International Journal of Information Retrieval Research
Volume 12 • Issue 3

Ensemble of Support Vector Machine and Ontological Structures to Generate Abstractive Text Summarization

Amita Arora, J. C. Bose University of Science and Technology, YMCA, India*
https://orcid.org/0000-0002-9190-6195

ABSTRACT

Automatic summarization systems are much needed to lessen the information overload which is being faced by people due to the exponential growth of data on world wide web. These systems choose the most significant part of the text from a single document or multiple documents and present the compressed surrogate form of the complete information which was intended to be conveyed. In this research paper, the authors propose an approach to generate summary from a given text first by extracting the most relevant sentences and then making further concise by creating ontological structures of these sentences and then generating the abstractive summary from these structures. The proposed system is evaluated with DUC 2002 data set, and it is found that the performance of this system as evaluated using ROUGE-1 is 58.175, which is better than other state-of-the-art systems. The values reported in the experimental process of the research report the significant contribution of this innovative method.

KEYWORDS

Abstractive Summary, Concepts, Extractive Summary, Machine Learning, Ontology, Semantic Similarity, Support Vector Machine

INTRODUCTION

Automatic text summarization is a process of making a coherent summary that retains the most important points of original document using a computer program. It is a method for data reduction which enables users to reduce the amount of text that must be read to gather the essential information. Summarization helps user to find meaningful and relevant information from large documents. It plays a significant role in information retrieval and information gathering. Headlines, table of contents, abstracts, reviews, highlights etc. give the summarized view of a large text.

Summarization of text is a necessity as there is a large amount of data on the web expressing the same ideas. It requires deciding which sentences or phrases are to be chosen such that they show the main ideas in the document. Summaries of documents may help readers to go through the most important aspects of the document instead of having to read the full-length document. In some cases, human generated summaries are not sufficient to produce the desired results. For example, in information retrieval systems, a user query may not match the human generated summary of some document. In such cases summaries are needed to be generated on the fly as per the user query

DOI: 10.4018/IJIRR.300294 *Corresponding Author

keywords. This requires generating the summaries automatically to meet the dynamic requirement of text summarization.

The goal of summarization is to achieve high similarity of the summary information to the original document and lesser redundancy. Two major categories of text summarization are (i) extractive and (ii) abstractive summarization. Extractive summarization techniques select important sentences from the text to be extracted for generating summary. Importance of sentences is calculated on the basis of some features such as position of the sentence in the document, term frequency, lexical chains etc. For contents such as news articles and reviews about a product, there is a lot of redundancy. Using extractive summarization for this kind of content may not be a good idea as extractive summaries may contain unnecessary information. For this kind of content abstractive summaries provide a concise and compact idea of the content.(Uçkana, 2020) Abstractive summarization is able to generate sentences other than the original sentences in the given text. These new sentences have to be grammatically correct and able to convey the summarized information in a consistent way. This technique requires deeper text understanding to build some representation of text before generating the summary.(W. Xu, 2020)

The type of summarization under these premises is extractive at first, as a summary is produced just by copying the most relevant sentences from the original text, without rewriting them or making any change to the produced text. Then we create ontology for this extractive summary. This is where semantic conceptualization plays the key role by identifying the concepts from each sentence, generating sub-ontologies from them and merging the sub-ontologies by taking care of semantic roles of the concepts. The ontology thus produced from this extractive text when used to rephrase the sentences produces an abstractive summary of that text. The proposed technique tries to get success in removing the redundant information from the extractive summary of the text giving semantically correct yet more concise information.

RELATED WORK

Significant achievements have been obtained in the area of text summarization(Gambhir, 2017) (Ibrahim Altmami & El Bachir Menai, 2020). Different researchers have proposed many techniques to generate summary using features, using graphs as a collection of sentences as nodes, the edges denoting the similarity among sentences or by using cluster as a similarity measure or by using knowledge base or by using maximum diversity among sentences(Manju, David Peter, & Mary Idicula, 2021). These approaches may be divided into several categories:

GRAPH BASED APPROACHES

Leskove(Leskovec, 2005)generated document summary by using a semantic representation of the document and machine learning to create semantic sub-structure that can be used for extracting summaries. This approach shows the importance of the document semantic structure attributes in the sentence selection process. This can be used for abstract summary creation for a single document as well as multi document where linguistic features optimize the performance when training data contains shorter summaries while semantic features do the same for longer summaries.

In(Canhasi, 2014)Archetypal analysis and weighted archetypal analysis is used by Canhasi, to compute the positive and negative sentences for a given graph representation of a document set. Clustering and matrix factorization are also used in this approach.

Parveen (Parveen D. &., 2015)(Parveen D. H.-M., 2015) features a method to extract single document summary by making bipartite graph consisting of sentence and entity nodes. Sentences are ranked using a graph based ranking algorithm. Redundancy is removed and the sentences are checked for their local coherence and summary is generated. Very little linguistic information is contained in

the entity graph. In this method human subjects are included as judges to analyze the performance instead of domain experts that could give better judgment.

Another approach discussed in(Banerjee, 2015) by Siddhartha Banerjee takes the sentences from important documents and are aligned to sentences in other documents generating clusters of sentences that are similar. A word-graph structure is made from the sentences in each cluster and K-shortest paths are generated for this graph. Integer linear programming is used to select sentences having the shortest paths. H. Van (H. Van Lierde, 2019) proposes a fuzzy hypergraph model where sentences are nodes and fuzzy hyperedges are topics. User-defined query is used to extract a set of sentences from the corpus by maximizing their relevance,centrality in the fuzzy hypergraph and their coverage of topics in the dataset.Uçkana (Uçkana ., 2020)also uses weighted graphs that are non-oriented to represent text. The accuracy of this method depends on how correctly the sentences are transformed into graphs.Text summarization is taken as optimization problem by Gupta(Gupta, 2014) and is tried to be solved as weighted minimum vertex cover problem using textual entailment to exploit relationships among sentences. Raj et al.(Raj, Haroon, & Sobhana, 2020) use self-organizing maps clustering and name entity recognition for extracting the summary.

ONTOLOGY BASED APPROACHES

Some researchers have made efforts to utilize ontology to make the process of summarization better. Documents related to same domain may share the same information and can use the same domain ontology for this purpose. Ontology helps to extract the entities and categorization of sentences. Baralis (Baralis, 2013)depended upon YAGO ontology to evaluate and select sentences from documents. Entity recognition and disambiguation steps in the process of generation of document summary were performed using YAGO. Henning et al.(Hennig, 2008) used a hierarchical ontology to generate summaries. Their work maps the sentences of original document to the nodes of the ontology using an SVM classifier which is trained using search engines for sentence classification. Ontological knowledge was used byVerma(Verma, 2009) also to generate document summary. Query based summary is generated which utilizes WordNet or UMLS ontological knowledge to revise the query and then calculating the distance of query from each sentence. The sentences having lesser distance than a threshold are included in set of candidate sentences to be included in summary. These sentences are again divided into groups by calculating the pair wise distances among them and then the highest ranked sentences are chosen for the final summary. Natural language programming techniques were not used here. Abstract statistical data was also not utilized. (A.P.S., 2012)gave an approach for single document summarization that uses two sentence importance measures: first is the frequency of the terms in the sentence and the other is the similarity to the other sentences. The sentences in the document were ranked according to their respective scores and the top ranked sentences are selected for summary. The statistical sentence selection measures include: Sentence position, Cue words, Document frequency, Inverse document frequency, Term frequency. Their approach used nearest neighbor search technique to find the neighbor documents that are similar to the specified document. The sentences were scored using global affinity graph. The highest scored sentences were then checked for redundancy at the document level. Ragunath(Ragunath, 2006)presented an idea for ontology-based summarization to compute a set of features for each sentence based on the output of the hierarchical classifier. A sentence was classified to a leaf node or to an internal node. Nodes sharing common sub trees are matched using the classifier. If a sentence was mapping to more than one sub tree in the hierarchy, all nodes from each sub tree were included. For each sentence confidence weights assigned by the classifier were used to compute a sub tree overlap measure.

