

# **CS5560 KNOWLEDGE DISCOVERY AND MANAGEMENT**

## **PROJECT REPORT-PHASE 3**

### **SUMMARIZATION OF GOOGLE/TWITTER TRENDING NEWS**

#### **TEAM MATES**

- 1. PRASANNA MUPPIDI(23)**
- 2. SANTHOSH MOHAN MURARISHETTI(24)**
- 3. ANUDEEP PANDIRI(29)**
- 4. VAMSHI RAJARIKAM(36)**

# Index

<b>1. Problem statements and Data collection</b>	<b>-----3</b>
<b>2. Tasks</b>	<b>-----4</b>
<b>3. Implementation Specifications</b>	<b>-----5</b>
<b>4. Results &amp; Evaluation</b>	<b>-----6</b>
<b>5. Project Management</b>	<b>-----16</b>
<b>6. References</b>	<b>-----19</b>

## **1. Problem statements and Data collection:**

### **a. Motivation:**

Every social networking site provides data about the trending news but only to the users who have an existing account in that particular website. Displaying and letting every user know about the latest or the trending news is very important sometimes. For example, Earthquake. It's very important for people to know about the status of the earthquake or status of people in particular locations. To let the users know about the latest or trending news with so much ease is the main motivation of our application.

### **Objective:**

Our application takes information regarding all the latest or trending news from various social networking sites using their particular APIs and displays the summarized data to the users. Summarization is done using various NLPs. The unstructured data which is collected from the social networking sites is processed using TF-IDF and the output data is structured. This structured data is easily understandable by the users. It's not required for the users to Login into the application and view the news. The news which is displayed is clear cut without any unnecessary information making it easy for the users. Collecting, Managing, Summarizing and Displaying all the trending news is the main objective of our application.

### **Expected Outcomes:**

When we give an unstructured data file as an input, summarized and easily understandable data would be the outcome of our application.

### **b. Domain**

- **Topic:** Summarization of trending topics in the world from various news articles.
- **Technologies Used:**
  - ✓ Languages: Java, Scala
  - ✓ IDE: IntelliJ Idea
  - ✓ Frameworks: Spark
  - ✓ Libraries: CoreNLP, SparkNLP, JSON, WordNet, SparkML, LDA,
  - ✓ APIS: Google Trends, Guardian, NYtimes, USAtoday, Intellixir

## c. Data Collection

There are many open source datasets available in the internet. There are mainly two types of data collection.

- i. Static Data
- ii. Real Time Data using APIs

### i. Static Data:

Static data is nothing but data which is immutable and not changes during time. It is a fixed data set. We collected static data from twitter using curl command

Example:

```
curl --get 'https://stream.twitter.com/1.1/statuses/sample.json' --header 'Authorization: OAuth oauth_consumer_key="TjIDZ6QX6TZOq64EZ49SatYb", oauth_nonce="d54db403fb54cc9e5a10a92bb2741e6e", oauth_signature="GqHWmZKgD8YO6rX5HgGKRuMFWGQ%3D", oauth_signature_method="HMAC-SHA1", oauth_timestamp="1457754994", oauth_token="453746488-vDRGN511Pk3g3tSvOhpgIdSRErFjXP5fClexkpWp", oauth_version="1.0"' --verbose> tweet.txt
```

### ii. Real Time Data Using APIs:

Real time data is the data which changes dynamically with time. This data is provided by some open source APIs like Google trends, Hawt trends, USAToday, etc.,

Presently, we are collecting data using google trends API which gives JSON output. We are storing those data in text file which we will provide as input for future analysis.

**API:**

**Google Trends:**

<https://www.google.com/trends/api/stories/latest?cat=m&fi=15&fs=15&geo=US&ri=300&rs=15&tz=300>

**Newyork Times:**

[https://api.nytimes.com/svc/search/v2/articlesearch.json?api-key=1069bc25bff24ebf8cf3dbae1133e000&q=tech&sort=newest&fl=lead\\_paragraph&page=0](https://api.nytimes.com/svc/search/v2/articlesearch.json?api-key=1069bc25bff24ebf8cf3dbae1133e000&q=tech&sort=newest&fl=lead_paragraph&page=0)

## 2. Tasks

### a. Rest API Service:

We are using Rest API service using HTTPURL connection in java. The API provides the top trending topics in the world with source name, article name and URL of the website.

