Saaraansh: Gujarati Text Summarization System

Jikitsha Sheth
Asst. Professor,
Shrimad Rajchandra Institute of Management and Computer Application, Uka Tarsadia University, Dist. Surat, Gujarat, India.

Bankim Patel
Director,
Shrimad Rajchandra Institute of Management and Computer Application, Uka Tarsadia University, Dist. Surat, Gujarat, India.

***Abstract*-The domain of natural language processing has moved from international language processing like English to national language processing like Hindi. In the world of information overload, end users are benefited with summaries of given text document. Summarization helps end users to get the core theme of given text. Relevant and wide variety of text summarization work is found for English language. Still for Indian languages, text summarization processing needs more efforts to achieve high recall. The proposed summarizer identifies relevant sentences from the given Gujarati text which can be chosen to create a summary. The recall of proposed work is 0.69.**

Keywords-Gujarati language, Text Summarization System, Natural Language Processing, Unsupervised learning

I. INTRODUCTION

A summary is defined as a text that is generated from one or more text passages, which contains a significant portion of the information in the original text(s), and is not longer than half of the original text(s)[1]. A computerized system that generates a summary from a given passage is called Automatic Text Summarization (ATS) system. Text summarization system which was one of the major application of Information Retrieval (IR) now also belongs to NLP domain.

There are two types of summaries: extractive and abstractive. The extractive summaries are developed by extracting the relevant lines of given passage based on word and sentence features. To develop abstractive summaries, the theme of the passage is rephrased and concised. This paper aims to develop extractive summary for the given passage.

For Hindi extractive summaries are generated using feature vector. As a part of post-processing, low coherence and redundancy is handled using a similarity measure[2]. For Tamil language, graph based ranking algorithm for text ranking is done, where for each term, its root is found and using weight of root, the graph is created. The graph is always a mesh topology, which makes the approach time and space consuming[11]. Using surface score, content score and PageRank, extractive text summarization is done for Englishand Tamil. It claims to perform better than those where synonyms and other complex similarity measures are used. A text summarization system is proposed for Punjabi language. It focuses on features namely Punjabi keywords identification, relative sentence length feature and numbered data feature. The weights of features are determined using regression. These weights further help to determine the score of the sentence so as to incorporate the sentence in automatic summary[3][4]. For Gujarati language, no full-fledged text summarizer exists yet. Also due to resource constraint and its characteristics, existing summarizers cannot be adopted for Gujarati language. Few of such points are discussed in Section 2. Further, section 3 describes the proposed model of text summarization for Gujarati language and Section 4 discusses the results obtained after experimenting the same. This paper aims to propose a text summarization system for Gujarati language with better recall.

II. CHALLENGES IN TEXT SUMMARIZATION FOR GUJARATI LANGUAGE

Nearly 50 million people of western part of India use Gujarati language[5]. About 65.5 million speakers of Gujarati exists worldwide, making it the 26th most spoken native language in the world.

Before the model for text summarization of Gujarati language was developed, a strong study related to why English text summarization system cannot be used for Gujarati language was done.

Some of the distinctions, with respect to computational linguistics, between Gujarati and English language were found which are specified below:

- Gujarati follows SOV as its default sentence structure. It has free word order i.e. words can move freely within a sentence without changing its meaning. For example, for an event described as ‘Ram gave a book to Shyam’, can be written in more than one ways in Gujarati as follows:

  રામેશ્યામનેપુસ્તકઆપી. (Ram gave a book to Shyam.)

  શ્યામનેરામેપુસ્તકઆપી.(Shyam was given a book by Ram.)

- It has relatively rich set of morphological variants. A word may appear with a number of inflections and each inflection can appear with several words. For example, ક૨(/kɾ/)+વું, ક૨(/kɾ/)+તું,ક૨(/kɾ/)+ેand ફ૨(/pʰɾ/)+વું,ફ૨/pʰɾ/)+તું, ફ૨/pʰɾ/)+ે

- Verbs undergo morphological changes depending upon the number and gender.

  રામ૨મતોહતો.(Ram is playing.)

  સીતા૨મતીહતી. (Sita is playing.)

  બાળકો૨મતાહતા.(The kids are playing.)

Here forms of રમ(/ɾm/-to play) keeps on changing based on the gender or number of the subject.

- It has complex predicates (CPs). The complex predicate combines a light verb with a verb, noun, or adjective, to produce new verb. For example

  રાધાઆવી.(Radha came.)

