# ReDefine - Research Definitions Summarized

## Second Increment Report

Team #7 - COGNITOS

Team Members:

Nikhita Sharma (37), Dig Vijay Kumar Yarlagadda (47), Harsha Komalla (14), Sai Madhavi Sunkari (39)

---

## 1. INTRODUCTION

### Motivation:

Our motivation for this project stemmed from having to read dozens of research papers just to understand all the terms in a single research paper. There is no simple way of understanding a research article, the information from the internet sources like Wikipedia doesn't provide a clear insight into many technical concepts. Having experienced this first-hand, we realized the importance of a system which can provide a concise summary of a research article.

We aspire to extend this system as a knowledge graph, which can be used as a handy search tool by students to find articles, summaries of articles, authors, and domains. This system has the potential to be useful in many different ways: a beginner in a field can explore research papers in a particular domain, a search engine can crawl through the knowledge graph for more relevant search results, etc.,

### Objectives:

The objectives of our project are:

- Build a scalable system for summarization of research articles which would give a graphical representation of the contents of the research paper including the important key terms and the relation between them. This would help understand the contents of the paper and the topic of discussion.

- Create an interactive knowledge graph of research articles where the user can view a summarized representation research papers in a field of research along with the sub categories in a specific research field. It would also includes author, co-author, publication information, keywords in the paper and other related papers written by same author or co-author.

## Features:

- User can provide a corpus of research papers related to a field as input.
- Users are presented with an interactive graph UI, which can be explored to view the interactions between research categories under the research field, authors and papers written, co-authors and interaction between related articles
- The summary generated can be read by clicking on a particular paper title which would give a summarization of the contents of the paper.

## Expected Outcomes:

1. **Knowledge Graph:**

We expect this project to generate a Knowledge graph representing all the publications in a specific field of study selected based on the requirement of the user. This graph would provide important results about the different papers written in the field, the paper titles, information about author of the paper, co-authors' information, other related papers an author or co-author has written and important results about each paper itself like focus of a paper, key terms in the paper and other important information.

2. **Publication Summary:**

This project will also generate a text summary of each research paper/publication. A graphical representation of a research article would show the most important keywords in the article and relation between them to help understand the focus of research of the author. It would also show the major fields and topics of discussion. The summary will give an overview of contents of each paper to the end user, without the user having to read the complete research paper.

## 1. DOMAIN OF THE PROJECT:

Our project focuses on building a knowledge graph model using NLP and ML techniques to automatically generate summaries from a dataset of research articles. Due to the difficulty of finding sources for obtaining research articles, we are limiting our dataset to a few thousand articles. However, our system is based on Apache Spark and we are confident that it can scale to large volumes.

Further we are concentrated on improving the quality of summaries generated, we are not particular about generating output in different formats like text summaries or graphical user interfaces; the output will be generated as a knowledge graph.

## 1. DATA COLLECTION:

Finding an API which provides a dataset of entire research articles proved to be a difficult task, metadata of articles is available from many sources, but the actual full content of an article is not open sourced. We opted for using IEEE Xplore XML API.

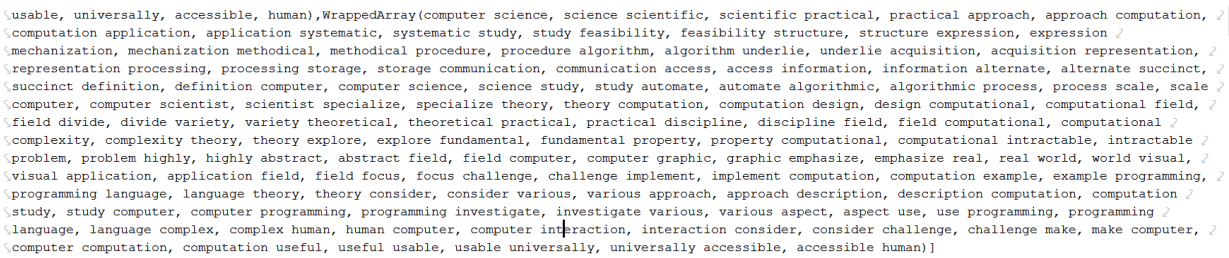
We have currently used IBM Watson PDF2Word conversion API to convert the research papers to text format. Initially we plan to perform summarization on 100 research articles and then extend to a larger dataset.

## 1. TASKS AND FEATURES IMPLEMENTED:

### a. NGram and Word2Vec

We performed tokenization, lemmatization and Stop words removal using an aggressive and updated stop words removal technique. We generated NGrams for the research articles as input. Below are the Word2Vec Vector values received for NGrams (BiGrams).

Bigrams created:




Word2Vec Vector Values with NGrams:

```
(dummies wiley,5.598421958998375)
(analysis technique,5.598421958998375)
([support analysis,5.598421958998375)
(propose framework,5.598421958998375)
(language implementation],5.598421958998375)
(volume november],5.598421958998375)
(tune digital],5.598421958998375)
(faculty comp,5.598421958998375)
([invesitage propose,5.598421958998375)
(hiveql sacrifice,5.598421958998375)
([issue november,5.598421958998375)
(sciences volume,5.598421958998375)
(functionality extensibility],5.598421958998375)
(educator school,5.598421958998375)
([cache mechanism,5.598421958998375)
([complexity today,5.598421958998375)
([paul harris,5.598421958998375)
(well datum],5.598421958998375)
```

### TFIDF and Word2Vec

Performed tokenization, lemmatization and Stop words removal using an aggressive and updated stop words removal technique. The keywords were extracted and the top 20 words were selected using the TFIDF method. These top words were given input to the Word2Vec Model.

```
TOP WORDS:

(location,22.310346711737093)
(dsm,19.47013780182233)
(val,18.07385740980107)
(singleton,17.848277369389674)
(parent,17.848277369389674)
(integration,16.697549079582547)
(recovery,14.602603351366746)
(prefer,14.602603351366746)
(compile,9.735068900911164)
(println,9.735068900911164)
(comparable,9.735068900911164)
(flatmap,9.735068900911164)
(scalable,9.735068900911164)
(hand,9.735068900911164)
(high,9.735068900911164)
(grad,9.735068900911164)
(group,9.735068900911164)
(func,9.735068900911164)
(resend,9.735068900911164)
(size,9.735068900911164)
```

Synonyms:

```
scalable  :
[vector,0.9988365059111689]
[state,0.9945299179615035]
[product,0.9921028498670936]
[input,0.9904648359607394]
[keyword,0.9898994283669985]
[solve,0.9898756730889848]
[note,0.9896879196798526]
[utilize,0.9878361197954827]
[filter,0.9845374755616015]
```

## b. Relation Extraction

Performed Relation extraction on the words received after the process of tokenization, lemmatization, Stop words removal and Top Words generation. We used Stanford CoreNLP OpenIE to extract relations between keywords.

