Question Answer System for Twitter and Wikipedia on US Election 2016

Project Report Phase 3

Submission Date: 07/25/2016

by

Ganesh Taduri (#40)

Swatvik Gunamaneni (#10)

Venkata Sasidhar kanumuri(#13)

Tejo Kumar Manchu (#19)

Motivation: There are many advantages in extracting the information from the slew of sources available on the internet. Not a decade ago, we realized that we can process this large amount of data and extract meaningful information from it. There is a growing importance on the development of information technologies that are capable of accessing and analyzing the data available via social media. Twitter is one such source of large amount of raw data is which contains worldwide news, public opinion on several events which can be processed and relevant information can be extracted. On average, every second around 6000 tweets are tweeted on twitter. This data can be used to a build a question answer system

Objectives:

The objective of the project is to learn machine learning algorithms, natural language processing through implementation of available services and APIs like Natural language classifier, Speech to text, speech to text etc. We plan to tap the twitter data and develop a question answering system which takes a text question as input and answers it in accordance with the data available on twitter. In addition, we wish to use Wikipedia data to answer questions on static affairs related to US election 2016

Data collection: Our data is related to US election 2016.We collected data from various sources like Wikipedia and twitter using python and their respective API. The twitter data is collected based on #election2016.The Wikipedia data is a text file but the twitter data is extracted in the form of JSON object. We extracted the tweet text from the object. The data is distributed among ten documents depending upon its content. The extracted data from Wikipedia is a meaningful one whereas most of data from twitter is inconsistent with various errors like spelling and grammatical mistakes and unrelated data. We collected data from Wikipedia pages like Donald Trump, Obama, Hillary Clinton etc.

Data collection URL: www.twitter.com & www.wikipedia.com

Input:

The text data we collected from various sources was given as input to the model.

Expected output: The expected output is a Question Answer system, where a user can ask a question in text format and the model answers the question in text format.

Sample Question1:

Who is the republican candidate for US elections 2016?

Sample Answer1:

Donald Trump

Sample Question2:

What is the public opinion on Hillary Clinton winning chances?

Sample Answer2:

She has 60% winning chance

Domain: Question answering system

Class diagram:

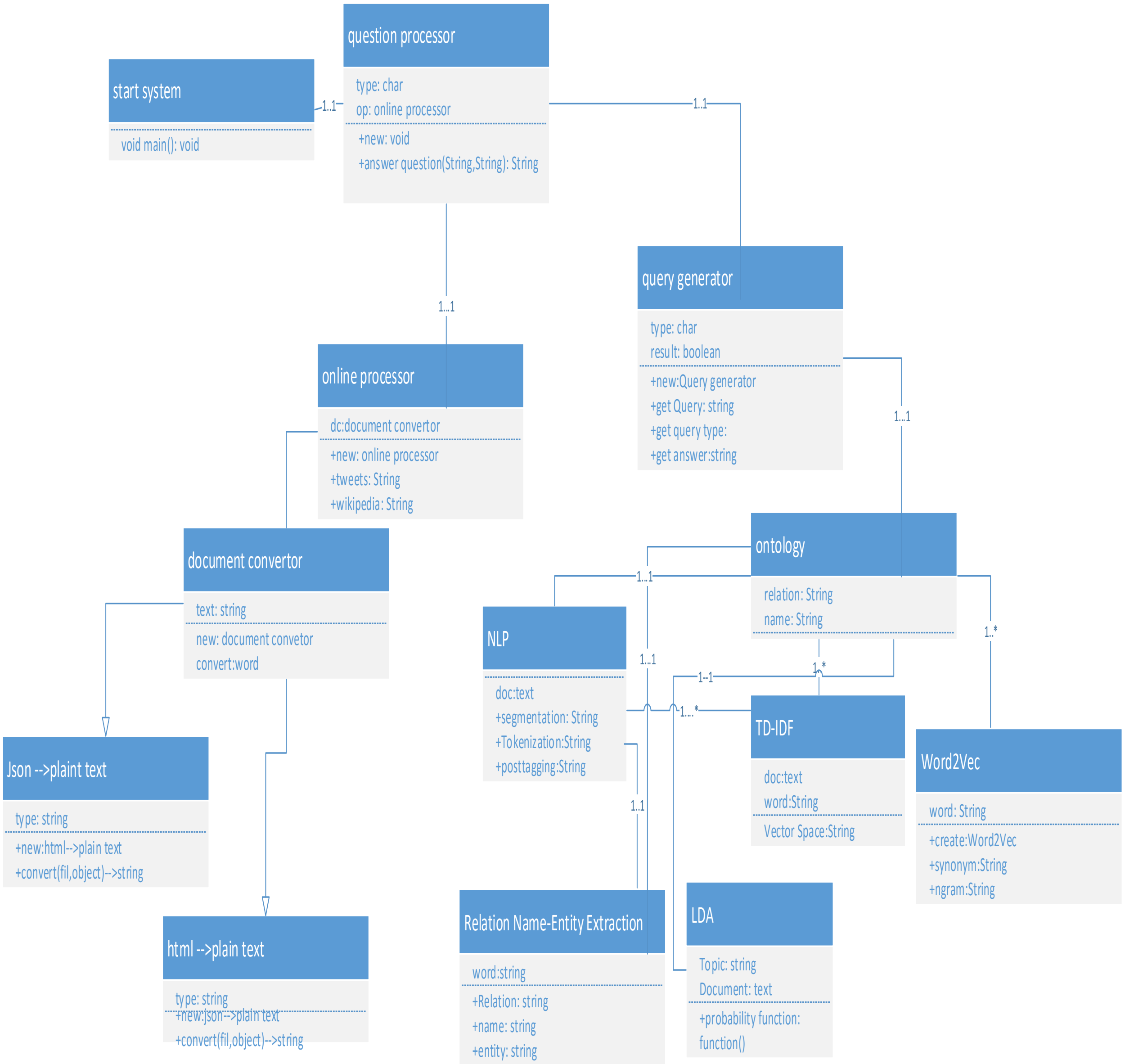
Class diagrams shows the classes in the system and their interrelationships between them

