An Effective Approach for AESOP and Guided Summarization Task

Niraj Kumar, Kannan Srinathan and Vasudeva Varma
IIIT-Hyderabad, Hyderabad, Andhra Pradesh, India - 500032
niraj_kumar@research.iiit.ac.in
srinathan@iiit.ac.in, vv@iiit.ac.in

Abstract. In this paper we, present (1) an unsupervised system for AESOP task and (2) a generic multi-document summarization system for guided summarization task. We propose the use of: (1) the role and importance of words and sentences in document and, (2) number and coverage strength of topics in document for both AESOP and Guided summarization task. We also use some other statistical features, simple heuristics and grammatical facts to capture the important facts and information from source document(s).

Our devised system for AESOP Task uses the above discussed features to judge the summary quality.

We believe that, the output of a good generic system may be able to answer the most of the general queries, which are used in guided summarization task. This is main reason for the development of a generic multi-document summarization system for guided summarization task.

Part-1: AESOP - Task

1 Introduction

Evaluation of machine generated summaries has been of importance both in TAC (Text Analysis Conference) and previously DUC (Document Understanding Conference). In this vein, Automatically Evaluating Summaries of Peers (AESOP) task in TAC 2010 focuses on developing automatic metrics to judge summary quality. The main goal of AESOP task is to produce two sets of numeric summary-level scores i.e.

All Peers case: a numeric score for each peer summary, including the model summaries. The "All Peers" case is intended to focus on whether an automatic metric can differentiate between human and automatic summarizers.

No Models case: a numeric score for each peer summary, excluding the model summaries. The "No Models" case is intended to focus on how well an automatic metric can evaluate automatic summaries.

Evaluation Process: Each AESOP run is evaluated for:

1. Correlation with the manual metric
2. Discriminative Power compared with the manual metric

1.1 Current Trends and Techniques

[1], Manual pyramid scores and [2], automatic ROUGE metric (considers lexical n-grams as the unit for comparing the overlap between summaries) are generally considered as current state-of-the-art techniques.

[5], [6] proposed basic elements based methods (BE, another state-of-the-art technique), which facilitates matching of expressive variants of syntactically well-formed units called Basic Elements (BEs).

The ROUGE/BE toolkit has become the standard automatic method for evaluating the content of machine-generated summaries, but the correlation of these automatic scores with human evaluation metrics has not always been consistent and tested only for fixed length human and machine generated summaries.

Donaway [12] proposed using sentence-rank-based and content-based measures for evaluating extract summaries, and compared these with recall-based evaluation measures.

1.2 Motivating Factors

This system is a slight enhancement of our work [15]. Most techniques in this field consider: (1) Co-occurrence of N-grams, or (2) overlapping of sequences, or (3) similarity at the level of sentences etc.

On the contrary, we believe that (1) every word of a document has different importance, so we cannot provide equal weight to every co-occurring words or N-grams or sequence in the evaluation process, similarly (2) a document may have more than one topic, (3) the importance and coverage strength of the topics of the same document may be different and (4) we cannot provide equal weight to every sentence of the document. Based on these facts, we concentrated our attention on the calculation of (1) role / importance of words, (2) importance of sentences and (3) number and importance of topics in given document. We employ all these facts in the calculation of information coverage in test document(s) w.r.t. given model document(s).

Role of Words: Generally, keyphrase extraction algorithms use such concepts, to extract terms which either have a good coverage of the document or are able to represent the document's theme. We use this concept and prepare a proper scheme to calculate the weight of words in a document, We employ features like: (1) Frequency, (2) Position of the word in the sentence, (3) position of the sentence in which a given word occurs, and (4) length of sentence etc. to calculate the importance of the word in a given model document / source document.

Importance of sentences: Another issue is that all the techniques mentioned above do not give any weight to the role of sentences in a document or human-written model summary.

In general, we cannot provide the same importance to every sentence of a document. The importance of sentences in same document depends upon a lot of factors including (but not limited to) (1) Information contents (i.e. weight or importance of words that occur in that sentence) and (2) order or position of sentences in document etc.

Number and importance of topics: As the number and strength of topics in any document may vary, so we use [15] community detection based approach to automatically identify the sentence communities in document, which represent the topics covered in document. We extended the weighting scheme applied to calculate the weight of words and sentences to calculate the weight or information coverage of every identified topic.

Based on above discussed facts, we developed an automatic evaluation technique for AESOP Task. The devised system, first of all identifies the number of topics covered in given model or source summary. For this it uses community detection scheme [9], [10] and identify all sentence communities in given model or source summary. Next, it calculates the importance of all identified topics and then uniquely maps the most matching sentences from machine generated / target summary to these identified topics. Later it calculates how much information related to every identified topic exists in the most uniquely matching sentences from machine generated / target summary. Thus, it evaluates not only the topics covered in machine generated / target summary, but also able to evaluate the strength of information coverage in machine generated / target summary, related to each identified topics in model / source summary.

