Using Machine Learning for Medical Document Summarization

Kamal Sarkar[1], Mita Nasipuri[2], Suranjan Ghose[3]
[1,2,3] *Computer Science and Engineering Department,*
Jadavpur University, Kolkata-700 032, India
jukamal2001@yahoo.com,
mitanasipuri@yahoo.com,
suranjanghose@yahoo.co.uk

Abstract

Summaries or abstracts available with medical articles are useful for the physicians, medical students and patients to know rapidly what is the article about and decide whether articles are suitable for in-depth study. Since all medical text documents do not come with author written abstracts or summaries, an automatic medical text summarization system can facilitate rapid medical information access on the web. We approach the problem of automatically generating summary from medical article as a supervised learning task. We treat a document as a set of sentences, which the learning algorithm must learn to classify as positive or negative examples of sentences based on summary worthiness of the sentences. We apply the machine learning algorithm called bagging to this learning task, where a C4.5 decision tree has been chosen as the base learner. We also compare the proposed approach to some existing summarization approaches.

Keywords: *text summarization; machine learning; decision trees; bagging; domain-specific features; medical document summarization.*

1. Introduction

These days, people are overwhelmed by the huge amount of information on the Web. The number of pages available on the Internet almost doubles every year. This is also the case for medical information [1] (Afantenos et.al., 2005), which is now available from a variety of sources. Medical Literature such as medical news, research articles, clinical trial reports on the web are an important source to help clinicians in patient treatment. Initially, clinicians go through author-written abstracts or summaries available with medical the articles to decide whether articles are relevant to them for in-depth study. Since all types of medical articles do not come with author written abstracts or summaries, automatic summarization of medical articles will help clinicians or medical students to find the relevant information on the web rapidly. Moreover, monitoring infectious disease outbreaks or other biological threats demand rapid information gatherings and summarization.

Text summarization is the process to produce a condensed representation of the content of its input for human consumption [2] (Mani, 2001). Input to a summarization process can be one or more text documents. When only one document is the input, it is called single document text summarization and when the input is a cluster of related text documents, it is multi-document summarization. We can also categorize the text summarization based on

the type of users the summary is intended for: User focused (query focused) summaries are tailored to the requirements of a particular user or group of users and generic summaries are aimed at a broad readership community [2] (Mani, 2001).

Depending on the nature of text representation in the summary, summary can be categorized as an abstract and an extract. An extract is a summary consisting of a number of salient text units selected from the input. An abstract is a summary, which represents the subject matter of the article with the text units, which are generated by reformulating the salient units selected from the input. An abstract may contain some text units, which are not present in the input text.

Based on the information content of the summary, it can be categorized as informative and indicative summary. The indicative summary presents an indication about an article's purpose and approach to the user for selecting the article for in-depth reading; informative summary covers all salient information in the document at some level of detail, i.e., it will contain information about all the different aspects such as article's purpose, scope, approach, results and conclusions etc. For example, an abstract of a medical research article is more informative than its headline.

In this paper we present a machine learning based model for a sentence extraction-based, single document, informative text summarization in medical domain. In our work, we approach automatic text summarization as a supervised learning task. We treat a document as a set of sentences, which must be classified as positive or negative examples of sentences based on the summary worthiness of sentences where a sentence is represented by a features set, which includes a number of features used in the summarization literature [3] (Radev et. al., 2004) and some other features specific to medical domain. Thus this summarization task can be formulated as the classical machine-learning problem of learning from examples. There are several unusual aspects to this classification problem. For example, the size of positive examples in the training set is relatively small compared to the size of the entire training set because a summary size is roughly less than one-fourth of the size of the source document. It has been generally thought that a summary should be no shorter than 15% and no longer than 35% of the source text[4] (Hovy, 2005).

C4.5 is typically applied to more balanced class distributions. In our experiment, we found that bagging improves the performance of C4.5 [5] (Quinlan, 1996). In general, bagging generates many different decision trees and allows them to vote on the classification of each example. But majority-voting procedure can only determine the class of the samples. For text summarization applications, we need to rank sentences based on its summary worthiness. So, for sentence ranking, we have to follow a new methodology to combine decisions of a bag of trees (discussed in section 4).

We adopted and designed seven features to characterize sentences (taken as basic linguistic units) in the documents. Out of seven features, some features are medical domain specific features. We used domain knowledge (glossary database) to extract domain specific features.

Though the proposed model has been tested on the medical news articles, we hope that it can be easily applied to summarize other types of medical literatures such as medical research articles, medical reports etc. with minor modifications.

The paper is organized as follows. Section 2 provides related work. In section 3 we discuss how to build and use domain knowledge. In section 4, the summarization method has been discussed. We present the evaluation and the experimental results in section 6.