1. Relation Extraction Results using CoreNLP OpenIE:

```
[1.0  degree  being reused across mobile apps, 1.0  degree  being reused across apps, 1.0  proportion  in  category, 1.0  degree  being  reused][1.0
 degree being reused across mobile apps, 1.0  degree  being reused across apps, 1.0  proportion  in  category, 1.0  degree  being  reused][1.0 degree
 being reused across  mobile apps, 1.0  degree  being reused across apps, 1.0  proportion  in  category, 1.0  degree  being  reused][1.0 degree
 being reused across  mobile apps, 1.0  degree  being reused across apps, 1.0  proportion  in  category, 1.0  degree  being  reused][1.0 category
 of reused  PCSR, 1.0  proportion  has complement obtain][1.0  category  of reused  PCSR, 1.0  proportion  has complement obtain]


[1.0  total number  =  −, 1.0  number  =  −][1.0  total number  =  −, 1.0  number  =  −]

[1.0  reuse  is  high][1.0  Figure 1  shows  PCSR for category, 1.0  Figure  shows  PCSR for category][1.0  Figure 1  shows  PCSR for category,
 1.0  Figure  shows  PCSR for category]

[1.0  62.72 percent  with  PCSR of 63.59 percent, 1.0  Comics category has lowest PCSR, 1.0  Comics category has PCSR][1.0  62.72 percent  with
 PCSR of 63.59 percent, 1.0 Comics category has lowest PCSR, 1.0  Comics category has PCSR][1.0  62.72 percent  with  PCSR of 63.59 percent,
 1.0  Comics category has lowest PCSR, 1.0  Comics category has PCSR][1.0  classes are unique to  mobile app, 1.0 few classes are unique to  app,
 1.0  few classes are unique to  mobile app, 1.0 few classes are unique, 1.0 classes are unique to  app, 1.0  classes are unique][1.0 classes are
 unique to  mobile app, 1.0 few classes are unique to  app, 1.0  few classes are unique to  mobile app, 1.0 few classes are unique, 1.0 classes are
 unique to  app, 1.0  classes are unique][1.0 classes are unique to  mobile app, 1.0 few classes are unique to  app, 1.0  few classes are unique
 to mobile app, 1.0 few classes are unique, 1.0 classes are unique to  app, 1.0  classes are unique][1.0 classes are unique to  mobile app, 1.0 few
 classes  are unique to  app, 1.0  few classes are unique to  mobile app, 1.0 few classes are unique, 1.0 classes are unique to  app, 1.0  classes
 are  unique][1.0 classes are unique to  mobile app, 1.0 few classes are unique to  app, 1.0  few classes are unique to  mobile app, 1.0 few
 classes  are unique, 1.0 classes are unique to  app, 1.0  classes are unique][1.0 classes are unique to  mobile app, 1.0 few classes are unique
```

2. Relation Extraction Results using CoreNLP OpenIE with Clause Extraction:

```
nmod:for(educated-109, instance-112)
nummod(percent-115, 33-114)
nsubj(had-116, percent-115)
acl:relcl(instance-112, had-116)
det(experience-120, some-117)
amod(experience-120, graduate-118)
compound(experience-120, school-119)
dobj(had-116, experience-120)

root(ROOT-0, attracted-59)
aux(attracted-59, has-58)
dobj(attracted-59, thousands-60)
case(survey-65, of-61)
compound(survey-65, developers-62)
compound(survey-65, .2-63)
det(survey-65, A-64)
nmod:of(thousands-60, survey-65)
case(developers-69, of-66)
nummod(developers-69, 352-67)
compound(developers-69, app-68)
nmod:of(survey-65, developers-69)
```

### c. WordNet

We tried using wordnet to generate Synonyms and relation between the Top Words based on the TFIDF calculation. Below are the results for the same.

Results for Synonyms and Relation for Top TFIDF Words:

```
Synonyms for learn (pos: v)
absorb
account
acquaint
acquire
advise
alternate
announce
apprise
apprize
ascertain


Relationship between: domain and major
domain and major are related by a distance of: 0.25
Common parents:
entity
```

### d. Topic Discovery

Performed the NLP process of tokenization, lemmatization and Stop words removal using an aggressive and updated stop words removal technique. The results after aggressive Stop words removal were improved greatly. Below are the results for topic discovery.

Results for Topic Discovery:

```
20 topics:
TOPIC 0
dataset 0.017526799933379276
spark   0.017489615266341154
variable    0.011559182597457212
rdd 0.009880042682733809
file    0.009421892830506694
operation   0.009391484575592017
```

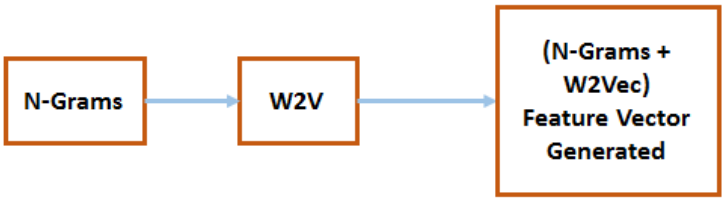
```
TOPIC 1
rdd 0.022523847657914535
spark   0.01988864458995418
node    0.01856463257447125
dataset 0.017563466086173585
variable    0.0167100004262386
partition   0.011374287336147693
operation   0.011155729577428037
file    0.009764023741866794
cache   0.009309112653297674
memory  0.008331650737572483
worker  0.00825846507562744
broadcast   0.008059326918577924
function    0.008009964446935481
parallel    0.007803224278069412
```

```
TOPIC 2
application 0.036515863821119994
system  0.031218520113038762
datum   0.03101560714568022
model   0.029451783937834042
flow    0.02729486723630875
fault   0.02596965143782975
machine 0.025882995908724026
set 0.02196057408931425
mapreduce   0.021193575618327194
tolerance   0.018308692843550245
data    0.01600468214935767
cluster 0.01427973475742479
```

