A MINI PROJECT REPORT

On

KEYWORD EXTRACTION

Submitted in partial fulfillment of the requirement of

University of Mumbai for the Course

Natural Language Processing

In

Computer Engineering (VIII SEM)

Submitted By

Akshaya Ramanathan

Chaitanya Nawathe

Ramita Shinde

Subject Incharge

Prof. Mamta Patil

Department of Computer Engineering

NEW HORIZON INSTITUTE OF

TECHNOLOGY AND MANAGEMENT

THANE

UNIVERSITY OF MUMBAI

Academic Year 2020 – 21

CERTIFICATE

This is to certify that the requirements for the project report entitled '**Keyword Extraction**' have been successfully completed by the following students:

Name	**Roll No.**
Akshaya Ramanathan	45
Chaitanya Nawathe	40
Ramita Shinde	58

in partial fulfillment of the course Natural Language Processing in Computer Engineering (VIII SEM) of Mumbai University in the Department of Computer Engineering, New Horizon Institute of Technology and Management, Thane (W) – 400615 during the Academic Year 2020 – 21.

_____

(Prof. Mamta Patil)

Subject Incharge

PROJECT APPROVAL

This project entitled “Keyword Extraction” by Akshaya Ramanathan, Chaitanya Nawathe, and Ramita Shinde is approved for the course Natural Language Processing in Computer Engineering (VIII sem) of Mumbai University in the Department of Computer Engineering.

Examiners:

1. _____

2. _____

Subject Incharge:

Mamta Patil

Date:

Place: Thane

DECLARATION

We declare that this written submission for Natural Language Processing mini-project entitled “Keyword Extraction” represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas/data/fact/source in our submission. We understand that any violation of the above will cause disciplinary action by the institute and also evoke penal action from the sources which have not been properly cited or from whom prior permission has not been taken when needed.

Project Group Members:

_____

Akshaya Ramanathan

_____

Chaitanya Nawathe

_____

Ramita Shinde

Date:

Place: Thane

Table of Contents

Abstract

Nowadays when we wake up, the first thing that we do in the morning is to check our messages. Your mind has trained to ignore the messages of those people and groups that you don't like. You decide the importance of these message by only checking the keywords of people and group name. Your mind will extract the keywords from WhatsApp group name or contact name and train to like it or ignore it. It also depends on many other factors. The same behavior can be visible while reading articles, watching tv or Netflix series, etc.

Machine learning can mimic the same behavior. It is known as keyword extraction in Natural Language Processing (NLP). The keyword extraction process not only separates the articles but also helps in saving time on social media platforms. Every article, post, comment has its own important word that makes them useful or useless.

The keyword extraction process identifies those words and categorizes the text data. Keyword extraction uses machine learning artificial intelligence (AI) with natural language processing (NLP) to break down human language so that it can be understood and analysed by machines.[1]

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Fundamentals

In this project, we have used python programming language to implement keyword extraction with Machine Learning and Natural Language Processing concepts. We'll be using a stack overflow dataset which is a bit noisy and simulates what you could be dealing with in real life. Fig 1.1 shows an example of keyword extraction.