2. Related work

Some previous works on extractive summarization used few or all of the features such as term frequency, positional information and cue phrases to compute sentence scores [6][7][8] (Baxendale, 1958; Edmundson, 1969; Luhn, 1958). MEAD (Radev et. al., 2004), [4] a popular summarization system ranks sentences based on its similarity to the centroid, position in the text, similarity to the first sentence of the article and length. It uses linear combination of features whose values are normalized between 0 and 1 for sentence ranking. Redundancy is removed by a variation of MMR (Maximal Marginal Relevance) algorithm [9] (Carbonell et. al., 1998). The SumBasic algorithm uses term frequency to compute the probability that a term appears in the summary and this probability is used as term weight, which contributes to identifying important sentences. After selecting top sentence in the summary, the next sentence is selected by re-ranking the rest. During each re-ranking operation, weights of the terms found in the previously selected sentences are reduced by multiplying its probability with itself to discourage information redundancy [10] (Nenkova and Vanderwende, 2005) in the summary.

Some machine learning approaches to extractive summarization have already been investigated. In (Kupiec et. Al., 1995) [11] sentence extraction is viewed as a Bayesian classification task. To our knowledge, there are few attempts to use machine learning algorithm for medical document summarization task.

Most of the researchers extend to the medical domain the summarization techniques already used in other domains. One of the projects in medical domain is MiTAP (Day, et al. 2002)[12]. MiTAP (MITRE Text and Audio Processing) monitors infectious disease outbreaks or other biological threats by monitoring multiple information sources. The work presented in (Johnson, et. al., 2002)[13] exploits extractive techniques, which ranks the extracted sentences according to the so-called cluster signature of the document. The abstracts and full texts from the Journal of the American Medical Association were used for their experiments. TRESTLE (Text Retrieval Extraction and Summarization Technologies for Large Enterprises) is a system, which produces single sentence summaries of pharmaceutical newsletters [14] (Gaizauskas, et. al., 2001). TRESTLE generates summaries by filling the templates by the Information Extraction process. The system HelpfulMed[15] (Chen, et. al., 2003) helps professional and advanced users to access medical information on the Internet and in medical related databases. An ontology based summarization approach has been proposed in [16] (Fiszman, et. al., 2004). A query based medical information summarization system that exploits ontology knowledge has been proposed in [17] (Chen and Verma, 2006).

The work presented in [17] (Chen and Verma, 2006) uses ontology to expand query words and assigns scores to sentences based on number of original keywords (query words) and expanded keywords.

Most recently a variation of lexical chaining method (Barzilay and Elhadad, 1999) [18] called bio-chain (Reeve et. al., 2007)[19] is used in biomedical text summarization.

Compared to the above-mentioned approaches, we develop a machine learning based model for medical document summarization that also exploits domain knowledge.

3. Domain knowledge preparation

In the domain specific text summarization, one of important tasks is to discover terms and phrases specific to the domain, because the cue phrases affect the probable summary worthiness of the sentences while summarizing an article. For example, the medical cue phrases such as “World Health Organization”, “Stem cell therapy”, ”Ataxia Telangiectasia” etc. affect the probable summary worthiness of the sentences in a medical article. So, the

sentence containing a cue phrase should get higher score than a sentence which has no cue phrase.

But, it is difficult to identify all domain specific cue phrases by hands. To decide whether a phrase is cue phrase or not, we have used one domain specific vocabulary which is built up by using MeSH (Medical Subject Headings), which is NLM's (U.S. National Library of Medicine) controlled vocabulary thesaurus. All the MeSH key terms have been treated as the cue phrases.

The cue phrases are stored in a knowledge-base. If a sentence contains *n* number of cue phrases, it is assigned a score of *n*.

In our work, all the cue phrases have been considered to be equally important because the discriminations between the cue phrases require more specific domain knowledge.

4. Summarization method

In extractive text summarization approach, the main task is to identify sentences in a source text, which are relevant to the users while simultaneously reducing information redundancy. Sentences are scored based on a set of features. The top-*n* highest scoring sentences in a text are then extracted where *n* is an upper bound, which is determined by the compression rate. Finally the selected sentences are presented to the user in their order of appearance in the original source text [20] (Barzilay et. al., 2001).

The proposed summarization method consists of three primary components: document preprocessing, sentence extraction using a meta-learner called *bagging* and summary generation.

4.1. Document preprocessing

The preprocessing task includes formatting the document, removal of punctuation marks (except dots at the sentence boundaries).

4.2 Using Bagging for sentence extraction

We apply a meta-learner called *bagging* for sentence extraction, where a C 4.5 decision tree has been chosen as the base learner.

Bagging-a name derived from "bootstrap aggregation" – uses multiple version of a training set D, each created by drawing n' < n samples from D with replacement. Each of these bootstrap data sets is used to train a different component learner (base learner) and the decisions of the component learner are combined to arrive at the final decision. Traditionally, the component learners are of the same general form. In our case, all component learners are decision trees. In general, decision tree induction algorithms have low bias but high variance. Bagging multiple trees improves performance by reducing variance and this procedure appears to have relatively little impact on bias.

To train a learning algorithm, we need to establish a set of features and a training corpus of document/extract pairs. In our work, the main goal is to train a bag of decision trees with the set of features and the multiple versions of a training set D and combine the decisions of those trained decision trees to classify a sentence as summary worthy (positive) or not (negative example). After completion of training, the trained learning algorithm is tested on unseen instances, which is not part of training corpus.

