Text Data Summarization of News articles

Department of Computer Science

Santhosh Mohan M,
Anudeep Pandiri,
Prasanna Muppidi,
Vamsi Rajarikam

Abstract:

In this paper, we present about summarization of text data of news articles using Apache Spark framework. Summarization is a very useful task [3] since it saves time, gives inshort information about data. This requires collection of data, its classification and then semantic analysis on it. Visualization of data is made using onto graphs which gives hierarchy of classes, relationships between objects and individuals. There is a common workflow for summarization, sematic search, recommendation system and question answering systems to some extent. We have used many natural language processing techniques such as TF-IDF, LDA, NER, etc., and also few machine learning algorithms such as naïve Bayes, random forest, etc., for analysis and classification of data. We are going to present workflow, architecture we used and few evaluation results.

Keywords:

Text summarization, Apache Spark, NLP, Scala, Protégé, WebOwl, Apache Fuseki, Intellexer, NewYork Times.

1 Introduction:

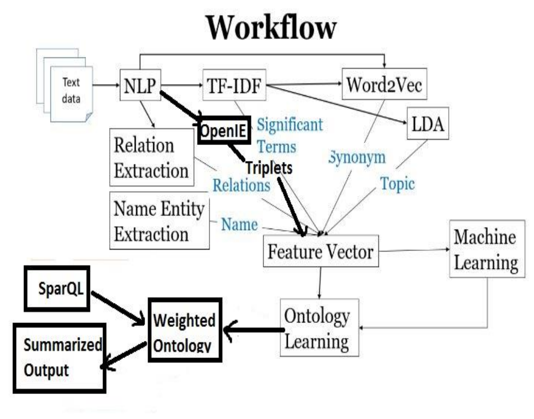
In the recent days, the usage of internet has increased exponentially from 1 billion users in 2005 to 4 billion users in 2015 [2]. The data usage in the internet is mostly unstructured data i.e., it does not have any schema, attributes and it does not contain any table or labelled data. Earlier it is very hard to analyze those data since it need lots of computation power and artificial intelligence which makes difficult to implement, classify and analyze the data. Originally artificial intelligence was introduced in late 1970s but it was not popular until the beginning of early 2000 with the introduction of deep learning and machine learning algorithms [4]. By the end of 2010, artificial intelligence and deep learning was popular and many applications of it are used by many parts of the world. With the advancements in the technology, computation has become cheaper, many parallel processing frameworks have been evolved. For example, Apache Hadoop, Apache Spark, OpenGL, etc., are some of the most used and popular frameworks for easy and flexible parallel computing. So we used Apache spark framework in the Intellij Idea IDE. Section 2 introduces about related work, section 3 explains about the approach we followed, whereas in section 4 we have mentioned implementation details, followed by evaluation results in section 5 and finally concluded the project in section 6.

2 Related Work:

Though, this is not a new idea research is going on from decades with improvements in accuracy, speed, precision, etc., New technique was proposed by Nagwani [6] where he calculates the shortest path for the graphs by using algorithms for the sentences and builds a graph. We though to implement similar concept by using Ontology with ranks to each object based on likelihood and how important the triplet in the document. You can also find similar work of text summarization by university of Michigan.

3 Proposed Solution:

To get summarized output, raw data has to be processed in many stages for relation extraction, triplets, lemmatization, topic discovery, classification, Name Entity recognition and a graph building using ontology. Finally, we used SparQL for querying the Ontograph for the corresponding output. Here each process has its own importance and few processes are optional.



NLP:

Natural Language Processing is a computer program useful for extracting grammatical structures such as parts of speech, Lemmatization, tokenization, correlation among sentences. This is first step of processing raw data to get tokenized output which is useful for further processing such as getting term frequency [7].

TF-IDF:

TF-IDF is nothing but it given information about how important a word in the entire corpus. To calculate TF-IDF, first we need to calculate term frequency, document frequency, inverse document frequency and finally we get TF-IDF by multiplying TF and IDF. Term frequency is the number of times a particular word occurs in the given document. We cannot depend only on TF because it just gives the count of each word, for instance words like “a”, “the” may appear many times which doesn’t convey any meaningful information about the document. So we need to calculate inverse document frequency which gives the count of a particular word appears in each document. So by multiplying both, it gives meaning information regarding the document.

$$IDF(t,D) = \log \frac{|D|+1}{DF(t,D)+1},$$

$$TFIDF(t,d,D) = TF(t,d) \cdot IDF(t,D).$$

Where t corresponds to term and D corresponds to document.