Amit et al. (Amit Vhatkar, 2020)uses subject-verb-object triples from sentences and forms a knowledge graph and the summary sentences are chosen by count of frequencies from these knowledge graphs.

MACHINE LEARNING BASED APPROACHES

These approaches learn a model that determines importance of sentences using a training corpus of full texts and their summaries. Different models such as Naive Bayes, Decision trees, SVM, HMM, CRF can be used Sujian(Sujian Li, 2007) (Amita Arora, 2017). These approaches operate at lexical level and provide good results for query-based summaries. The features considered by these models can be Sentence length, presence of indicator phrases, Sentence position (first/medium/final), Highly weighted content words, Containment of (important) named entities, Containment of specific topic words. A large amount of text is needed for learning purpose as human-generated summaries are required to train a classifier for the given text. These approaches are unable to manipulate information at abstract level.

Patil (Patil, 2014) showed that the choice of the classifier influences the performance of the trainable summarizer strongly. The procedure of automatic trainable summarization employed statistical and linguistic features which were extracted directly and automatically from the original text.Ramanujam(Ramanujam N., 2016)used the Naïve Bayesian Classification and the timestamp concept. This summarizer may work on many domains as it did not use knowledge base. The user could specify the compression rate so that amount of information to be extracted from the documents can be chosen. Singh(Singh, 2016) had presented a technique using unsupervised deep learning approach to summarize documents from Hindi and English. A set of eleven features were extracted from each sentence of document to generate the feature matrix which was passed through Restricted Boltzmann Machine to increase accuracy of choosing relevant sentences.Babara(Babara, 2015) used latent semantic analysis and fuzzy logic system to extract the summaries from the original text. A set of features was used which includes the title sentence, sentence length, sentence position, numerical data, proper nouns etc. Each feature was given a score using fuzzy logic. Based on this score each sentence was classified into three classes of important, average and unimportant and are thus selected to create the summary. This approach is not applicable for multi document summary.Xu J.(Xu J., 2019) uses syntactic compression using neural networks to achieve text summarization

Some other works include use of genetic algorithms(E. Vázquez, 2018), markov clustering algorithm(Conroy, 2001)(D. Sahoo, 2018).

PROBLEM STATEMENT

In this work we try to resolve the following problems which have been found in previously done work by other researchers.

Overlapped Information

In most of the contributions focusing on extractive summarization where whole sentences are included in summary may lead to overlapped information in summary. As semantic structure of sentence and semantic relationships between sentences is not taken into account, so despite being successful to an extent, these methods may not be able to identify sentences which are semantically equivalent. Thus, the final summary would contain redundant information.

Dangling Co-References

Coreference resolution corresponds to two entities referring to the same object. It is the task of identifying all coreferents in a text document that refer to the same entity(Balaji J., 2014) If during the text summarization process, the pronouns are extracted out of context, they may lose their relevance and thus generating the problem of ‘dangling co-references’ as given by(Christian Smith, 2012).

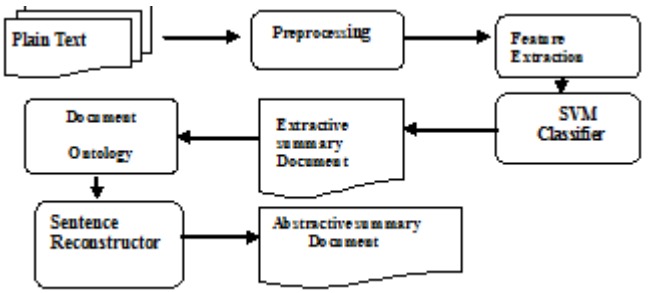
Ignoring the Semantic Structure Of Sentence

Aforementioned methods also treat sentences as bag of words and are unable to understand text deeply (Furu Wei, 2020).Despite learning to capture the document level context(Hong Wang, 2019) these are unable to capture the full semantic structure of the sentence. Some approaches use only hierarchical ontologies as discussed by(Hennig, 2008) or hierarchical classifiers for mappings as used by (Ragunath, 2006)ignoring the non-hierarchical semantic structure of the sentence.

PROPOSED APPROACH

The approach proposed here for summarizing a document is a hybrid technique that involves few sub-steps: i) extracting some statistical features from the text and using SVM classifier to generate extractive summary. ii) generating a document ontology for this extractive summary keeping into account the hierarchical and non-hierarchical relationships among the constituents of sentences in the extractive summary document. This is done by identifying and merging semantically similar sentences and concepts. Afterwards the sentences are reworded or reconstructed from the ontology to attain an abstractive summary. The architecture as shown in Figure 1 depicts how abstractive summary is being obtained from the plain text document by getting processed through different components of the system. After some pre-processing of the document to be summarized such as tokenization, stop words removal etc., some features are extracted from sentences of the document which are used by SVM classifier which extracts the sentences to be included in summary. This extractive summary may contain overlapped information such as different sentences but containing semantically similar information. This extractive summary document is further processed to construct an ontology. For constructing ontology from that extractive summary document, we identify the concepts in the document, their properties and relations among concepts. Some additional information i.e. semantic roles these concepts are playing in each sentence of the text is also attached while constructing ontology. For this the sentences of extractive summery document are parsed using Stanford dependency parser. The parser provides dependency tags attached with each term. By utilizing these dependency tags new tags are formed which specify the concepts, relations, semantic roles and properties. The concepts and relations are represented in an ontological structure with additional information such as semantic roles and properties of concepts in the sentence.

Figure 1. Proposed Architecture of Automatic Document Summarization using Ontological Structures



DETAILED DESIGN OF SYSTEM

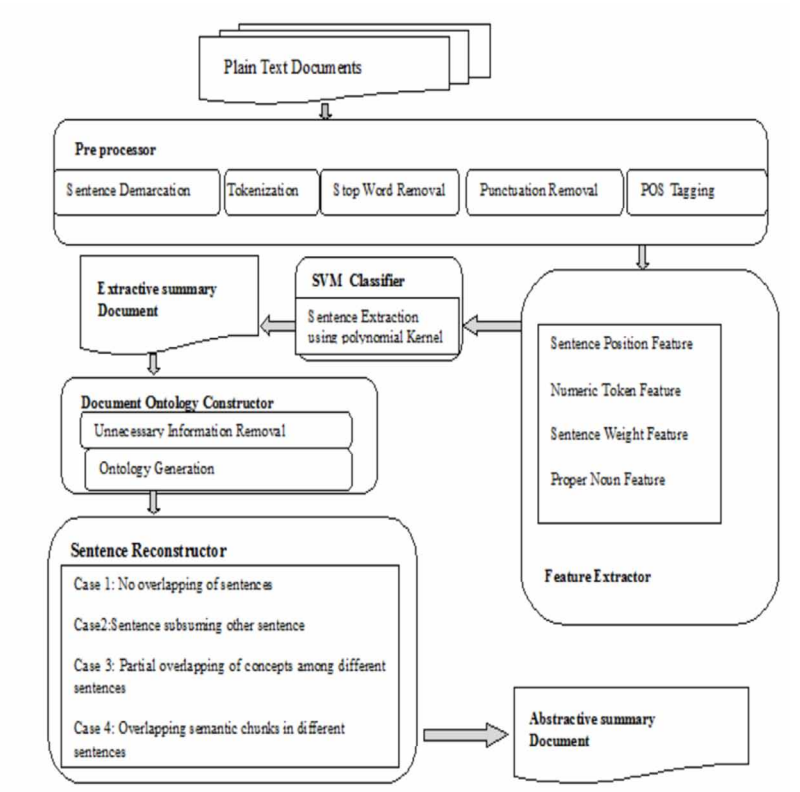
The input text documents are first processed by a natural language processor which performs the tokenization at sentence level, removes punctuation and stop words and performs part of speech

tagging, anaphora resolution and transform these sentences into tagged structures. Figure 2 shows the detailed design of the proposed system. The detail of each module of the system is as follows:

PRE-PROCESSOR

The text to be summarized has to go through some pre-processing steps so that this text can be used for extracting features. This module is having following components:

Figure 2. Detailed Design of the System



i)**Sentence demarcation**
ii)**Tokenization**
iii)**Stop Word Removal**
iv)**PunctuationRemoval**
v)**Part Of Speech Tagging**

A brief explanation of each step is given as follows:

SENTENCE DEMARCATION

In this step the complete text is divided into sentences using NLTK python library.

TOKENIZATION

The sentences are tokenized to generate tokens that are used to detect Keywords and Key phrases.

STOPWORD REMOVAL

Stop word are highly frequent words that do not carry any information. These are filtered out before processing the text. We have filtered out stop words list from the NLTK corpus.

PUNCTUATION REMOVAL

Punctuation marks are also removed to be not included in text features count.

PART OF SPEECH TAGGING

In part of speech tagging according to the category of words i.e. noun, verb, adjective etc. words are tagged. We have used Stanford Parser to perform the part of speech tagging.

The algorithm for pre-processing the text document to be summarized is given in Fig2.

The algorithm takes the text document as input and uses NLTK python library for its processing. In step 1, the sentence is marked for its beginning and ending. Step 2 tokenizes the sentence. Stop words and punctuation marks are removed from the sentence in step 3 and 4 respectively. Each token identified in step 2 of the algorithm is provided part of speech tag in step 5. The output of the algorithm is the pre-processed document which is given to the Feature Extractor module.

FEATURE VECTOR CALCULATOR

Feature vectors are generated for each sentence in the pre-processed text document. The value of each feature lies between 0 and 1. These feature vectors are used to form the feature matrix. Following features are extracted from the text.

SENTENCE POSITION FEATURE

Sentence relevance can be checked on the basis of its position in document. As first sentence of text document is supposed to contain important information, it is given a score 1. Also, last sentence of the sentence is the concluding sentence, it is also given score 1.

Sentence_Pos = 1, if sentence is the first or last sentence of text.
Sentence_Pos =cos((Sentence_Pos-minv)*((1/maxv)-minv)), for the rest of the sentences.

Where, *Sentence_Pos* is position of sentence in the text.

minv is calculated as (th*N)
maxv is calculated as (th*2*N)
N is total number of sentences in document.
th is threshold calculated as (0.2*N)

NUMERIC TOKEN FEATURE

This feature is calculated for the sentences containing numeric tokens by dividing total number of numeric tokens in that sentence with total number of words in that sentence.

Num_token_feature^{S}i = *num_numeric^{S}i /length*

where, *num_numeric^{S}i* is number of numeric tokens in ***i****th* sentence.

length is total number of words in sentence

WEIGHT OF THE SENTENCE:

Freq_word = Freq_word/Maximum Frequency Value

where, *Freq_word* is the frequency of a word occurring in a sentence.

Maximum Frequency Value is the frequency of the word occurring the maximum times in the document

Sentence_Weight= Sum(Freq_word of all words in the sentence)/length of sentence

PROPER NOUN FEATURE

Proper nouns refer to the named entity. Proper Nouns are calculated on basis of part-of speech tagging of each sentence. This feature is used to give importance to those sentences, which has proper nouns.

OUTPUT DATA FORMAT OF FEATURE VECTOR CALCULATION

The output of this module will be a set of feature vectors. The format of a feature set corresponding to a sentence will contains numeric values calculated as discussed.

X=[Numeric token Feature, Proper noun Feature, Sentence length Feature, Weight of the sentence, Unique term Feature]

For example, if following sentence is taken which is first sentence of the text document from the Implementation section of this paper.

The highest flood peaks on the Xijiang and Beijiang Rivers have passed, said Zhu Senlin, governor of south China's Guangdong Province.

The feature vector for this sentence will be calculated by this module is as follows:

X=[1.0, 0.0, 0.32, 0.5102040816326531, 0.1366666666666667, 0.24]

Here, sentence position feature=1, for being the first sentence of text.

Numeric token Feature = num_numeric /length

= 0/25=0

Proper noun Feature = number of terms tagged as nouns/length

= 8/25=0 .32

Sentence length Feature = length of sentence/length of the longest sentence

=25/49=0.5102040816326531

Weight of the sentence =sum (Freq_word of all words in the sentence)/length of sentence

= 3.41/25 = 0.1366666666666667

Unique term Feature = number of unique terms/length= 6/25= 0.24

For each document having N sentences in total, feature data will be having N feature sets. These feature sets are stored in a text file.

SVM CLASSIFIER

Support Vector Machines given by(Vladimir N. Vapnik, 1995) are among the best supervised learning algorithms which provide a powerful approach even in case of high dimensional feature space. (T., 1998)has been used by SVM classifiers for text categorization as shown in Figure3.

Considering its linear form for a binary problem with feature x and label $y \in \{-1, 1\}$

Training data is represented as $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$. We define the maximum margin hyperplane as given in (1)

$$\vec{w}.\vec{x} + b = 0 \tag{1}$$

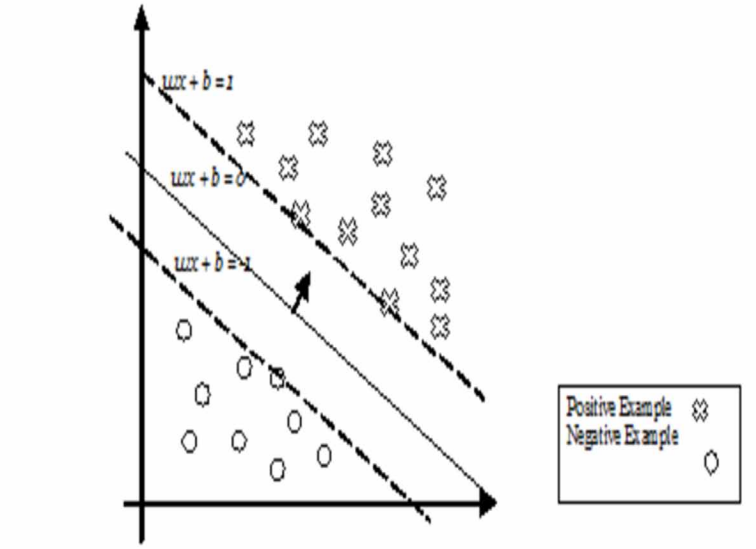
Where $\vec{w}$ is the normal vector to the hyperplane.