**Input:**

**"<https://www.google.com/trends/api/stories/latest?cat=m&fi=15&fs=15&geo=US&ri=300&rs=15&tz=300>"**

**Output:** <https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/src/gTrends>

The useful information is extracted from the response and stored in text file.

### b. CoreNLP:

CoreNLP provides natural language processing tools which does grammatical analysis on the words.

**Input:** <https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/src/gTrends>

**Output:** [https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/documentation/CoreNLP TF-IDF output.docx](https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/documentation/CoreNLP%20TF-IDF%20output.docx)

### c. TF-IDF:

**TF-IDF** of a word says how important a word is in a document. It is an important factor in determining the weightage of the word in the document or a collection of documents.

**Input:** <https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/src/gTrends>

**Output:** [https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/documentation/CoreNLP TF-IDF output.docx](https://github.com/murarishetty/KDM-SM2016-TEAM-9/blob/master/documentation/CoreNLP%20TF-IDF%20output.docx)

#### d. Name Entity Extraction:

Name entity extraction specifies and tags words such as persons, places, organizations in the given input corpus.

##### News Article 1:


<https://drive.google.com/file/d/0B4VHwW192C9HZ0s1QmU3dlp0VVE/view>

##### News Article 2:

<https://drive.google.com/file/d/0B4VHwW192C9HZmQ2VTBCaHdsNTA/view>

Output: <https://drive.google.com/file/d/0B4VHwW192C9HaXAzeXpoelk1WWM/view?ts=57805ff1>

##### Output Screenshot:



```
Run SparkNER
16/07/08 22:08:45 INFO DAGScheduler: Finished task 0.0 in stage 1.0 (TID 2). 4525 bytes result sent to driver
16/07/08 22:08:45 INFO DAGScheduler: ResultStage 1 (collect at SparkNER.scala:30) finished in 36.344 s
16/07/08 22:08:45 INFO DAGScheduler: Job 1 finished: collect at SparkNER.scala:30, took 36.454136 s
(Mr. ,O )
(Malek , ,PERSON O )
(who ,O )
(play ,O )
(a ,O )
(hacker ,CS)
(wage ,O )
(war ,O )
(on ,O )
(corporate ,O )
(culture , ,O O )
(and ,O )
(Sam ,PERSON )
(Email , ,PERSON O )
(the ,O )
(show \ u2019 ,O NUMBER O )
(creator , ,O O )
(discuss ,O )
(why ,O )
(the ,O )
(first ,ORDINAL )
(season \ u2019 ,O NUMBER O )
(big ,O )
(reveal ,O )
(be ,O )
(merely ,O )
(a ,O )
(setup ,O )
(for ,O )
```

### e. SparkLDA:

In SparkLDA, each word in each document is tagged under a specific topic.

#### News Article 1:

<https://drive.google.com/file/d/0B4VHwW192C9HZ0s1QmU3dlp0VVE/view>

#### News Article 2:

<https://drive.google.com/file/d/0B4VHwW192C9HZmQ2VTBCaHdsNTA/view>

#### Output:

<https://drive.google.com/file/d/0B4VHwW192C9HaXAzeXpoelk1WWM/view?ts=57805ff1>

#### Output Screenshot:

```
* Created by Mayanka on 21-Jun-16.
*/
object SparkTFIDF_FVMain {
  def main(args: Array[String]) {

    // Configuration

    Run SparkLDA_Main

    16/07/08 22:38:21 INFO BlockManagerMasterEndpoint: Registering block manager localhost:59639 with 1125.8 MB RAM, BlockManagerId(driver, localhost, 59639)
    16/07/08 22:38:21 INFO BlockManagerMaster: Registered BlockManager
    Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [3.4 sec].

    Corpus summary:
      Training set size: 40 documents
      Vocabulary size: 152 terms
      Training set size: 384 tokens
      Preprocessing time: 37.712660111 sec

    16/07/08 22:39:07 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
    16/07/08 22:39:07 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
    Finished training LDA model. Summary:
      Training time: 31.875923187 sec
      Training data average log likelihood: -49.16508836832392

    4 topics:
    TOPIC 0
      0.08680608755409154
      0.06069670771500531
      0.054941353081149204
```