  રાધાઆવીગઈ.(Radha arrived.)

  રાધાઆવીપહોચી.(Radha came (suddenly).)

  Here, આવીગઈand આવીપહોચીare the complex predicates.

- The complex predicates change the functional structure of the sentence.
- Instead of prepositions, post-position case markers (Karaks) are used. 'To market' is represented as બજારે (/bəd͡ʒaɾe/), where ે (/e/) is part of the word બજાર (/bəd͡ʒaɾ/).
- There exists a sequence of verbs. For example, ખાતીરહેછે (keeps eating), ફરતોરહેછે (keeps roaming). The gender information is contained in the verb group.
- Adjectives may appear with variations to agree with gender. For example, સારોછોકરો (good boy) and સારીછોકરી (good girl).
- In English language, the pronouns reflect gender information like He, She and It represents a singular masculine, feminine and neuter respectively. He, She and It are mapped with તે (/t□e ː /) in Gujarati language, which is as such associated with all three genders.

Thus morphology of the language plays vital role in overall processing of the language and due to the differences discussed above, an NLP system that works outstanding for language processing in English language, might not perform even average when applied for Gujarati language. Gujarati is one of the Indo-Aryan languages. Each language that belong to Indo-Aryan group share some common features. Even then, each language is unique in itself. The uniqueness is not just with respect to script of the language but also due to other grammatical and orthographical characteristics. Unlike its sister-language Hindi, Gujarati doesn't uses the explicit Karaks. For example in Hindi sentence,

राम ने सीता को पुष्पदीया ।(Ram gave flower to Sita.)

ने and को are vibhakti that appear as explicit post-positions.

Its translated sentence in Gujarati shall be

રામેસીતાનેપુષ્પઆપ્યું.(Ram gave flower to Sita.)

Here, ે and ને are part of the words રામે/rame/ and સીતાને/sitane/ respectively as the endings.

Hence, the text summarizers of language like Hindi, Marathi, etc. cannot be used for Gujarati text summarization.

To imbibe the morphological level processing in the proposed text summarization, the need of the stemmer and string similarity was found. For this, DHIYA stemmer[6] and GUJStringSimilarity[7] are used at morphological level. These are the core components modelled on the base of Gujarati grammar.

III. PROPOSED MODEL

The development of text summarization system is done with morphological and semantic level analysis. Its classification model is statistical, while the core components are still rule-based in nature. Thus, a hybrid approach is used with the purpose of scalability and robustness. A language, whether Indian or non-Indian can reproduce this text summarization system by replacing the core-components of systemin respective language.

For a given text, the concepts represented in it are described by words. But due to morphologically rich and karak being part of the word, in Gujarati, multiple words may represent same concept. For example, a text on Mahatma Gandhi may have words like ગાંધીજી (/gʱan□d□ʱi:d͡ʒi:/-Gandhiji),ગાંધીજીએ (/gʱan□d□ʱi:d͡ʒi:e ː /-by Gandhiji) and ગાંધીજીનાં (/gʱan□d□ʱi:d͡ʒi:n□ɑ/-of Gandhiji), all of them are formed by different karaks(here, એ andનાં) combined with word Gandhiji(/gʱan□d□ʱi:d͡ʒi:/-ગાંધીજી). Instead of considering them as three unique words, they should be treated as one word with three occurrences. Then only it can be identified by a machine that the text is about Gandhiji. So stems and their weightages are core for identification of concepts.

Once the concepts are identified, based on their presence in sentence one can find the sentences that are related to each other. Also it can be known that which sentences are related to the theme of the text. This helps to also rank the sentences so as to classify whether a given sentence should be added to summary text or not. Hence, the present text summarization has focused on two basic features:

i) Stem weights: It represents the importance of the concept described in the given text. Other text summarization work focuses on term-vector while the proposed work emphasizes on stem-vector because number of morphological variants exists for a given term in Gujarati language.

ii) Sentence centrality score: It represents the score that determines the relationship of sentence with the theme of the text given. This is derived using LexRank algorithm.

The proposed work is designed by considering text summarization as a classification problem. The model classifies whether the given sentence should be chosen for summary text or not. The model proposed for extractive text summarization is as Figure 1.