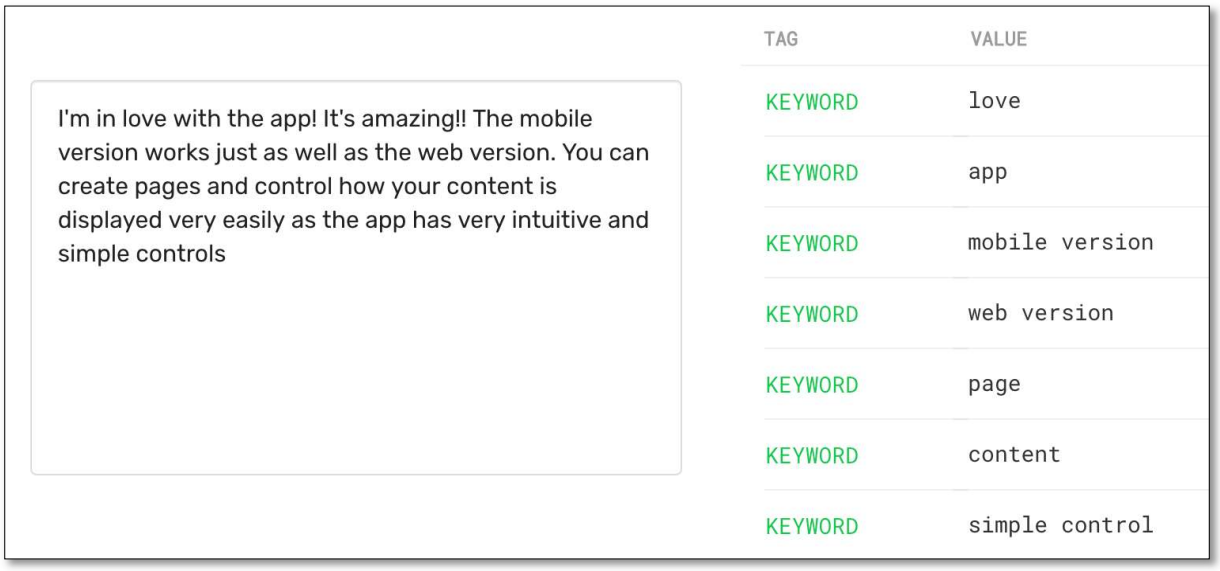

Fig 1.1: Keyword Extraction

1.2 Objectives

With keyword extraction, you can find the most important words and phrases in massive datasets in just seconds. And these words and phrases can provide valuable insights into topics your customers are talking about. In the academic world, keyword extraction may be the key to finding relevant keywords within massive sets of data (like new articles, papers, or journals) without having to read the entire content. Keyword extraction tools are the key to help you automatically index data, summarize a text, or generate tag clouds with the most representative keywords.

1.3 Scope

1. Automated keyword extraction allows you to analyze as much data as you want. Keyword extraction acts based on rules and predefined parameters. You don't have to deal with inconsistencies, which are common in manual text analysis.
2. Automating this task gives you the freedom to concentrate on other parts of your job.
3. Keyword extraction can automate workflows, like tagging incoming survey responses or responding to urgent customer queries, allowing you to save huge amounts of time.
4. It also provides actionable, data-driven insights to help make better business decisions. They are easy to set up and implement. [2]

1.4 Organization of the Report

The report is organized as follows: The introduction is given in Chapter 1. It describes the fundamental terms used in this project. It motivates us to study and understand the different techniques used in this work. This chapter also presents the outline of the objective of the report. Chapter 2 describes the review of the relevant various techniques in the literature systems. It describes the pros and cons of each technique. Chapter 3 presents the Theory and proposed work. It describes the major approaches used in this work. The societal and technical applications are mentioned in Chapter 4. The summary of the report is presented in Chapter 5.

Chapter 2

Literature Survey

2.1 Introduction

This chapter provides an overview of recent research in evaluating keywords. The current literature roughly classifies and shows the types of words that can be predicted to complete the sentences and to increase the speed of the typing.

2.2 Literature Review

In the paper "Automatic Keyphrase Extraction based on NLP and Statistical Methods" by Martin Dostal and Karel Je˘zek, Automatic key phrases are important for automatic tagging and clustering because manually assigned keyphrases are not sufficient in most cases. Keyphrase candidates are extracted in a new way derived from a combination of graph methods (TextRank) and statistical methods (TF*IDF). Keyword candidates are merged with named entities and stop words according to NL POS (Part Of a Speech) patterns. Automatic keyphrases are generated as TF*IDF weighted unigrams. Keyphrases describe the main ideas of documents in a human-readable way. Evaluation of this approach is presented in articles extracted from News websites. Each article contains manually assigned topics/categories which are used for keyword evaluation [3]

Rohith P, Sidharth Sasi Kumar, Anju RC in the paper "Keyword Extraction From Malayalam News Articles Using Conditional Random Fields" Keywords are those words of a document that can describe the meaning of the document precisely and effectively. This paper is putting forward a novel automatic keyword extraction method for Malayalam news articles using a CRF model. An automatic keyword extraction system for Malayalam documents has not been implemented previously, and manually assigning high-quality keywords uses fixed taxonomy, is expensive, and error-prone. The Conditional Random Fields (CRF) model is a highly advanced, probabilistic sequence labeling model, which can effectively use the features of a document for the task of keyword extraction. [4]