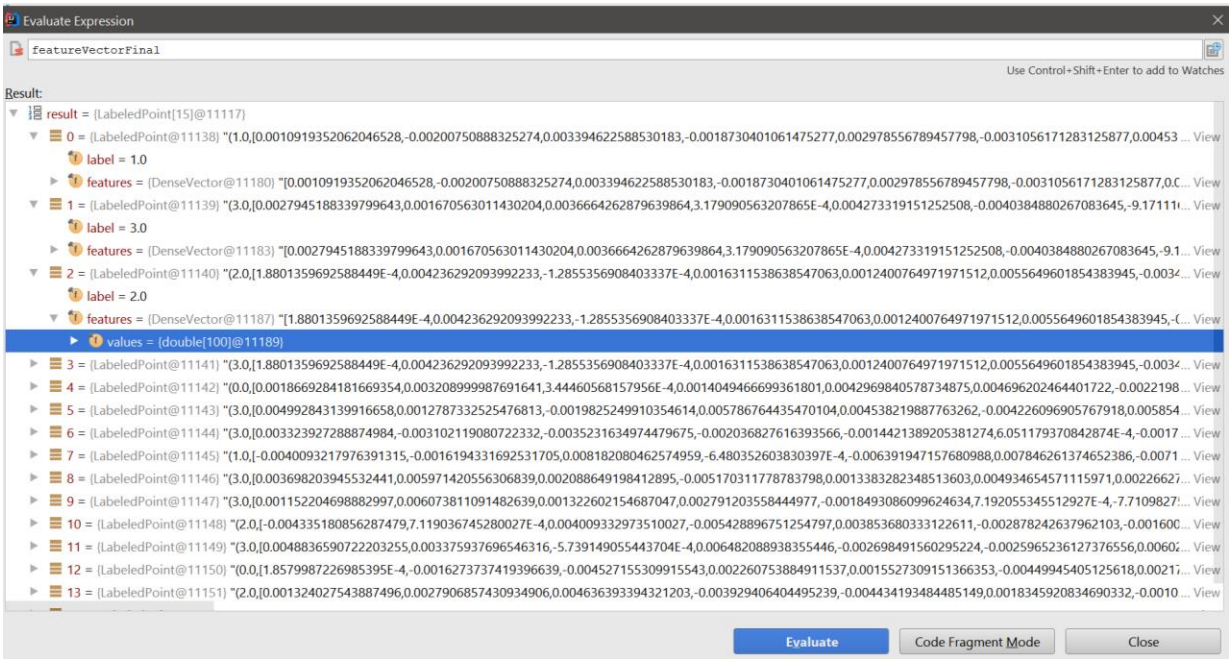
## e. Feature Vector for Machine Learning

1. Feature Vector 1:

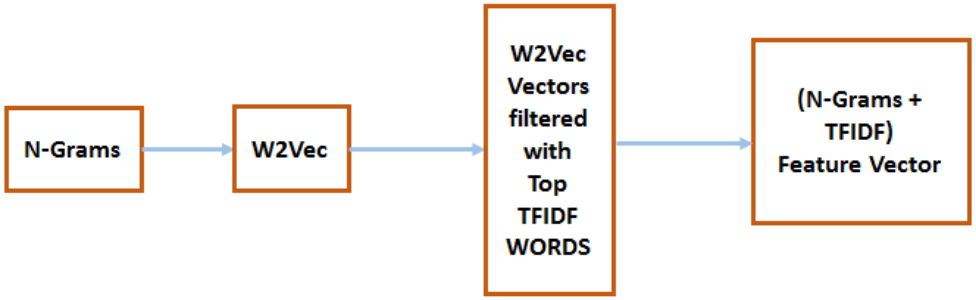
Below is the feature vector for Word2Vec model. The Word2Vec was performed after generating NGrams and this result was input to the Word2Vec computation.