## f. Feature Vector:

### News Article 1:

<https://drive.google.com/file/d/0B4VHwW192C9HZ0s1QmU3dlp0VVE/view>

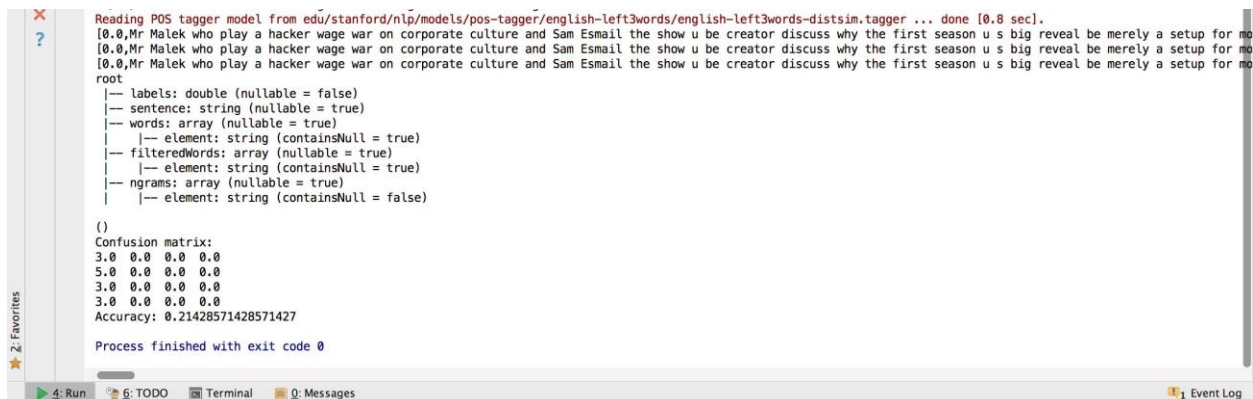
### News Article 2:

<https://drive.google.com/file/d/0B4VHwW192C9HZmQ2VTBCaHdsNTA/view>

### Output:

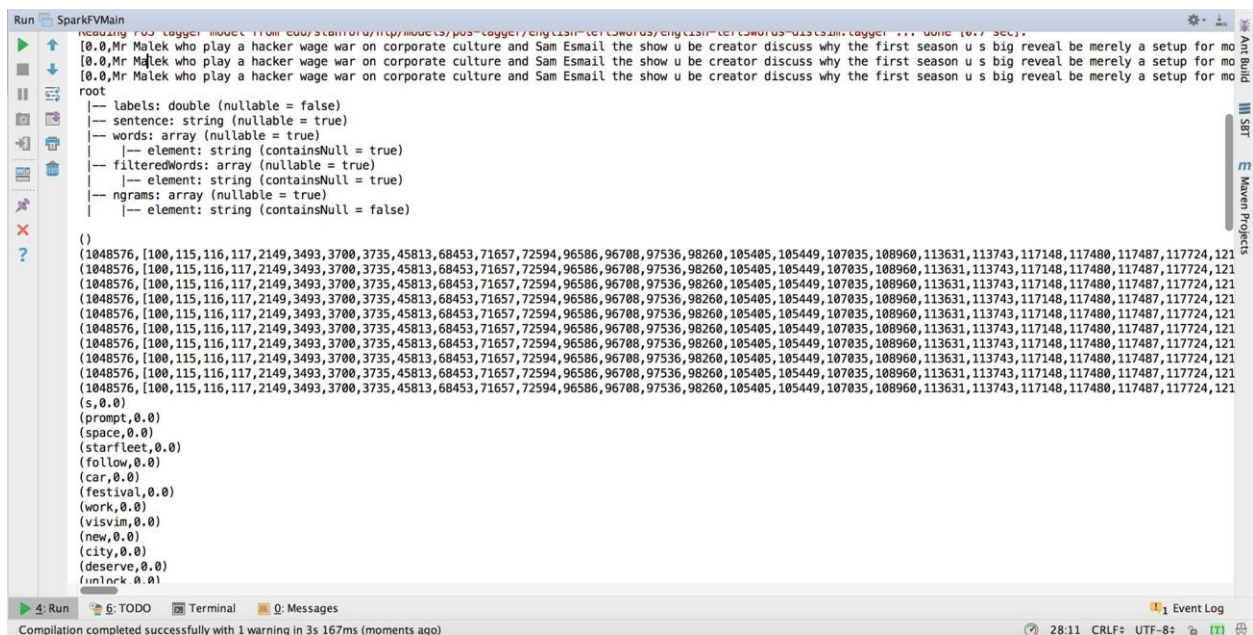
<https://drive.google.com/file/d/0B4VHwW192C9HaXAzeXpoelk1WWM/view?ts=57805ff1>