4.2.1. Building corpus: The training and test corpus are built by downloading medical news articles from a number of online sources. Then the summaries are manually created for those articles. A total of 75 medical news documents are downloaded from the online sources such as medical news today, MedLinePlus, NIH news etc. The downloaded articles are related to the topics such as new findings about heart diseases, heart surgery, stroke, diabetes, vitamin deficiency related problems, stem cell treatment, discovery of new genes for genetic diseases, gene therapy, reports on Drugs and their side effects, outcomes of Anti HIV Gene Therapy Trials and other clinical trials, Control of antibiotic resistant bacteria, outbreaks of infectious diseases, government's health policies, survey on mental and psychological diseases.

For each article two manual summaries (model summaries) are created by human abstractors. Since summaries are very subjective and user sensitive, for each article we decide to have two different model summaries created by two different human abstractors. Human abstractors are faculty members and postgraduate students of our institute. Though human abstractors have been instructed to create abstracts for each article by copying material from the original text, they freely used their own sentence construction while summarizing articles. But, to apply machine learning algorithm, we need to have extracts instead of abstracts because for extracting features for summary sentences we need to match the manual summary sentences to the sentences in the original document. So, for each manually created abstract, we create an extract by selecting the sentences from the original document that best match the sentences in the abstract. The average size of an extract is 25% of the source document.

We choose relatively long medical news documents in our study because the summarization becomes more useful for long news documents. The reason behind choosing medical news articles for our experiment is that the medical news articles do not come with any abstract or summaries. Though some features of the medical news articles and general newspaper articles may overlap, we found that the medical news articles have some features such as medical terms, medical cue phrases which may be absent in the general newspaper articles.

It is a common practice to evaluate a machine learning algorithm using k-fold cross validation, where the entire data set is divided into *k* subsets, and each time, one of the *k* subsets is used as the test set and the other *k-1* subsets are put together to form a training set. Thus, every data point gets to be in a test set exactly once, and gets to be in a training set *k-1* times.

For the evaluation of the proposed learning based system, the entire data set are divided into 3 folds where each fold consists of one training set of 50 documents and a test set of 25 documents.

4.2.2. Features: To characterize the sentences in the medical documents we have designed and adopted a number of features such as: centroid overlap, sentence position, first-sentence overlap, sentence length, domain specific cue phrases, acronyms. For normalization of each feature value, we divide the value by the maximum of the scores obtained by the sentences due to the feature. The following we discuss the features in detail.

Centroid: The centroid value for a sentence is computed as the sum of the centroid values of all words in the sentence. A centroid is a pseudo-document which consists of words which have TF*IDF scores above a predefined threshold. It is hypothesized that sentences containing words from the centroid are more indicative of the topic of the document [3] (Radev et. al., 2004). Here, TF is term frequency, which is local to a document. IDF is inverse document frequency computed using the formula: log (N/df) where df (document frequency)

= number of documents containing a term and N=number of documents in a corpus. Since IDF is computed over a corpus of text documents it can be treated as a global parameter indicating rarity or commonness of a term in a corpus.

Similarity to first sentence: The similarity of a sentence S to the first sentence of the document is computed by the inner product of the sentence vectors for the current sentence S and the first sentence of the document. Each sentence is represented by an *n*- dimensional sentence vector, whereby the value at position i of a sentence vector indicates the number of occurrences of that word in the sentence [3] (Radev et. al., 2004) and *n* indicates the length of the sentence vector which is computed by counting the number of distinct words in our corpus.

Sentence position: The positional value is computed as follows:

$$P_i = \frac{1}{\sqrt{i}},$$

where i is the position of the sentence in the document. When i =1, that is, the first sentence of the document gets maximum positional value and as the sentence number increases, the positional value decreases. This heuristic is derived from the fact that author of a news article may summarize the main concepts within the first few paragraphs before further elaboration.

Medical domain specific cue phrases: A sentence gets score of *n* if it contains *n* number of the domain specific cue phrases (discussed in section 3) from our knowledge base.

Position of cue phrases: If the sentence contains a cue phrase at the beginning or at the end of the sentence it gets an additional score of 1. But, if the sentence does not contain any cue phrase at all, it gets a score of 0.

Acronyms: A sentence gets a score based on the number of acronyms it contains. In medical articles, authors frequently use acronyms for important complex medical terms, perhaps it help them memorize the things better. So, we consider acronym as an important feature for medical document summarization task. We have used a shallower approach for the detection of an acronym and the following is the rule used for this purpose:

If some letters (at least two letters) of a term are capital, we treat the term as an acronym (gene names, medical instruments etc.) For example, fMRI is term, which is found in our test document, is recognized as an acronym by this rule.

Though the above-mentioned two rules are not fool proof, it works fairly well in improving summarization performance.

For these features, we calculate the score of a sentence by the number of acronyms it contains.

Sentence length: Sometimes a long sentence may get a higher score due to the fact that it contains more number of words. Similarly, a short sentence may get relatively low score due to the fact that it contains less number of words. So, we also consider the length of a sentence as a feature.