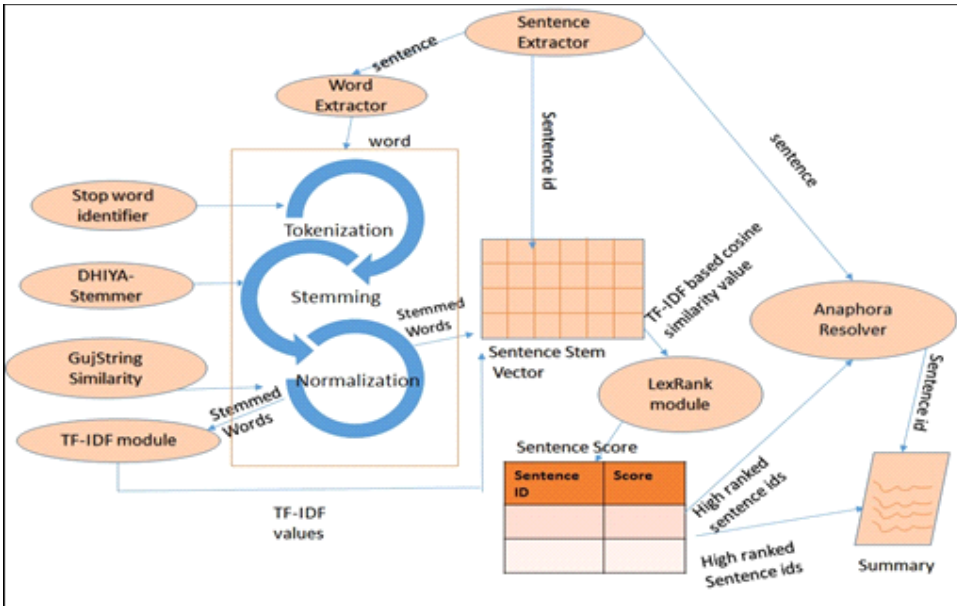

Figure 1. Text summarization model for Gujarati language

Following are the components of text summarizer for Gujarati language:

- Sentence Extractor
- Word Extractor
- Stopword identifier
- DHIYA-stemmer
- GujStringSimilarity module
- Stem weightage module
- LexRank module
- Anaphora resolver

A. *Sentence Extractor*

The module extracts each sentences from text and assigns a sentence id to the same. The extractor is made sensitive toward end markings like fullstop(.), comma(,), question mark(?) and other punctuations. Also, abbreviations are identified, so that incorrect sentence generation can be prevented.

B. *Word Extractor*

The module extracts words from given sentence which are smallest unit of concept representation. These words are further used by similarity measuring module and stemmer.

C. *Stop word identifier*

As our point of interest is keywords, the stop words are removed using stop-word list. A list of 58 stop-words was created for the same. Some of the stop words are છે(/tʃʰe/-is), માત્ર(/mātɾə/- only), વગેરે(/ʋəgere/-et cetera), હતા(/ɦətāṁ/-was), etc.

D. *GSoundex and GujStringSimilarity*

In Gujarati, the vowels can be dependent (e.g.ા,િ,ી,ો, etc.) in case to give a sound to the consonant or it can be independent (e.g. આ, ઇ, ઈ, ઓ, etc.). Also, the sound of matraિ/ɪ/ andી/i:/ is same, only degree of elongation varies. Similar is the case forુ/ʊ/ andૂ/u:/. So words that have these characters are often misspelled. For example ભૂલand ભુલ, both the strings sound as /bʰʊlə/ and mean 'error'. Further, similar sounding characters form words that may have different meaning. For example દિન (day) and દીન (poor); both sound nearly as /dɪnə/. To handle such cases in effective manner while processing, GSoundex and GujStringSimilarity as proposed in [7]are used. GujStringSimilarityis used to find lexical similarity between given two Gujarati words. This module normalizes the given word to a common form that can be further processed by the stemmer.

E. *Dhiya-stemmer*

As several inflections exist with noun, verb and adjectives, these inflections are to be stripped off. To carry out this task, a Gujarati stemmer *Dhiya* is used[6]. Thus words like કમિટીનાં(/kmɪʈi:na:/-of committee) and કમિટીમાં(/kmɪʈi:ma:/-in committee) would result to કમિટી(/kmɪʈi:/-committee). The stemmer identifies the stem of given Gujarati word and adds it to the sentence-word vector. The calculation of stemweightage is discussed further.