### Output Screenshot:



```
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [0.8 sec].
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
root
  |-- labels: double (nullable = false)
  |-- sentence: string (nullable = true)
  |-- words: array (nullable = true)
  |   |-- element: string (containsNull = true)
  |-- filteredWords: array (nullable = true)
  |   |-- element: string (containsNull = true)
  |-- ngrams: array (nullable = true)
  |   |-- element: string (containsNull = false)
  ()
Confusion matrix:
3.0  0.0  0.0  0.0
5.0  0.0  0.0  0.0
3.0  0.0  0.0  0.0
3.0  0.0  0.0  0.0
Accuracy: 0.21428571428571427
Process finished with exit code 0
```

### Feature Vector with TF-IDF:



```
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [0.7 sec].
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
[0.0,Mr Malek who play a hacker wage war on corporate culture and Sam Esmail the show u be creator discuss why the first season u s big reveal be merely a setup for no
root
  |-- labels: double (nullable = false)
  |-- sentence: string (nullable = true)
  |-- words: array (nullable = true)
  |   |-- element: string (containsNull = true)
  |-- filteredWords: array (nullable = true)
  |   |-- element: string (containsNull = true)
  |-- ngrams: array (nullable = true)
  |   |-- element: string (containsNull = false)
  ()
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
[1048576,[100,115,116,117,2149,3493,3700,3735,45813,68453,71657,72594,96586,96708,97536,98260,105405,105449,107035,108960,113631,113743,117148,117480,117487,117724,121
(s,0.0)
(prompt,0.0)
(space,0.0)
(starfleet,0.0)
(follow,0.0)
(car,0.0)
(festival,0.0)
(work,0.0)
(visvim,0.0)
(new,0.0)
(city,0.0)
(deserve,0.0)
(unlock,0.0)
Compilation completed successfully with 1 warning in 3s 167ms (moments ago)
```





## h. SPARQL:

Sparql is a query language to retrieve and manipulate the data in Resource Description Framework (RDF).

### News Article 1:

<https://drive.google.com/file/d/0B4VHwW192C9HZ0s1QmU3dlp0VVE/view>

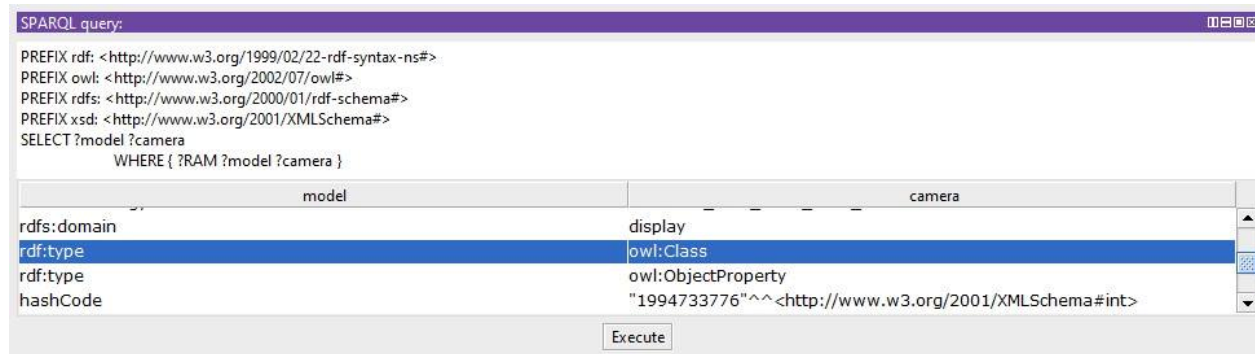
### News Article 2:

<https://drive.google.com/file/d/0B4VHwW192C9HZmQ2VTBCaHdsNTA/view>

### Jena Fuseki:



### Output Screenshot:

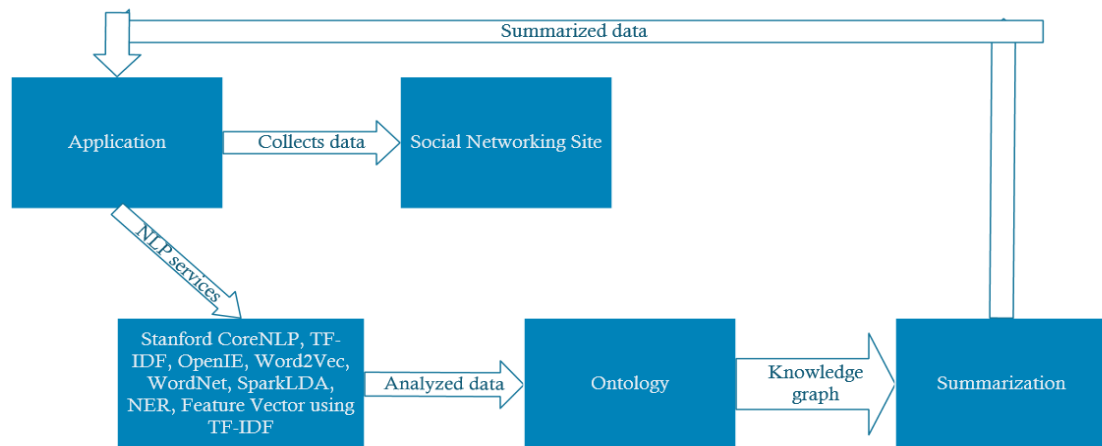


### C. Application:

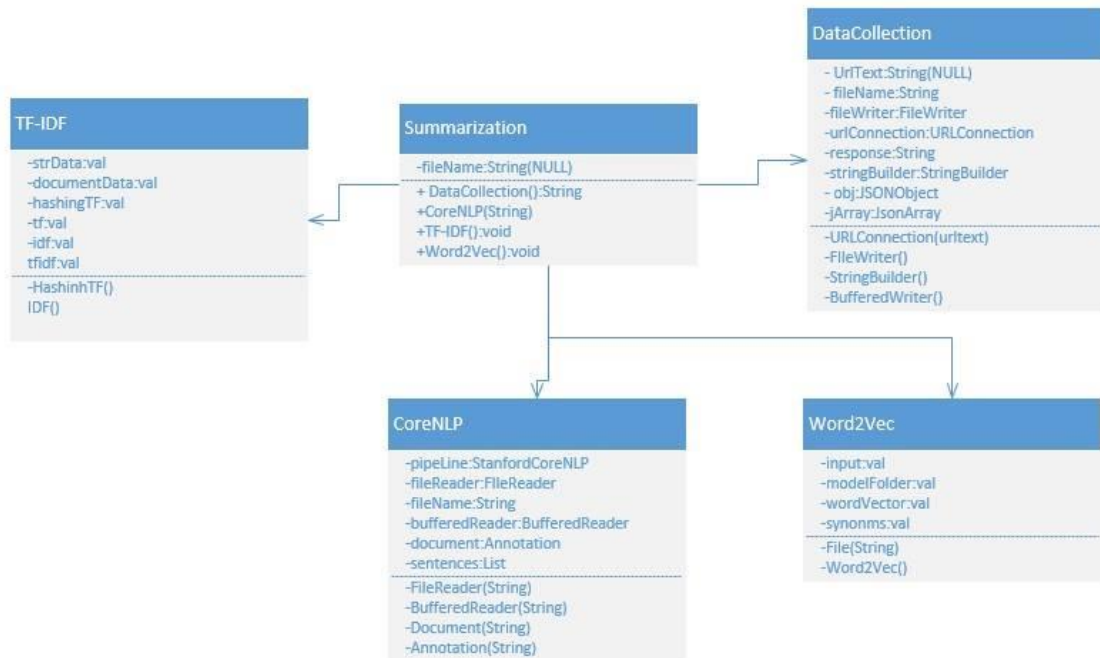
Summarization: Our project is the summarization of news articles from various news sources. Combine related topics together and summarize them as a one. Ontology helps use to relate a specific topic from different sources and joins them. By which we can summarize them easily.