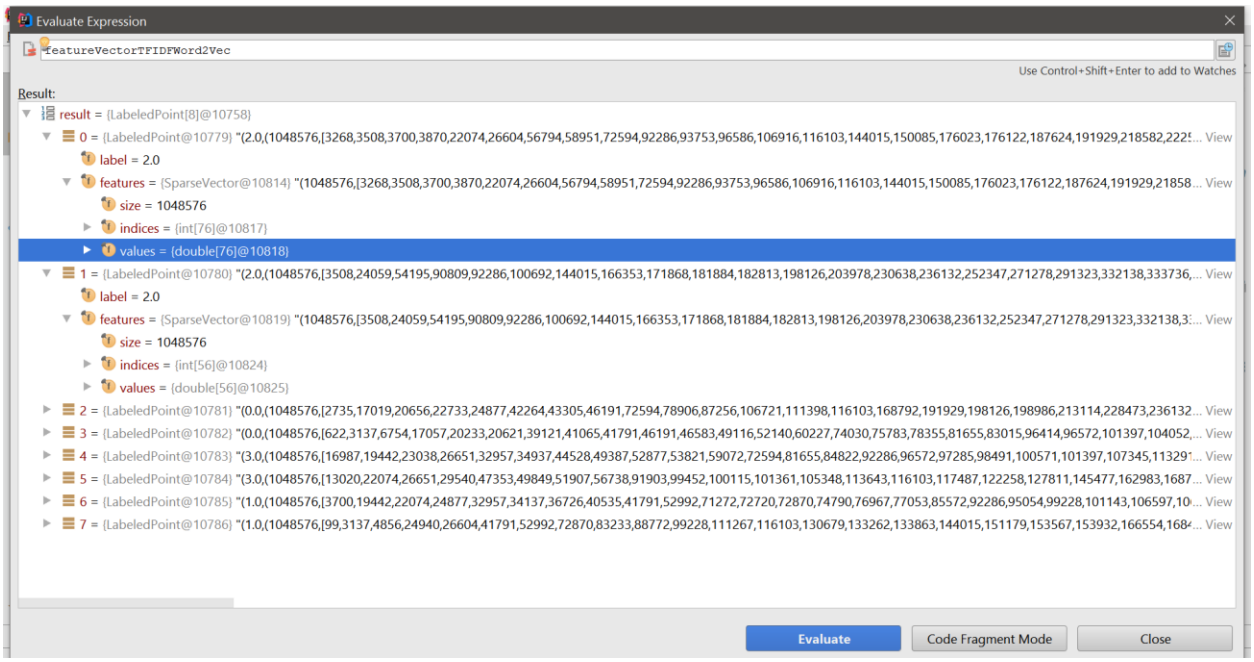

Feature Vector Results for N-Grams+ Word2Vec



2. Feature Vector 2:




Feature Vector Results for N-Grams+ Word2Vec + Vectors filtered with TFIDF:



1. **IMPLEMENTATION SPECIFICATION:**

## a. Software Architecture & UML Model

### System Software Architecture:

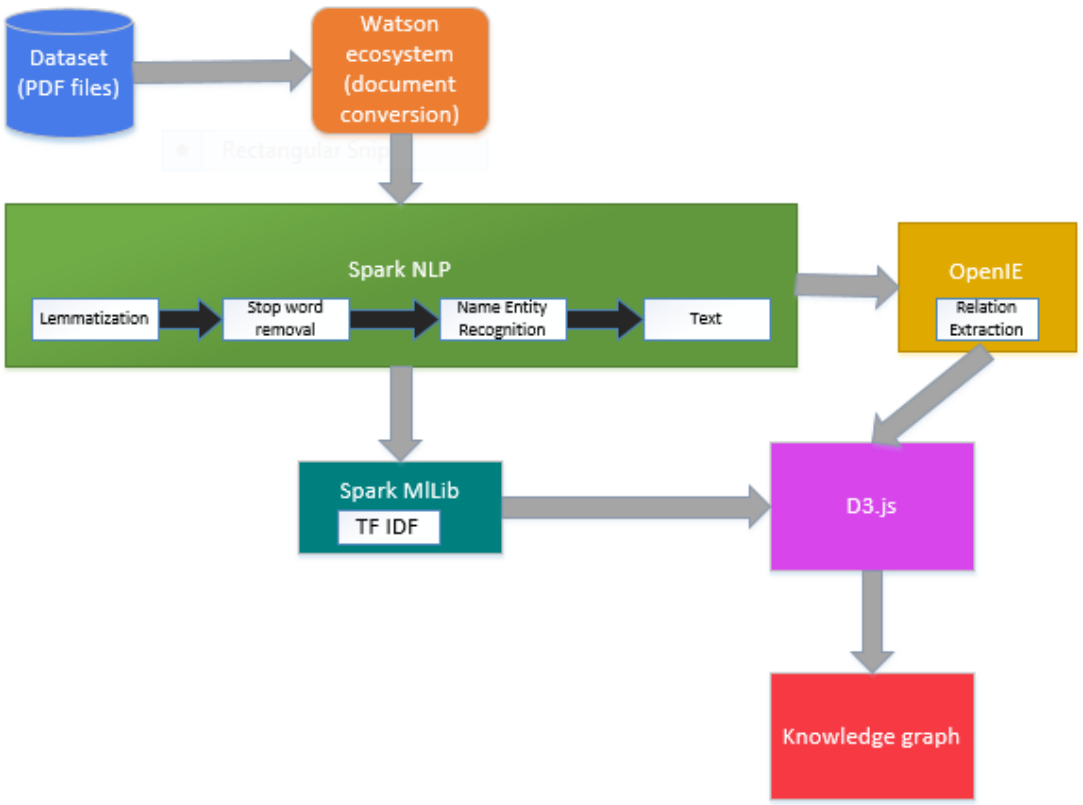



Fig 1: Architecture Diagram

### Architecture Components:

1. Watson Ecosystem SDK (PDF2Word):

Watson Ecosystem SDK's PDF2Word Converter is one of the services provided by IBM to convert a PDF or a HTML document into plain text or JSON answer units. This document conversion tool detects the words using Optical Character Recognition (OCR) and also supports content in various languages.

2. Spark NLP:

Natural Language Processing is a part of Artificial Intelligence that has the ability to understand human speech. NLP tasks such as Sentence segmentation, tokenization, lemmatization, POS tagging, Named Entity extraction and Coreference resolution are provided thorugh the Spark Core NLP package.

3. OpenIE (Relation Extraction):

OpenIE is used to extract relation triples from the document representing subject, relation and object triples. First a sentence is divided into clauses and each clause is minimized to small sentences and each sentence is divided into triples. We have used multiple OpenIE versions and extensions to extract relations like Stanford CoreNLP OpenIE, Ollie, Reverb and AllenAI OpenIE. We are planning to select the method which gives best triples results.

4. SparkMlLib:

Spark's Machine Learning Library is a framework on top of Spark Core. It includes learning algorithms and utilities -regression, clustering and collaborative filtering and higher-level pipeline APIs, which makes machine learning easy and scalable. We will be using this package for Feature Vector generation and Machine learning

5. D3.js:

D3.js is a JavaScript library which provides dynamic visualizations using SVG, HTML5 and CSS standards. SVG-objects are created by pre-built JavaScript functions and styled using CSS. Rich graphic charts are resulted on JSON/CSV as input file by using simple D3.js functions and SVG-objects. We will be using this tool to represent our summary and Knowledge Graph.