Classes involved in this system are

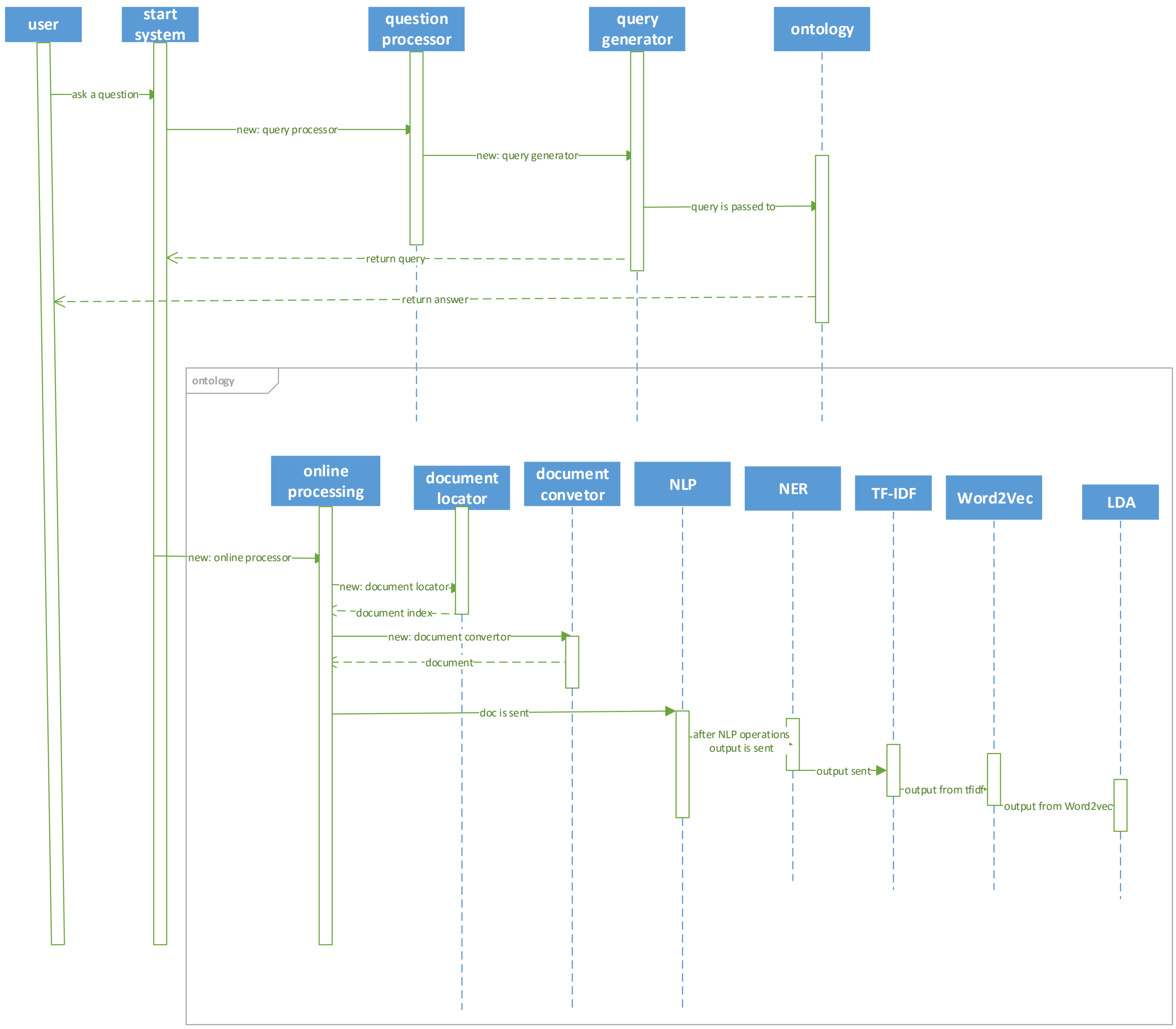
- Start system
- Online processor
- Query generator
- NLP
- Question processor
- Document convertor
- Word2Vec
- Json_plain text
- Html_plain text
- Ontology
- LDA
- Name Relation Entity Extraction

We have the following relations for classes

- Dependency relation shows class is dependent on or uses other class
- Generalization which shows the inheritance of classes like which class is a subset of or what is the superclass of this subclass
- Generalization is placed from document convertor to Json_plain text and html_plain text where document convertor is the superclass of both classes
- Dependency is placed between word parser (NLP, Word2vec,) and Topic discovery (LDA)as to parse the data we need text split into sentences and we get this from sentence extraction
- Other classes are linked with association relation that shows the connection of two classes


question processor
type: char
op: online processor
+new: void
+answer question(String,String): String
start system
void main(): void
1..1
1..1
1..1
query generator
type: char
result: boolean
+new:Query generator
+get Query: string
+get query type:
+get answer:string
online processor
dc:document convertor
+new: online processor
+tweets: String
+wikipedia: String
1..1
document convertor
text: string
new: document convetor
convert:word
ontology
relation: String
name: String
1..1
1..1
1..*
1--1
1..*
1...*
NLP
doc:text
+segmentation: String
+Tokenization:String
+posttagging:String
TD-IDF
doc:text
word:String
Vector Space:String
Word2Vec
word: String
+create:Word2Vec
+synonym:String
+ngram:String
Json -->plaint text
type: string
+new:html-->plain text
+convert(fil,object)-->string
1..1
html -->plain text
type: string
+new:json-->plain text
+convert(fil,object)-->string
Relation Name-Entity Extraction
word:string
+Relation: string
+name: string
+entity: string
LDA
Topic: string
Document: text
+probability function:
function()

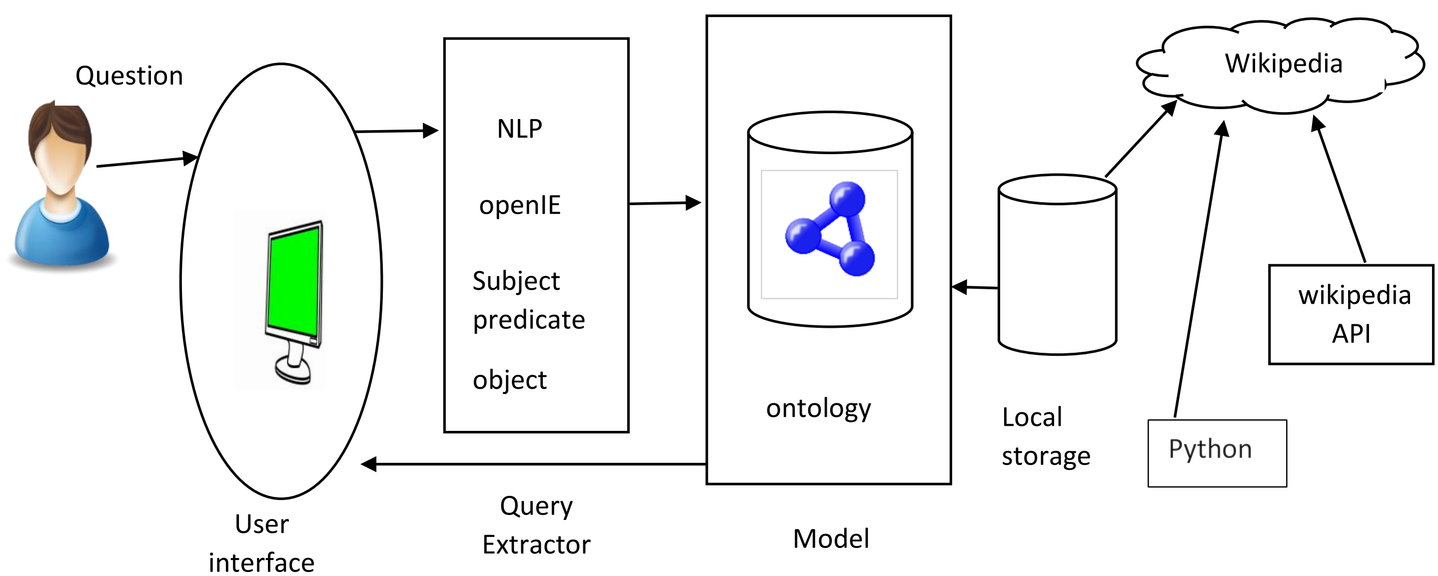
<u>Sequence diagram:</u> Sequence diagram gives the information about the interactions between the objects. The communication between the objects is done by passing messages from one object to another.



