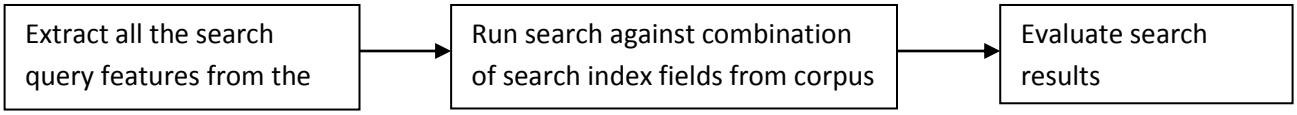
Stanford NLP(Verson 3.8): stanford-corenlp, stanford-parser

Architectural Diagram:

Search Index Creation:



Query Parsing and Search:



Results and Error Analysis:

All the results are evaluated by calculating **MRR**. The mean reciprocal rank is a statistic measure for evaluating any process that produces a list of possible responses to a sample of queries, ordered by probability of correctness.

Task1: Created corpus with 1003 articles and 100,000 word

Task2:

Following are the list of sentences that worked as the words of the sentences and article sentences matched

- Community tries to battle floods due to cyclone (Rank = 3)
- Decrease in production of wheat (Rank = 2)
- Virus affects Australian tomato production (Rank = 4)
- What has been affecting tomatoes production? (Rank = 6)
- With which company did potato farmers have an issue? (Rank =2)
- International demand for wheat processors have decreased with decreasing export of wheat (Rank = 5)
- Water shortage blamed for lower export licenses issued (Rank =3)
- Drought in NSW gets worse (Rank=3)

Sentences that didn't work:

- Drought blamed for reduced grain export
- Unnamed retailers help increase sale. It is said that lack of producer-consumer requirements knowledge is affecting sales. Seasonal conditions make it hard for buyers to select good product

Above sentences has worked after modifying them as below

- Drought blamed for lower grain export (Rank = 4)
- Unnamed retailers help increase wool sales. It is said that lack of producer-consumer requirements knowledge is affecting sales. Seasonal conditions make it hard for buyers to select good quality wool (Rank = 1)

MRR = ((1/3) + (1/2) + (1/4) + (1/6) + (1/2) + (1/5) + (1/3) + (1/3) + (1/4) + (1/1)) /10 =0.28

Task3:

Following are the list of sentences that worked as the words of the sentences and article sentences matched

- Prediction of drop in grain prices (Rank=1)
- Awaiting head of investigation's response (Rank = 4)
- Decrease in production of wheat (Rank =1)
- Drought blamed for lower grain export (Rank =2)

- With which company did potato farmers have an issue? (Rank=3)
- International demand for wheat processors have decreased with the decreasing export of wheat (Rank =3)
- Initiative to teach children about animal welfare (Rank=2)
- Water shortage blamed for lower export licenses issued (Rank=2)
- Virus affects Australian tomato production (Rank = 4)

MRR = ((1) + (1/4) + (1) + (1/2) + (1/3) + (1/3) + (1/2) + (1/2) + (1/4)) /9=0.51

Task4:

Following are the list of sentences that worked as the words of the sentences and article sentences matched

- Water shortage blamed for lower export licenses issued (Rank =2)
- Drought in NSW gets worse (Rank=1)
- Decrease in production of wheat (Rank=1)
- International demand for wheat processors have decreased with the decreasing export of wheat (Rank=2)

MRR = ((1/2) + 1 + 1 + (1/2))/4 = 0.75

Challenges:

Following are the issues that were faced:

Difficulty in finding the best combination of features for Task 4 to improve the results. This was because every input query worked well with a different combination.
Overcome: We tried different combinations of features and compared the results to find the best that worked for most of the sentences.

Pending Issues:

Improving Task 4 to give better search results.

Potential Improvements

Make it possible to automatically decide which features are best for the provided sentence and hence use it to get the best result.