A Novel Approach for Patent Document Summarization

[1] Avinash Thakur, [2]V.Hepsiba Mabel
[1]Research Scholar, [2]Assistan[t] Professor,
[1,2,]Deparment of Computer science and engineering,
[1,2]School of Computing science & Engineering Vellore Institute of technology, Chennai India
Email id: avinash.thakur2017@vitstudent.ac.in, hepsiba.mabel@vit.ac.in

Abstract

Patent papers are valuable intellectual assets intended to safeguard individual interests and creativity. In a recent study field, patent retrieval helps to assist patent analysts in the collection, compilation, and review of patent records. Patent applications are extensive and full of scientific and legal terms, so it requires much time to translate, understand so review a single patent application, particularly for domain experts. It is also important to streamline this process and to assist the analysts in evaluating the relationship between the form and the patents obtained. Typical patent retrieval activities also involve analyzing whether various patent records are similar/different in several respects. This approach analyzes and summarizes a patent document in terms of similarities and differences. Patent documents are lengthy and consumes time for analysis; this approach automates the process and gives a comparative summary of multiple patent reports. The algorithm takes two patent documents as input and provides a comparative overview of the two patents. The final output is a summary of two patents that cover both discriminative and similar aspects of the documents.

Keywords: *Patent, Intellectual resource, summary generation, Patent retrieval.*

1. Introduction

Patent papers are valuable intellectual tools for the defense of the rights of corporations. The purpose of patent retrieval as a new research field is to assist patent analysts in the collection, processing, and review of patent documents. Patent papers are lengthy and full of scientific and legal jargon, so it requires a great deal of time for specialized specialists to interpret, evaluate, so review a specific piece of material. It is also important to streamline this approach and to assist analysts in determining the relation between the application and the patents obtained. Typical procedures for the collection of patents sometimes include analyzing how similar/different two patent applications are in various respects. The growing amount of patent-related records and the ever-increasing demand for exposure to this knowledge by multiple categories of users stimulates researchers to establish strategies and methodologies for reliable and successful patent retrieval. Such customers could be patent practitioners, corporate and scientific testing groups, executives, venture capitalists, developers, patent lawyers, etc. There are various research areas in patent retrieval and mining, such as evaluation of patent retrieval, automated patent classification, patent text retrieval, image-based patent retrieval and classification, multilingual patent retrieval, etc. [12].This approach analyzes and summarizes a patent document in terms of similarities and differences. Patent documents are lengthy and consume time for analysis; this approach automates the process and gives a comparative summary of two patent documents.

The algorithm takes two patent documents as input and provides a comparative overview of the two patents. The final output is a summary of two patents that cover both discriminative and similar aspects of the documents.

2. Background of Research

2.1 Existing system

With the growth of digitalization and the easy availability of electronic resources, a lot of textual data is available all over the world. Documents are being made digitally, with the increase in complex documents containing a lot of information in them and due to exponentially increasing volume of data, understanding them has in-turn increased the popularity of text summarization and classification techniques, and there is need for efficient tools to handle such data, some of the existing methods are elaborating here,.

Changjian Fanget al.[1]This paper recommends a novel co-ranking word-sentence concept called Co-Rank for automated extractive text summarization. Co-Rank incorporates the interaction between the word phrase and the graph-based rating pattern. The Co-Rank method can be seen as mutual support of terms and sentences on the basis that specific words will have prejudice weights

Ferreiraet al. [2] Authors described three separate formulations used to test the strategies. The authors selected the five best results obtained from the different test sets, one of which would achieve a combination of four methods as the best: Word Frequency, TF / IDF, Lexical Similarity, and Sentence Length. The "Text Rank Value" technique was also selected to have reasonable outcomes for two of the three test data sets. The findings given by ROUGE for the quantitative evaluation of the summaries were very similar to those obtained by the qualitative study. The TF / IDF equation is by far the most computationally expensive of all the methods evaluated. Methods Phrase Duration and Sentence Length have the optimal compromise between the execution period and the number of appropriate sentences.