Two parallel hyperplanes are determined that separate the two classes of data, so that the distance between them is maximum. The region bounded by these two hyperplanes is called the "margin", and the maximum-margin hyperplane lies halfway between them as shown by (2) and (3).

$$\vec{w}.\vec{x} + b = 1 \tag{2}$$

$$\vec{w}.\vec{x} + b = -1 \tag{3}$$

Figure 3. SVM Classifier



To make the algorithm work for non-linearly separable datasets, the optimization is explicated as:

$min_{w,b} \frac{1}{2} \| w \|^2 + C\sum_{i=1}^{m} \xi_i$ where C is a scalar regularization hyperparameter.

Such that

$$y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 - \xi_i, \quad i = 1, \ldots, m \tag{4}$$

$\xi_i \geq 0, i = 1,\ldots,m$ where ξi is called slack variable.

The $\frac{1}{2} \| w \|^2$ specifies size of the margin and second $C\sum_{i=1}^{m} \xi_i$ specifies misclassification.

Support vectors are data points where the margin inequality constraint is active (i.e., an equality): There can be following types of support vectors:

i) $\xi_i = 0$. This implies $y^{(i)}\left(w^T x^{(i)} + b\right) = 1$. These are points that are on the margin.

ii) $\xi_i < 1$. These are points that can be classified correctly but do not satisfy the large margin constraint, i.e., distance to the hyperplane is less than 1

iii) $\xi_i > 1$. These are points that are misclassified.

This inequality ensures that all sample points that don't violate the margin are treated the same; they all have $\xi_i = 0$.

Training an SVM involves the reduction of above equation to a NP problem from which decision function can be derived by equation (5)

$$g\left(x\right) = \sum_{i=1}^{l} \lambda_i y_i x_i . \; x + b \tag{5}$$

where the parameter x_i determines the trade-off between increasing the margin-size and ensuring that it lie on the correct side of the margin.

SENTENCE EXTRACTION USING POLYNOMIAL KERNEL

For a non-linear decision surface, kernel trick is applied to maximum-margin hyperplane and the dot product is replaced by the kernel function as given by (6)

$$k\left(\vec{x}_i, \; y\right) = \left(\vec{x}_i y + 1\right)^2 \tag{6}$$

This polynomial kernel has been very effective when applied to several tasks of natural language processing (Joachims, 1998) of a second degree with a value of C as 0.0001. We have used the same for extracting summary from text. This extractive summary is further cleansed and an ontology graph is constructed with the cleansed extractive summary.

ALGORITHM DESIGN FOR SVM CLASSIFIER

The algorithm as shown in Figure 4begins with generating the training data file by labeling the sentences from the dataset as positive by providing them label +1 and some other sentences as negative by providing them label -1 in Step 1. This training file is used to train the SVM classifier in Step 2.

Figure 4. Algorithm for SVM classification

```
Algorithmsvm_classifier()
Input: feature vectors of Trainfile,ManualSumFile, test text file
Output: extractive summary of test text documents
Begin
Step1:  //generate training text file  train-5000.txt
        for each sentence S having feature vector x[i] of TrainFile in ManualSumFile
        y=+1,corresponding to feature vector x[i]
        else
        y= -1,corresponding to feature vector x[i]

Step 2: // train the svm classifier
        X_train, y_train = load_svmlight_file("train-5000.txt")
Step 3: load test text file
Step 4: Call  svm.SVC(kernel='poly', C=0.0001, degree=2, probability=False)
Step 5: Rank sentences according to their distance from hyperplane
Step 6: Take top 7 ranked sentences
End
```

The test text file whose summary is to be generated is loaded in step 3. In the next step SVM polynomial kernel of degree 2 is applied with value of C=0.0001. Step 5 ranks the sentences according to their distance from maximum margin hyperplane. Top 7 sentences are picked as extractive summary in step 6.

TEXT DOCUMENT ONTOLOGY CONSTRUCTOR

To remove redundancy among sentences in extractive summary we construct an ontological structure for the same by following these two steps.

UNNECESSARY INFORMATION REMOVAL

Information which is redundant and unnecessary in the extractive summary document is processed to be removed from this document(A, 2020). For this, different kind of transition words or phrases are identified and treated according to their type. Some of the types of transition words are given in Table 1.

Table 1. Transition Words Table

Transition word/ phrase type	Example words	Action Taken
Addition	indeed, further, as well (as this), either (neither), not only (this) but also (that) as well, also, as a matter of fact, moreover, what is more, in all honesty, and, furthermore, in addition (to this), besides (this), to tell the truth, or, in fact, actually, to say nothing of, etc.	Transition word/phrase removed
Introduction	such as, as, particularly, including, as an illustration, like, for example, in particular, for one thing, to illustrate, for instance, especially, notably, by way of example, etc.	Transition word/phrase removed. Also the text following the transition word/phrase in the sentence is removed.
Reference	speaking about (this), considering (this), regarding (this), with regards to (this), as for (this), concerning (this), the fact that, on the subject of (this), etc.	Transition word/phrase removed
Similarity	similarly, in the same way, by the same token, in a like manner, equally, likewise, etc	Transition word/phrase removed
Clarification	that is (to say), namely, specifically, thus, (to) put (it) another way, in other words,etc.	Transition word/phrase removed. Also the text following the transition word/phrase in the sentence is removed.
Conflict	but, by way of contrast, while, however, (and) yet, on the other hand, whereas, though (final position), in contrast, when in fact, conversely, etc.	Transition word/phrase removed
Emphasis	even more, above all, indeed, more importantly, Besides, etc.	Transition word/phrase removed
Result	as a result (of this), consequently, hence, for this reason, thus, because (of this), in consequence, so that, accordingly, as a consequence, so much (so) that, so, therefore, etc.	Transition word/phrase removed
Purpose	for the purpose of, in the hope that, for fear that, so that, with this intention, to the end that, in order to, Lest, with this in mind, in order that, so as to, so, etc.	Transition word/phrase removed
Consequence	under those circumstances, then, in that case, if not, that being the case, if so, otherwise	Transition word/phrase removed
Sequential Transition	in the (first, second, etc.) place, initially, to start with, first of all, thirdly, (&c.), to begin with, at first, for a start, secondly, etc.	Transition word/phrase removed
Continuation	subsequently, previously, eventually, next, before (this), afterwards, after (this), then, etc.	Transition word/phrase removed
Conclusion	to conclude (with), as a final point, eventually, at last, last but not least, in the end, finally, lastly, etc.	Transition word/phrase removed
Degression	to change the topic, incidentally, by the way, etc.	Transition word/phrase removed
Resumption	to get back to the point, to resume, anyhow, anyway, at any rate, to return to the subject, etc.	Transition word/phrase removed
Consession	but even so, nevertheless, even though, on the other hand, admittedly, however, nonetheless, despite (this), notwithstanding (this), Albeit (and) still, although, in spite of (this), regardless (of this), (and) yet, though, granted (this), be that as it may, etc.	Transition word/phrase removed

Table 1 continued on next page

Table 1 continued

Transition word/ phrase type	Example words	Action Taken
Summation	as was previously stated, so, consequently, in summary, all in all, to make a long story short, thus, as I have said, to sum up, overall, as has been mentioned, then, to summarize, to be brief, briefly, given these points, in all, on the whole, therefore, as has been noted, hence, in conclusion, in a word, to put it briefly, in sum, altogether, in short, etc.	Transition word/phrase removed

ONTOLOGY GENERATION

After removing the transition words and unnecessary information, the researchers identify the Elementary semantic Chunks (ESCs) (concept-verb-concept) from the text that comprise of concepts in the extractive document, their properties and relations among concepts. These ESCs contain some additional information also i.e. semantic roles these concepts are playing in each sentence of the text. Including these semantic roles in ESCs leads to deeper text understanding which leads to a rich Ontology generation. The ontology graph is being generated in this work using the method described by(A, 2020)(Arora, 2017) which includes constructing sub ontologies for each sentence and then merging these sub ontologies using the rules proposed in ontology mapping and merging module. The algorithm shown in Figure 5 generates the overall ontology for a given text document by calling the methods defined for the all the modules of the system. The algorithm takes extractive summary document from SVM as input and gives a final ontology of that document.