F. *Stem weighting module*

In text summarization systems, the importance of key terms for the given document is identified by TF-IDF(Term Frequency-Inverse Document Frequency). Here TF is the raw frequency of term t in the given sentence s, while IDF is the inverse document frequency which denotes how common the term is across all sentences.

Given a document collection D, a word w, and an individual document d ϵ D, weightage of word w is calculated as follows:

$$w_d = f_{w,d} * \log (|D| / fw,D) \quad (1)$$

where $f_{w,d}$ equals the number of times w appear in d, |D| is the size of the corpus, and $f_{w,D}$ equals the number of documents in which w appears in D.

TF-IDF is not able to define the relationship between words that are variant to each other. TF-IDF could not equate

the word 'drug' with its plural 'drugs', categorizing each instead as separate words and slightly decreases the word's w_d value.

So if we are able to identify the terms that represent the concept, we are able to derive the important sentences of a given text and thus can generate a summary. But as Gujarati language has several morphological variants to represent the same concept, in the present work, we have identified stems of these terms. For this, a word has to be first stemmed to its base form. Here we have established a relationship between word and its respective stem termed as *WordStem* relationship denoted by z. This has been formulated as follows:

$$z(w_i,ST)=\begin{cases}1, \text{ if } w_i \text{ is a conflated term of stem ST} \\ 0, \text{otherwise}\end{cases} \quad (2)$$

To find the weight of the stem, a relationship between stem and sentence is established based on the occurrence of the morphological variants in the sentence for a given stem. Assume that a sentence s has set of words W= $\{w_1, w_2,\ldots w_n\}$. *Stem frequency* is formulated as

$$sf(ST, s)=\{|w_i||z(w_i,ST)=1\} \quad (3)$$

This indicates number of times a word is found in sentence s whose stem is ST. As we are interested in only those words of sentences that conflate to the given stem, the criteria of $z(w_i,ST)=1$ becomes mandatory.

Other critical information to derive is whether a sentence has the given stem or not. It helps to identify the existence of a stem in given text with number of sentences. It has been coined as *StemSentence* relationship and denoted by r. This relationship has been formulated as follows:

$$r(ST,s)=\begin{cases}1, \text{ if } sf(ST, s) >0 \\ 0, \text{otherwise}\end{cases} \quad (4)$$

As Gujarati language has words with large number of inflections, a word and its variants may appear several times. So a stem gets matched several times in a given text and stem frequency raises. Apart of the description power, a stem also possesses power of discrimination. The discriminative power of a stem is to identify which sentence is more relevant in the given text. Thus, to normalize the relevance of a stem ST in a given text t with sentences s, *inverse text frequency of the stem* ST is formulated as follows:

$$itf(ST,t)=\{\log(|t|/|s_t|)|r(ST,s_t)=1\} \quad (5)$$

where |t| denotes size of text t in terms of total number of sentences and $|s_t|$ denotes the number of sentences for which stem ST is one of the stems.

Hence the weightage of a stem ST in text t with the set of sentences S = $\{s_1, s_2, s_3, \ldots, s_n\}$ is found using equation 3and 5 as follows:

$$sw(ST, s_i, t) = sf(ST, s_i) \text{ X } itf(ST, t) \quad (6)$$

where s_i represents the sentences of text t.

Hence, the weightage of each stem for each sentence across the given text was identified. This had resulted to creation of stem-sentence matrix named d (ST_i, s_i) that is similar to a term-sentence vector used in automatic text summarization system. This matrix is representation of a sentence in an N-dimensional vector.

G. LexRank module

The vectors created by the stem weightage module, are the core to identify similarity between two sentences. The sentences that are similar to many of the other sentences in a cluster are more central (or salient) to the topic[8].

So, at first instance, there is a need to establish similarity between two sentences. As we had represented the sentences as a vector, cosine-similarity was used to find similarity between given two sentences. The cosine-similarity [9] is defined as

$$\text{Cosine\_similarity}(v_1,v_2)=\frac{v1.v2}{|v1||v2|}$$

where, v_1 and v_2 are vectors. As the vectors created by us are based on stem weightage, we formulated the cosine similarity as follows:

$$\text{Stembased}_{cosines(x,y)}=\frac{\sum_{w\in x,y} sf_{w,x}sf_{w,y}[(itf)_w]^2}{\sqrt{\sum_{x_i\in x}\left(sf_{x_i,x}itf_{x_i}\right)^2} X \sqrt{\sum_{y_i\in y}\left(sf_{y_i,y}itf_{y_i}\right)^2}}$$

where x and y represents the sentences. $sf_{w,x}$ represents the stem frequency of word w in sentence x, while itf_w represents the itf of stem w in the document containing x and y sentences. The numerator is meant to calculate the numerical overlap between the vectors representing sentence x and sentence y. Few words and their morphological variants occur invariably. This result to high stem occurrence. So those sentences which have such stems shall be defined more similar. To overcome this phenomenon, length normalisation is done. The numerator is divided by the respective vector lengths.

