Abstraction Based Text Summarization using NLTK

Mr. Anand Tilagul1, Santhosh T M2, Prajwal M B3, Nikhil R4
Assistant Professor, Department of Information Science and Engineering1
Student, Department of Information Science and Engineering2,3,4
S J C Institute of Technology, Chikkaballapur, Karnataka, India

Abstract: As there is huge amount of content is produced in our day-to-day life using electronic devices in various form fields. The main problem arises from here huge form information once to analyzing and understanding the meaning of text become difficult and time taking, so the Text summarization is introduced. There are two types of text summarization types one is Extraction based text summarization and Abstraction based text summarization.

Keywords: NLTK

I. INTRODUCTION
An abstraction-based text summarization is a method of generating summary for the huge input text given by the user. Basically, in this project user can give desired length of text to summarizer so the machine is trained such a way that it will produce summary for input text which will in reduced in length, reduced in words and meaningful sentences. By this summary it will make user to read the content of the huge text easily and also the main motive of the summarizer is to reduce the time of user without stuck in understanding inner meaning of the sentences. And summary should contain in the form human readable format with the most needed words which help to impact the meaning of the summary. Already many models are built to summarize the text using the neural form networks and using machine learning to get the meaning summary. Natural Language Toolkit (NLTK) which is toolkit build for purpose working with the Natural Language Processing with Python Programming language.

II. PROBLEM IDENTIFICATION
An Investigating Officer may sometimes be required to refer to online news articles to obtain further information about a case beyond what is already known through on-ground sources. Due to the proliferation of news websites on the internet, it is not uncommon for a simple search on a topic or suspect of interest to return thousands, and even lakhs, of relevant news articles. It would take an Investigating Officer hours and hours of manual effort to go through these news articles, understand them and assimilate key findings. Often information would be spread out and not available in a single article.

III. METHODOLOGY
STEPS INVOLVED TO GENERATE THE ABSTRACTATION BASED SUMMARY

- **STEP 1**: Here the text is used collected from the user as input for summarizer.
- **STEP 2**: In this step collected text is cleaned, means deleting the stop words, special characters, numbers which is irrelevant to text and punctuations
- **STEP 3**: In this step word token and sentences token are created this process is called Tokenization
- **STEP 4**: In this step by those tokens created in pervious step, frequency is found for every word in the users input text.
- **STEP 5**: Here in this step weights are assigned to words.
- **STEP 6**: Based on the weights, most top rated 20% weighted sentences are called final summary.
IV. IMPLEMENTATION

The pseudocode of a NLTK TEXT- SUMMARIZER mechanism

Start

INPUT: Un-summarized Text

Output: Summary For given input Text

1. Import nltk , (import by typing “ PIP INSTALL NLTK” command)
2. import Stopwords
3. Def nltk_summarizer(TEXT)
4. SW = set(stopword.word(“English”))
5. words = word_tokenize(TEXT)
6. freqTable = dict()
7. // Removing Stop Words
    for word in words
    word = word.lower()
    if word not in stopWords
    if word in freqTable
        freqTable[word] += 1
    else
        freqTable[word] = 1
    end for
8. sentence_list = sent_tokenize(docx)
9. max_freq = max(freqTable.values())
10. for word in freqTable.keys()
11. freqTable[word] = freqTable[word]/max_freq
12. sentence_scores = {}  
13. for sent in sentence_list
14. for word in nltk.word_tokenize(sent.lower())
15. if word in freqTable.keys()
16. freqTable[word] = freqTable[word]/max_freq
17. sentence_scores = { }
18. for sent in sentence_list
19. for word in nltk.word_tokenize(sent.lower())
20. freqTable.keys()

Figure 1: Summary Generation Process
V. TESTING.

<table>
<thead>
<tr>
<th>C</th>
<th>TEST CASE</th>
<th>EXPECTED OUTPUT</th>
<th>OBTAINED OUTPUT</th>
<th>RESULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Passing the text input</td>
<td>Text should be read and displayed in input area</td>
<td>Text should be read and displayed in input area</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Passing the text input as null</td>
<td>Show alert messages “Enter the text”</td>
<td>Show alert messages “Enter the text”</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Selecting the NLTK method</td>
<td>Ready to create summary</td>
<td>Ready to create summary</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>Without selecting the NLTK method</td>
<td>An error should be thrown specifying &quot;importing NLTK &quot;</td>
<td>An error is thrown</td>
<td>Pass</td>
</tr>
<tr>
<td>5</td>
<td>The model should return reduced and meaningful summary</td>
<td>Abstractive summary will be displayed</td>
<td>Abstractive summary will be displayed</td>
<td>Pass</td>
</tr>
</tbody>
</table>

VI. RESULTS

1. The solution should take the desired length of summary from the user as an input should return summarized output.
2. The most important output of these Abstraction based text summarizer is to reduce the reading time.
3. Abstraction based text summarization produces meaningful sentences.
4. It makes the user to read the summarized output easily.
5. It gives short, exact and more content full summary without repetitive summary.
REFERENCES


[4]. 978-1-7281-9744-9/20/$31.00 ©2020 IEEE