Y. Zhang et al. [3] A text summarization method focused on Convolutionary Neural Networks is proposed in this paper for learning sentence features and jointly performing sentence ranking. It transforms the ranking function into a process of regression. Our proposed structure requires no prior knowledge, and can, therefore, extended with different
Writing styles to various document review tasks.

P. Sethi et al. [4] authors were able to auto-summarize news articles and compare the summaries produced by them to determine what score parameters would contribute to better performance. In the procedure, modified approaches used to exploit the idea that only news reports are dealing. They were just connecting nouns to lexical strings.

A. T. Al-Taani et al. [5] In this article, the authors analyzed and examined ATS extractive-based strategies built for Arabic texts. Such methods shall include 1. Statistics Approaches. 2. Graphic-based methods. 3. Approaches for Artificial Learning. 4. Methods of clustering 5. Meta – Heuristic Analysis Approaches.

P. P. Tardan et al. [6]It is noted that the writers used computational methodology techniques for this study, including sentence classification and sentence location functions. It describes the weak magnitude of the average outcome between the mathematical method and the qualitative research, plus the positive performance of the calculation of subjectivity. One of the unimplemented elements of this research is the elimination of sentences concerning trash. Junk sentences are sentences that have little connection with the context of the text.

J. N. Madhuriet al. [7] Recognizing the required subsections of the text in discussion. In this research, Researchers suggested extractive-based document summarization using a statistically novel approach driven on sentence classification sentences are picked by the abstract. The collected sentences are created as a summary text and translated into audio formats.

M. N. Uddin et al. [8] Study research has performed on document summarization, and the Bangla language summary Extraction has carried out. The critical drawback of the Bangla Interpretation is that it merely removes specific phrases from the text in the discussion that is far more distinct from the human summarizing. Another disadvantage is that often sentences that arrive early in the document have a higher probability of being in the description.

X. Sun et al. [9] Reinforcement ranking of various representation units within the scientific paper on the Semantic Connection Network will substantially boost the paper's extractive description. It not only lays out a description methodology focused on semantic modeling but also verifies the significance of the Semantic Connection Network in the portrayal and interpretation of the material of the document. The proposed approach has stable quality in single document summarization on both scientific papers and short news

A. P. Patil et al. [10]In actuality, they aim to extract a single English article, not exceeding 300 sentences in length, to a fraction of its original size while retaining cohesion and then use a lexical database to abstract the Summary produced. The program uses the external tool WordNet To obscure the review created. WordNet is a lexical database grouping words through semantic relationships. The Natural Language Toolkit (NLTK) for Python is used to access the database through the program. ROUGE used for evaluating the summarization.

H. T. Le et al. [11] This paper proposed an approach to general text summarization, consisting of 2 stages: the elimination of sentences and the combination of sentences. The sentence reduction stage is based on rules of debate to delete unnecessary words at the beginning of a sentence, and syntactic restrictions to complete the end of the reduced sentence. The combination stage of the sentence is based on a word graph to present relationships from input text between terms, clauses, and penalties. Using a word graph generates new sentences that combine information from several sentences. Experimental findings suggest our approach to solving the AS problem is promising.

Avinash et al. [13] This paper has done a comprehensive critical review on patent document mining, analysis and evaluation. It has covered all the dimension of literature available in state of the art.

2.2 Research Gap

we discussed in the earlier section; research is done on text summarization as growing demand for extractive summary generation results in various methods invented and applied for a summary generation. All these methods shows its application mainly for a document related to news, article, and content belongs to the specific regional language. Not a single system exists which can do summarization of multiple patent documents. All order has the feasibility to process a single document at a time.

Our proposed system covers limitations observed in the existing system, i.e., uploading multiple documents for summary generation, Documents maybe belonging to different categories, all document summary generated with simplification.

3. Proposed System Methodology

Figure 1 shows the main phases of generating Summary based on this proposed measurement, such as Preprocessing, Feature Graph Generation, Steiner Tree Generation, and Generating Summary. The document is preprocessing for eliminating the nouns which exist in the original text so that the result from the preprocessing phase can be processed for further steps.