4.2.3. Sentence extraction: Training a learning algorithm for summary sentence extraction requires document sentences to be represented as feature vectors. For this purpose, we write a computer program for automatically extracting values for the features characterizing the sentences in the documents. For each sentence in the given document we extract the feature values from the source document using the measures discussed in the sub-section 4.2.2. If the sentence under consideration is found in both the extracts, extract1 and extract2, which are created from the human abstracts (discussed in 4.2.1), we label the sentence as "summary worthy" sentence. If it is found in one of these extracts, we label the sentence as "moderately summary worthy" and if it is not found in any one of these extracts we label the sentence as "summary unworthy". Thus each sentence vector looks like {<a_1 a_2 a_3 ….. a_n>, <label>} which becomes an instance (example) for a base learner C4.5 decision tree, where a_1, a_2 . . .a_n, indicate feature values for a sentence. All the documents in our corpus are converted to a set of instances of the above form.

We divide the entire data set into 3 folds where each fold consists of one training set corresponding to a set of training documents and a test set corresponding to a set of test documents. After preparation of a training set, the multiple decision trees are trained with the different versions of the training set and the decisions of those trained decision trees are combined to classify a sentence as one of three categories: "summary worthy", "moderately summary worthy" and "summary unworthy". For each fold, a model is built from a training set using the bagging technique and then the learned model is applied to the test set.

For our experiments, we have used Weka (www.cs.waikato.ac.nz/ml/weka) machine learning tools. Initially, for each fold, we submit the training data set and the test data set to Weka. Then we select the option "bagging" under meta-classifier folder in Weka. Then the various configurable parameters such as bagsizePercent (percentage of the training set size), number-of-base learners etc are set. We chose J48 (Weka's implementation of Quinlan's C4.5 [21] (Quinlan, 1993) decision tree) as a base learner and set the number-of-base learners to the default value which is 10. Then we vary the bagsizePercent from 1% to 100% and finally, the parameter, bagsizePercent is set to the value for which we get the best performance. Though all the attribute values of the instances in the training and test sets are continuous, we did not apply any separate discretization algorithm because C4.5 is capable of handling continuous attribute values.

We configure WEKA in such a way that for each test instance, we can get the predicted class and the probability estimate for the class. The trained learning algorithm will assign one of three labels: "Summary Worthy" (SW), "Moderately Summary Worthy" (MSW) or "Summary Unworthy" (SU) to a test instance corresponding to a sentence in a test document. It is possible to save the output in a separate file. We save the output produced by WEKA in a file and then collect the classification output for the sentences belonging to each test document. Then we design a sentence- ranking algorithm based on the classification output and the probability estimates for the classes. The algorithm for sentence ranking is given below.

Sentence ranking algorithm:

Input:
An output file produced by WEKA, which contains the sentences of a test document with their classifications and the probability estimates of the classes to which the sentences belong.

Output: A file containing the ranked sentences

Begin

Read the input file.

- Select those sentences, which have been classified as "Summary Worthy" (SW) and reorder the selected sentences in decreasing order of the probability estimates of their classes. Save the selected sentences in the output file and delete them from the input file.

- Select those sentences, which have been classified as "Moderately Summary Worthy" (SW) and reorder the selected sentences in decreasing order of the probability estimates of their classes. Save the selected sentences in the output file and delete them from the input file.

- For the rest of the sentences, which are classified as "Summary Unworthy", we order the sentences in increasing order of the probability estimates of the class labels. In effect, the sentence for which the probability estimate is minimum (that is, the sentence is minimum "Summary Unworthy") comes at the top. Append the ordered sentences to the output file.

- Close the output file

End of the algorithm

The sentence-ranking algorithm has three major steps. At the first step, the sentences, which are classified as "summary worthy", we undoubtedly select those sentences in the summary.

If the number of sentences selected at step 1 is less than the desired number of sentences, we consider those sentences which are not selected in the summary at the first step. At the second step, the sentences, which are classified as "Moderately Summary Worthy" are considered.

If the number of sentences selected at step 1 and step2 are less than desired number of sentences, we consider the sentences, which have been classified as "Summary Unworthy" and order those sentences in increasing order of the probability estimates of the class labels, that is, the sentences are ordered from minimum summary unworthiness (maximum summary worthiness) to maximum summary unworthiness (minimum summary worthiness). They are selected in this order one by one in the summary until the desired summary length is reached.

4.3 Summary generation

After ranking the sentences, *n* top ranked sentences are selected to generate the final summary. Value of *n* depends on the compression rate. But, the summary produced in this way may contain some redundant information, that is, some sentences in the summary may entail partially or fully the concept embodied in other sentences. This restricts the summary to be more informative when the summary length is a restriction. Moreover, a user who is used to just looking at first few sentences representing the same concept will prefer to see something different information, though marginally less relevant. To keep the sentences in the summary sufficiently dissimilar from each other, the diversity based re-ranking method called Maximal Marginal Relevance (MMR) is a well-known measure [9] (Carbonell and Goldstein, 1998). This approach uses a ranking parameter that allows the user to slide between relevance to the query and diversity from the sentences seen so far. The MMR algorithm is most suitable to apply in query-focused summarization where the summary will be focused toward the user's query. But in our generic summarization environment where only one generic summary will be produced for a text document, we have used a variant of the MMR algorithm to remove redundancy in the summary. This algorithm works as follows:

- Rank the sentences using the ranking algorithm discussed in the sub-section 4.2.3

- Select the top ranked sentence first.
- Select the next sentence from the ordered list and include into the summary if this sentence is sufficiently dissimilar to all of the previously selected sentences.
- Continue selecting sentences one by one until the predefined summary length is reached.