In the paper "Simple Unsupervised Keyphrase Extraction using Sentence Embeddings", Kamil Bennani-Smires, Claudiu Musat, Andreaa Hossmann, Michael Baeriswy, Martin Jaggi said that Keyphrase extraction is the task of automatically selecting a small set of phrases that best

describe a given free text document. Supervised keyphrase extraction requires large amounts of labeled training data and generalizes very poorly outside the domain of the training data. At the same time, unsupervised systems have poor accuracy, and often do not generalize well, as they require the input document to belong to a larger corpus also given as input. Addressing these drawbacks, in this paper, we tackle keyphrase extraction from single documents with EmbedRank: a novel unsupervised method, that leverages sentence embeddings. EmbedRank achieves higher F-scores than graph-based state-of-the-art systems on standard datasets and is suitable for real-time processing of large amounts of Web data. [5]

In the paper "A Graph-based Approach of Automatic Keyphrase Extraction" by Yan Yinga, Tan Qingpinga, Xie Qinzhenga, Zeng Pinga, Li Panpana, it is said that existing graph-based ranking techniques for keyphrase extraction only consider the connections between words in a document, ignoring the impact of the sentence. Motivated by the fact that a word must be important if it appears in many important sentences, it proposes to take full advantage of the reinforcement between words and sentences by melting three kinds of relationships between them. Moreover, a document is grouped with many topics. The extracted keyphrases should be synthetic in the sense that they should deal with all the main topics in a document. Inspired by this, they considered the topic model. Experimental results show that this approach performs better than the state-of-the-art keyphrase extraction method on two datasets under three evaluation metrics. [6]

Lu´ıs Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W. Black, Anatole Gershman, David Martins de Matos, Joao P. Neto, and Jaime Carbonell in the paper "Automatic Keyword Extraction on Twitter" said that build a corpus of tweets from Twitter annotated with keywords using crowdsourcing methods and identify key differences between this domain and the work performed on other domains, such as news, which makes existing approaches for automatic keyword extraction not generalize well on Twitter datasets. These datasets include the small amount of content in each tweet, the frequent usage of lexical variants, and the high variance of the cardinality of keywords present in each tweet. They proposed methods for addressing these issues, which leads to solid improvements on this dataset for this task. [7]

2.3 Summary of Literature Survey

SN	Techniques	Author and Year of Publication	Advantages and Disadvantages
1.	Automatic Keyphrase Extraction based on NLP and Statistical Methods	Martin Dostal, Karel Je˘zek 2011	Advantages- This approach is very useful in cases where we don't have manual keywords assigned by the author or where these keywords are not sufficient Disadvantages - These corpuses were not available at the moment the precision was checked so we had to use our data collection for the first evaluation tests. Data training is less.
2.	Keyword Extraction from Malayalam News Articles Using Conditional Random Fields	Rohith P, Sidharth Sasi Kumar, Anju RC 2019	Advantages - given satisfactory performance with scope for improvement in the future. The experimental results show that a CRF model can perform considerably well in the task of sequence labeling compared to other models. Disadvantages - he extracted keywords can effectively convey a short description of the content of the article, however, with the help of better lemmatizers and parsers for Malayalam, the performance of the model can be improved considerably
3.	Simple Unsupervised Keyphrase Extraction using Sentence Embeddings	Kamil Bennani-Smires, Claudiu Musat, Andreaa Hossmann, Michael Baeriswy, Martin Jaggi 2018	Advantages – This method is entirely unsupervised, corpus-independent, and they only require the current document itself, rather than the entire corpus to which it belongs (that might not exist at all). Disadvantages - Unsupervised systems have poor accuracy, and often do not generalize well, as they require the input document to belong to a larger corpus also given as input.
4.	A Graph-based Approach of Automatic Keyphrase Extraction	Yan Yinga, Tan Qingpinga , Xie Qinzhenga, Zeng Pinga ,Li Panpana	Advantages- Experiments show that this method outperforms other baseline methods on two datasets Disadvantages - The method only takes consideration of a single document next it can make full use of corpus which has a bunch of documents similar to the specific document.