Word2Vec:

Words can be represented in the vector format. The main advantage of model is similar words are grouped together which makes simpler and easier for building natural processing applications. By using these data feature vector can be built. We use some model like skip-gram model to build feature vector

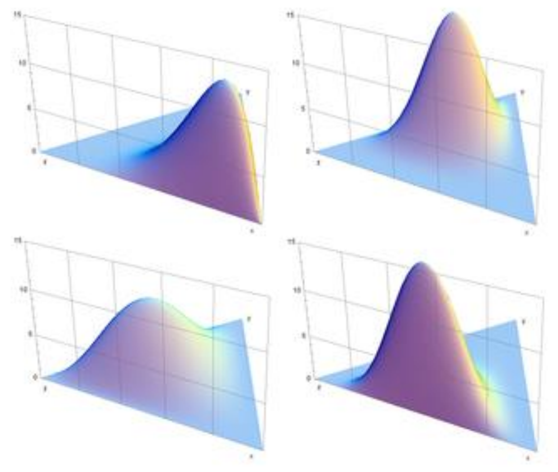
$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j=-k}^{j=k}\log p(w_{t+j}|w_t)$$

$$p(w_i|w_j) = \frac{\exp(u_{w_i}^{\top} v_{w_j})}{\sum_{l=1}^{V}\exp(u_l^{\top} v_{w_j})}$$

It can be useful to display synonyms, hyponyms for the specified word by using the similarity of words. To increase speed, we used softmax model, by which complexity has been reduced from logp(wi/wj) to O(log(V)).

LDA:

Latent Dirichlet Allocation (LDA) is a way of discovering or grouping sentences to different topics. LDA takes the sentences as input which is a mixture of many different topics. By using “EM” algorithm probability for a word is calculated and based on the probability value, it is grouped into different topics. The main disadvantage of LDA is we need to predefine the number of topics and number of iterations. Dirichlet is nothing but a parameter α corresponding to each topic j, so we calculate αj for each sentence assigns to corresponding topic based on the value.



Here, it indicates that there are 4 different topics [10] where y axis indicates Dirichlet score. We predefined 2 topics such as design and features and max iterations as 20 and algorithm “EMLDAOptimizer”.

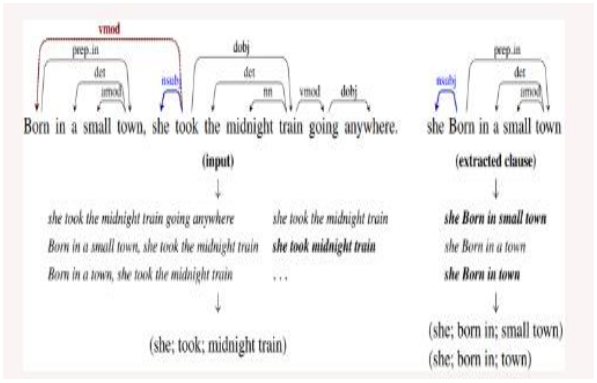
Name Entity Recognition:

Words in sentences may be company names, person names, currency, date, duration, protein names, gene, etc., Name Entity Recognition is nothing but detecting the name entity [12] for that particular word. This can be done by using Stanford Core NLP API. It is a java implementation which takes sentence as input and returns Name Entity for that word. It may not give Name entity for all the words, so whenever absence of entity, it returns zero for the corresponding word.

For example, “Santhosh is pursuing Masters at UMKC in Kansas City”. The output is as follows [Name, 0, 0, 0, 0, 0, Organization, Location]. But we should not only depend on the NLP API results, we have to define our own NER tags by using domain based words for the domain you are working. I have collected some tech related words in the internet and by using those words, I have assigned NER tag and classified into categories. I have filtered these NER tags and passed those tags to create classes in ontology.