The similarity between two sentences is measured using cosine similarity metric. If the cosine similarity between two sentences is greater (less) than a threshold, we say that the sentences are similar (dissimilar). The cosine similarity between two sentences is measured by the following formula as stated in [22] (Erkan and Radev, 2004)

idf-modified-cosine(x,y)=

$$\frac{\sum_{\omega \in x,y} tf_{\omega,x} * tf_{\omega,y} * (idf_{\omega})^2}{\sqrt{\sum_{x_i \in x} (tf_{x_i,x} * idf_{x_i})^2} \times \sqrt{\sum_{y_i \in y} (tf_{y_i,y} * idf_{y_i})^2}}$$

where $tf_{\omega,S}$ is the number of occurrences of the word ω in the sentence S, idf_{ω} is the inverse document frequency of the word ω and x_i is the i-th word in the sentence x and y_i is the i-th word in the sentence y. idf value of a word is computed on a corpus of documents using the formula: log(N/df) where N is the number of documents in the corpus and df is the number of documents in the corpus that contain the word.

Finally, the sentences selected in the above-mentioned manner are reordered using text order (sorted in the order in which they appear in the input texts)[20] (Barzilay et. al., 2001) to increase the readability of the summary.

5. Comparison to an existing summarizer

We have compared the performance of the proposed summarization system to one popular summarization system called MEAD [3] (Radev et. al., 2004).

MEAD is a single and multi-document summarizer that ranks sentences using the linear combination of normalized values of the features such as position of a sentence, similarity of a sentence to the centroid etc.. Finally, the information redundancy is reduced using a variant of Maximal Marginal Relevance (MMR). For comparison of MEAD to our work, we have implemented MEAD with two important features: position and centroid. We have discussed later in this paper the performance of our implementation of MEAD.

6. Evaluation, experimental results and discussion

To evaluate our summarization system, 75 medical news articles have been downloaded from a number of online medical news sources such as medical news today[1], MedLinePlus[2], NIH news[3] etc. From the downloaded articles, the images and other links are

[1] http://www.medicalnewstoday.com/
[2] http://www.nlm.nih.gov/medlineplus/newsbydate.html

manually removed and only the news content is considered. Average size of news content of the articles is 630 words (approx). Details on corpus development have been discussed in section 4.2.1.

Traditionally, for each system generated summary, more than one model summaries are used for evaluation because the human abstractors may disagree with each other in producing the summary of the document. But, manual summary creation is a tedious task. In our experiments, we have used two reference summaries for evaluating a system generated summary.

6.1. Evaluation

For system evaluation, we have used two measures: the first one is the traditional precision and recall and the second one is the ROUGE (Recall-Oriented Understudy for Gisting Evaluation, version 1.5.5) [23] (Lin and Hovy, 2003).

Precision and recall: Precision and recall are the well known evaluation measures in the information retrieval settings. Since our system extracts sentences from the source document to form a summary, we define precision and recall as follows:

$$\text{Precision} = \frac{N}{K}$$

Where, N = number of extracted sentences matched with a reference summary and K = number of sentences extracted by the system.

$$\text{Recall} = \frac{N}{M}$$

Where, N = number of extracted sentences matched with a reference summary and M = number of sentences in the reference summary.

Since we have used two reference summaries for evaluating a system generated summary, we have compared the system summary to each of the reference summaries and computed the precision and recall. Thus for each system generated summary, we get one pair of precision and recall values for the first reference summary and another pair of precision and recall values for the second reference summary. We define the average precision_{R1R2} and the average recall_{R1R2} as follows:

[3] http://www.nih.gov/news/

Average Precision$_{R1R2} = \frac{P_{R1}+P_{R2}}{2}$ Average Recall$_{R1R2} = \frac{R_{R1}+R_{R2}}{2}$

Where P_{R1}= average precision of a system, where the precision is computed by comparing the system generated summary and the first reference summary for a document, P_{R2} = average precision of a system, where the precision is computed by comparing the system generated summary and the second reference summary for a document, R_{R1}= average recall of a system, where the recall is computed by comparing the system generated summary and the first reference summary for a document, R_{R2} = average recall of a system, where the recall is computed by comparing the system generated summary and the second reference summary for a document.

For evaluating the system using precision and recall, we set the compression ratio to 15% and 20%. Compression ratio r% means r percent of the total sentences in the source documents are extracted as a summary.

ROUGE: For system evaluation we also used an automatic summary evaluation tool, ROUGE [24] (Lin, 2005) developed by the Information Science Institute at the University of Southern California. ROUGE is an automated tool, which compares a generated summary from an automated system with one or more ideal summaries. The ideal summaries are called models. ROUGE is based on n-gram overlap between the system-produced and reference summaries. ROUGE was used in the 2004 and 2005 Document Understanding Conferences (DUC) (National Institute of Standards and Technology (NIST), 2005) as the evaluation tool.