For the given sample text, stem-based cosines are calculated.

TABLE 1. SAMPLE NEWS ARTICLE

S1	નવી દિલ્હી, તા. ૩૦
S2	અગાઉના ૫.૯ ટકાના અંદાજીત કુલ રાષ્ટ્રીય ઉત્પાદનનાવૃદ્ધિ દરમાં પ્રગતિકારક સુધારો કરવા છતાં ૧૯૯૯-૨૦૦૦ના વર્ષ દરમ્યાન ભારતીયઅર્થતંત્રનો વિકાસ ગત વર્ષની તુલનામાં નજીવો ઘટીને ૬.૪ ટકા નોંધાયો હતો.
S3	૧૯૯૮-૯૯ના વર્ષમાં દેશના કુલ રાષ્ટ્રીય ઉત્પાદનોનો વૃદ્ધિદર ૬.૮ ટકા નોંધાયોહતો.

S4	સેન્ટ્રલ સ્ટેટીસ્ટિકલ ઓર્ગેનાઈઝેશન (સીએસઓ) તરફથી પ્રસિદ્ધ કરાયેલી સુધારેલી અંદાજીત વાર્ષિક રાષ્ટ્રીય આવક મુજબ ૧૯૯૯-૨૦૦૦નાવર્ષમાં મુખ્યત્વે ઉત્પાદક અને કૃષિ ક્ષેત્રે થોડી રીકવરી થઈ હોવાથી કુલ રાષ્ટ્રીયઉત્પાદનોના વૃદ્ધિદરને સુધારીને પ્રગતિકારક અંદાજવામાં આવ્યો હતો.
S5	૧૯૯૯-૨૦૦૦ના વર્ષમાં ઉત્પાદન ક્ષેત્રેસુધારેલો વૃદ્ધિદર ૮.૫ ટકા હતો જે ગત વર્ષના ૭ ટકાના વૃદ્ધિ દર કરતાં ૧.૫ ટકોવધારે હતો ૧૯૯૮-૯૯ના વર્ષમાં ઉત્પાદન ક્ષેત્રે ૩.૬ ટકાનો વિકાસદરનોંધાયો હતો.
S6	ચાલુ વર્ષે કૃષિ ક્ષેત્રનો વિકાસ દર ૧.૩જેવો નજીવો હોવા છતાં તેમાં અગાઉના વર્ષના ૦.૮ ટકાની તુલનામાં નજીવોસુધારો અવશ્ય થયો હતો.
S7	૧૯૯૮-૯૯ના વર્ષમાં કૃષિ ક્ષેત્રે ૭.૨ ટકાનો વિકાસ દરહાંસલ કરી શકાયો હતો જેની તુલનામા ચાલુ વર્ષે નોંધપાત્ર ઘટાડોનોંધાયો હતો એમ સી.એસ.ઓ.ના આંકડાકીય માહિતીમાં કહ્યું હતું.
S8	કુલ રાષ્ટ્રીય ઉત્પાદનના સુધારેલા વિકાસમાંયોગદાન આપનાર અન્ય ક્ષેત્રોમાં વેપાર, હોટેલ ટ્રાન્સપોર્ટ અને સંદેશા વ્યવહારક્ષેત્રનો સમાવેશ થતો હતો.