### 3. Implementation Specification:

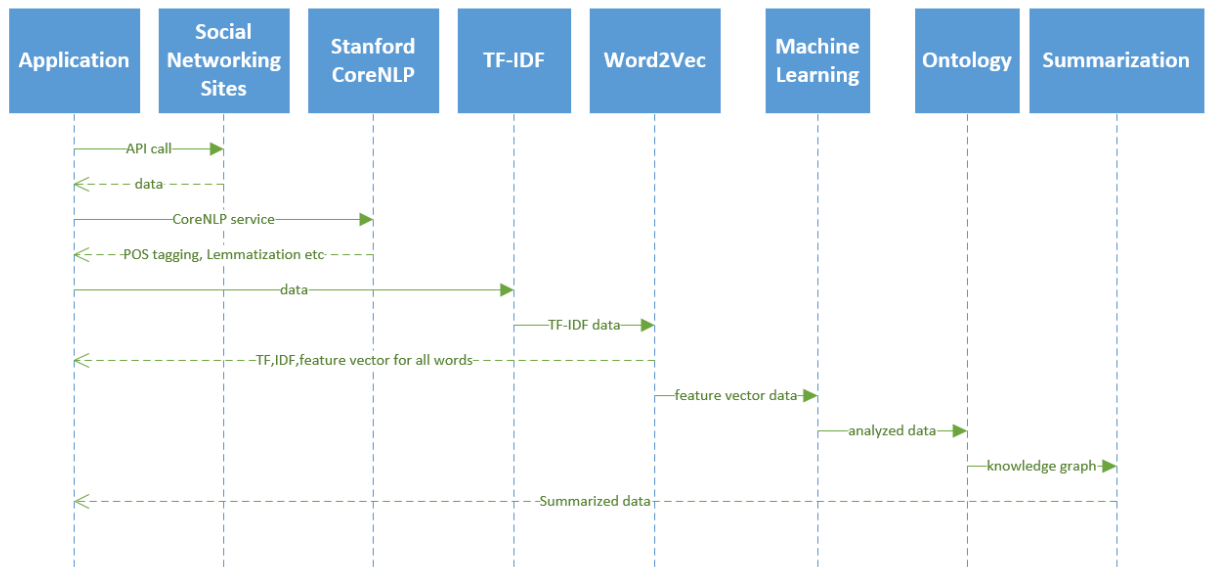
#### a. Architecture Diagram



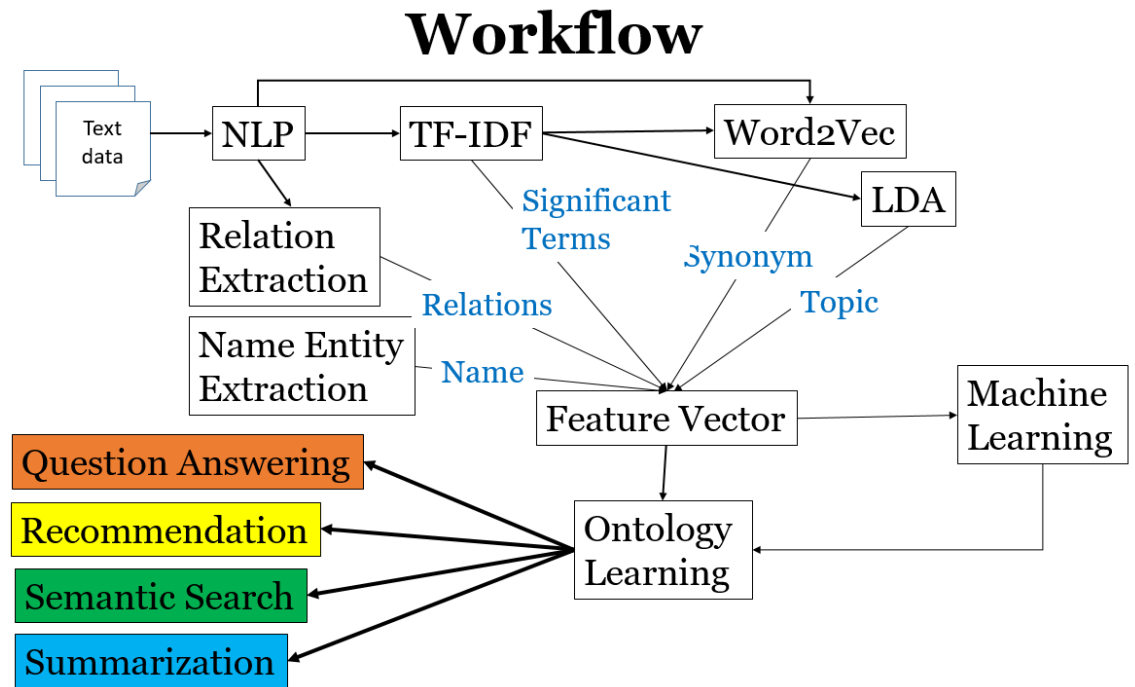
## b. Class Diagram



## c. Sequence Diagram



#### d. Workflow



#### e. Existing Services Used

- Stanford CoreNLP
- SparkNLP(TF-IDF)
- OpenIE
- Word2Vec
- SparkLDA
- Name Entity recognition
- WordNet
- Feature Vector
- RestAPI

#### f. new services/feature you implemented:

- Intellexer API

**g. Application:**

Our project is a java application which will summarize the data from various news sources and display as summary.