ROUGE was initially used in DUC 2004 for automatic evaluation of the summaries. It was the older version of the ROUGE package which was based on n-gram overlap between the system-produced and reference summaries. The older version of ROUGE, such as ROUGE 1.4.2, reports separate scores for 1, 2, 3, and 4-gram matching between the model summaries and the generated summary (here n-gram is a sequence of *n* consecutive words in a summary sentence). This version of ROUGE evaluates summaries using a recall-based measure that requires that the length of the system generated summary and the reference summary should be the same. The score ROUGE-N is computed using the following formula:

$$ROUGE-N = \frac{\sum_{S \in (\text{Reference Summaries})} \sum_{gram_n \in S} \text{Count}_{\text{match}}(gram_n)}{\sum_{S \in (\text{Reference Summaries})} \sum_{gram_n \in S} \text{Count}(gram_n)}$$

Where ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries, *n* stands for the length of the n-gram, $gram_n$ and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries [23] (Lin and Hovy, 2003). Jonas Sjo¨bergh (Sjöbergh, 2007)[25] has shown that older versions of ROUGE, such as ROUGE 1.4.2, used in the DUC 2004 evaluations produces only the recall score is not always useful to distinguish between a good summary and a bad summary because one can easily generate texts that score highly using the ROUGE-measurements without being good summaries. They have also shown that recent versions of ROUGEeval such as ROUGE 1.5.5, which calculates both precision and recall values, is a good way to detect whether the system generated summaries are good or bad and it seems to be much harder to achieve high precision and recall than just a high recall.

ROUGE 1.5.5 calculates both the precision and recall. For a single summary, the recall is the percentage of n-grams in the models that also occur in the peer (system generated summary). The precision is the percentage of n-grams in the peers that also occur in the models. For a system, the average recall and precision are the averages over all the summaries in the test set when multiple reference summaries are available for comparing each system generated summary.

The following discusses some important ROUGE scores:

- Rouge-N: counts contiguous n-grams, where n ranges from 1 to 4.
- Rouge-L: computes longest common subsequence. Given two sequences X and Y, a longest common subsequence of X and Y is a common subsequence with maximum length.
- Rouge-W: is like Rouge-L but uses a weighting factor for longest number of consecutive matching words. ROUGE-W favors strings with consecutive matches.
- Rouge-S: uses skip bigrams: pairs of words in sentence order, ignoring gaps. It allows arbitrary gaps in matches as LCS (Longest Common Subsequence) but count all in sequence pairs; while LCS only counts the longest subsequences. For example, the sentence, “police killed the gunman” has the following skip bigrams: (“police killed”, “police the”, “police gunman”, “killed the”, “killed gunman”, “the gunman”). One can limit the maximum skip distance, between two in-order words that is allowed to form a skip-bigram. If the maximum skip distance is set to 4 then only word pairs of at most 4 words apart can form skip-bigrams.
- Rouge-SU: includes unigrams in the skip bigrams.

ROUGE-1, ROUGE-2 and ROUGE-SU metrics have been widely used in the NLP (Natural Language Processing) community for automatic summary evaluation. For these reasons, these three summary evaluation metrics have been used to evaluate the summarization methods presented in this paper.

Summarization system evaluation using ROUGE requires that both the system generated summary and the reference summary should be of equal length and the summary length should be specified in terms of words or bytes. This setting differs from the setting used for evaluating summaries using precision and recall. So, we have defined two tasks for evaluating the systems using ROUGE: 100-word summary generation task and 150-word summary generation task. For these tasks, to evaluate a system generated summary, we need to have one or more reference summaries. So, a 100-word reference summary and a 150-word reference summary have been created by selecting the first 100 and the first 150 words from each extract (discussed in subsection 4.2.1) used for evaluating the system using precision and recall.

6.2. Results

To measure the overall performance of the proposed learning based summarization system, our experimental dataset consisting of 75 documents are divided into 3 folds for 3-fold cross validation where each fold contains two independent sets: a training set of 50 documents and a test set of 25 documents. For each fold, a separate model is built from 50 documents and the learned model is applied to the test set of 25 documents. Thus, for each task, if we consider all three folds, we can get a summary for each of 75 documents in our corpus. For other systems such as MEAD (discussed in section 5) and the lead baseline system (which simply takes the first n words or n sentences of the document) to which the

proposed system is compared, we run the systems on the entire 75 documents in our corpus to collect 75 summaries for each task. Then the evaluation metrics discussed in the section 6.1 are used to compute the final results.

Table 1 shows the results in terms of precision and recall for the compression ratio set to 15 % and Table 2 shows the results for the compression ratio set to 20%.

Table 1 Precision and recall for 15% summary generation task on the test data set

	Average Precision_{R1R2}	Average Recall_{R1R2}
Proposed learning based summarization approach	0.63	0.29
MEAD	0.54	0.24
Baseline-lead	0.58	0.25

By analyzing table 1, we find that for 15% summary generation task, the learning based system performs better than the lead baseline and MEAD, but MEAD performs worse than the lead baseline.

Table 2 Precision and recall for 20% summary generation task on the test data set

	Average Precision_{R1R2}	Average Recall_{R1R2}
Proposed learning based summarization approach	0.59	0.35
MEAD	0.54	0.31
Baseline-lead	0.47	0.27

Table 2 shows that for 20% summary generation task, MEAD performs better than the lead baseline whereas the learning based system performs better than MEAD and the lead baseline.