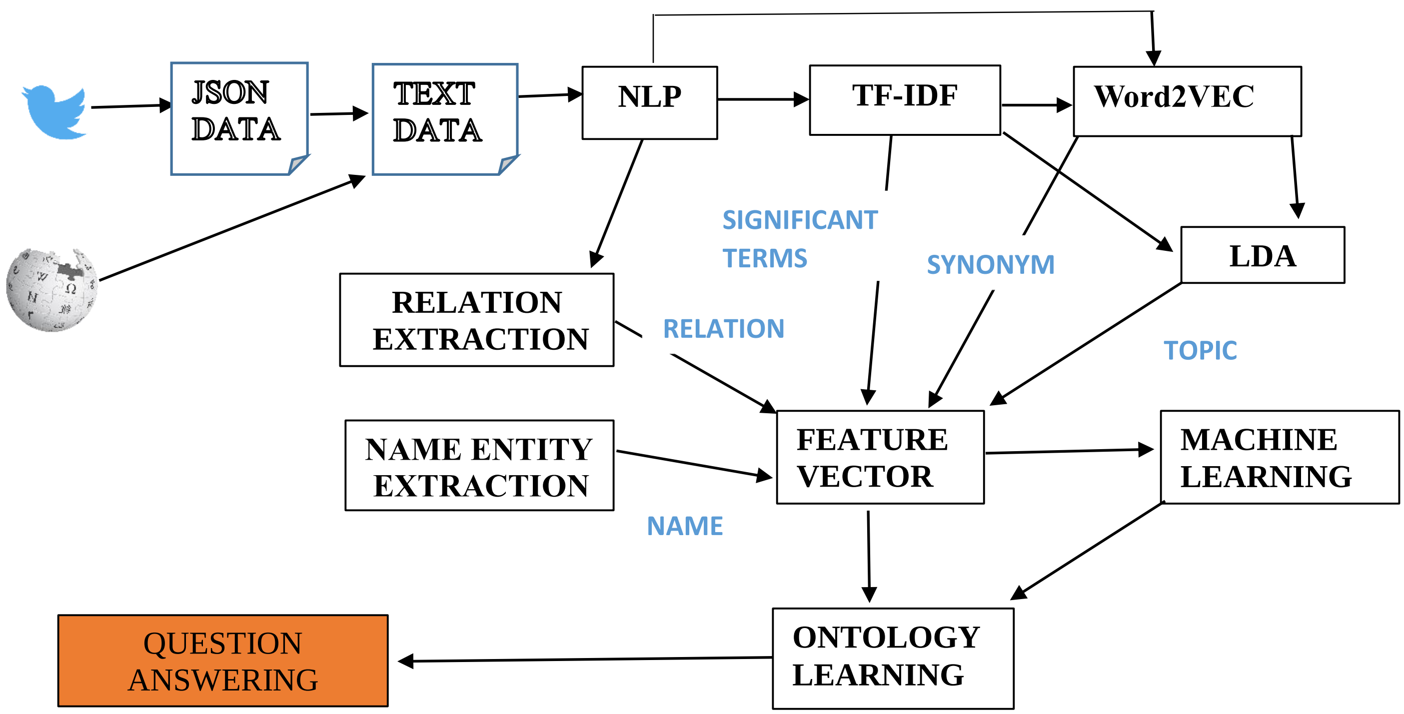
The objects involved in this model are:

- User: who interacts with the system by giving a question
- Start system: this takes the input from the user and interacts with our model
- Question processor: processes the question and classifies the question based on what, when, and other classifications
- Query generator: generates a machine readable query of the question from the user
- Online processing: streaming for dynamic data that comes from the twitter API
- Document locater: locates the document like this gives the index of the document in the local storage
- Document convertor: converts the html file or Json file
- Sentence extractor: gives the sentences from the paragraphs in the document
- Word parser: generates the tokens from the sentences
- Answer generator: generates the answers based on machine learning algorithms provided

System Architecture



WORKFLOW



<u>Services used:</u>

<u>Natural language Classifier (NLC):</u> this service uses the deep learning techniques and by using these techniques we make predictions about the predefined classes for sentences that we give as input for this service. This makes use of some machine learning algorithms to make predictions

Example of his service could be IBM Watson

<u>MongoDB</u>: Mongo DB is an open source database which has the convenience for the developers in terms of scalability and ease of use. Data is stored in the form of a Json file in mongo DB

<u>Alchemy API</u>: we make use of alchemy API like named entity extraction and other keyword extraction

- This twitter API forms as an interface for the developer to use data coming from twitter, here data is in the form of tweets. Data is collected in Json format.
- We will also use speech to text API to convert the speech that we give as input from the device into text format

Natural classifier demo: We plan to use Natural language classifier to determine the question type. In this section, the demo of the same classifier is shown. In this demo we train the classifier with twenty-five male names and twenty-five female names. Then we give a test name which the classifier takes as input and gives the percentage confidence of male and female classification.

Features developed for phase 1:

Cognitive services you want to use for project:

Implemented natural language classifier service from alchemy API. We are planning to use this service to classify questions into various categories.As of now we practiced implementing this service for classifying male and female names.

Screen shot of giving training data:

```
C:\>curl -i -u "f6de402c-0f6e-4c26-bbdd-b5e314767f72":"agBMTwd7DPnF" \ -F training_data=@C:/Users/tmcx4/Desktop/names.csv
//gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers"
curl: (6) Could not resolve host: \
curl: (6) Could not resolve host: \
curl: (6) Could not resolve host: \
HTTP/1.1 100 Continue
X-Note: Gateway Ack

HTTP/1.1 200 OK
X-Backside-Transport: OK OK
Connection: Keep-Alive
Transfer-Encoding: chunked
Date: Sat, 25 Jun 2016 00:06:41 GMT
Content-Type: application/json
```

Classifier response:

```
{
  "classifier_id" : "2373f5x67-nlc-7103",
  "name" : "TutorialClassifier",
  "language" : "en",
  "created" : "2016-06-25T00:06:40.763Z",
  "url" : "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/2373f5x67-nlc-7103",
  "status" : "Training",
  "status_description" : "The classifier instance is in its training phase, not yet ready to accept classify requests"
}
C:\>
```

Classifier Status:

```
C:\>
C:\>curl -u "f6de402c-0f6e-4c26-bbdd-b5e314767f72":"agBMTwd7DPnF" \ "https://gateway.watsonplatform.net/natural-lang
curl: (6) Could not resolve host: \
{
  "classifier_id" : "2373f5x67-nlc-7103",
  "name" : "TutorialClassifier",
  "language" : "en",
  "created" : "2016-06-25T00:06:40.763Z",
  "url" : "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/2373f5x67-nlc-7103",
  "status" : "Available",
  "status_description" : "The classifier instance is now available and is ready to take classifier requests."
}
C:\>
```

Classifier Output for test data: The name “Suresh” is classified as male with 85% confidence.

```
C:\>curl -G -u "f6de402c-0f6e-4c26-bbdd-b5e314767f72":"agBMTwd7DPnF" \ "https://gateway.watsonplatform.net/natural
resh"
curl: (6) Could not resolve host: \
{
  "classifier_id" : "2373f5x67-nlc-7103",
  "url" : "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/2373f5x67-nlc-7103",
  "text" : "suresh",
  "top_class" : "male",
  "classes" : [ {
    "class_name" : "male",
    "confidence" : 0.8489180884346
  }, {
    "class_name" : "female",
    "confidence" : 0.15108191156540002
  } ]
}curl: (6) Could not resolve host: \
```

The name “Jessica” is classified as female with 95% confidence.

```
C:\>curl -G -u "f6de402c-0f6e-4c26-bbdd-b5e314767f72":"agBMTwd7DPnF" \ "https://gateway.watsonplatform.net/natural-
ssica"
curl: (6) Could not resolve host: \
{
  "classifier_id" : "2373f5x67-nlc-7103",
  "url" : "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/2373f5x67-nlc-7103",
  "text" : "jessica",
  "top_class" : "female",
  "classes" : [ {
    "class_name" : "female",
    "confidence" : 0.9571645890651067
  }, {
    "class_name" : "male",
    "confidence" : 0.042835410934893264
  } ]
}curl: (6) Could not resolve host: \
```

Word count count on the data collected:

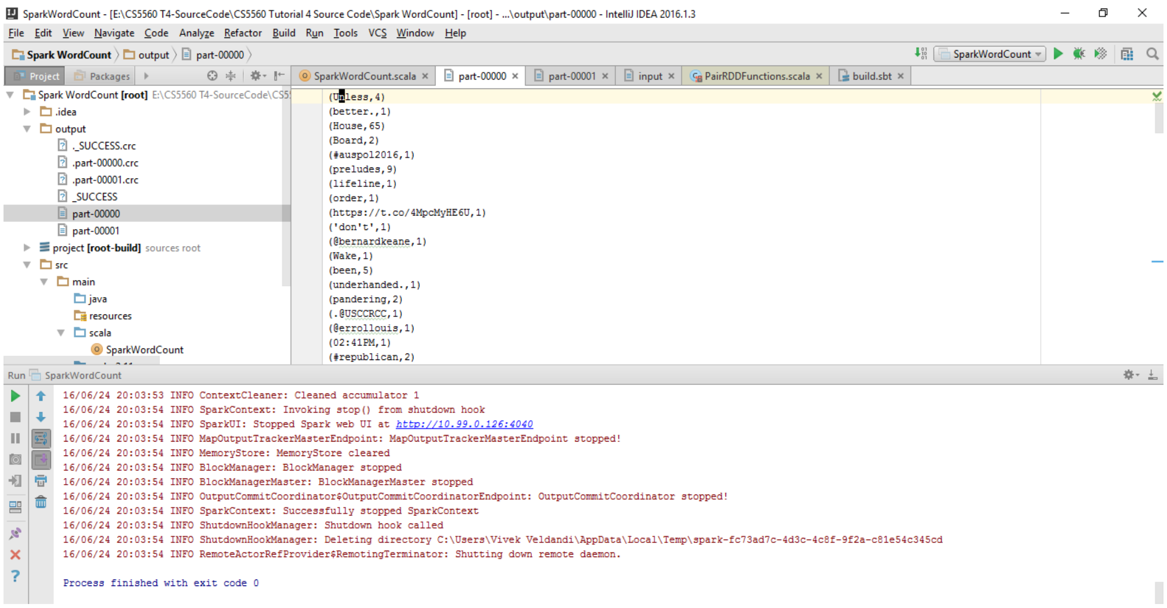
Input:

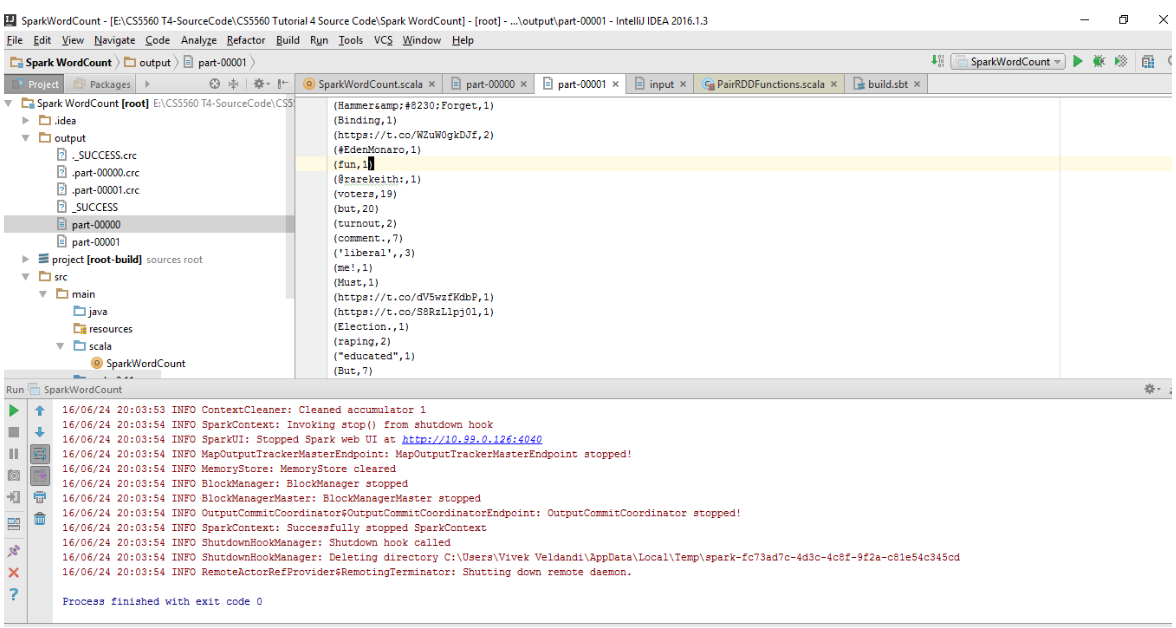
Text data from twitter.

Output:

Word count for given input

Sample screen shots:





TF-IDF on the data:

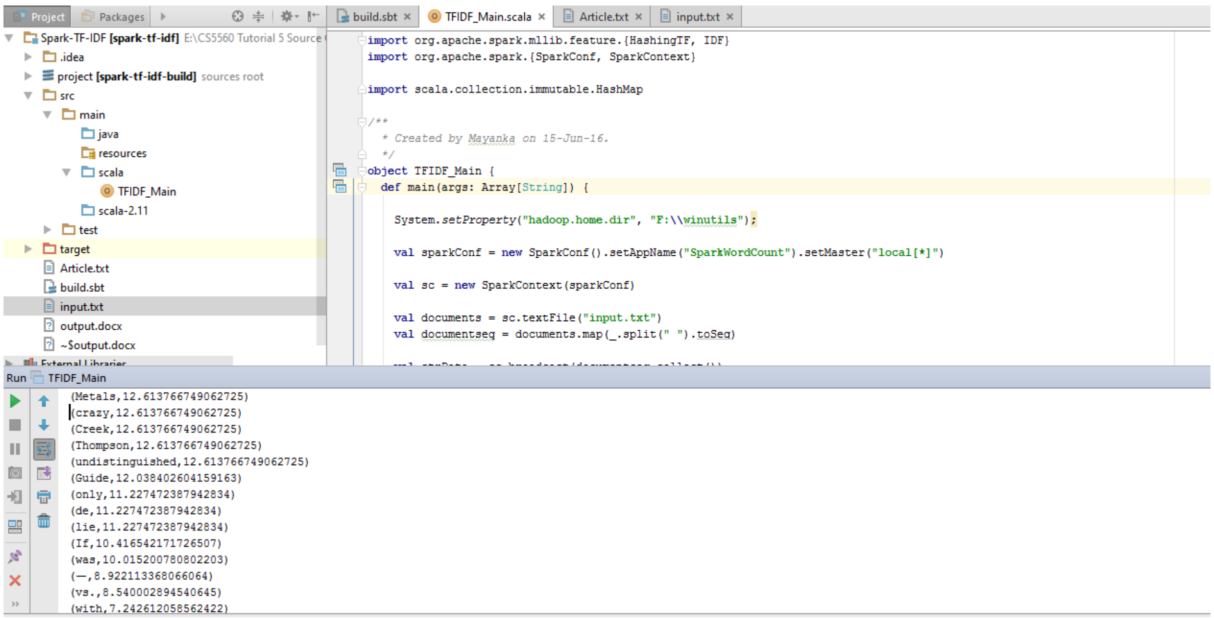
Input:

Text data from Wikipedia

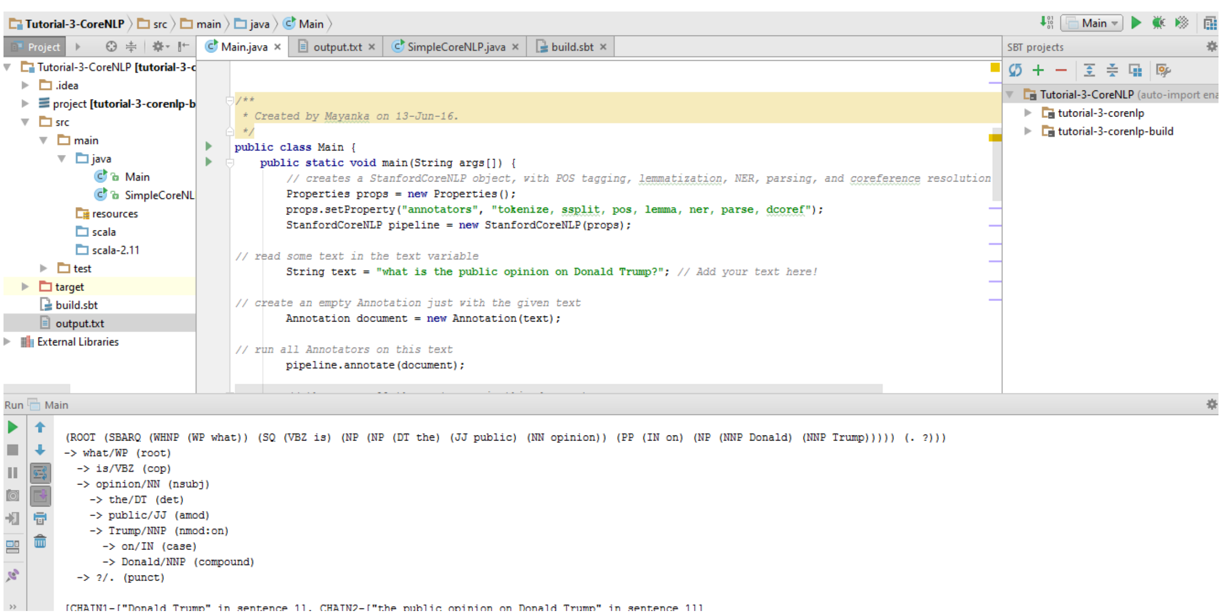
Output:

TF –IDF score for each word and top TF – IDF words.

Sample Screen Shots:



Sample screen shots for CoreNLP:

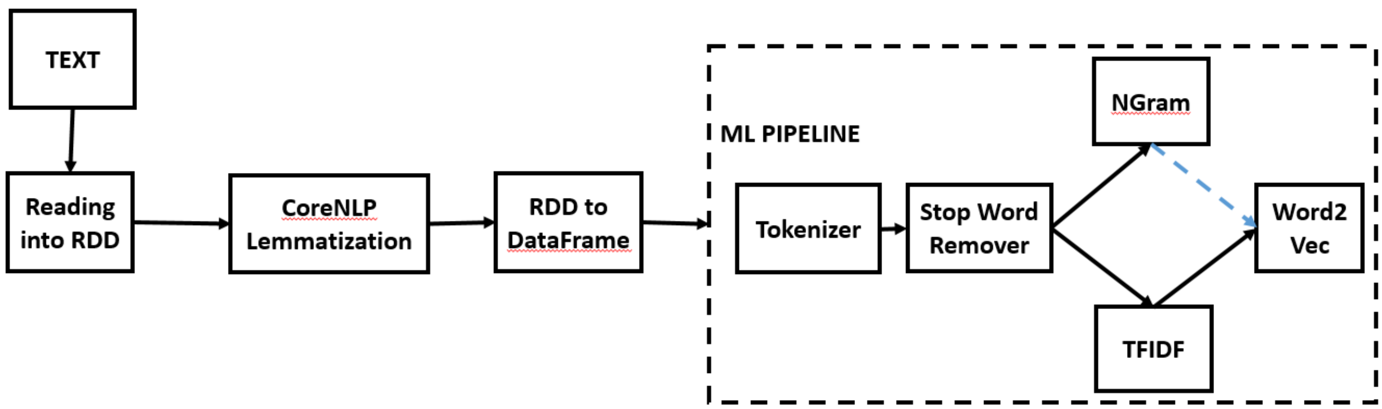


Features developed for phase 2:

NGram and Word2Vec Exercise on the data:

For the data we collected from Wikipedia and twitter we collected NGram and Word2Vec. The NGram data frame we took was 5 (Greater the number of data frame better is the sense of the output). We also collected top TF-IDF words and found synonyms for them using Word2Vec.

workflow



Input:

Text data collected from Wikipedia and twitter.

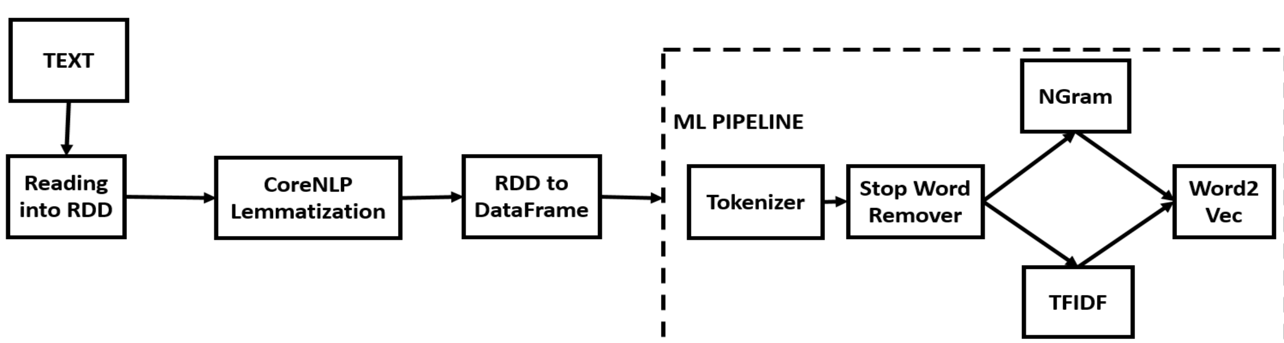
Output:

N Grams with DataFrame = 5 and synonyms for top TF-IDF words.

Name Entity Extraction and Relation Extraction on our Data:

Next we extracted structured relation triples Using OpenIE. Then by using ConceptNet5 we tried to retrieve some concepts and relationships for a few of the wiki pages (The pages where we collected out raw data).

Workflow:



The Sample output we got using OpenIE with coreNLP on Twitted data is

[(This,is,brilliant,1.0)]

[(they,call,election,1.0)]

Usage of WordNet on the data:

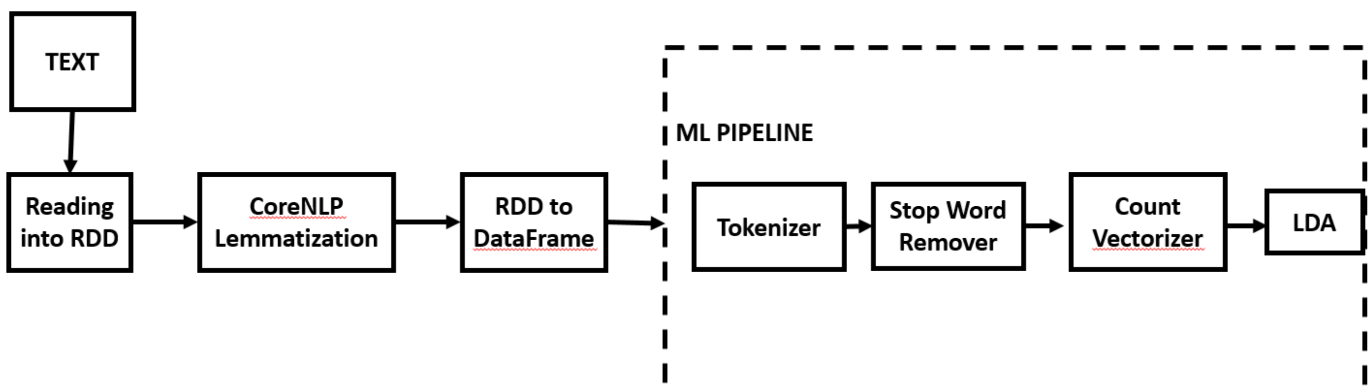
Then with the help of WordNet we tried to extract some Synonyms for some of the sample words in our data. By comparing these Synonyms with the synonyms extracted from Word2Vec we identified that synonyms from WordNet are more accurate and related than that are form Word2Vec.