## 4. Results & Evaluation

### a. Accuracy:

- **Precision:** exactness – what % of tuples that the classifier labeled as positive are actually positive.

$$precision = \frac{TP}{TP + FP}$$

For our project, Precision = 0.5

- **Recall:** completeness – what % of positive tuples did the classifier label as positive

$$recall = \frac{TP}{TP + FN}$$

For our project, Recall=0.5

- **F measure:** harmonic mean of precision and recall

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

For our project, F measure=0.5

### b. Runtime Performance:

Preprocessing time: 0.728231933 sec

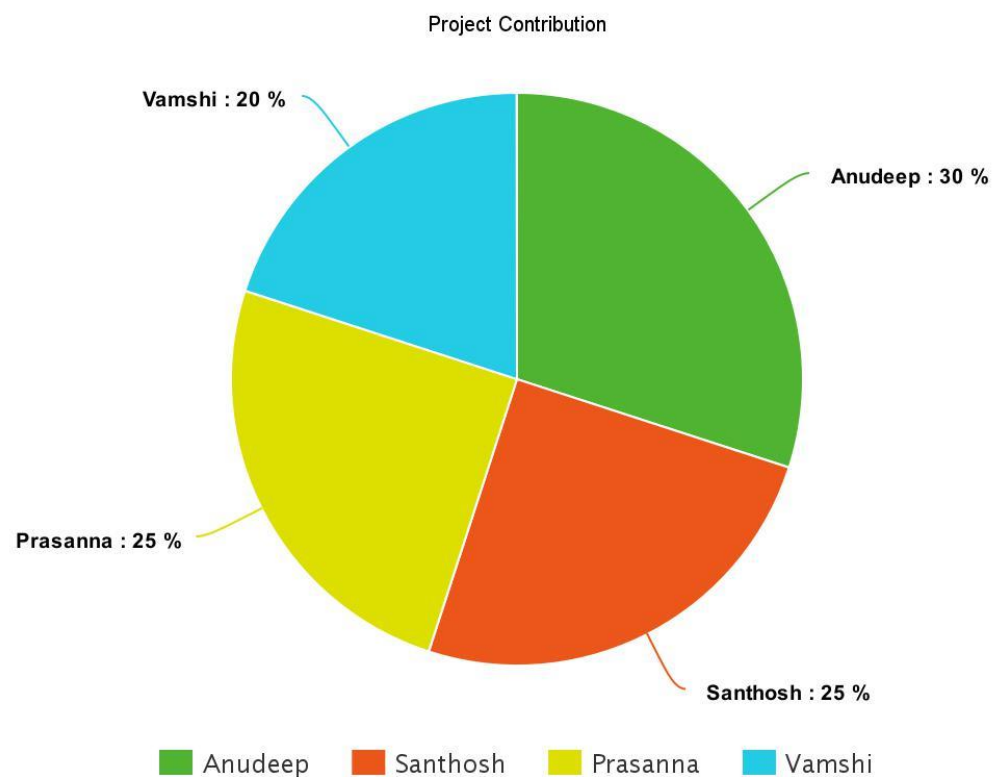
Training time: 2.975362895 sec

Compilation time in 21s 865ms

## 5. Project Management:

### a. Contribution

Name	Work
Santhosh Mohan Murarishetti	Data Collection using API (Real Time), Design of Application Workflow, WordNet, NER Implementation, Intellexer API
Prasanna Muppidi	CoreNLP, NGram, Word2Vec Implementation
Anudeep Pandiri	TF-IDF, SparkLDA, Ontology Implementation
Vamshi Rajarikam	Feature Vector implementation