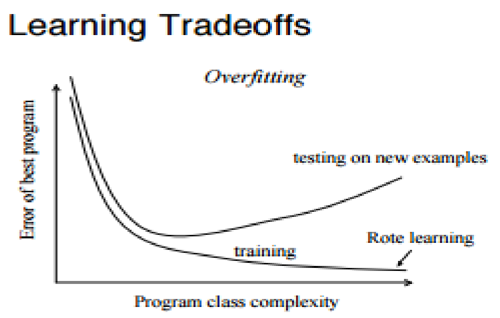
OpenIE:

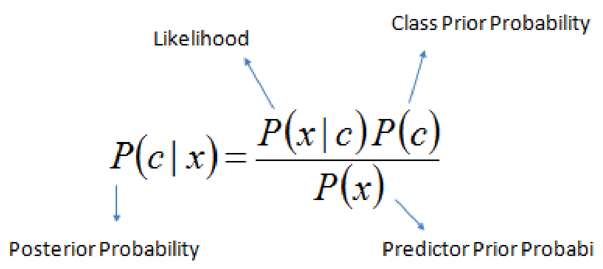
Till now we have dealt with words and sentences. Now we want some other form to extract relations in the sentence, how words in the sentences are related to each other. So we tried to use OpenIE (Information Extraction) which takes input as sentences and gives different possibility triplets which has subject, predicate and object. By using this we can get relation between subject and object which is helpful in building knowledge graph using ontology.


vmod
prep.in
det
amod
nsubj
dobj
det
nn
vmod
dobj
Born in a small town, she took the midnight train going anywhere.
(input)
she took the midnight train going anywhere
Born in a small town, she took the midnight train
Born in a town, she took the midnight train
she took the midnight train
she took midnight train
(she; took; midnight train)
she Born in a small town
(extracted clause)
she Born in small town
she Born in a town
she Born in town
(she; born in; small town)
(she; born in; town)

Machine Learning:

Till now, we have used many NLPs which may not need artificial intelligence or any deep learning algorithms for training our data. Now comes the most important part where actual intelligence resides. The use of machine learning is to extract hidden information from the data and we need to train the data first, so that data will be classified automatically while testing. We can classify learning as unsupervised learning and supervised learning. We used classification algorithms such as naïve Bayes, Random Forest, Decision Tree, PCA, etc., each algorithm has its own importance, for instance naïve bayes has good accuracy for text data than other algorithms. These classification algorithms are not 100% accurate, they have error probability based on the training data.





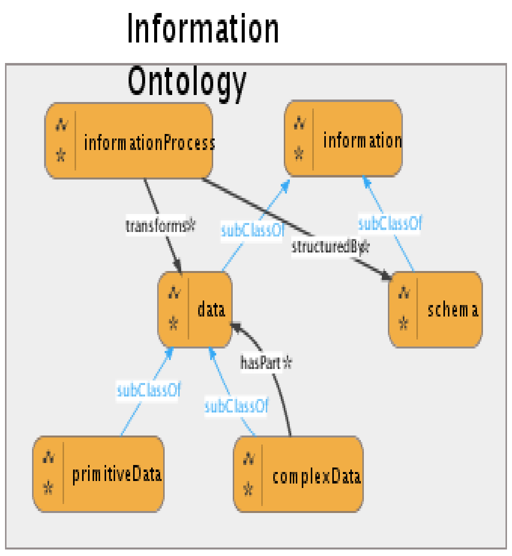
$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P$$

Ontology Learning:

Formal way of representing knowledge is called Ontology. In Ontology, point of view is more important than all the requirements it is supposed to follow. The main purpose of ontology is to integrate data and knowledge. The amount of data on web is growing rapidly, which makes the structured access to data more important. We used ontology to represent the already processed data.

The below is the sample ontology graph.



Apache Fuseki:

Apache Fuseki is a Java framework to develop Semantic web applications. It provides a programmatic environment for OWL and SPARQL [14]. Owl file is used to draw knowledge graph. SPARQL is used to extract required data from the knowledge graph.