## Class Diagram:

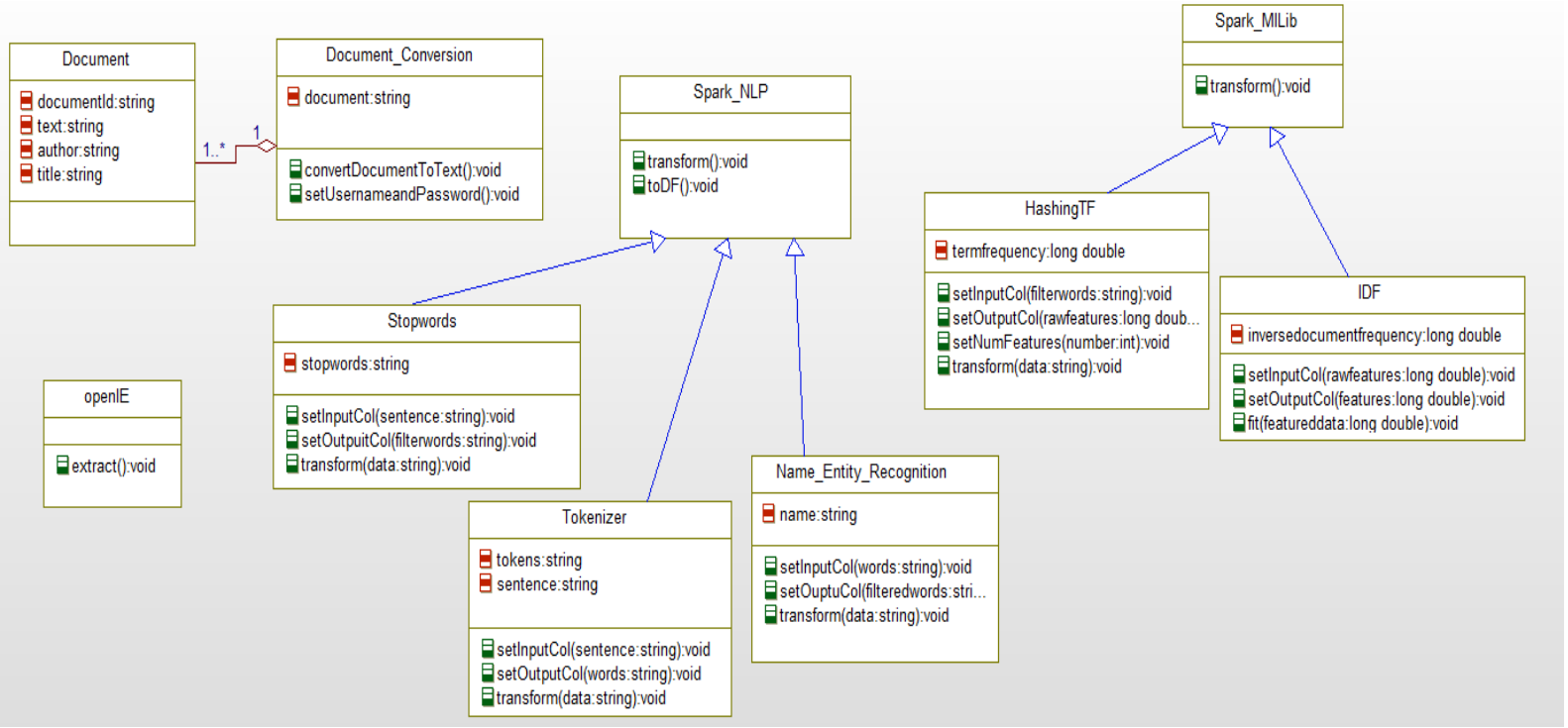



Fig 2: Class diagram

## Sequence Diagram:

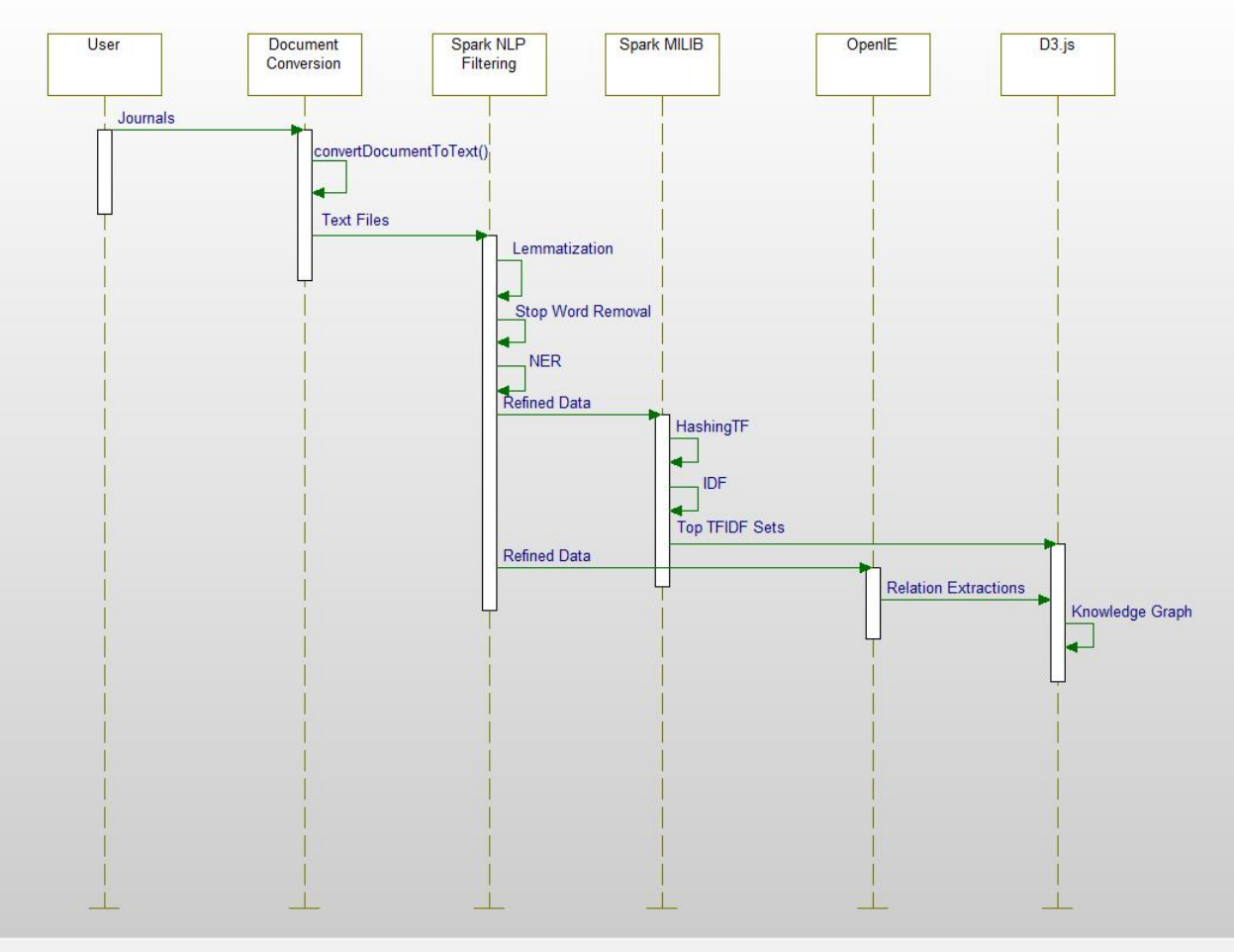



Fig 3: Sequence Diagram

Use Case Diagram:

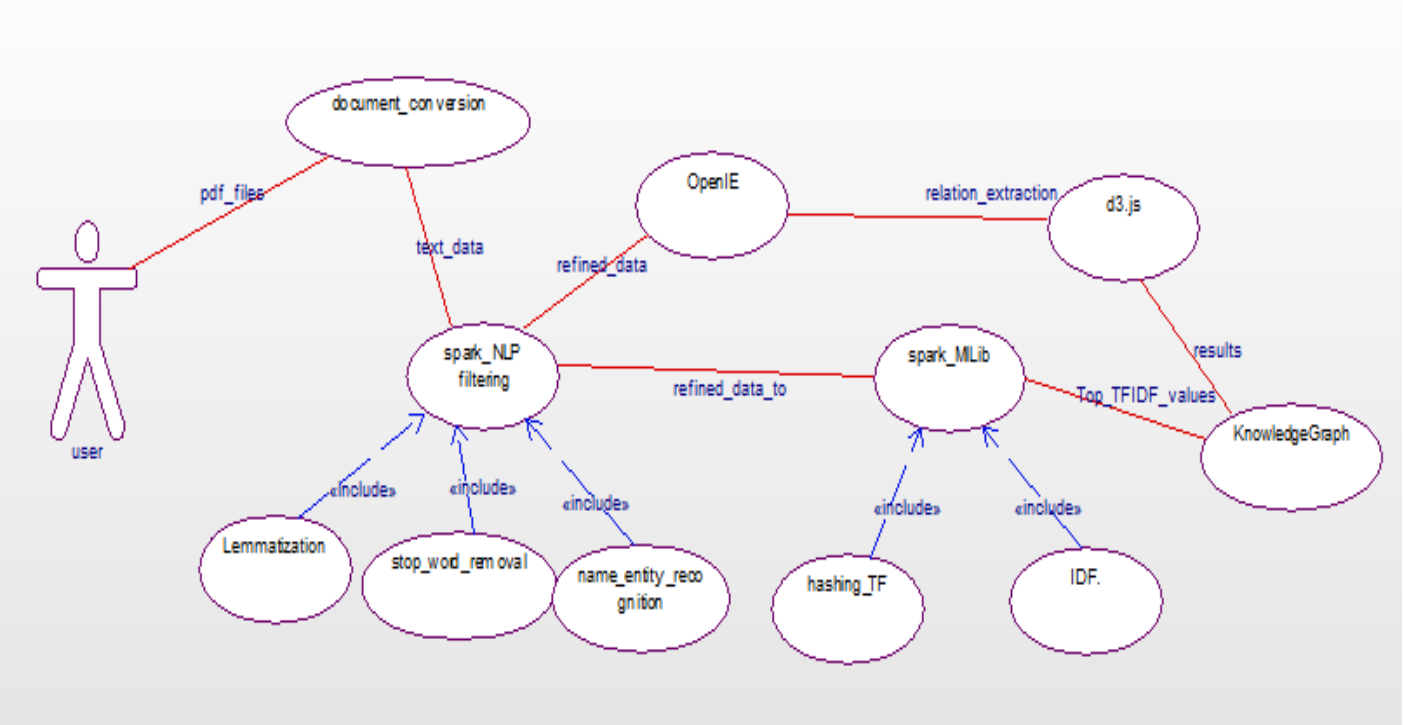



Fig 4 : Use case diagram

## b. System Workflow

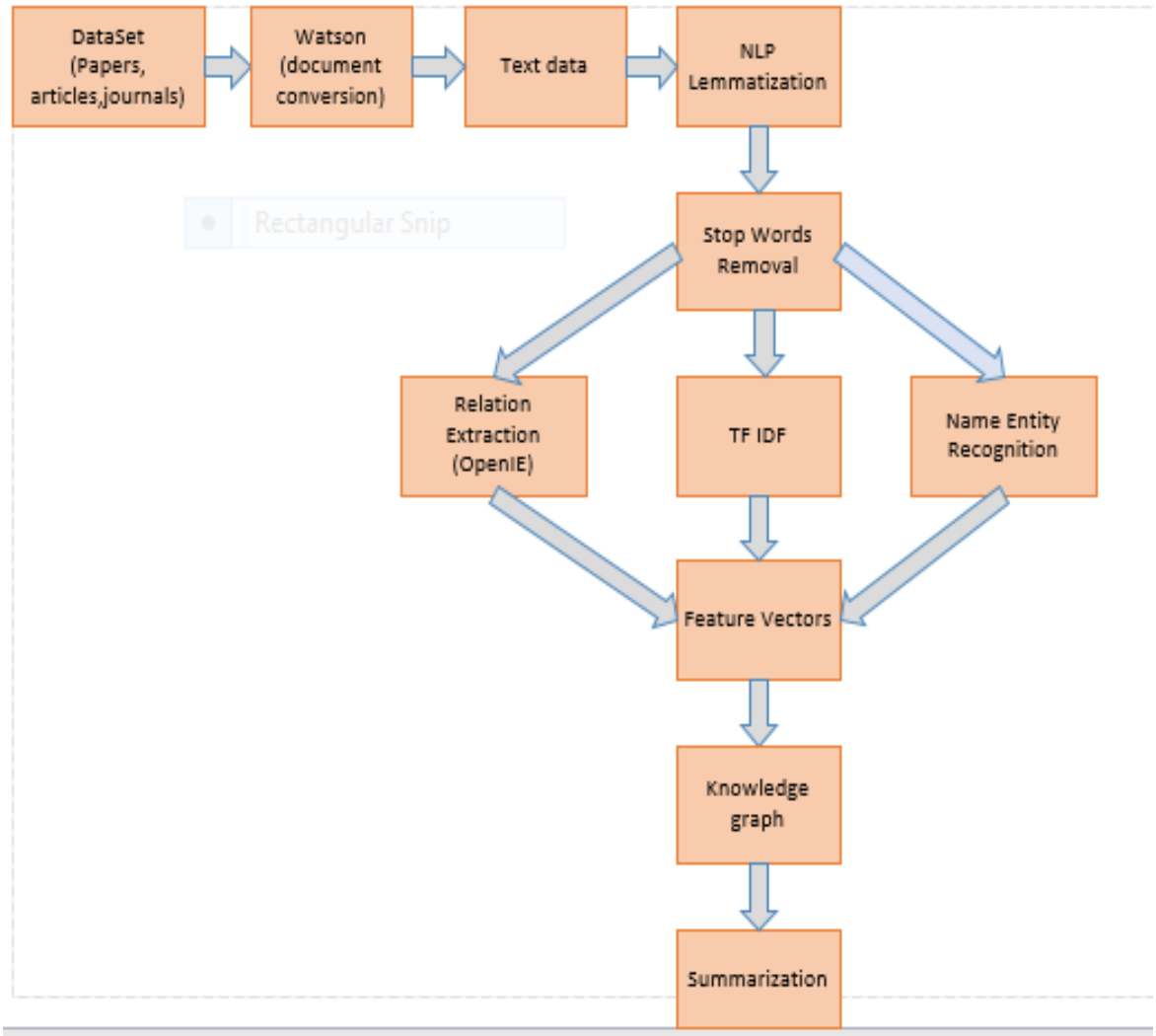



Fig 5: Workflow Diagram

### c. Existing Services/APIs:

ReDefine uses different services and APIs for NLP and Information Extraction:

1. Document Conversion API in Watson Developer Cloud Java SDK using IBM Bluemix, for converting research documents in image or pdf formats into readable text files.
2. Open IE for Relation Extraction
3. D3.js for visualizing knowledge graph

### d. New services implemented:

1. Created module to pick a text research paper file as input and perform NLP and Information Extraction on the data