TABLE 2. INTRA-SENTENCE COSINE SIMILARITIES

	S1	S2	S3	S4	S5	S6	S7	S8	S9
S1	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
S2	0.00	1.00	0.55	0.43	0.56	0.56	0.42	0.26	0.55
S3	0.00	0.55	1.00	0.44	0.52	0.29	0.24	0.32	0.24
S4	0.00	0.43	0.44	1.00	0.40	0.29	0.41	0.48	0.39
S5	0.00	0.56	0.52	0.40	1.00	0.48	0.60	0.24	0.55
S6	0.00	0.56	0.29	0.29	0.48	1.00	0.44	0.14	0.43
S7	0.00	0.42	0.24	0.41	0.60	0.44	1.00	0.31	0.64
S8	0.00	0.26	0.32	0.48	0.24	0.14	0.31	1.00	0.31
S9	0.00	0.55	0.24	0.39	0.55	0.43	0.64	0.31	1.00

S9	આ તમામ ક્ષેત્રોમાં અગાઉના વર્ષના ૫.૯ ટકાના અંદાજોનીતુલનામાં ૬.૭ ટકાનો વિકાસ દર નોંધાયો હતો જ્યારે બાંધકામ ક્ષેત્રમાંઅગાઉ ૯ ટકાના વિકાસદરનો અંદાજ મુકવામાં આવ્યો હતો જેની સરખામણીમાં ૯.૧ ટકાનોવિકાસ નોંધાયો હતો.

The matrix inTable 2shows intra sentence cosine similarities among the 9 sentences of the article shown in Table 1.Based on this matrix, a sentence similarity graph is constructed as shown inFigure 2, where edge exists if threshold is 0.5 or more.

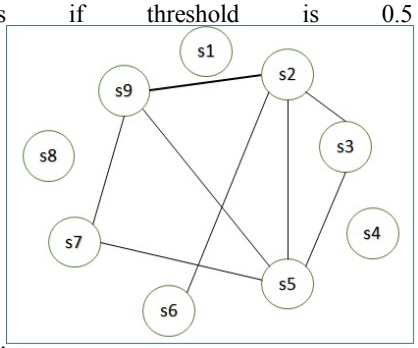

Figure 2. Similarity graph with threshold 0.5

For the proposed work, the threshold considered for finding LexRank between two sentences is 0.1 as recommended by Erkan and Radev(2004). The LexRank is calculated by distributing the score of sentence to its neighbor sentences.

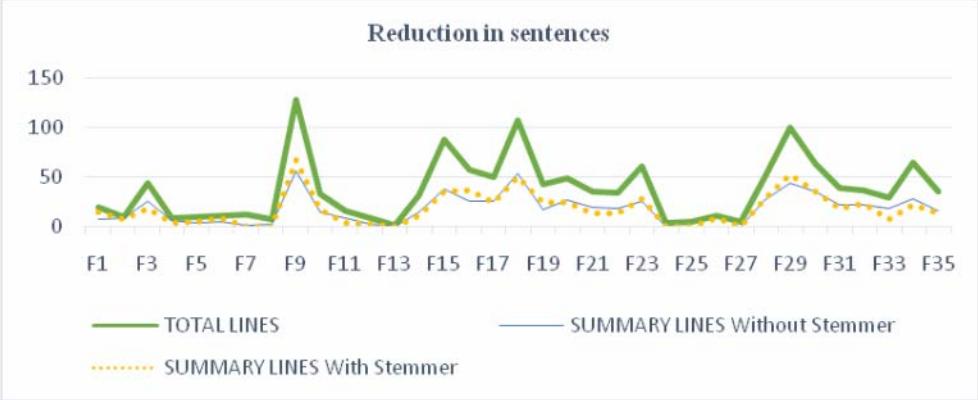

Figure 3. Sentence reduction after summary generation

$$p(u) = \sum_{v \in adj[u]} \frac{p(v)}{deg(v)}$$

where p(u) is the centrality of sentence u and adj[u] is the set of neighbor sentences of u and deg(v) is the degree of node v. The matrix representation of the graph is considered as Markov chain[8]. The stationary distribution of this chain gives a vector, which is LexRank of each sentence. If the sentence is similar to many other sentences, its LexRank shall be higher. Also, those sentences to which such high scored sentences are similar, shall also have high LexRank. The sentences that scored higher than the average score were chosen as a part of summary.

H. Anaphora resolver

Anaphora are forms of reference in written or spoken speech in which a word, most commonly a pronoun is used in place of a previously mentioned item (most often a noun or noun phrase) where both refer to the same entity[10]. A sentence which had anaphora is dependent on its respective noun. According to Denber, the most anaphora does not refer back more than one sentence in any case. So, to handle inter-sentential anaphora, we have restricted our focus to two-sentence window and a presumption that the antecedent of a given anaphora shall be present either in that same sentence or at the most to its previous one. Thus, if the sentences bearing anaphora were selected in the summary, then their previous sentences were also included in the summary. The list of pronouns considered are:

તેમણે, તેઓ, તેમને, તેમના, તે, આ and એ.