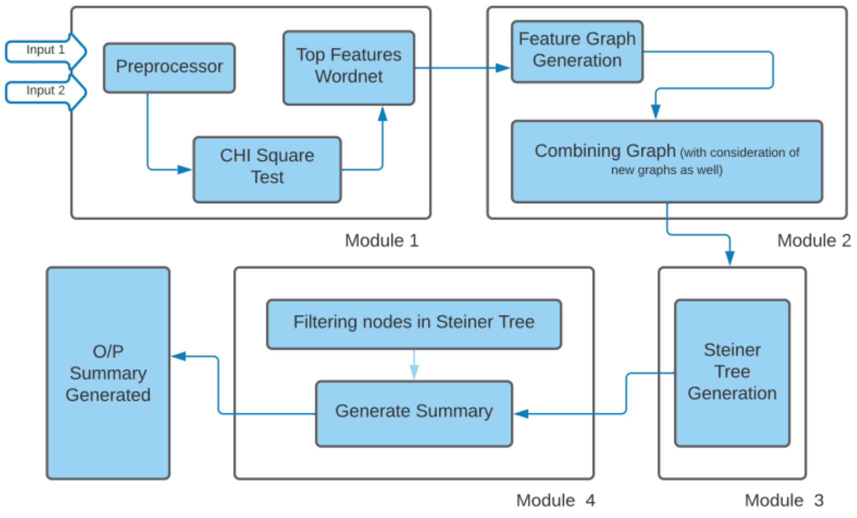


Figure:1 Architecture diagram

3.1 Discriminative feature selection

3.1.1 Pre-processor

Data preprocessing is done on the given input documents to remove objectionable content from them using parsing techniques. Due to the data preprocessing, the data to be processed further gets reduced, which makes the system viable to use. Pre-Processing involves the extraction of features from the document as we are working with the patents' focus is on noun extraction. These nouns extracted from the preprocessing step can be used for further processing. Preprocessing can be done using basic parsing techniques.

We have to obtain feature set F, from the patent document, a patent document consisting of sentences, where S be the set of words in the patent document, and N be the set of all nouns, the feature set is nothing but the intersection of the game of words and terms in the sentence, i.e. $F = S \cap N$. SW be set of all Stop words and words having a length less than

3. In proceeding step we omit to stop words from the feature set, so we get more optimized feature set.
$F2 = F - SW$.f(x) be a function that lemmatizes a noun.
Then, we build function $F3_1$for document 1
$F3_1 = f(x)\ \forall\ x \in F2$.
The above procedure is Applied to Document 2 to obtain,$F3_2$
$F3_2 = f(y\ )\forall\ y \in F2$.
$F3_1$, $F3_2$ Used in next for applying CHI Square Test.

3.1.2 CHI Square Test

The chi-square method is using to identify discriminative word characteristics. We use $\chi 2$ statistics as a function selection method as it has been widely extending to text mining fields.

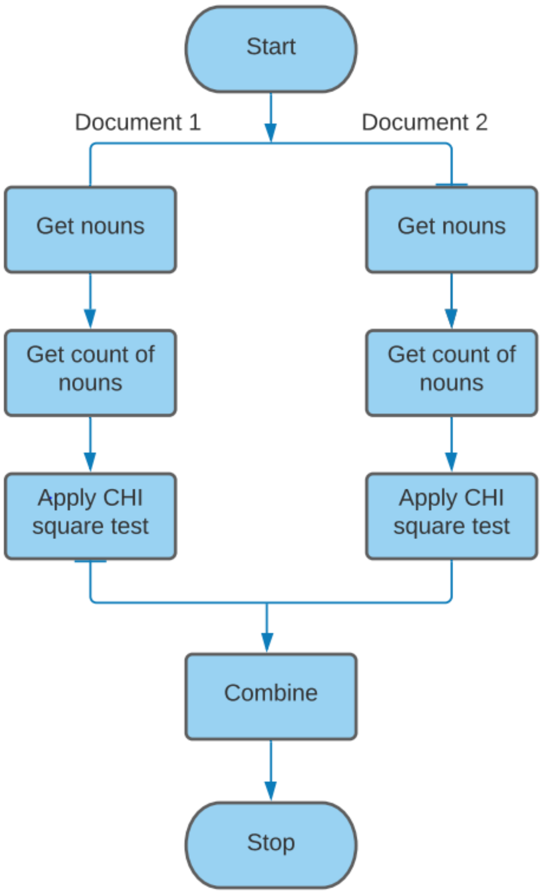


Figure 2: CHI Square test process

Let,
n(x) be a function that maps the count of features/noun to a document.
C1 be the class document 1.
C2 be the class document 2.
The standard formula for CHI Square test as follows:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}.$$

C - s the n(x) ∀ x ∈ C1 and x ∉ F31.
D - is the n(x) ∀ x ∉ C1 and x ∉ F31.