Topic Discovery on the data:

We collected around 9 documents of data from Wikipedia and one document of twitter data and we classified all the words in all the documents into 4 topics using Latent Dirichlet allocation (LDA) method.

Later, we also used K- means algorithm to classify the words into groups again.

Workflow for LDA:



Input:

All the documents of data from Wikipedia and twitter.

Output:

Output has group of 4 topics each topic has words and its respective probability value for belonging to that group.

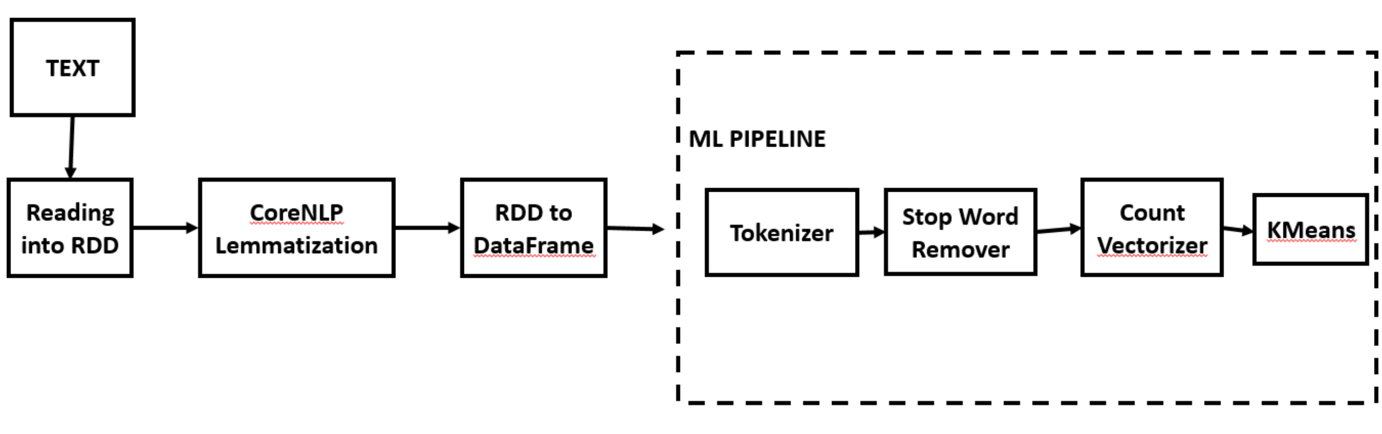
Summary of LDA execution:

```
Corpus summary:
    Training set size: 2473 documents
    Vocabulary size: 8006 terms
    Training set size: 45460 tokens
    Preprocessing time: 27.113088039 sec
```

```
Finished training LDA model.  Summary:
    Training time: 29.763379458 sec
    Training data average log likelihood: -178.32006826792988
```

<u>K – Means Clustering:</u>

Now we applied K -means Clustering method to group the data into clusters using K – Means algorithm.



Input:

1. All the 9 documents containing Wikipedia data
2. Document containing tweets from the twitter

Output:

The K-means clustering has divided all the words in all the documents into 4 groups. The output has 4 vectors with their group words and their respective cluster IDs.

Sample Corpus Summary from K- means:

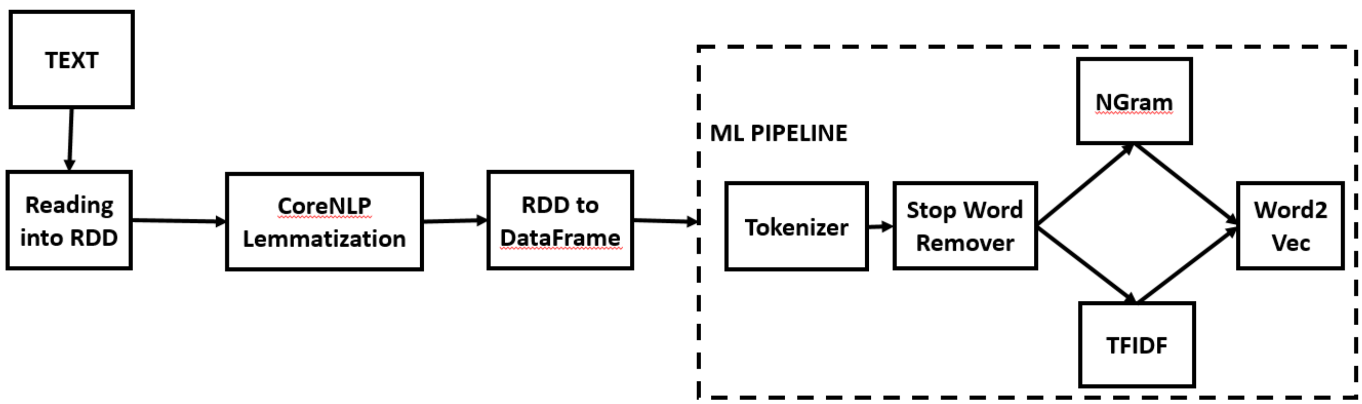
```
Corpus summary:
    Training set size: 9 documents
    Vocabulary size: 6889 terms
    Preprocessing time: 41.883770357 sec

Finished training KMeans model.  Summary:
    Training time: 2.212250886 sec
```

Feature Vector Generation: We generated two feature vectors for machine learning one with TFIDF and other with TFIDF + Word2vec.

Input: set of documents containing the text data.

Output: Feature vectors

Work flow:

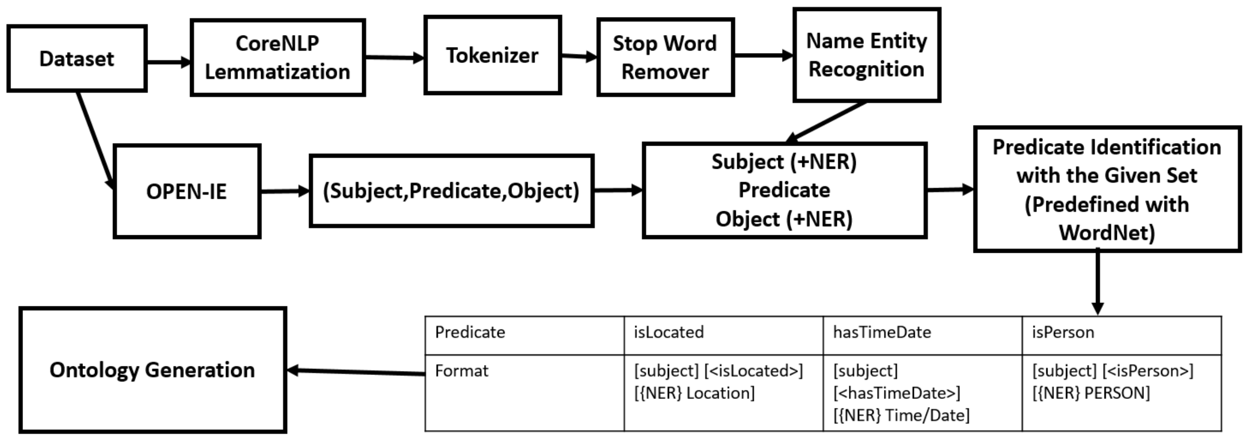


Features developed for phase 3:

We created ontology using our data both statically and dynamically

Dynamically developed Ontology: To create dynamic ontology we pruned our data to maximum extent to make the ontology meaningful. We developed two ontologies dynamically using twitter and Wikipedia data. Of the two, ontology based on Wikipedia data is more meaningful. Even though the ontology is meaningful it is not useful enough to extract sensible data.