Figure 5. Algorithm for Ontology Generation

```
Algorithm Ontology_Generator
Software Tools Used: Stanford Dependency Parser, Senna, JavaRAP,
Input: A set of text documents,  WordNet
Output: Ontology of documents
Begin
for each document
for each sentence
    1. pre_processor();
    2. anaphora_resolver();
                                                    //output is pronouns resolved to their
                                                                    noun mentions
    3. dependency_parser();
                                        //output is part of speech tagged sentences along with
                                                                    //their dependencies
    4. ner_tagger();
                                    // output is named entity tagged nouns
    5. concept_relation_role_extractor();
//output is concept, roles, object properties
    6. property_extractor();// output is data properties of concepts
    7. hierarchy_extractor();// output is hierarchies of concepts if exist
    8. Generate sentence ontologies using Jena API for ontology
                // generate sentence  ontologies
    end for
    9. ontology_matcher_merger();// match and merge ontologies to generate full ontology
    10. Generate the ontology in rdf format using Jena API and GraphViz
end for
End
```

Here, contextual relationships among sentences are described by the semantically similar concepts or the ESCs that are common in different sentences. This is shown in the ontology graph in Figure 7 having multiple edges among semantic chunks from different sentences or having same concept but different or same roles in multiple sentences.

SENTENCE RECONSTRUCTOR

The ontology thus constructed after removing unnecessary words or phrases or parts of sentences gives the concise representation of the extractive summary. In this process of ontology creation, as discussed in approach(A, 2020), similar sentences or parts of sentences are merged by finding the semantic and syntactic similarity among different sentences based on hierarchy and roles of the concepts in the sentences. This ontology is used now to reframe the sentences to obtain the abstractive summary. There can be following cases while rewording the sentences:

Case 1: No overlapping of sentences

Sentences are taken as such in the summary.

Case 2: Sentence subsuming another sentence

The longer sentence is taken into summary and the subsumed sentence is discarded.

Case 3: Partial overlapping of concepts among different sentences

This can be the case where a concept is having same role in different sentences but is part of different semantic chunks (concept-verb-concept). Connector such as *and* or *then* is used.

Case 4: Overlapping semantic chunks in different sentences

A semantic chunk can be overlapping in two sentences and can be used as a connector for merging two sentences.

Figure 6. *Algorithm for Sentence Constructor*

```
Algorithm sentence_reconstructor()
Input  each sentence of extractive summary,
Output reframed sentences for abstractive summarization
Begin
1:  for each sentence Si of extractive summary
        match each row to object property table of final ontology
2:    if all rows are matched                          //case 1:subsumption
                discard the sentence Si from summary
    3:    if no rows are matched                    // case 2: no overlapping
                keep the sentence in summary
    4:    if some rows have  matching concept –verb-concept triplets //case 3
            merge the sentences by keeping overlapped concept-verb- concept triplet
5:    if concepts are matching in but relations and semantic roles are different
            merge the sentences by keeping that sentence and using connector
        "and"                    with    the    other    matched    sentence
        //case 4
 6: perform inverse co reference resolution
End
```

This algorithm takes the extractive summary of the text document, the object property tables of each sentence of the document and the object property table of the final ontology of the document.

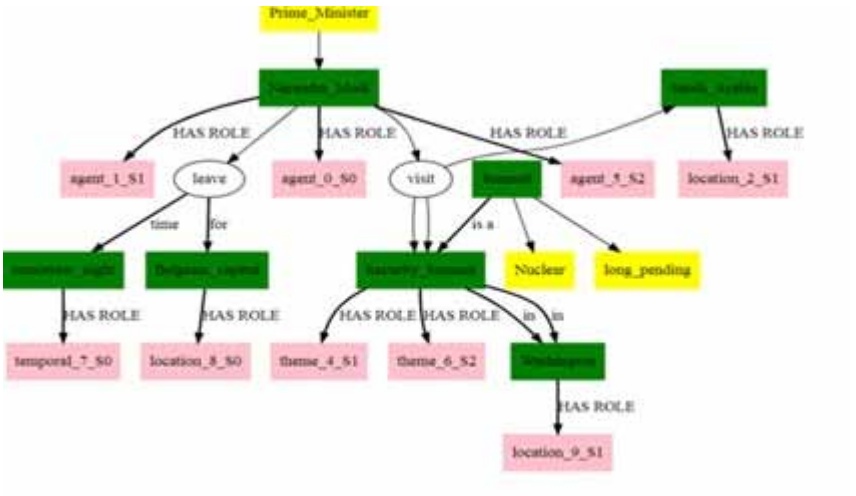
The algorithm begins by processing the object property table of each sentence as the step 1. Step 2 ensures if all rows of object property table of a sentence are matching with those of final object property table then this sentence is being subsumed by another longer sentence and is not taken in final summary. Step 3 checks If there is no overlapping of sentences then the sentence is kept as such in the summary. Step 4 finds whether there is overlapping among sentences. Step 5 merges the parts of sentences which are matching. Step 6 merges the sentences by using the connector word "and". Step 7 performs the inverse co reference resolution to convert repeated entity names in consecutive sentences to suitable pronoun.

An example of sentence reconstruction from text document ontology is shown as follows:

Text Document: *Prime minister Narendra Modi will leave for the Belgium capital tomorrow night. He will attend the Nuclear Security Summit in Washington and visit Saudi Arabia. Prime minister will take part in the long-pending Summit for the first time.*

The final ontology will be shown graphically in the Figure 6.

Figure 7: Ontological Structure for the Example Text



Applying the sentence reconstruction algorithm, the abstractive summarized text will be:

Abstractive Summary: *Prime Minister Narendra Modi will leave for the Belgium capital tomorrow night to visit the long-pending Nuclear Security Summit for the first time in Washington and visit Saudi Arabia.*

IMPLEMENTATION

For a text document shown below, the feature matrix is generated after pre-processing and extracting features described in section for each sentence. This feature matrix is given as input to the trained SVM classifier. These sentences are ranked according to their distance from the hyper-plane.