To compare the performance of the systems using ROUGE, for each medical document we assume first n words as the baseline summary (lead baseline) where n = 100 for 100-word summary generation task and n=150 for 150-word summary generation task. When evaluation is done, we set ROUGE configuration parameter as: ROUGE-1.5.5.pl -a -n 2 -x -m -2 4 –u, this means it will compute ROUGE-1, ROUGE-2 (option: -n 2) and ROUGE-SU4 (option: 4 –u) recall scores by comparing the reference summaries and the system generated summaries. The system generated summaries and the reference summaries are stemmed (option: - m) for comparisons. The ROUGE scores shown in the tables are basically the average F-measure values.

Table 3 shows the results in terms of ROUGE scores after testing the systems on all the medical articles for 100- word summary generation task.

Table 3 ROUGE scores for 100-word summaries on test data set with 95% Confidence interval

	ROUGE-1 score	ROUGE-2 score	ROUGE-SU score
Proposed learning based summarization approach	0.5751	0.4110	0.4150
MEAD	0.5262	0.3221	0.3376
Baseline-lead	0.5720	0.4054	0.4113

By analyzing the results shown in table 3 for 100-word summary generation task, we find that the learning based approach performs slightly better than the lead baseline, but MEAD performs worse than the lead baseline.

Table 4 shows results for 150-word summary generation task where we assume that the summary length restriction is 150 words. Here we find an improvement over the lead-baseline. MEAD performs better than the lead baseline whereas the proposed learning based approach outperforms both MEAD and the lead baseline.

Table 4 ROUGE scores for 150-word summaries on test data set with 95% confidence interval

	ROUGE-1 score	ROUGE-2 score	ROUGE-SU score
Proposed learning based summarization approach	0.5830	0.3770	0.3916
MEAD	0.5712	0.3623	0.3810
Baseline-lead	0.5507	0.3532	0.3692

From the results presented in table 1-4 we can conclude that when the target summary length is set to a low value (that is, the system generated summary is short), it is very difficult to outperform the lead baseline, but when the target summary length is set to relatively high value, other systems perform better than the lead baseline.

To know the reasons, we analyze the human summaries and find that almost all human summarizers have selected materials in the summaries from the first two paragraphs of the articles in concise manner and then they have selected other information if summary length permits. For the 100-word and 15% summary generation tasks, the chance of selecting more diverse information is less than that of the150- word and 20% summary generation tasks.

6.3. Discussion

We have presented an approach for text summarization in medical domain. We approach the problem of automatically generating summary from a medical article as a supervised learning task. Our approach was to apply the machine learning algorithm called bagging that uses a number of C4.5 decision trees as the base learners for detecting summary sentences from a text. Our experiments show that bagging is helpful for this task.

For system evaluation, we have used two types of evaluation measures: one is precision and recall and another is ROUGE, because we believe that precision and recall evaluate the system performance, but not the quality of the summaries produced by the systems whereas ROUGE evaluates the system performance by judging the quality of the summaries produced by the systems.

6.4. Future work and limitations

The performance of the system can be further improved by exploring more number of domain specific features and improving the methods for medical entity detection. Though the learning model proposed in this paper has been tested on the medical news articles, it can be easily applied to summarize other medical literatures such as medical research articles, reports etc. with minor modifications.

Although our approach uses the bagging procedure for identifying summary sentences from a test, the framework for applying machine learning algorithm to text summarization task, presented in this paper can be useful for the approaches which will use other machine learning algorithms such as random forest, support vector machines etc.

We did not incorporate stemmer in our summarization system. Apparently, it seems that stemming of the input documents and other information from knowledge base may improve the summarization performance, but one important question is: which stemmer is suitable for medical domain- porter's algorithm or some other special purpose stemmer useful for medical domain!

We also believe that the results could be improved by adding some kind of synonym detection to the sentence extraction algorithm.

7. Conclusion

This paper discusses a machine learning based model for text summarization in medical domain. Most of previous works on text summarization in medical domain extends the various features used in the other domains to the medical domain. In our work, we have combined several medical domain specific features with some other features used in the state-of-art summarization approaches. A machine-learning tool has been used for effective feature combination. The proposed approach performs better than the systems it is compared to.

References

[1.] S. D., Afantenos, V. Karkaletsis, P. Stamatopoulos "Summarization from Medical Documents: A Survey", Journal of Artificial Intelligence in Medicine, vol. 33, 2005, Pages 157-177.

[2.] Mani I., "Automatic summarization", Book, Volume 3 of Natural language processing, Amsterdam/Philadelphia: John Benjamins Publishing Company, 2001.

[3.] D. R. Radev, H. Jing, M. Sty, D. Tam, "Centroid-based summarization of multiple documents", Journal of Information Processing and Management, Elsevier, Volume 40, Issue 6, 2004, Pages 919-938.

[4.] E. H. Hovy, "Automated text summarization", In R. Mitkov (Ed.), The oxford handbook of computational linguistics, Oxford: Oxford University Press, 2005, pp. 583–598.

[5.] J.R. Quinlan, "Bagging, boosting, and C4.5", In Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96), AAAI Press, 1996, pp. 725–730.

[6.] P. B Baxendale, "Man-made index for technical literature—An experiment". IBM Journal of Research and Development, 2(4), 1958, pages 354–361.