2 Framework and Algorithm

The entire scheme is given in sequel.

2.1 Input Cleaning and Pre-processing

Input cleaning task includes: (1) removal of unnecessary symbols, (2) stemming and (3) sentence filtration. To stem the document we have used Porter Stemmer [13].

2.2 Calculating Weight

At this phase we calculate the weight of every distinct words of given model / source summary. The weight calculation scheme depends upon the following features.

Frequency: It is the most widely used feature, but several times, the direct dependency on frequency can misguide us; as some noisy word may have very high frequency, or some useful word may have low frequency. So, instead of applying the direct occurrence frequency of any word, we, inherited the scheme used in [15] which, collect the information content of that word in a given document. In order to achieve this, it applies the concept of entropy and calculates the information content of word in document. In this case the weight does not depend directly on frequency, but depend on the probability of word and thus reduce the chances of giving more weight to highly frequent words. The scheme is given below:

$$W_1 = \frac{F}{N}\log_2\left(\frac{F}{N}\right) \qquad \text{-- (1)}$$

Where:

W_1 = Entropy of word in given document.

F = Occurrence frequency of word in document

N = Total Number of words in document.

This scheme is different from the technique used in [8], and considers all words in the given document.

Position of sentence in document in which the given distinct word exists: Here, we utilize the well known fact that the word which comes earlier is more important [11]. To convert this fact in weighting, we use the sentence index in which the given distinct word appears first and total number of sentences in given document. The calculation scheme is given as:

$$W_2 = \left(\frac{S_{total}+1}{S_f+1}\right) \qquad \text{-- (2)}$$

Where

W_2 =weight of given word, due to index position of the sentence in which the given word occurs first.

S_{total} =Total number of sentences in given document.

S_f =Sentence Index in which the given word occurs first.

The value of this ratio will be high, if the sentence index position i.e. S_f will be less.

Strength due to combined effect of length of sentence and position related strength: Length of sentence is also a deciding factor. For example: if a distinct word exists at same index position at two different sentences and length of both sentences varies, than their importance in both sentences will also vary. Generally with the increase in length of sentence the importance of given distinct word which, exist at subject position increases. So we combine the both features in calculation of weight of word.

The overall calculation scheme is given below:

$$W_3 = \log_2\left(\Sigma\left(\frac{L(S)+1}{P(K)+1}\right)\right) \qquad \text{-- (4)}$$

Where:

W_3= Weight value of given distinct word, calculated by using position related strength of word in sentence and length of sentence in which it exist.

$L(S)$= Length of sentence 'S' in which the given distinct word 'K' is present. This can be calculated by finding count of the number of words in 'S'.

$P(K)$= Index position of given distinct word 'K' in sentence 'S'.

Description: The value of this scheme depends on the ratio $((L(S)+1)/(P(K)+1))$ i.e. depends on both, length of sentence and index position related strength of the given distinct word in that sentence. It achieves our motivation behind using this scheme because

(1) If the length of sentence increases w.r.t. the index position related strength of given distinct word, then its importance increases.

(2) If a given distinct word comes early in the sentence and hence, its index position related strength is less, then its importance increases.

Final Weight calculation scheme: To calculate the weight of every distinct word in given document, we deploy all the above discussed features, i.e. (1) Frequency, (2) Position of sentence in document in which the given distinct word exists and (3) Strength due to combined effect of length of sentence and position related strength. The overall scheme to calculate the weight is:

$$W(K) = (W_1 \times W_2 \times W_3) \qquad \text{-- (5)}$$

Where

$W(K)$= weight of distinct word 'K' in given document.

For W_1, W_2 and W_3 refer to equation (1), (2) and (4) respectively.

2.3 Sentence Community Detection

We take pre-processed document and filter the sentences. Next, we prepare the graph of sentences by treating every sentence as node of the graph. An undirected graph of sentences is created, in which the connection between any two nodes or sentences, depends upon the occurrence of common words between them. The weight of edge of this graph is calculated using the following scheme:

$$W(E_{(S1,S2)}) = \frac{1}{count\_common\_word(S1,S2)} \qquad \text{-- (6)}$$

Where

$W(E_{(S1,S2)})$=weight of edges between sentences S1 and S2

$count\_common\_word(S1,S2)$=count of common words between sentences, S1 and S2.

Finally, we apply the shortest path betweenness strategy, as applied in [9]; [10] to calculate the sentence community. We use the faster version of community detection algorithm [9] which is optimized for large networks. This algorithm iteratively removes edges from the network to split it into communities. The edges removed being identified using graph theoretic measure of edge betweenness. The edge betweenness can be defined as the number of shortest paths between vertex pairs that go along an edge. To estimate the goodness of the certain graph partition, the authors of [9] propose the notion of modularity. Modularity [9] is a network" s property which refers to a specific proposed division of that network into communities. It measures whether the division is a good one, in the sense that there are many edges within communities and only a few between them. We inherited this scheme from our previous work [15].