meta-chart.com



## b. Github

URL: <https://github.com/murarishetty/KDM-SM2016-TEAM-9>

Jun 19, 2016 – Jul 25, 2016

Contributions: **Commits** ▾

Contributions to master, excluding merge commits

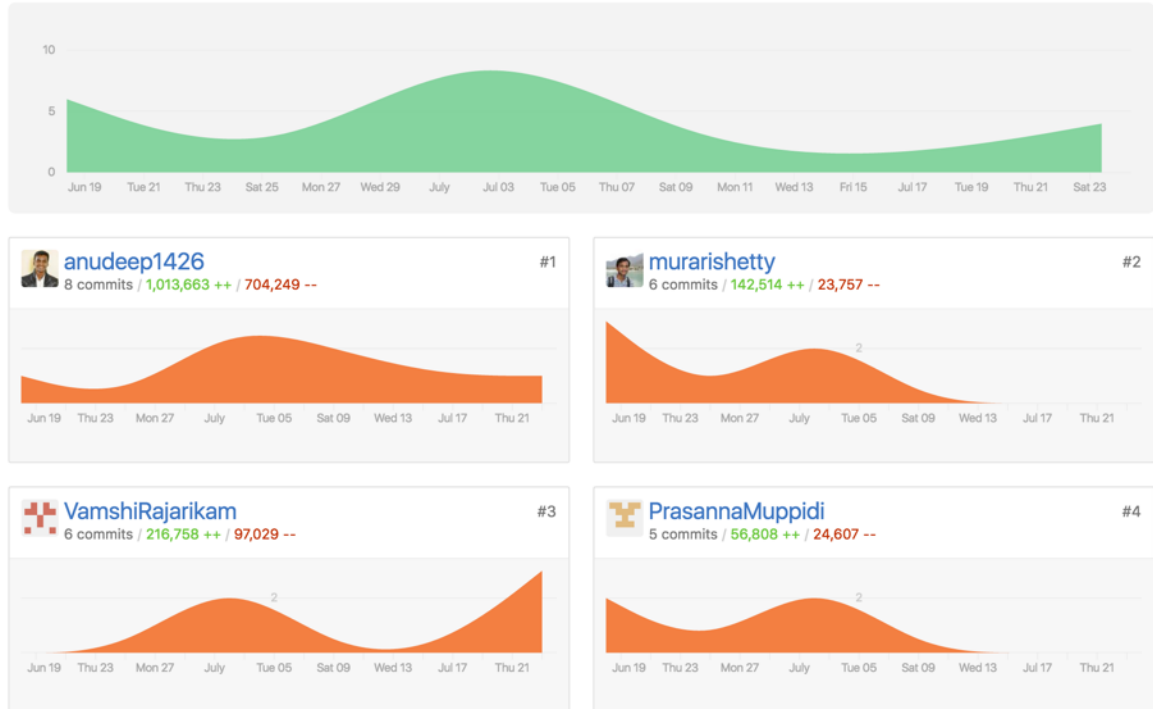


Fig: Code Contribution

### c. Zenhub

[Pull requests](#)
[Issues](#)
[Gist](#)
[↑ To Do](#)

[murarishetty / KDM-SM2016-TEAM-9](#)

Unwatch

 4
 

Star

 0
 

Fork

 0

[Code](#)
[Issues 10](#)
[Pull requests 0](#)
[Boards](#)
[Burndown](#)
[Wiki](#)
[Pulse](#)
[Graphs](#)

[Repos \(1/1\)](#)
[Show one](#)

Labels

Milestones

Assignees

New Issue

9 New Issues 4

0 Icebox 0

0 Backlog 0

5 In Progress 3

0 Review/QA 1

KDM-SM2016-TEAM-9 #21

video

2

KDM-SM2016-TEAM-9 #22

Report

1

KDM-SM2016-TEAM-9 #23

ontology

3 enhancement

KDM-SM2016-TEAM-9 #24

LDA

3 enhancement

KDM-SM2016-TEAM-9 #10

SparkLDA

2

KDM-SM2016-TEAM-9 #14

NER

2

KDM-SM2016-TEAM-9 #15

Phase 2 Report

1

KDM-SM2016-TEAM-9 #

CoreNlp

## 6. References

- <https://www.google.com/trends/>
- <https://www.quora.com/Does-Google-Trends-have-a-publicly-available-API>
- <http://spark.apache.org/documentation.html>
- <https://spark.apache.org/docs/1.1.0/mllib-feature-extraction.html>
- <http://stanfordnlp.github.io/CoreNLP/>
- <https://developer.usatoday.com/>