N – No. Of documents.

We obtain,

X1 2 (t,c) ∀ x ∈ F31

Similarly,
- find X 2 (t,c) for F32 and C2.

We obtain,
- X2 2(t,c) ∀ x ∈ F32- X 2 = X1 2 ∪ X2 2

Let
- t(m,threshold) be a threshold function.
- The threshold is a value of 0.01.

Final feature set FS = t(x, threshold) ∀ x ∈ X 2.

FS is forwarding to WordNet Module.

3.1.3 Top features wordnet A wordNet is a lexical dictionary of semantic connections between terms in more than 200 languages, according to Wikipedia. WordNet translates phrases into semantic associations that involve synonyms, hyponyms, and metonyms. The synonyms are organizing into sentences containing brief meanings and instances of usage. Therefore, WordNet can be used as a mixture of dictionary and thesaurus and its extension. While it is available through a web browser to human users, its primary purpose is in predictive text processing and artificial intelligence applications.

1. In the patent documentation, we use Wordnet to evaluate the similarities of nouns.
2. WordNet is the English language focused lexical website. It groups English terms into synonym sets named sentence, includes brief meanings and descriptions of use, and tracks a variety of relationships between such synonym sets or their representatives.
3. It forms a tree-like structure in which nodes are positioning according to their relation. Their disparity in position is relevant when defining the resemblance between two terms, and their difference in degree is often taken into consideration.

Let, L1 be a list of words

For each word says Wx, its weight would be:

$$\text{Weight of Wx} = \frac{\sum_{i=0}^{i=N} \mathit{similarity}(Wi,Wx) \sum_{i=0}^{i=N} \mathit{similarity}(Wi,Wx)}{N \qquad\qquad N}$$

Let
t1 () and t2() be a threshold function
Threshold be a value like 0.1

Outlier feature set O = t1 (weight(x), threshold1) ∀ x ∈ L1.
General feature set G = t2 (childCount(x), threshold2) ∀ x ∈ L1.
Return L1 - O, G
Where L1- O is a set of top features

3.2 Feature graph

3.2.1 Feature Graph Generation

The feature selected from the Chi-square is further classifying into:

1. **Discriminative features**

Suppose there are't' feature variables from the two patent documents, denoted by {xi|xi∈ F},
Where F is the full feature index set, having |F| = t.
There is class variable, C = {c1, c2}.
The problem of feature selection is to select a subset of features, S ⊂ F, to predict the target class variable C accurately.

2. **Outliers**

Outliers are the feature that may not be important to describe the document, and their frequency is comparatively less. Generic functions are the words that are shared by or related to a group of elements. Discriminative features are the words that describe or are necessary to describe the document.

A comparative summary of two patent documents should include both different and universal aspects. It obtains the common elements and links them to the differences; the graph-based approach is used. Here an undirected graph G is constructed to represent two patent documents, where G = (V,E).
G contains a set of vertices (i.e., features) V, where each vertex represents the nouns in patent documents.

Let v1 and v2 be featured present in document A and B
Let |{v1|v1 ∊A}| and |{v2|v2 ∊ A}| denote the frequencies of v1 and v2 in document A, respectively
Let |{(v1,v2)|v1 ∊ A,v2 ∊ A}| represents the number of times that v1 and v2 appear in the same sentence of document A.
The w_A(v1,v2) models the co-occurring probability of v1 and v2 in document A. be The w_B(v1,v2) models the co-occurring probability of v1 and v2 in document B.