Intellexer API:

Intellexer API provides us with the natural language solutions such as Sentiment Analyzer, Name Entity Recognition, Multi Document Summarizer, Comparator, Natural Language Recognizer etc. [15]. We can use Intellexer API with any software component with HTTP requests.

4 Implementation:

News is everywhere now. We have numerous sites to know the news around. But to know exactly what the news is, in short, we don't have any application or a website. We have created an application which will collect news articles from various sources, analyze them, relate them and summarize them. The key parts of application are 1. Collection of data, 2. NLP service,

3. Analyzed data, 4. Knowledge graph, 5. Summarization

Architecture Diagram

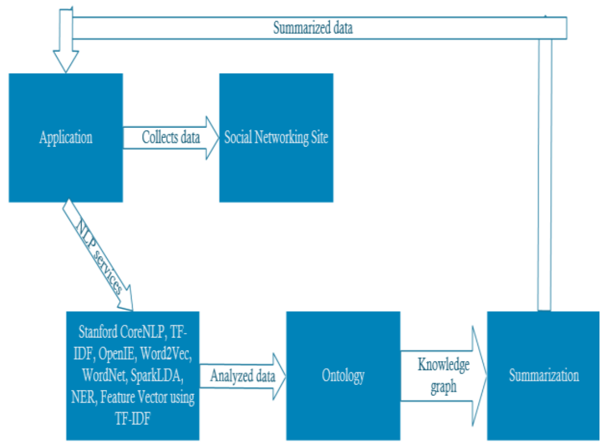


Figure 1: Architecture Diagram

Collection of Data:

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

a) Static Data
b) Real Time Data using APIs

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

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

New York Times:

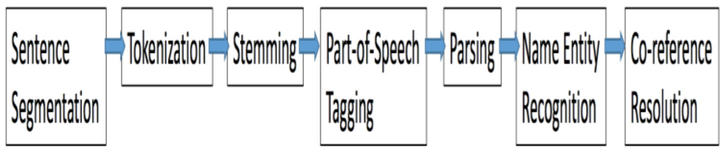
https://api.nytimes.com/svc/search/v2/articlesearch.json?api-key=1069bc25bff24ebf8cf3dbae1133e000&q=tech&sort=newest&fl=lead_paragraph&page =0

For the collection of data, we have used real time data using Google trends and New York times news API's. The data which collected is a real time data (dynamic data), when the application is running it will stream the data from those API's. This streamed data is used further.

NLP Service:

Natural language processing (NLP) is a stream of computer science, which is developed to communicate between computers and the people in a natural language. NLP is to create human-computer interaction.

Workflow



NLP service is the key part of the application, where the text which is collected goes into various stages and form as a structured data.

Sentence segmentation is the stage where the paragraph is divided into numerous sentences based on the punctuations. In this stage, the application will detect the punctuations and split the text into sentences.

Tokenization is a stage in which words, numbers, etc., are identified. The text from social networking sites like Facebook, Twitter as special characters in the text like hashtags, alphanumeric characters were explicit.

Stemming cut down the ending words i.e. greatest hits-> great hit. This process is important for identifying the words for further process.

Part-of-speech tagging (PoS) is process of assigning the part of speech i.e. noun, verb or adjective to each word. This process is important to find the relations between the words. The application will detect PoS based on the word position where it is used instead of assigning the PoS for each word preinstalled.

Parsing gives use the syntactic structure of a sentence which used to relate the subjects and predicate of a sentence.

Named entity recognition is to detect the word entities like persons, locations and times in the document.

Co-reference Resolution used to relate the references such as 'he, she, it, them, etc.,' with the entities to make real sense of the sentence.

Analyzed data:

After the NLP processing the next stage is the analyzing the text further. In the process of analysis, the text goes into various steps:

TF-IDF (Term Frequency-Inverse Document Frequency) is step where we calculate term frequency, document frequency. This step gives the info about which entities are used the most. This step helps the application to find the meaningful information about the document. This helps in analyzing the document like an overview.

Word2vec is a process of building a vector of words and values. This steps bring down the similar words together, thus finding the relations between them. It will simplify the process of NLP modeling. This data is further used to build Feature vector, a crucial role in NLP, which fastens the process.