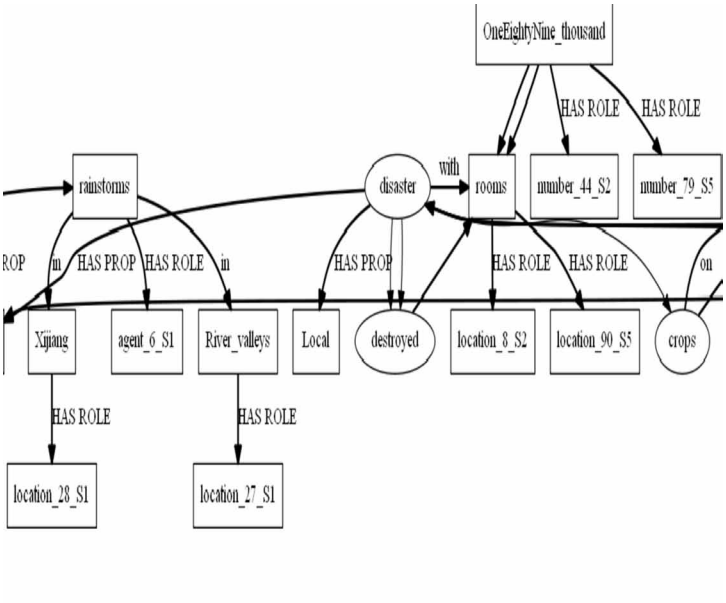
Original Text: *The highest flood peaks on the Xijiang and Beijiang Rivers have passed, said Zhu Senlin, governor of south China's Guangdong Province. While inspecting flood control and relief work in Qingyuan city on the Beijiang River yesterday, he attributed Guangdong's success in combating this flood -- almost the biggest in 100 years -- to concerted efforts by the armymen stationed in Guangdong and local residents. More than 200 people lost their lives in the natural disaster, which destroyed 189,000 rooms and ruined crops on 1.2 million hectares. The flood was caused by successive torrential rainstorms in the Xijiang and Beijiang River valleys in early and middle June. Major flood monitoring stations on the two rivers recorded their highest water levels, all four meters above the danger mark. Local governments at various levels in the province have paid close attention to flood control work, and leading government and communist party officials of different localities have gone to the flood-fighting front. No breaches of major embankments or reservoirs were reported despite the most serious flood in a hundred years, effectively protecting the safety of the provincial capital, Guangzhou, and the Pearl (Zhujiang) River Delta. But the losses caused by the flood were quite serious, said the governor. According to him, 11 million people in the province's nine cities and 55 counties were affected, and more than 200 people died in the natural disaster, with 189,000 rooms destroyed and 1.2 million hectares of crops ruined. The direct economic losses were set at 10.2 billion yuan. The governor warned that though the flood danger had receded, the determination to fight possible further floods could not slacken, as this is just the beginning: the main flood season, which usually begins in late July and early August, has not yet arrived. He urged local officials to be on constant alert against further possible floods and be meticulous about flood prevention and control measures, while doing their utmost to help flood victims, assisting them to resume production as soon as possible and maintaining social stability.*

The farther the sentence from hyper-plane, more confidently the classifier predicts it a positive example. From the sentences which were labeled positive we have taken top 35% of ranked sentences as extractive summary as shown here:

Extractive Summary: *But the losses caused by the flood were quite serious, said the governor. The flood was caused by successive torrential rainstorms in the Xijiang and Beijiang River valleys in early and middle June. More than 200 people lost their lives in the natural disaster, which destroyed 189,000 rooms and ruined crops on 1.2 million hectares. Local governments at various levels in the province have paid close attention to flood control work, and leading government and communist party officials of different localities have gone to the flood-fighting front. According to the governor, 11 million people in the province's nine cities and 55 counties were affected, and more than 200 people died in the natural disaster, with 189,000 rooms destroyed and 1.2 million hectares of crops ruined. The direct economic losses were set at 10.2 billion yuan. The governor warned that though the flood danger had receded, the determination to fight possible further floods could not slacken, as this is just the beginning: the main flood season, which usually begins in late July and early August, has not yet arrived.*

This extractive summary text is converted into text document ontology using technique discussed by (A, 2020)and a small part of that ontology is shown using GraphViz tool in Figure 8.

Figure 8. A part of Ontology Structure



After applying the cases discussed in the *Sentence Reconstructor* module, Final summary is obtained as shown below:

Abstractive Summary: *The flood was caused by successive torrential rainstorms in the Xijiang and Beijiang River valleys in early and middle June. Local governments at various levels in the province have paid close attention to flood control work, and leading government and communist party officials of different localities have gone to the flood-fighting front. 11 million people in the province's nine cities and 55 counties were affected, and more than 200 people died in the natural disaster, with 189,000 rooms destroyed and 1.2 million hectares of crops ruined. The direct economic losses were set at 10.2 billion yuan. The governor said the losses caused by the flood were quite serious and warned that though the flood danger had receded, the determination to fight possible further floods could not slacken as this is just the beginning the main flood season which usually begins in late July and early August has not yet arrived.*

The manual (reference) summary as given in the DUC2002 dataset for the given document is shown here:

Manual Summary: *The highest flood peaks on the Xijiang and Beijiang Rivers have passed, said Zhu Senlin, governor of south China's Guangdong Province. No breaches of major embankments or reservoirs were reported despite the most serious flood in a hundred years. Eleven million people in the province's nine cities and 55 counties were affected, more than 200 people died, 189,000 rooms were destroyed and 1.2 million hectares of crops were ruined in this natural disaster. Economic losses were set at 10.2 billion yuan. The really bad news is that this is just the beginning. The main flood season has not yet started.*

It can be analyzed here that proposed system generated summary contains most of the sentences that are present in the manual summary.

EXPERIMENTAL EVALUATION

This section describes the experimental results of the proposed approach with appropriate simulations.

CORPUS

In this research, DUC 2002(Over P., 2007) corpus is used by the researchers for the evaluation of the proposed approach. The corpus contains 533 different sets of newspaper articles. and it contains up to three human summaries for each document. These manually-created extracts, that have a length of approximately 100 words are used to generate training set for our SVM classifier. These human summaries are considered as model summaries for our evaluation. Here 5000 random positive and negative example sentences from DUC2002 dataset of 533 documents are generated with positive examples indicating the presence of sentence in the extractive summary and negative sentence indicate absence of sentence in the extractive summary. The reason to use this dataset is that it is the most widespread corpus for single-document summarization and also allows to compare our proposed method with other state-of-the-art approaches under the same conditions.

EVALUATION ENVIRONMENT

The conducted evaluation verifies the performance of the proposed approach by determining the accuracy of the generated summaries and getting the measure of their quality with respect to the information contained. The SVM classifier is trained and evaluated using Python on dataset using polynomial kernel of second degree with a value of c as 0.0001.

For ontology generation and sentence reconstruction the tools used are: Stanford Dependency Parser for dependency parsing, Senna for name entity recognition and JavaRAP for anaphora resolution.

EVALUATION METRICS

The evaluation is performed using ROUGE measure (the measure adopted by DUC as the standard for assessing the summary coverage) that calculates the intersection of n-gram, word pairs and word sequences between candidate summaries and the reference or human-generated summaries (e.g., unigrams, bigrams, or word sequences). ROUGE is recall-oriented, based on n-gram overlap, and correlates well with human evaluations as discussed by (Lin, 2003)(Lin., 2004). This measure allows the automatic evaluation of text summaries where the content of these summaries is compared to a model one, (human generated summary). Based on these, ROUGE implements different metrics, such as ROUGE-1, ROUGE-2, ROUGE-L etc.

ROUGE returns the commonly used NLP measures: Precision, Recall and F-measure.

The evaluation method, in which an automatic summary compared with a reference/manual summary, is based on the n-grams of words of the automatic summary coinciding in the manual summary.(Carlos-Francisco Méndez-Cruz, 2017). We get a score of recall if the number of co-occurrences are divided by the total $\boldsymbol{n}$-grams of the manual summary, whereas we get a score of precision, if the number of co-occurrences are divided by the total $\boldsymbol{n}$-grams of the automatic summary. These two scores can be further combined to obtain an $\boldsymbol{F}$-score for the automatically generated summary. We can say that the recall tells how much relevant information is obtained from the manual summary, and the precision depicts how much relevant information we get in the automatically generated summary. ROUGE scores range between 0 and 1, where 1 is better. Here, reference summaries were manually generated by experts, so they are considered as gold-standard summaries.