The algorithm followed by this component is as follows:

Step 1. Populate a data structure originalText that represents allsentences (s) of given text and $status_{si}$ to false where s_i represents i^{th} sentence of originalText.

Step 2. If LexRank score of s_i (where i>1) is above average score, then

Step 2.1 Create a window of two consecutive sentence-window (s_{i-1},s_i).

Step 2.2 If s_i consists of any of the pronouns from pronounlist, then

Step 2.2.1 if $status_{si}$ is false then set it to true.

Step 2.3 Set $status_{i-1}$ to true.

Step 3. Repeat step 2, for all s_i in originalText.

Step 4. End

Because most of the pronouns are part of the stop word list, they get removed during the process of stop-word removal. So this module finds anaphora from summary text and matches required line to be added to the summary text from the main text.

IV. EXPERIMENT AND RESULT DISCUSSION

The model was implemented in Java. The proposed Gujarati text summarization system was analyzed with the input of 140 news articles from EMILLE corpus. The average number of sentences in each file were 31.07 with each sentence of approximately 18 words. As text summarization is defined as “the process of condensing a source text into a shorter version preserving its information content”, the performance of this system is measured in terms of two parameters: condensation and content preservation.

A. Condensation

Text condensation was defined as

$$Tc = \frac{Number of sentences in summary text}{Number of sentences in original text}$$

The results found were as follows:

TABLE 3 . CONDENSATION IN TEXT

Data	Original text	Summary text by proposed summarizer	Reduction (%)
Lines	4350	2392	45
Words	76960	30784	60

Overall 45% reduction is found in number of lines in the summary text compared to the original lines of the given text. The reduction in number of lines in summary textincreases when stemmer is absent. This is because, stemmer is able to map one word to several sentences and thus increasing weightage of more sentences, results to more sentences in the summary. This phenomenon is represented in Figure 3.

To rank the sentences, stem weightage based Sentence – Stem vector was created. The size of this data structurewas observed in presence and absence of the morphological components present in the proposed text summarization. In the presence of String Similarity, Stemmer and usage of stop words, the vector size has been reduced by 50%. The condensation found with respect to data-structure created to generate the summary is shown in Figure 4. This implied that by using linguistic aspects the data structures created for processing can be significantly reduced.

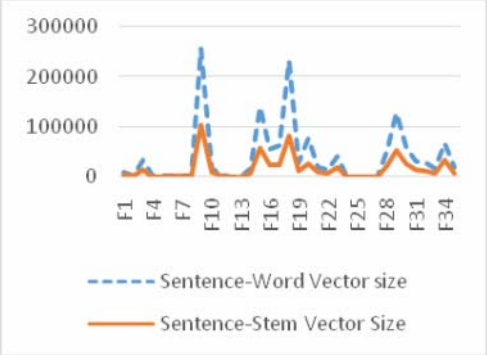

Figure 4. Reduction in Vector size

Majority of the text summarization system has predefined compression ratio, but the present work does not limits summary to certain length or ratio. We analyzed the results keeping in mind the text file size and the condensation achieved. The corpus on which we have worked had files ranging from 8 lines to 128 lines. It was found that as the size of file increases, the algorithm achieve greater condensation. This implies that the algorithm itself tends to reduce the lines from summary as original or given article's length is more. The present system is able to do so because of the threshold determined in average of all scores.

B. Content preservation

To evaluate content preservation, gold summaries are required. DUC provides document collections with known extracts to evaluate the performance of summarization system submitted at TREC conference. No such dataset is available for Gujarati text summarization. So human judges were involved to prepare gold summaries for these Gujarati articles. Later on, these gold summaries were used to measure the effectiveness on the proposed the text summarization.

Each article was given to 30 human judges. Each human judge had to select the best N lines that represent the text itself. For better comparison, N was defined as N=L*40/100 where L is total sentences in given document. Judges were allow to ceil or floor the value N.The results after comparing the top selected lines of human judges with lines selected by proposed summarizer is given in Table 5.

TABLE 4. PERFORMANCE OF PROPOSED SUMMARIZER SYSTEM

Measure	Summary text using Stemmer	
	With stemmer	Without stemmer
Precision	0.51	0.49
Recall	0.69	0.63
F-Score	0.59	0.54

As shown in theTable 4, the usage of stemmer improves the performance of a summarization system in terms of both parameters i.e. precision and recall. It can be easily derived that more improvement is seen in recall as compared to the precision, because stem has descriptive power. Hence, for the present text summarization system, the stemmer has enabled it to generate summaries similar to gold summaries.