5.	Automatic Keyword Extraction on Twitter	Lu´ıs Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W. Black, Anatole Gershman, David Martins de Matos, Joao P. Neto, and Jaime Carbonell 2015	Advantages- A baseline system is defined using existing methods applied to our dataset and improvement significantly using unsupervised feature extraction methods. Furthermore, an additional component to predict the number of keywords in a tweet is also built. Disadvantages - The first problem is the existence of many lexical variants in Twitter. The proposed model does not have this information and will not be able to generalize properly

Table 2.1: Literature Summary

Chapter 3

Implementation Details

3.1 Overview

Keyword Extraction is a simple project which is implemented using python programming language and it uses some Machine Learning (ML) as well as Natural Language Processing (NLP) concepts to break down human language so that it can be understood and analyzed by machines. It's used to find keywords from all manner of text.

3.1.1 Existing Methodology and Systems

The existing method involves just reading through a whole chunk of text. This method is very exhausting and time taking. Considering that more than 80% of the data we generate every day is unstructured; i.e., it's not organized in a predefined way, making it extremely difficult to analyze and process – businesses need automated keyword extraction to help them process and analyze customer data more efficiently.

Existing methods for automatic keyword extraction can be divided into four approaches namely, statistics, linguistics, machine learning, and hybrid approaches. We will use the machine learning approach for the implementation.

3.1.2 Proposed Methodology and System

Keyword extraction simplifies the task of finding relevant words and phrases within the unstructured text. This includes emails, social media posts, chat conversations and any other types of data that are not organized in any predefined way.

Keyword extraction can automate workflows, like tagging incoming survey responses or responding to urgent customer queries, allowing you to save huge amounts of time. It also provides actionable, data-driven insights to help make better business decisions. But the best thing about keyword extraction models is that they are easy to set up and implement.

There are different techniques you can use for automated keyword extraction. From simple statistical approaches that detect keywords by counting word frequency, to more advanced machine learning approaches that create even more complex models.

3.2 Implementation Details

1. The dataset is first pre-processed to remove tags, special characters, and digits. All the stopwords are eliminated.
2. Using CountVectorizer and TfidfTransformer, for a given document TF_IDF is calculated.
3. Next, we sort the words in the vector in descending order of tf-idf values and then iterate over to extract the top-n items with the corresponding feature names.

3.2.1 Methodology

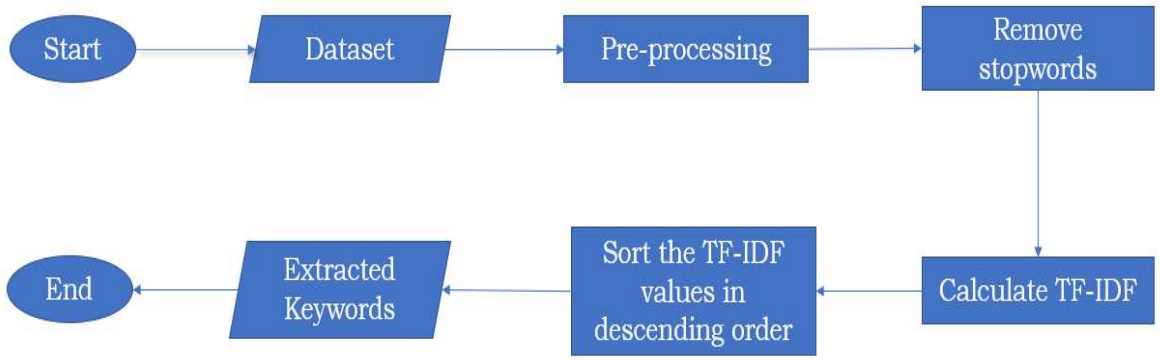

Fig 3.1: Workflow

Fig 3.1 shows the methodology of our project.

3.2.2 Details of packages, data set

The packages that are used for the implementation of Keyword Extraction are ‘pandas’, ‘re’, and ‘sklearn’. Pandas is a Python library. Pandas is used to analyze data. We use the pandas library to load the dataset into the data frame.

A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern. RegEx can be used to check if a string contains the specified search pattern. Python has a built-in package called re, which can be used to work with Regular Expressions.

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction.