Given two patent documents A and B, we connect v1 and v2 having weight w_{Total} (v1,v2) if their averaged linkage score w_{Total} (v1,v2) exceeds a predefined threshold τ1.

$$w_A(v_1, v_2) = 2\frac{|\{(v_1, v_2)|v_1 \in A, v_2 \in A\}|}{|\{v_1|v_1 \in A\}| \times |\{v_2|v_2 \in A\}|},$$

$$\omega_{Total}(\nu_1, \nu_2) = \frac{\omega_a(\nu_1, \nu_2) + \omega_b(\nu_1, \nu_2)}{2}$$

3.3 Summary Generation

3.3.1 Filtering nodes in Steiner Tree

Input for nodes filtration in the Steiner tree is Steiner Tree and List of generic features. Then the output will be Filtered Steiner tree with marked nodes, where Similar nodes that are merging into module 2 are separating in this phase. – The nodes which represent the generalized feature of the document are marked so that they are not considering while generating Summary. It creates room for features that are specific to the patent hence making comparative Summary more precise and less general.

Let N be the set of nodes in the graph generated by Graph Generation sub module.
Let C be the set of nodes that represents standard features among the two documents. Where $\{C \mid C \in N\}$
Let R be the set of nodes using which comparative summaries are to be generating. Where $R = N - C$.
Let E be the set of edges in the resultant graph.
Let D be the set of documents of size 2 in which 1st element is a set of sentences in the first document, and the 2nd element is a set of sentences in the second document.

Let F (Set of documents) be the function that returns the comparative summaries.
Let S (set of sentences) be the function that returns the final game of sentences in the Summary of the respective document.

3.3.2 Generate Summary

The Steiner tree obtained from this provides us with the foundation for producing comparative summaries of two patent articles. Selecting from the original documents the minimum set of sentences by which the elements in the Steiner tree can be fully covered. Each sentence can be interpreted as a subset of the overall feature graph, while the Steiner tree can also be viewed as a sub graph. Thus, the question is to pick the minimal collection of sub graphs covering the Steiner tree. The union of two graphs is formally describing as
$Ga = (Va,Ea)$ and
$Gb = (Vb,Eb)$ as the union of their vertex and edge sets,
i.e., $Ga \cup Gb = (Va \cup Vb, Ea \cup Eb)$.
Each sentence is denoting as $Gi = (Vi,Ei)$.It is the G (V, E) sub graph. The question of creating comparative summaries is then developing as the question of identifying the smallest subset of sub graphs of which the Steiner tree encompasses the union. The list of words generated by sub-module Word net of module 1 is not considering while generating Summary as those words represent the general concept of the patent. These words are at the top of the hierarchy in the word net tree. Hence, they are more general words and less valuable when it comes to comparing patents as in comparative Summary, more focus is giving on the specifics of the patent rather than a general view of the patent.

4. Our Designed Algorithms

```
Module 1:
Read sentences:
Document = The patent to be analysed.
Words_in_document = {}
  for page in document
      for sentence in page
          for word in sentence
              Add word in words_in_document
Return words_in_document
Get Lemmatized Nouns
      # get All nouns from document
      nouns = {}
      words = {}
      for lines in document
          for word in lines
              # remove stop words and words having length less than 2
      if len(word) > 2 and word is not a stop word
              Add word into words list.
          end if
      # tagged_words are words tagged as nouns, Adjective etc.
      tagged_words = tag all words in words list.
      for x in tagged_words:
          if1 x is noun
              if2 x is an english word
                  Lemmatize x and add it to nouns list
              end if2
          end if1
      return nouns
```

```
Calculate chi-square value for nouns
      Chi_nouns ={}
      for all nouns in document
          Find chi-square values for noun
          if chi-square value > CHI_THRESHOLD
              Add noun to Chi_noun
          end if
      Return Chi_noun
```

Chi-Square Module

```
      nouns_doc1 = getLemmatizedNouns(doc1)
      nouns_doc2 = getLemmatizedNouns(doc2)
      chi_sq_doc1 = get key value pair for noun and its chi-square value
      chi_sq_doc2 = get key value pair for noun and its chi-square value
      return chi_sq_doc1, chi_sq_doc2
```

Forward these filtered chi square nouns to the Wordnet module.