We calculate the scores for ROUGE-N as follows in equations (7), (8) and (9):

$$Recall = \frac{n-gram\left(system\ \ generated\ \ summary, reference\ \ summary\right)}{number\ \ of\ \ n\ \ gram\ \ in\ \ reference\ \ summary} \tag{7}$$

$$Precision = \frac{n-gram\left(system\ \ generated\ \ summary, reference\ \ summary\right)}{number\ \ of\ \ n\ \ gram\ \ in\ \ system\ \ generated\ \ summary} \tag{8}$$

$$F-score = \frac{2RecallPrecision}{Recall + Precision} \tag{9}$$

COMPARISON OF DIFFERENT TECHNIQUES

The proposed system is compared to the following systems which are also implemented and experimented upon here on same data used for evaluation for the proposed system

- The lead baseline summaries which are approximately the first 150 words of the original text
- Automatic summarization tool TextRank that uses an unsupervised and undirected graph-based extractive technique that uses the structure of document to find important topics for constructing a graph whose vertices are text units and the lexical similarity between two vertices form the edges. Each vertex is ranked using the random walk algorithm for generating summary based on the eigenvector(Mihalcea, 2004).

Table 2. Comparision of Implemented Systems with Baseline and TextRank

System	Recall	Precision	F-score
Baseline	45.8888	44.8892	45.3853
Textrank	42.0053	48.2547	45.3091
Without-ontology	58.1751	37.7685	45.8016
With-ontology	52.9745	46.0319	49.25978

As shown in the Table 2 the system that generates abstractive summary using ontological structures has higher precision than Baseline summaries. Recall for the proposed system is also higher than both TextRank and Baseline summaries. Using ontological structure with this system generated extractive summaries improves the precision but lowers recall. F-score of this system generating abstractive summaries is highest among baseline, TextRank and extractive summaries.

COMPARISION WITH OTHER MODELS

To validate the effectiveness of the proposed model, we conducted a comprehensive comparison with other state of art methods Here instead of precision or F-measure, only ROUGE recall metric is used for the comparison with other models. As besides the proposed system (extractive and abstractive summary) and TextRank, other summarization methods have not been re-implemented here. Instead, published results are being relied upon for all the compared approaches so the only common metric across all the approaches was the recall metric for ROUGE. (J. Rodríguez-Vidal, 2020). ROUGE-1 is chosen here as according to Lin (Lin., 2004), for concise summaries ROUGE-1 may suffice.

The proposed system is compared in terms of Recall only to the following systems:

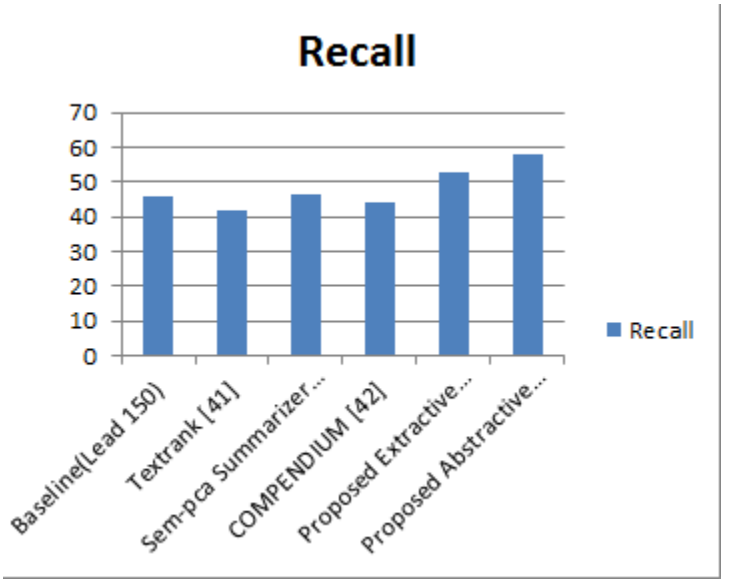
- The lead baseline summaries
- Automatic summarization tool TextRank
- COMPENDIUM(Lloret, 2013) a Text summarization tool that generates informative summaries for generic, extractive and abstractive summaries.
- Sem-PCA Summarizer(Alcon O, 2018) which uses Principal Component Analysis to reduces the dimension of a document enriched with semantic information. PCA is used to identify and rank the relevant concepts from a concept text matrix used for selecting the most important sentences to generate summaries.
- Proposed Extractive Summarization using SVM
- Abstractive summaries by the proposed system which are generated after using automatically generated ontological structures..

Table 3. Comparison of Different Systems

System	Recall
Baseline(Lead 150)	**45.888**
Textrank(Mihalcea, 2004)	**42.005**
Sem-pca Summarizer(J. Rodríguez-Vidal, 2020)	**46.688**
COMPENDIUM(Lloret, 2013)	**44.022**
Proposed Extractive Summarization using SVM	**52.974**
Proposed Abstractive Summarization using SVM and ontological structures	**58.175**

As shown in the Table 3, the proposed system that generates abstractive summary using ontological structures surpasses the Baseline summaries and all other systems in terms of Recall. The same is shown using a bar graph in Figure 9.

Figure 9. Result Analysis of Different Approaches



CONCLUSION

The objective of this research was to design, implement and evaluate a framework for single document text summarization. In this research, extractive text summarization approach using Support Vector Machines technique enhanced with semantic information using ontological structures is proposed and analyzed. In this approach, specifically, SVM is used as a detector of important sentences in the text. Further the information contained in these extracted sentences is condensed by constructing the ontological graphs which relates the sentences from extractive summary semantically and leads to discarding the unnecessary information and then generating the abstractive summary. Thus, the proposed framework is capable of generating a new summarized logical coherent sentence. The experimental study manifested that the results confirmed that our model produces good summaries and shows encouraging and stimulating performance which is more than the state of the art and baseline approaches. In particular, the best results were obtained for the ROUGE metrics, when ontological structures are being used that exploit the semantic knowledge present within the text considering the context of the concepts. So, it can be remarkably helpful for providing key information from the text.

FUNDING AGENCY

Publisher has waived the Open Access publishing fee.

REFERENCES

A, A. (2020). Automatic Ontology Construction: Ontology From Plain Text Using Conceptualization and Semantic Roles. In *Critical Approaches to Information Retrieval Research.*

Alcon, O. L. E., & Lloret, E. (2018). Sempca-summarizer: Exploiting semantic Principal component analysis for Automatic summary generation. *Computer Information*, *37*(5), 1126–1148. doi:10.4149/cai_2018_5_1126

Amit Vhatkar, P. B. (2020). Knowledge Graph and Deep Neural Network for Extractive Text Summarization by Utilizing Triples. *Proceedings of the 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation*, 130-136.

Amita Arora, M. S. (2017). Machine Learning Approach for Text Summarization. *International Journal of Database Theory and Application*, *10*(8), 83–90. doi:10.14257/ijdta.2017.10.8.08

A.P.S., R. Y. (2012). An Efficient Approach for Web document summarization by Sentence Ranking. *International Journal of Advanced Research in Computer Science and Software Engineering*, *2*(7).

Arora, A. S., Singh, M., & Chauhan, N. (2017). Automatic Ontology Construction using Conceptualization and Semantic Roles. *International Journal of Information Retrieval Research*, *7*(3), 62–80. doi:10.4018/IJIRR.2017070104

Babara, S. P. (2015). Improving Performance of Text Summarization. *Procedia Computer Science,* 354-363.