The workflow used to create dynamic ontology is:



Predicate	isLocated	hasTimeDate	isPerson
Format	[subject] [<isLocated>] [{NER} Location]	[subject] [<hasTimeDate>] [{NER} Time/Date]	[subject] [<isPerson>] [{NER} PERSON]

ScreenshotsVisualization using VOWL:

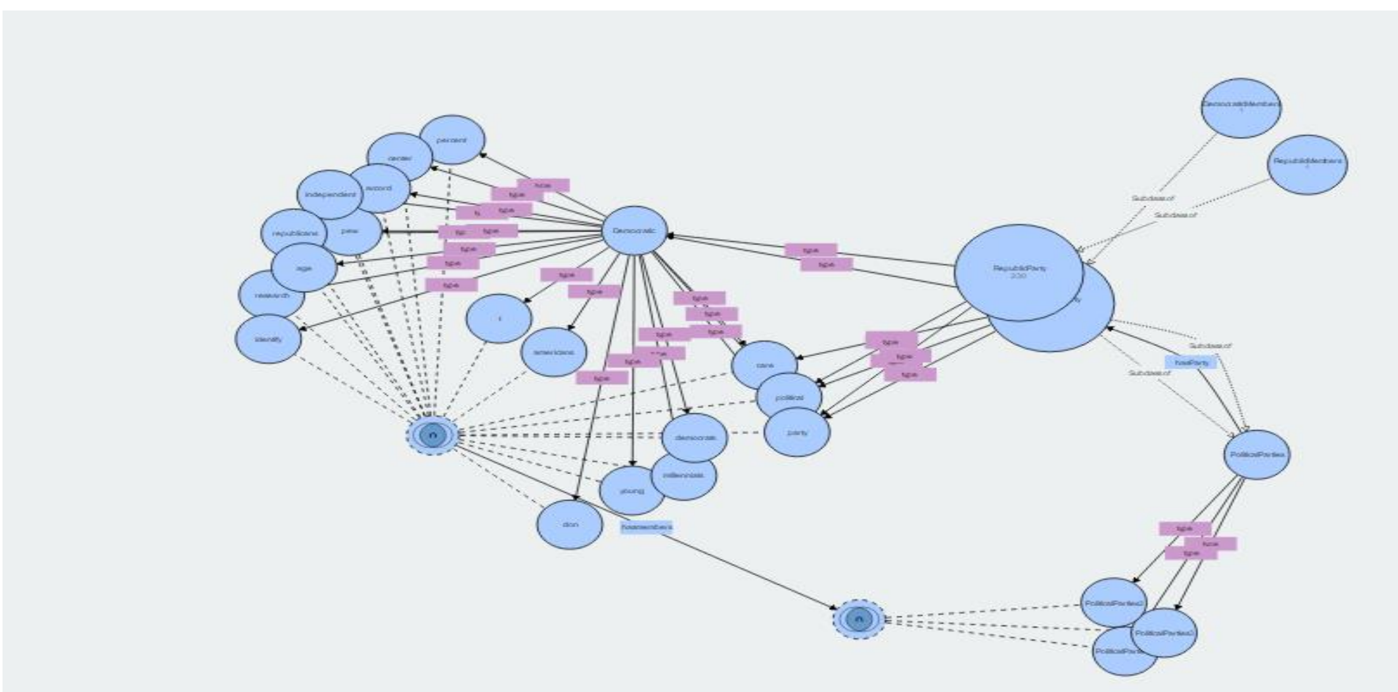


Fig: *Screenshot of Dynamic Ontology*

Note: The ontology created using the whole data could not be loaded into VOWL tool as the file was very huge. So, only part of data is used to develop and visualize the ontology

Statically developed ontology: we used protégé tool to develop static ontology.

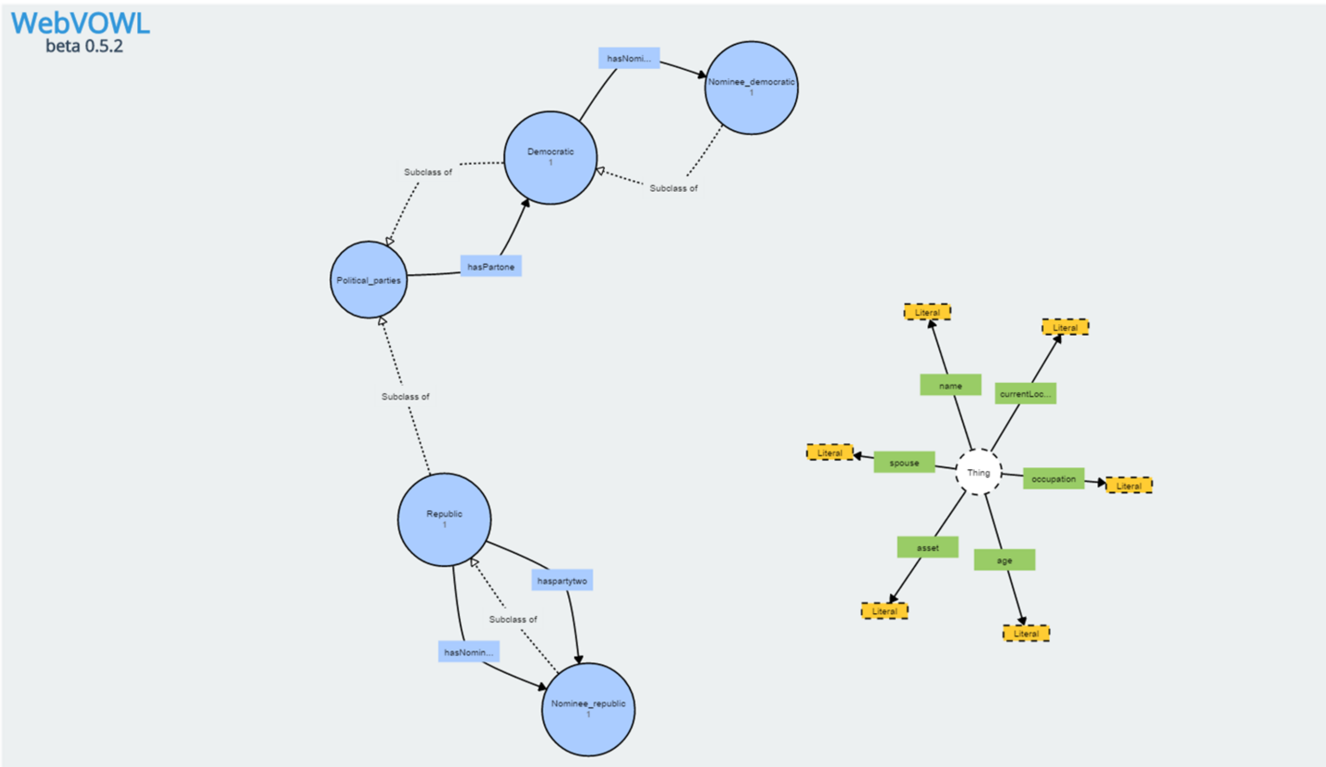


Fig: *Screenshot of static Ontology using VOWL*

Fig: *Screenshot Data Property Hierarchy*

Fig: *Screenshot Object Property Hierarchy*

Fig: *Screenshot class Hierarchy*

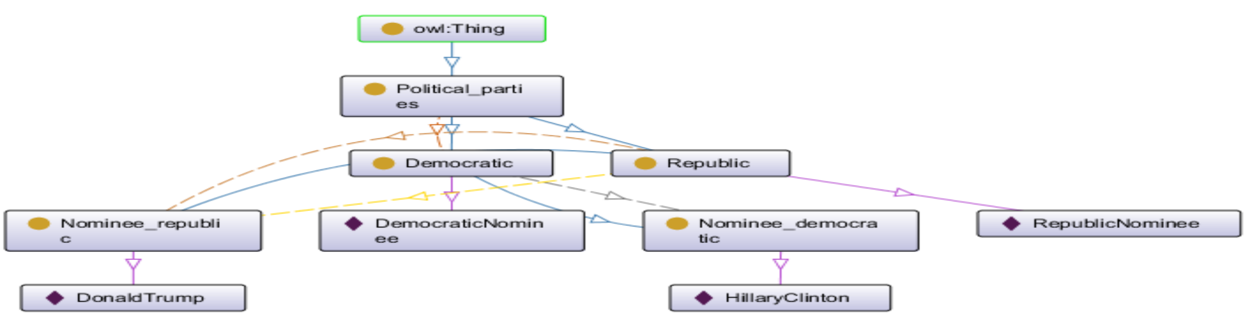


Fig: *Screenshot of static Ontograph using protege*

Question Anwering System

Workflow of the system is as shown below:

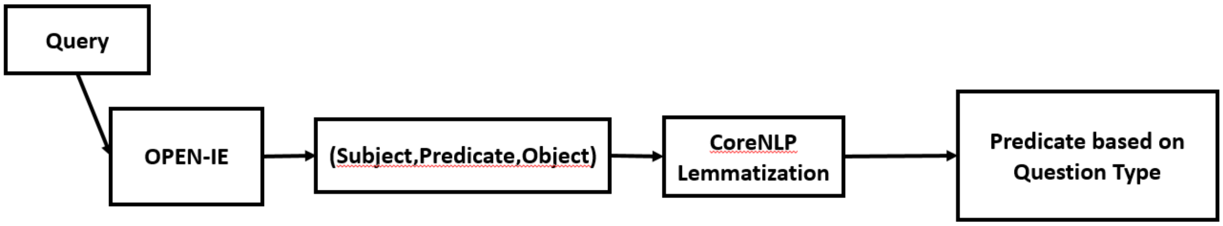


Fig: *Workflow of Question answering system*

From the developed ontology, we extracted data by asking some questions such as:

who is RepublicNominee?

Who is DemocraticNominee?

Where is DonaldTrump? Etc.

These questions are converted to SPARQL queries as per the workflow diagram. Sample converted queries and their respective answers

<u>Question</u>:Who is RepublicNominee?

<u>Answer:</u> Donald Trump

<u>Converted Query(Screenshot)</u>

```
PREFIX elec: <http://www.kdm.com/OWL/elections2016#>
SELECT ?name
WHERE {
  elec:RepublicNominee  elec:name ?name
}

```

Answer Screenshot:

QUERY RESULTS

Table | Raw Response

Showing 1 to 1 of 1 entries

	name
1	"Donald Trump"

Showing 1 to 1 of 1 entries

Question: Who is DemocraticNominee?

Answer: Hillary Clinton

Converted Query(Screenshot)

```
PREFIX elec: <http://www.kdm.com/OWL/elections2016#>
SELECT ?name
WHERE {
  elec:DemocraticNominee  elec:name ?name
}
```

Answer Screenshot

QUERY RESULTS

Table | Raw Response

Showing 1 to 1 of 1 entries

	name
1	"HillaryClinton"

Showing 1 to 1 of 1 entries

Question: Where is DonaldTrump?

Answer: New York

Converted Query(Screenshot)

```
PREFIX elec: <http://www.kdm.com/OWL/elections2016#>
SELECT ?city
WHERE {
elec:DonaldTrump elec:currentLocation ?city
}
```

Answer Screenshot

QUERY RESULTS

Table Raw Response

Showing 1 to 1 of 1 entries

	city
1	"NewYork"

Sample output screenshots:

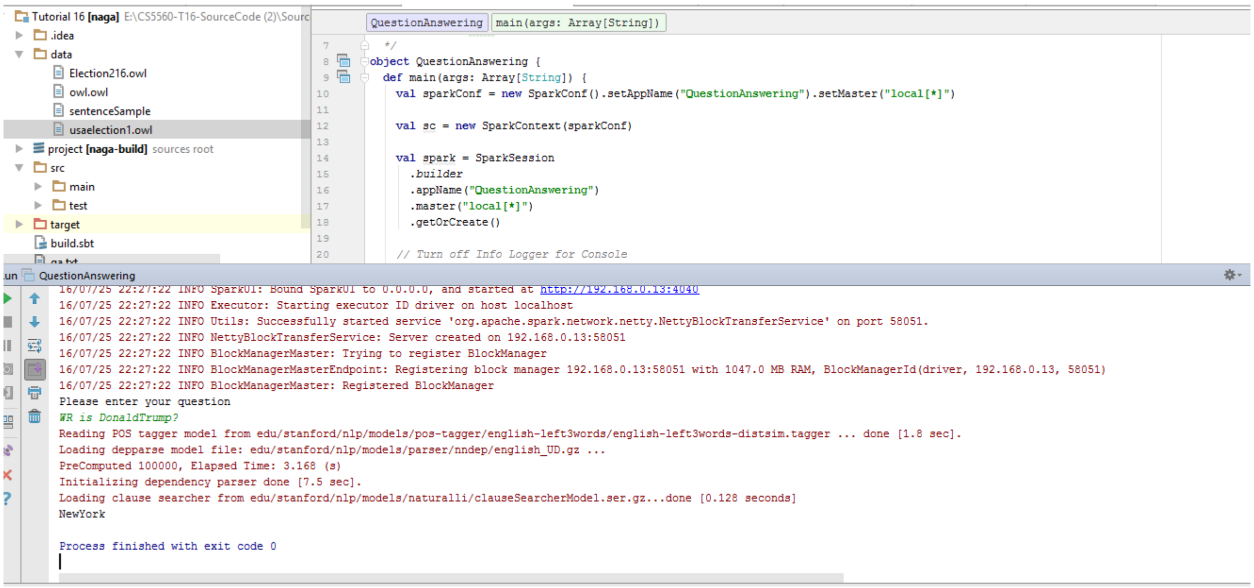


Fig: *Output for question "Where is DonaldTrump"*

Fig: *Output for question "Who is DemocraticNominee"*

Fig: *Output for question "Who is RepublicNominee"*

Contribution of Team members:

Ganesh Taduri (Team Leader) & Tejo Kumar: Handled the data collection part. The data is collected using Twitter API, Tweepy from Twitter and using Wikipedia API from Wikipedia. The twitter data is in Json format. We then extracted the tweet text using Json parser. We tried using the natural language classifier for the sample data. Dynamic Ontology developmet.

Swatwik & Sasidhar: Conceived the project idea, designed the project architecture and necessary UML diagrams. Handled the documentation part. Implemented the tutorial concepts on the collected data. Static ontology development using Protégé tool

All the teammates are involved in developing features for this phase project

Project GitHub Link: https://github.com/ganeshtaduri/KDM

Concerns/Issues:

- Although we started with the idea of collecting data from Twitter and LinkedIn, we had major glitches while collecting data. We had trouble in getting authorization keys.
- The execution of a few programs from tutorials is constrained by system memory and would occasionally throw errors like 'Job aborted due to stage failure'.

Future Work:

- Data extraction from Wikipedia using DBpedia framework helps in extracting the data from info boxes
- Online twitter streaming for real time data can be incorporated
- The ontology size can be scalable
- Developing GUI for the application
- Implementing machine learning algorithms for question classification and query extraction

References:

Class Lectures and Tutorial demos

https://docs.google.com/spreadsheets/d/1TjrVH8BxRmmvMlm3ig3X4DnwrIF3QR7ZpyUzjPbGCQ0/edit#gid=742202525

http://natural-language-classifier-demo.mybluemix.net/

http://adilmoujahid.com/posts/2014/07/twitter-analytics/

www.developers.twitter.com

www.stackoverflow.com

www.wikipedia.com