WORDNET Module :

```
1. Word Similarity(WORD, wordList):
  wordsyn = all synonyms of WORD(input argument #1) present in wordnet
  similarity = Array for each synonym, value initialised to 0
  threshold = 0.3  # minimum similarity index required

  for each synonym of WORD say SYN:
    for jword in wordList:
      if jword != word:
        temp = Average similarity of SYN with jword synonym set
        if temp >= threshold:
          similarity[SYN] += 1
  SYN = synonym having maximum value in similarity array
  Return SYN, Similarity[SYN]
```

```
2. countChild(SYN, wordList):
  threshold = 0.3  # minimum similarity index required
  child = 0

  for each synonym in wordList say feature:
    if SYN is higher in hierarchy than feature:
      temp = average similarity of SYN with feature.hypernyms set
      if temp >= threshold:
        child += 1
  return child
```

3. Wordnet Module(wordList):

```
threshold = 0.1   # if 10 percent of words are similar to a word then it is considered
filtered = []  # list without outliers

temp = list of nouns that are present in wordnet having at least 1 synonym
noNounInWordnet = list of nouns that are not in wordnet but are there in wordList
wordList = temp
# determine outliers, by calculating how many words are similar to given word
For each word in wordList:
    temp = wordSimilarity(word, wordList)
    if similarity score >= threshold:
        filtered.append(temp)

# determine general or common features
generalFeature = []
general_feature_threshold = 0.1   # if 20 percent features in document are children of any feature then it is general feature
# check how many childrens does a word have
for each SYN in filtered:
    child = countChild(SYN, filtered)
    if child / len(wordList) >= general_feature_threshold:
        generalFeature.append([SYN, child])

gen_and_comp_features = [x[0] for x in filtered]+noNounInWordnet
generalFeature = [x[0] for x in generalFeature]
return gen_and_comp_features,generalFeature
```

Module 2:

Functions in the module:

1. **Linkage score :: Linkage_score(feature1,feature2, document, count1, count2):**
    1. The linkage score is calculated by the below formula

$$w_A(v_1, v_2) = 2\frac{|\{(v_1, v_2)|v_1 \in A, v_2 \in A\}|}{|\{v_1|v_1 \in A\}| \times |\{v_2|v_2 \in A\}|},$$

```
"""
:param feature1: feature 1
:param feature2: feature 2
:param document: list containing lines of document
:param count1: occurrence of feature 1 in the document
:param count2: occurrence of feature 2 in the document
:return: linkage score between feature1 and feature2
"""
combine_occurance_count = occurrence of both the feature in same sentence
for each sentence in document:
  if (feature1 is present in sentence) and (feature2 is present in sentence) then:
      combine_occurance_count = combine_occurance_count + 1
score = (2 * combine_occurance_count) / (count1 + count2)
return score
```

2. **Linkage score :: find_score(features,document):**

```
"""
:param features: list of feature
:param document_lines: array if document lines
:return: list containing occurrence of each feature in the document
"""
features_count = list of number of occurrence of each feature in the document
For each feature in features:
    count = number of occurrence of feature in the document
    features_count[feature] = count
return features_count
```

3. **Graph :: 8feature_union(feature1,feature2)**

```
"""
:param feature_1: list containing features of document 1
:param feature_2: list containing features of document 2
:return: list containing features of document 1 and document 2
"""
features = feature_1 + feature_2
unique_features = initialize the list
for each feature in features:
  if feature not present in unique_features then:
      unique_features.append(feature)
return unique_features
```

4. **Graph :: generate_graph(document1,document2)**

```
features = features_union(features1, features2)
score1 = find_scores(features, document1)
score2 = find_scores(features, document2)
Add each feature as a node into the graph
for i in range(len(features)):
  for j in range(i + 1, len(features)):
    link_score1 = linkage_score(features[i], features[j], document1, score1[features[i]],
                                score1[features[j]])
    link_score2 = linkage_score(features[i], features[j], document2, score2[features[i]],
                                score2[features[j]])
    link_score = (link_score1 + link_score2) / 2
    if link_score > threshold_value:
        add edge between features[i] to features[j] with weight of edge equal to link_score
return graph
```

Module 3:

Methods involved in Module 3 : a) To form the steiner tree, we are using the "stiener_tree()" from the NetworkX module. There are two arguments to this function - (i) terminal nodes(discriminative features) and (ii) feature graph. Thus, the method "Module3()" in the main code is executing these functions.