Balaji, J. T. G., Geetha, T. V., & Ranjani, P. (2014). Graph-Based Bootstrapping for Coreference Resolution. *Journal of Intelligent Systems*, *23*(3), 293–310. doi:10.1515/jisys-2013-0056

Banerjee, S. M. (2015). Multi-Document Abstractive Summarization Using ILP Based Multi-Sentence Compression. *Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015).*

Baralis, E. C., Cagliero, L., Jabeen, S., Fiori, A., & Shah, S. (2013). Multi-document summarization based on the Yago ontology. *Expert Systems with Applications*, *40*(17), 6976–6984. doi:10.1016/j.eswa.2013.06.047

Canhasi, E. (2014). *Graph-based models for multi-document summarization.* PhD thesis.

Carlos-Francisco Méndez-Cruz, S. G.-C.-A.-P.-V.-J.-R.-V. (2017). First steps in automatic summarization of transcription factor properties for RegulonDB: Classification of sentences about structural domains and regulated processes. *Database (Oxford)*, *2017*, bax070. Advance online publication. doi:10.1093/database/bax070 PMID:29220462

Christian Smith, H. D. (2012). A More Cohesive Summarizer. COLING 2012.

Conroy, J. M. (2001). Using HMM and Logistic Regression to Generate Extract Summaries for DUC. *Proceedings of the DUC 01*.

Gambhir, M. G., & Gupta, V. (2017). Recent Automatic Text Summarization Techniques: A Survey. *Artificial Intelligence Review*, *47*(1), 1–66. doi:10.1007/s10462-016-9475-9

Gupta, A. K. (2014). Text Summarization Through Entailment-Based Minimum Vertex Cover. *Proceedings of the Third Joint Conference on Lexical and Computational Semantics (SEM 2014)*. doi:10.3115/v1/S14-1010

Hennig, L. W. (2008). An Ontology-based Approach to Text Summarization. *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops.*

Ibrahim Altmami, N., & El Bachir Menai, M. J. (2020). Automatic summarization of scientific articles: A survey. *Journal of King Saud University - Computer and Information Sciences* .

Leskovec, J. M.-F. (2005). *Extracting Summary Sentences Based on the Document Semantic Graph.* Microsoft Research MSR-TR-2005-07.

Li, S. Y. O. (2007). Multi-document summarization using support vector regression. *Proceedings of DUC.*

Lin, C.-Y. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. *HLT-NAACL-PARALLEL '03: Proceedings of the HLT-NAACL 2003 Workshop on Building and using parallel texts: data driven machine translation and beyond, 3.*

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*. Association for Computational Linguistics.

Lloret, E. P., & Palomar, M. (2013). COMPENDIUM: A Text Summarisation Tool for Generating Summaries of Multiple Purposes, Domains, and Genres. *Natural Language Engineering*, *19*(2), 147–186. doi:10.1017/S1351324912000198

Manju, K., David Peter, S., & Mary Idicula, S. (2021). A Framework for Generating Extractive Summary from Multiple Malayalam Documents. *Information (Basel)*, *12*(1), 41. doi:10.3390/info12010041

Mihalcea, R. &. (2004). Textrank: Bringing order into texts. In *Proceedings of EMNLP* (pp. 404-411). Academic Press.

Over, P. D. H., Dang, H., & Harman, D. (2007). DUC in Context, Information Processing and Enhanced Graph Based Approach for Multi Document Summarization. *Information Processing & Management*, *43*(6), 1506–1520. doi:10.1016/j.ipm.2007.01.019

Parveen, D. &. (2015). Integrating Importance, Non-Redundancy and Coherence in Graph-Based Extractive Summarization. *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015).*

Parveen, D. H.-M. (2015). Topical coherence for graph-based extractive summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. doi:10.18653/v1/D15-1226

Patil, M. S. (2014). A Hybrid Approach for Extractive Document Summarization Using Machine Learning and Clustering Technique. *International Journal of Computer Science and Information Technologies, 5*(2).

Ragunath, R. (2006). Ontology Based Text Document Summarization System,Using Concept Terms. *Journal of Engineering and Applied Sciences (Asian Research Publishing Network).*

Raj, M., Haroon, R., & Sobhana, N. (2020). A novel extractive text summarization system with self-organizing map clustering and entity recognition. Indian Academy of Sciences, 45(32).

Ramanujam, N. K. M., & Kaliappan, M. (2016). An Automatic Multidocument Text Summarization Approach Based on Naïve Bayesian Classifier Using Timestamp Strategy. *TheScientificWorldJournal*, *2016*, 1–10. doi:10.1155/2016/1784827 PMID:27034971

Rodríguez-Vidal, J., Carrillo-de-Albornoz, J., Amigó, E., Plaza, L., Gonzalo, J., & Verdejo, F. (2020). Automatic generation of entity-oriented summaries for reputation management. *Journal of Ambient Intelligence and Humanized Computing*, *11*(4), 1577–1591. doi:10.1007/s12652-019-01255-9

Sahoo, D., Bhoi, A., & Balabantaray, R. C. (2018). Hybrid approach to abstractive summarization. *Procedia Computer Science*, *132*, 1228–1237. doi:10.1016/j.procs.2018.05.038

Singh, S. K. (2016). Bilingual Automatic Text Summarization Using Unsupervised Deep Learning. *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)* – 2016. doi:10.1109/ICEEOT.2016.7754874

T., J. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. In *Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)* (pp. 137-142). European Conference on Machine Learning.

Uçkana, K. A. (2020). Extractive multi-document text summarization based on graph independent sets. *Egyptian Informatics Journal*, *21*(3), 145–157. doi:10.1016/j.eij.2019.12.002

Van Lierde, T. C. (2019). Learning with fuzzy hypergraphs: A topical approach to query-oriented text summarization. *Information Sciences, 496.* 10.1016/j.ins.2019.05.020

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer. doi:10.1007/978-1-4757-2440-0

Vázquez, E., Arnulfo García-Hernández, R., & Ledeneva, Y. (2018). Sentence features relevance for extractive text summarization using genetic algorithms. *Journal of Intelligent & Fuzzy Systems*, *35*(1), 353–365. doi:10.3233/JIFS-169594

Verma, R. C. (2009). A Semantic Free-text Summarization System Using Ontology Knowledge. *IEEE Transactions on Information Technology in Biomedicine*, *5*(4), 261–270.

Wang, H. X. W. (2019). Self-supervised learning for contextualized extractive summarization. Association for Computational Linguistics, 2221–2227.

Furu Wei, M. Z. (2020, October). At Which Level Should We Extract? An Empirical Analysis on Extractive Document Summarization Qingyu Zhou. *Tencent Cloud Xiaowei Beijing*.

Xu, W., Li, C., Lee, M., & Zhang, C. (2020). Multi-task learning for abstractive text summarization with key information guide network. *EURASIP Journal on Advances in Signal Processing*, *2020*(1), 16. doi:10.1186/s13634-020-00674-7

Xu J., D. G. (2019). Neural extractive text summarization with syntactic compression. Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing.

Amita Arora is presently working as an Assistant Professor at Department of Computer Engineering in J.C. Bose University of Science and Technology, YMCA Faridabad. She has fifteen years of experience in teaching. She has supervised ten M.Tech Thesis. Her current research interests are Machine Learning, Semantic Web, Information Retrieval, Natural language Processing. She has been teaching subjects like Data Analytics, Machine Learning, Analysis and Design of Algorithm, Computer Graphics. She has published ten articles in International/National Journals and Conferences.