If you need the term frequency (term count) vectors for different tasks, use Tfidftransformer. With Tfidftransformer you will systematically compute word counts using CountVectorizer and then compute the Inverse Document Frequency (IDF) values and only then compute the Tf-idf scores.

For this implementation, we'll be using a stack overflow dataset which is a bit noisy and simulates what you could be dealing with in real life.

We will be using two files, one file, stackoverflow-data-idf.json has 20,000 posts and is used to compute the Inverse Document Frequency (IDF), and another file, stackoverflow-test.json has 500 posts and we would use that as a test set for us to extract keywords from. This dataset is based on the publicly available stack overflow dump from Google's Big Query.

Chapter 4

Project Inputs and Outputs

4.1 Input Details

We'll be using a stack overflow dataset which is a bit noisy and simulates what you could be dealing with in real life. We will be using two files, one file, stackoverflow-data-idf.json has 20,000 posts and is used to compute the Inverse Document Frequency (IDF), and another file, stackoverflow-test.json has 500 posts and we would use that as a test set for us to extract keywords from.

```
In [2]: import pandas as pd
        # read json into a dataframe
        df_idf=pd.read_json("stackoverflow-data-idf.json",lines=True)

        # print schema
        print("Schema:\n\n",df_idf.dtypes)
        print("Number of questions,columns=",df_idf.shape)

        Schema:

         id                        int64
        title                     object
        body                      object
        answer_count               int64
        comment_count              int64
        creation_date             object
        last_activity_date        object
        last_editor_display_name  object
        owner_display_name        object
        owner_user_id            float64
        post_type_id               int64
        score                      int64
        tags                      object
        view_count                 int64
        accepted_answer_id       float64
        favorite_count           float64
        last_edit_date            object
        last_editor_user_id      float64
        community_owned_date      object
        dtype: object
        Number of questions,columns= (20000, 19)
```

Fig 4.1: Fields in the dataset

This stack overflow dataset contains 19 fields including post title, body, tags, dates, and other metadata which we don't quite need for this project. What we are mostly interested in is the body and title which will become our source of text for keyword extraction.

4.2 Evaluation Parameter Details

We will try to extract keywords from a given paragraph and the dataset is split into two parts– training and testing. Evaluation can be done by the accuracy of the model to check if the keywords extracted are correct and as needed by the user.

4.3 Output Details and Screenshots

This model will extract the keywords that have a lot of importance in the dataset. The importance of a specific word in a dataset is identified using TF-IDF values which are calculated. The more the TF-IDF value of the word, the more is the probability of that word appearing in the extracted keyword list. We compute the tf-idf value for a given document in our test set by invoking tfidf_transformer.transform(...). This generates a vector of tf-idf scores. Next, we sort the words in the vector in descending order of tf-idf values and then iterate over to extract the top-n keywords. The sort_coo(...) method essentially sorts the values in the vector while preserving the column index.

```
In [2]: import re
        def pre_process(text):

            # lowercase
            text=text.lower()

            #remove tags
            text=re.sub("</?.*?>"," <> ",text)

            # remove special characters and digits
            text=re.sub("(\\d|\\W)+"," ",text)

            return text

        df_idf['text'] = df_idf['title'] + df_idf['body']
        df_idf['text'] = df_idf['text'].apply(lambda x:pre_process(x))

        #show the first 'text'
        df_idf['text'][2]

Out[2]: 'gradle command line i m trying to run a shell script with gradle i currently have something like this def test project tasks create test
        exec commandline bash c bash c my file dir script sh the problem is that i cannot run this script because i have spaces in my dir name i
        have tried everything e g commandline bash c bash c my file dir script sh tokenize commandline bash c bash c my file dir script sh comman
        dline bash c new stringbuilder append bash append c my file dir script sh commandline bash c bash c my file dir script sh file dir file c
        my file dir script sh commandline bash c bash dir getabsolutepath im using windows bit and if i use a path without spaces the script runs
        perfectly therefore the only issue as i can see is how gradle handles spaces '
```

Fig 4.2: Text after preprocessing

```
=====Title=====
Integrate War-Plugin for m2eclipse into Eclipse Project

===Keywords===
eclipse 0.599
war 0.32
integrate 0.284
maven 0.276
tomcat 0.273
project 0.241
plugin 0.217
automate 0.159
jsf 0.153
deploy 0.132
```