Module 4:

1 . Summaries :: Create_Summary(graph, common_features, document_1, document_2)

```
'''
    param graph : An steiner tree generated in module 3
    param common_features : List of common features present in both the document
    param document_1 : List of strings , where each string is an each sentence in document_1
    param document_2 : list of strings , where each string is an each sentence in document_2
    return : list of strings , where first string is an summary of document_1 and second string is an summary of document_2 .
    '''
    graph.remove_nodes_from( common_features )
    edges = list( graph.edges )
    summary = array of summaries of two documents
    text = document_1
    ret = final_sentences( text , edges )
    for i in ret:
        summary[0] += i
    text = document_2
    ret = final_sentences( text , edges )
    for i in ret:
        summary[1] += i
    return summary
```

2. Sentences in the summary of an document :: final_sentences(text, edges)

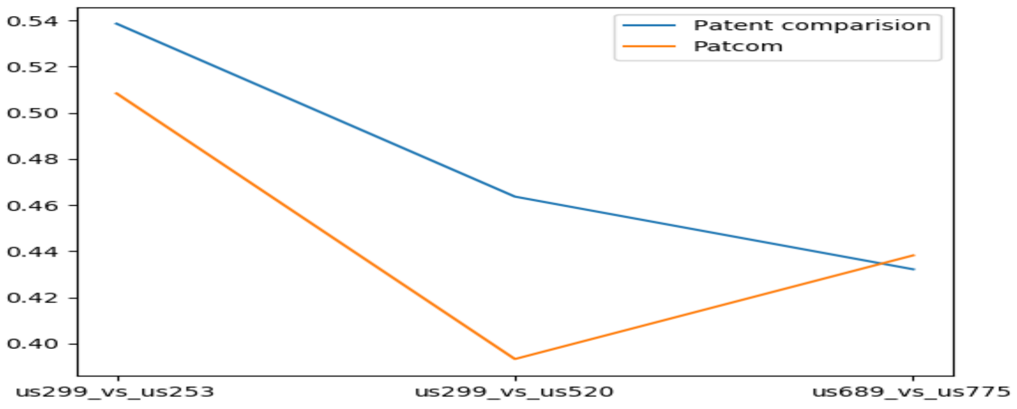
```
'''
    param text : It is the list of strings .
    param edges : It is the list of egdes .
    return : list of strings , where each string is an sentence which is to be present in summary of particular document .
    '''
    output = list of sentences in the final summary of text
    sentence_list = list which maintains a record for all sentences that how many edges are present in the particular sentences
    for i in text:
        temp_list = list which maintains a record that how many edges are present in the sentence i
        for j in edges:
            if( both the nodes of edge j are present in sentence i ):
                temp_list[0] += 1
                temp_list[1].append( j )
        sentence_list.append( list( temp_list ) )
    sentence_list.sort( key = lambda x : x[1] , reverse = True )
    hash_map = hash_map to check whether a particular edge has been covered or not
    for i in sentence_list:
        flag = flag to check whether there contains atleast one edge in sentence i, which was uncovered before
        for j in i[1]:
            if( j not in hash_map ):
                hash_map[ j ] = 1
                flag = 1
        if( flag is set ):
            output.append( i[2] )
    return output
```

5. Results & Discussions

Description of Results got from Proposed system

For comparison purposes, we used the Rouge score. With the help of that, we can say that our approach is closer to a human summary than the existing system. The primary reason contributing to the effectiveness of our method is the use of a word net module to identify a common feature set and we are assigning weight to the nodes present in the graph.