[7.] H. P. Edmundson, "New methods in automatic extracting", Journal of the Association for Computing Machinery, 1969, 16(2):264–285.

[8.] H. P. Luhn, "The automatic creation of literature abstracts", IBM Journal of Research Development, 1958, 2(2):159–165.

[9.] G. J. Carbonell, J. Goldstein, "The use of MMR, diversity-based re-ranking for reordering documents and producing summaries", In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 1998, pages 335–336.

[10.] A. Nenkova, L. Vanderwende, "The impact of frequency on summarization", No. MSR-TR-2005-101. Redmond, Washington: Microsoft Research, 2005.

[11.] J. Kupiec, J. O. Pedersen and F. Chen, "A trainable document summarizer", In Research and Development in Information Retrieval, 1995, pp 68–73.

[12.] L. D. Day, L. Hirschman, R. Kozierok, S. Mardis, T. McEntee, et al., "Real users, real data, real problems: the MiTAP system for monitoring bio Events", In the Proceedings of the Conference on Unified Science & Technology for Reducing Biological Threats & Countering Terrorism (BTR 2002), 2002, PP 167—77.
[13.] D. B. Johnson, Q. Zou, J. D. Dionisio, V. Z. Liu, W. W. Chu, "Modeling medical content for automated summarization", Ann NYAcad Sci, 2002, PP 247—58.
[14.] R. Gaizauskas, P. Herring, M. Oakes, M. Beaulieu, P. Willett, H. Fowkes, et al., "Intelligent access to text: integrating information extraction technology into text browsers", In Proceedings of the Human Language Technology Conference (HLT 2001), 2001, PP. 189—93.
[15.] H. Chen, A. Lally, B. Zhu, M. chau, "HelpfulMed: Intelligent Searching for Medical Information over the Internet", Journal of American Society for Information Science and Technology (JASIST), volume 54, Issue 7, 2003, pages 683-694.
[16.] M. Fiszman, T. Rindflesch, H. Kilicoglu, "Abstraction Summarization for managing the Biomedical Research Literature", In the Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics, 2004.
[17.] P. Chen, R. Verma, "A query-based medical Information summarization system Using Ontology Knowledge", In the proceedings of the 19th IEEE Symposium on Computer based Medical Systems, 2006.
[18.] Barzilay R., M. Elhadad, "Using Lexical Chains for Text Summarization", In Mani, I., & Maybury, M. T. (Eds.), Advances in Automatic Text Summarization, The MIT Press, 1999, PP 111–121.
[19.] L. H. Reeve, H. Han, A. D. Brooks, "The use of domain-specific concepts in biomedical text summarization", Journal of Information Processing and Management, Elsevier, Vol. 43, 2007, Pages 1765–1776.
[20.] R. Barzilay N. Elhadad, K. McKeown, "Sentence ordering in multi-document summarization". In Proceedings of the Human Language Technology Conference, 2001.
[21.] Quinlan J.R., "C4.5: Programs for Machine Learning", Morgan Kaufmann, California, 1993.
[22.] G. Erkan, D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization", Journal of Artificial Intelligence Research (JAIR), Volume 22, 2004, pages 457-479.
[23.] C.-Y. Lin, E. Hovy, "Automatic evaluation of summaries using n-gram co-occurrence", In Proceedings of Language Technology Conference (HLT-NAACL 2003), Edmoton, Canada, 2003.
[24.] C. Lin, "Recall-oriented understudy for gisting evaluation (ROUGE)", Retrieved in 2005, from http://www.isi.edu/~cyl/ROUGE/.
[25.] J. Sjöbergh, "Older versions of the ROUGEeval summarization evaluation system were easier to fool", Information Processing & Management, 43(6), 2007, Pages 1500-1505.

Authors

Kamal Sarkar received his B.E degree in Computer Science and Engineering from the Faculty of Engineering, Jadavpur University in 1996. He received the M.E degree in Computer Science and Engineering from the same University in 1999. In 2001, he joined as a lecturer in the Department of Computer Science & Engineering, Jadavpur University, Kolkata, where he is currently an associate professor. His research interest includes text summarization, natural language processing, machine learning, web mining, knowledge discovery from text data.

Mita Nasipuri received her B.E.Tel.E., M.E.Tel.E., and Ph.D. (Engg.) degrees from Jadavpur University, in 1979, 1981 and 1990, respectively. Prof. Nasipuri has been a faculty member of Jadavpur University since 1987. Her current research interest includes image processing, pattern recognition, and multimedia systems. She is a senior member of the IEEE, U.S.A., Fellow of I.E (India) and W.B.A.S.T, Kolkata, India.

Suranjan Ghose received his B.E. degree in Electronics & Telecommunication Engineering and M.E.and Ph.D. degrees in Computer Science & Engineering from Jadavpur University, Calcutta, India in 1978, 1981 and 1988 respectively. From 1982 to 1987 he was a faculty member at the Indian Statistical Institute, Calcutta. In 1988 he joined the Department of Computer Science & Engineering, Jadavpur University, Calcutta,
where he is currently a professor. His research interests include parallel algorithms & architectures, graphics algorithms, computer networks and fault-tolerant architectures.