Fig 4.3: Top 10 extracted keywords

We have extracted 10 important keywords from the extract here. Similarly, we can also extract 5 or even 20 top keywords from the extract.

```
keywords=extract_topn_from_vector(feature_names,sorted_items,20)

# now print the results
print("\n=====Title=====")
print(docs_title[0])
print("\n===Keywords===")
for k in keywords:
    print(k,keywords[k])
```

```
=====Title=====
Integrate War-Plugin for m2eclipse into Eclipse Project

===Keywords===
eclipse 0.599
war 0.32
integrate 0.284
maven 0.276
tomcat 0.273
project 0.241
plugin 0.217
automate 0.159
jsf 0.153
deploy 0.132
tutorial 0.12
couldn 0.116
explain 0.111
automatically 0.111
small 0.11
process 0.093
link 0.088
web 0.084
read 0.082
button 0.081
```

Fig 4.4: Top 20 extracted keywords

From the keywords above, the top keywords make sense, it talks about “eclipse”, “maven”, “integrate”, “war” and “tomcat” which are all unique to this specific dataset. There are a couple of keywords that could have been eliminated such as “automatically” and perhaps even “project” and we can do this by adding more common words to your stopwords list and we can even create your own set of stopwords list, very specific to our domain.

Chapter 5

Summary and Future Scope

5.1 Summary

Keyword extraction is an excellent way to find what's relevant in large sets of data. This allows businesses in any field to automate complex processes that would otherwise be extremely time-consuming and much less effective (and, in some cases, completely impossible to accomplish manually). By keyword extraction, we could easily find the words that are highly signified and of great importance. You can get valuable insights to make better business decisions. Keyword extraction can be made use of in summarization and organization, database indexing, and search engine optimization (SEO).

5.2 Future Scope

1. In the future other parameters such as purity, entropy could be considered of great importance and sentiments can be used to extract the keywords from the dataset.

2. The sentiments and emotions of the text can also be considered as well as some heuristic approach can be used to eliminate the noisy data which could further reduce the computational time of the algorithm.

3. This system can also be integrated with text mining, sentiment analysis, chatbots, etc, to improve their performance.

References

[1] https://towardsdatascience.com/keyword-extraction-process-in-python-with-natural-language-processing-nlp-d769a9069d5c

[2] https://monkeylearn.com/keyword-extraction/

[3] Martin Dostal and Karel Je˘zek (2010, May). Automatic Keyphrase Extraction based on NLP and Statistical Methods. Semantics Scholar, 6834431.

[4] Rohith P, Sidharth Sasi Kumar, Anju RC (2019, Feb). Keyword Extraction from Malayalam News Articles Using Conditional Random Fields. 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP). ISBN: 978-1-5386-7990-6

[5] Kamil Bennani-Smires, Claudiu Musat, Andreaa Hossmann, Michael Baeriswy, Martin Jaggi (2018, September). Simple Unsupervised Keyphrase Extraction using Sentence Embeddings.

[6] Yan Yinga, Tan Qingpinga , Xie Qinzhenga, Zeng Pinga ,Li Panpana (2017, May). A Graph-based Approach of Automatic Keyphrase Extraction. Science Direct.

[7] Lu´ıs Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W. Black, Anatole Gershman, David Martins de Matos, Joao P. Neto, and Jaime Carbonell (2015, July). Automatic Keyword Extraction on Twitter. International Joint Conference on Natural Language Processing (Short Papers).

Acknowledgment

We would like to especially thank our principal, Dr. Prashant D. Deshmukh, at New Horizon Institute of Technology and Management for his support and guidance. Making this project was possible because of your modified ways of making education interesting. You have made our college a better place.

I am highly indebted to our HOD, Dr. Sanjay Sharma for his kind cooperation and encouragement which helped us in the completion of this project, his guidance and constant supervision helped us in learning new things.

We would like to express our special thanks of gratitude to our subject teacher and our guide Prof. Mamta Patil, who gave us this golden opportunity to do this wonderful project on the topic of “Keyword Extraction”, which also helped us in doing a lot of research and learning new things. Thank you for giving us such attention, encouragement and inputs from time to time.

Akshaya Ramanathan

Chaitanya Nawathe

Ramita Shinde