- For the rouge score, we have considered human-generated comparative Summary for six documents.
- Along with that, we had generated a summary of patcom (existing system) as well as of this system.
- The results of this system are more accurate than those of previous systems like patcom.



rouge_score_us299_vs_us253()	hypothesis_patcom_us299
The present disclosure relates to a fabricating method of light guiding plate, and a backlight ' 'module and a display device. The embodiment of the present disclosure provides a fabricating ' 'method of grid points on a light guiding plate. The method includes following steps a layer of ' \ 'photosensitive resin is formed on a mold for the light guiding plate. The layer of photosensitive ' \ 'resin is subjected to a photolithography	Claim 1. A fabricating method of grid points on a light guiding plate, comprising following steps ' \ 'of: S1, forming a layer of photosensitive material on a mold for the light guiding plate; and S2, ' \ 'performing photolithography on the photosensitive material in order to form grid points on the ' 'light guiding plate.

6. Conclusion

We have successfully implemented the Patent Document Comparison System approach for comparative summarization of two patent documents and found excellent results. Some gaps have been suggested based on the actual outcomes and the required outcomes. However, these gaps may depend on the further refinement of the methods used in the code. So, it is the scientific art to re-check the Patent Document Comparison System approach to the suggested gaps. Moreover, the statistical methods of the Patent Document Comparison System are somehow easy to implement if appropriately understood and the results we got based on our implementation are entirely satisfactory.

Future Scope

Here we successfully implemented a basic model to compare two patent documents simultaneously. The system is effective and advancement in an existing order. In the future, we will work on

1) Multiple (More the two) Patent document comparative summarization.
2) The Implemented system is static as there is no learning involved in the process. It will be required.
3) The Summary generated will be more specialized than generalized.

Acknowledgement: -

We would like to thank Longhui Zhang, Lei Li, Chao Shen, and Tao Li
School of Computing and Information Sciences, Florida International University, PatentCom: A Comparative View of Patent Document Retrieval. Our research is inspired from this work and contribution for next steps of research.

Reference

1. Changjian Fang, Dejun Mu, Zhenghong Deng, Zhiang Wu, Word-Sentence Co-Ranking for Automatic Extractive Text Summarization, Expert Systems With Applications (2016),
2. Ferreira, Rafael & Cabral, Luciano & Lins, Rafael & Silva, Gabriel & Freitas, Fred & Cavalcanti, George & Lima, Rinaldo&Simske, Steven &Favaro, Luciano. (2013). Assessing sentence scoring techniques for extractive text summarization. Expert Systems with Applications. 40. 5755–5764. 10.1016/j.eswa.2013.04.023.
3. Y. Zhang, M. J. Er and M. Pratama, "Extractive document summarization based on convolutional neural networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 918-922.
4. P. Sethi, S. Sonawane, S. Khanwalker, and R. B. Keskar, "Automatic text summarization of news articles," 2017 International Conference on Big Data, IoT and Data Science (BID), Pune, 2017, pp. 23-29.
5. A. T. Al-Taani, "Automatic text summarization approaches," 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), Dubai, 2017, pp. 93-94.
6. P. P. Tardan, A. Erwin, K. I. Eng and W. Muliady, "Automatic text summarization based on semantic analysis approach for documents in Indonesian language," 2013 International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, 2013, pp. 47-52.
7. J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 2019, pp. 1-3.
8. M. N. Uddin and S. A. Khan, "A study on text summarization techniques and implement few of them for Bangla language," 2007 10th international conference on computer and information technology, Dhaka, 2007, pp. 1-4.
9. X. Sun and H. Zhuge, "Summarization of Scientific Paper Through Reinforcement Ranking on Semantic Link Network," in IEEE Access, vol. 6, pp. 40611-40625, 2018.
10. A. P. Patil, S. Dalmia, S. A. A. Ansari, T. Aul and V. Bhatnagar, "Automatic text summarizer," 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), New Delhi, 2014, pp. 1530-1534.
11. H. T. Le and T. M. Le, "An approach to abstractive text summarization," 2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR), Hanoi, 2013, pp. 371-376.
12. Khode, Alok&Jambhorkar, Sagar. (2017). A Literature Review on Patent Information Retrieval Techniques. Indian Journal of Science and Technology. 10. 1-13. 10.17485/ijst/2017/v10i37/116435.
13. Avinash & Hepsiba A Critical Literature Review on Patent document mining, analysis and evaluation.https://www.jardcs.org/backissues/archives-special.php?year=2018&issue=13-26