There were 3% news articles, where LexRank was unable to generate summary due to inability of sentence-graph formation. When stemmer was applied to these articles and then the text was given for LexRank scoring, the summarizer was successfully able to generate a valid summary.

Anaphora resolver did not just improved sentence selection, but also corrected the false references in the summary. It helped to achieved coherency in the summary text. At the same time, the condensation percentage has decreased by 10% due to anaphora resolver. In general, we have found a trade-off between condensation and content preservation. As they are inversely proportional to each other, one of them should be prioritized before the actual text summarization model is developed.

C. Comparison with other summarizers

The proposed work is compared with other Indian language text summarization. The Tamil summarizer is evaluated using ROUGE-1 metrics (i.e. unigram based recall). For 150 summaries, their ROUGE score is 0.4723[11]. Similarly Bengali text summarization by sentence extraction is done. The summaries of 38 articles are evaluated using unigram based recall. They

have used only one reference summary. The average unigram based recall obtained is 0.4122[12]. The proposed Gujarati text summarizer has 0.69 recall. The proposed system is a desirable means to retrieve data automatically as retrieved by human judges for summarization.

V. CONCLUSION

The proposed text summarization system has recall of 0.69 with nearly 50% compression of given Gujarati text. The performance of text summarization improves by adding linguistics components to it. Though human generated summaries are difficult to achieve by automatic text summarization, still using linguistic components like Stemmer and String similarity measure, summary with good recall can be achieved.

REFERENCES

[1] Mitkov, R. 2003. The Oxford Handbook of Computational Linguistics, OUP Oxford, New York.

[2] Patel, A., Siddiqui, T., & Tiwary, U. 2007. A language independent approach to multilingual text summarization, RIAO2007, Pittsburgh PA, USA.

[3] Gupta, V. & Lehal, G. 2012. Complete Preprocessing Phase of Punjabi Language Text Summarization, International Conference on Computational Linguistics COLING'12, IIT Bombay, India, pp. 199-205.

[4] Gupta, V. & Lehal, G. 2013. Automatic Text Summarization for Punjabi Language," International Journal of Emerging Technologies in Web Intelligence, vol. 5, pp. 257-271.

[5] Kayasth, M. and Patel, B. 2009. "Offline typed Gujarati Character Recognition", National Journal of Science and Technology. 2(1). pp. 73-82.

[6] Sheth, J., & Patel, B. 2014. Dhiya: A Stemmer for morphological level analysis of Gujarati language, Proceeding of International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT) pp.151-154, 10.1109/ICICICT.2014.6781269.

[7] Sheth, J., & Patel, B. 2015. Gujarati Phonetics and Levenshtein based String Similarity Measure for Gujarati Language, In Proc. of 5th National Conference on Indian Language Computing, pp. 41-44, ISBN: 978-93-80095-61-5.

[8] Erkan G. & Radev, D. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research, Vol. 22, pp. 457–479.

[9] Huang, A. 2008. Similarity Measures for Text Document Clustering, Proceedings of New Zealand Computer Science Research Conference, Christchurch New Zealand, pp. 49-56.

[10] Denber, M., 1998. Automatic resolution of anaphora in English. Technical report, Eastman Kodak Co.

[11] Sarkar, K., Ram V. and Devi, S. 2011. Text Extraction for an Agglutinative Language. Language in India. vol. 11, no. 5, pp.56-59.

[12] Sarkar, K. 2012. Bengali text summarization by sentence extraction, Proceeding of ICBIM, pp. 233-245.

AUTHORS PROFILE

Jikitsha Sheth: She has 12 years of experience in teaching profession. Her area of interest includes Natural Language Processing and Machine Learning. She has published several research papers in national and international journal. She has research grants based on Gujarati language NLP projects too.

Bankim Patel: He is in teaching profession from more than 25 years with 24 years of research experience. His area of interest includes Natural Language Processing and Intelligent Information System. He has 50+ research papers published in several reputed national and international journals. Many students have pursued research under his able guidance. He has also penned books in computer science field. He has received many awards from national and international bodies. He has been invited for expert talks at reputed organizations and has been session chair and reviewer for different national and international conferences.