LDA is used to discover different topics of given corpus. In the application, there will be several categories and sub-categories of a news article. In LDA, by using EM algorithm, it will discover the different topics and grouping them accordingly.

OpenIE (Information extraction) is the main process when comes to summarization. Because the main aspect of summarization is detecting the relationship between the words. OpenIE will extract the relationships of words. The news articles has numerous entities all these are related with OpenIE.

Knowledge graph:

After analyzing the data, it is important to make it should have some précised structure/schemas. So for this OWL (Web ontology Language) is used. It is similar to RDF schemas. It has precisely defined classes, properties and their relations. In short, it is a format of storing the data which is analyzed. Through OWL data, the knowledge graph can be formed.

Summarization:

The final output of analyzed data which is in the format of OWL is a summarized data of combination of different news sources together on a particular topic. The lengthy news article which has redundant data is summarized to short form. All the news articles, which we have streamed is now displayed as single summary on each topic.

Github: https://github.com/murarishetty/KDM-SM2016-TEAM-9/

Video:https://www.youtube.com/watch?v=adQifLxeywA&feature=youtu.be

Conclusion:

The amount of information or the data the world wide web is holding is increasing every second and this rate has called for a need to develop summarization systems. Even though there is a lot of research work going on text, single document and multiple document summarization, there is a long way to go before we really understand the concept of summarization. There are different bottlenecks in developing a summarization system. One of them is the shortage of training data [8], the other one is the shortage of knowledge in worldwide web which is required to perform the topic interpretation and the third one is the lack of resources and the funds for the development.

In this paper we have explained the summarization of News articles on Apache Spark framework. We have used multiple news articles and their corresponding APIs to do the summarization. We have done an integration of all the news articles and collected the top trending news of all the news and summarized the data. We have used various NLPs like TF-IDF, N-gram, Word2Vec, NER, SparkLDA to process the collected data, we have used Naïve Bayes algorithm to classify the words in the documents to different topics and have used Intellexer API to do the final summarization of the news. To draw the knowledge graph, we have used Ontology and to view them we have used Protégé tool. SparQL query language is used to extract the data from the knowledge graph drawn.

Future Work:

We haven't used any of the ranking algorithms to weigh the importance of each triplet in the documents. Sentence ranking algorithm computes the importance of each sentence in the document and we should make sure that the sentences with high importance should be included in the summarized text. In future we are planning to implement sentence ranking algorithm and sentence filter on the data to improve the sense of the summarized text.

Acknowledgements:

We would like to thank Dr. Yugyung Lee for valuable suggestions during the course and we would also like to thank Mayanka Chandra Shekar and Naga Krishna Vadlamudi for all the insightful discussions.

References:

[1] Elena Lloret, Dept. Lenguajes y Sistemas Informaticos Text Summarization: An Overview.

[2] www.statistica.com

[3]http://www.isi.edu/natural-language/projects/SUMMARIST.html

[4]https://en.wikipedia.org/wiki/Artificial_intelligence

[5] Gunes ¸ Erkan, Dragomir R. Radev The University of Michigan at DUC 2004

[6] N K Nagwani Summarizing large text collection using topic modeling and clustering based on MapReduce framework

[7] Lin J, Dyer C (2010) Data-Intensive Text Processing with MapReduce. Morgan & Claypool Publishers 3(1):1–177

[8] https://www.cs.cmu.edu/~afm/Home_files/Das_Martins_survey_summarization.pdf

[9] Wang Y, Bai H, Stanton M, Chen WY, Chang EY (2009) Plda: Parallel latent dirichlet allocation for large-scale applications. 5th International Conference, AAIM (Algorithmic Aspects in Information and Management), San Francisco, CA, USA, pp 309–322.

[10] https://www.quora.com/What-is-a-good-explanation-of-Latent-Dirichlet-Allocation

[11] http://nlp.stanford.edu/software/CRF-NER.shtml

[12] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 363-370.

[13] Michael Collins MIT CSAIL Machine Learning Methods in Natural Language Processing

[14] https://jena.apache.org/documentation/serving_data/

[15] http://www.programmableweb.com/api/intellexer