2. Performed an aggressive and updated stop words removal technique. The results after aggressive Stop words removal were improved greatly.

1. **PROJECT MANAGEMENT:**

## a. Team members contribution: (overall)

Nikhita Sharma - CoreNLP and Topic Discovery - 25%

Dig Vijay Kumar Yarlagadda - Relation Extraction - 25%

Harsha Komalla - Extending TFIDF and Text extraction - 25%

Sai Madhavi Sunkari - Extending Word2vec - 25%

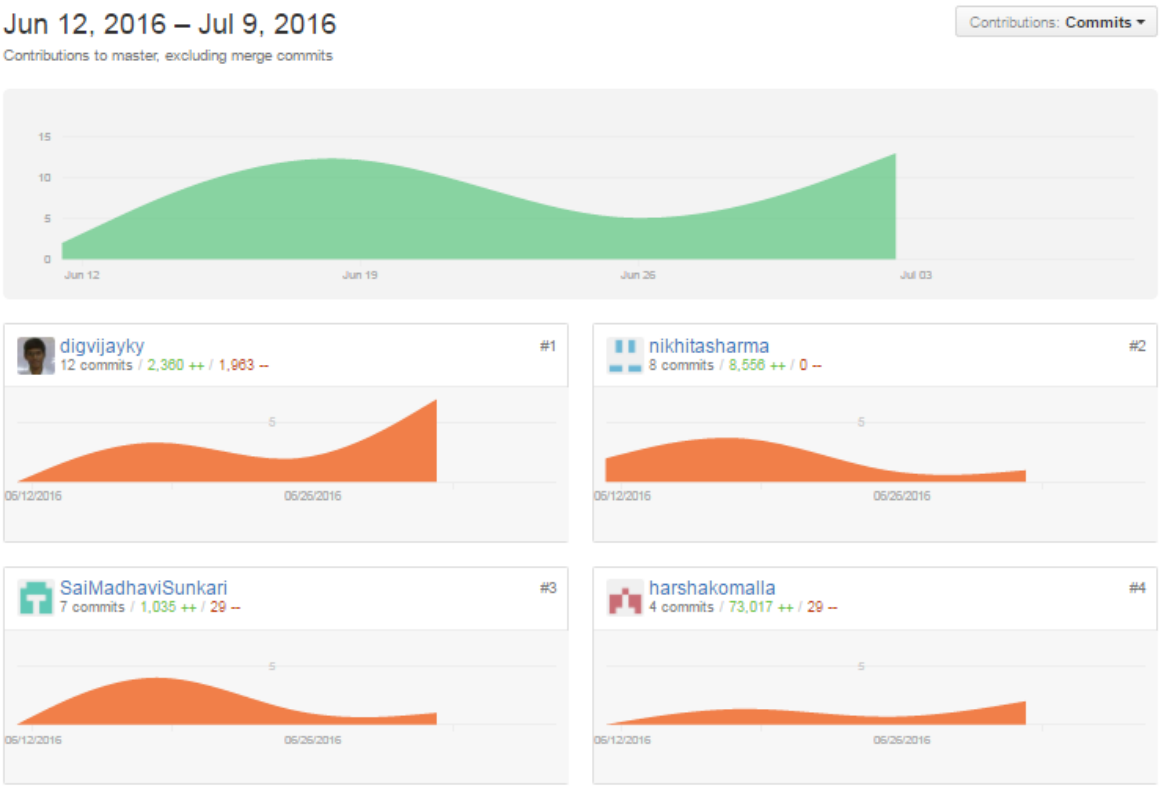
## b. Zenhub and Github URL/Statistics

Project GitHub repository:

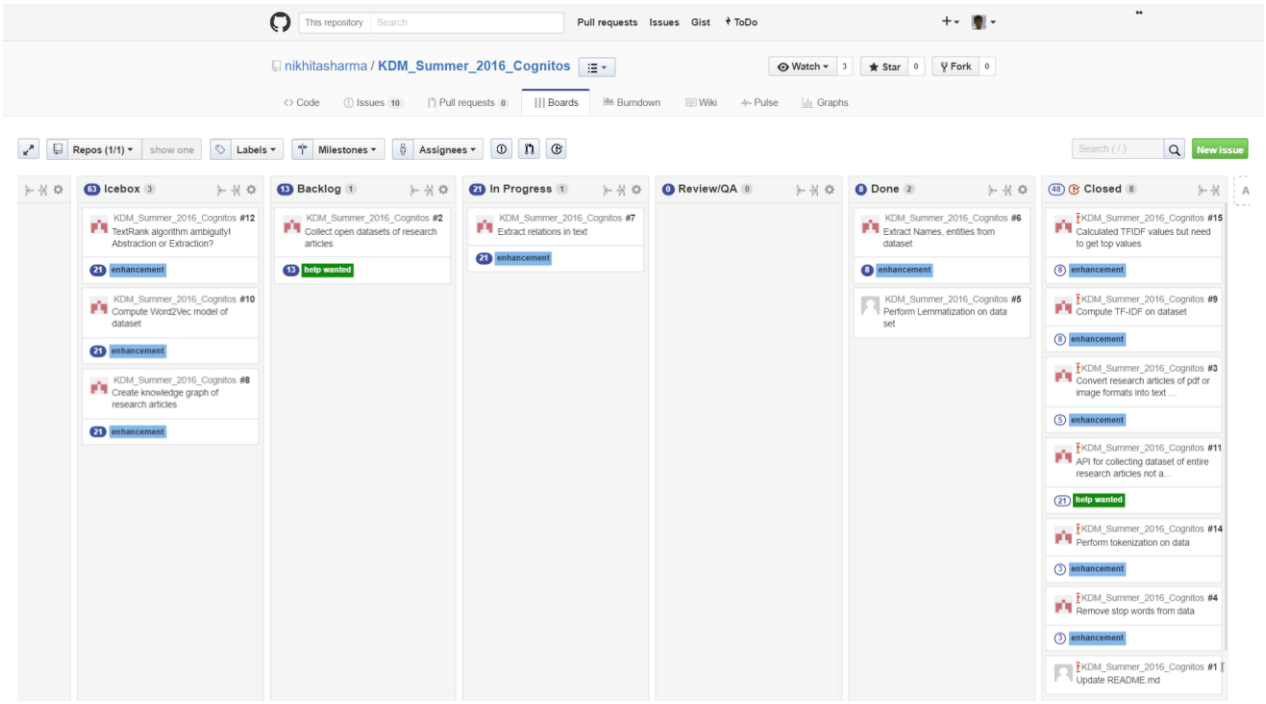
https://github.com/nikhitasharma/KDM_Summer_2016_Cognitos

Zenhub statistics:

Team members contribution:



Zenhub Boards:



Burndown Chart:


## c. Concerns or Issues:

- The main issue we faced in developing the system was that we are unable to access entire research article datasets. Metadata of research articles is available for access, but the actual content of a research article is not accessible. Microsoft Academic Search API used to provide a good interface for such data, but it is deprecated.

- Another issue we faced is while performing optical character recognition. Tesseract provides a good interface for performing OCR, but it uses up most of our computing resources. So, we chose Document Conversion API in Watson Developer Cloud Java SDK, running OCR on IBM Bluemix, thereby transferring the computational load to the cloud.

- In a knowledge graph, relationships between entities play a key role in understanding the gist of the concept discussed in a paper. In our case, after removing stop words from the data, we are performing lemmatization. We observed that this resulted in losing some important relationships between key terms. Since TextRunner values verb forms, it will not deliver sensible results on lemmatized dataset. We are mulling over this issue and trying with different articles to assess the impact of lemmatization on results generated by TextRunner.

### d. Future Work:

- The system is designed to be scalable, but never been tested for large datasets due to lack of data and computing resources.
- Also, our knowledge graph can be further extended to represent different sub-fields in a field of research.

**REFERENCES:**


[1]. http://apache.org/
[2]. http://spark.apache.org/docs/latest/mllib-guide.html
[3]. https://spark.apache.org/docs/1.3.1/api/java/org/apache/spark/rdd/RDD.html
[4]. https://spark.apache.org/docs/1.5.1/api/java/org/apache/spark/sql/DataFrame.html
[5]. https://spark.apache.org/docs/1.1.0/mllib-feature-extraction.html
[6]. https://github.com/watson-developer-cloud
[7]. https://github.com/stanfordnlp/CoreNLP
[8]. http://nlp.stanford.edu/software/openie.html
[9]. http://spark.apache.org/docs/latest/mllib-feature-extraction.html
[10].http://openie.allenai.org/
[11].http://reverb.cs.washington.edu/
[12].http://knowitall.github.io/ollie/
[13].http://www.sciencedirect.com/
[14].https://journalofbigdata.springeropen.com/
[15].http://bigdata-madesimple.com/research-papers-that-changed-the-world-of-big-data/
[16].http://www.engpaper.net/big-data-research-papers-2013-2014.htm
[17].http://www.journals.elsevier.com/big-data-research/recent-articles
[18].https://www.researchgate.net/directory/publications
[19]. Open Information Extraction from the Web Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead and Oren Etzioni, Department of Computer Science and Engineering University of Washington.