2.4 Calculating Weighted Importance of Communities

After step 2.3, we have sentence communities for given reference summary or model document. These sentence communities are referred as topics covered in document. Now the issue is to calculate the weighted importance of every identified Topic.

To calculate the weighted importance of any topic or sentence community we depend on the Sum of weighted importance of all words in the given sentence community. The calculation of weighted importance of any community can be given as:

$$W(C) = \sum W_{wd} \qquad \text{-- (7)}$$

Where
$W(C)$= weight of given community 'C'
$\sum W_{wd}$ =weight of all words in given community. (see sec 2.2 for calculation of weight of words).

Next, we calculate the percentage of weighted information of every community in all identified community. The percentage weighted importance of any identified sentence community can be calculated as:

$$\%W(C) = \left(\frac{W(C)}{\sum W(C)} \times 100\right) \qquad \text{-- (8)}$$

Where:
$\%W(C)$=percentage weight of given community 'C'.
$\sum W(C)$=sum of weighted importance of all identified communities.
$W(C)$= weight of given community 'C'
The scheme is similar to our previous work [15].

2.5 Preparing Evaluation Sets

At this stage, we uniquely map the sentences from target summary / machine generated summary to sentence communities. For this, we consider only the most matching sentences from target summary / machine generated summary. This mapping may be one-to-one, one-to-many. This depends upon the topics covered in the target summary / machine generated summary w.r.t. corresponding source or model summary. Thus finally each evaluation set contains an identified topic (i.e.

sentence community from source or model summary) and uniquely mapped set of sentences from target summary / machine generated summary).

2.6 Evaluation Scheme

The main aim of this evaluation set is to calculate the strength of information coverage by machine generated summary w.r.t. corresponding identified topics. We apply this scheme to convert this coverage strength into score.

At this step, we take every evaluation set one by one and check, if it contains uniquely mapped sentence(s) from machine generated summary then we calculate the matching score for every such set. For this, first of all we calculate the weighted score for matching words in both i.e. sentence community of model summary and uniquely mapped sentences from machine generated summary. For non-matching words, we check if any non matched word of mapped sentences is synonyms of any existing word in given sentence community (or topic). If such match occurs then we consider this also as matching entry. To check for synonym match, we depend upon oxford dictionary synonym list. Now we apply following formula to calculate the weighted score in any given evaluation set S_i.

$$Score(S_i) = \left(\frac{\sum Count_{match}(word)}{\sum Count(word)} \times 100 \right) \times \left(\frac{\%W(C)}{100} \right)$$

This implies:

$$Score(S_i) = \left(\frac{\sum Count_{match}(word)}{\sum Count(word)} \right) \times (\%W(C)) \quad \text{-- (9)}$$

Where:

$Score(S_i)$ = Evaluation score obtained at set S_i. This is a percentage score.

$\sum Count_{match}(word)$ =count of all such words in sentence community, which co-occur in both i.e. Sentence community (topic) and uniquely mapped sentence(s) from machine generated summary. As described earlier, we use synonym list to broaden our vision of matching entries.

$\sum Count(word)$ = Count of all words in given sentence community.

Note: In any given evaluation set, if there does not exist any mapped sentences for given sentence community, then we set the evaluation score of that set to zero. i.e.

$$Score(S_i) = 0; \quad \text{-- (10)}$$

Calculating Final Score: For this we just add the score of all evaluation sets. This can be given as:

$$Final\_Score = \sum_{i=1}^{k} Score(S_i) \quad \text{-- (11)}$$

Where:

$Final\_Score$=sum of percentage scores obtained from all evaluation sets.
K = Denote the total number of evaluation sets. This scheme is similar to our previous work [15].

Applying This System for AESOP Task: At the time of peer evaluation, we check, if any source / model summary, (obtained by peer evaluation scheme as discussed above) contains initial percentage score, then we use this score as reference and update the final percentage score of current target summary (on peer evaluation with source / model summary). In no model case source may be other no-model summaries.

3 Pseudo code

The pseudo code for entire system can be given as:
Input: CASE 1: (1) source / model summary, (2) target / machine generated summary, both in ASCII format.
Output: %score, which can be further normalized to “0-1” scale.
Algorithm:

1. Apply pre-processing and input cleaning for source / model summary and target / machine generated summary. If source / model summary contains final reference score (% score) all other information as given in step 2, 3 and 3, than apply pre-processing and input cleaning for target / machine generated summary only and go to step 5.
2. Calculate the weight of every word of source / model summary. (see step 2.2).
3. Identify the sentence community(s) in source / model document (also addressed as topic(s); see sec-2.3).
4. Calculate the weighted importance of every identified sentence community (see sec-2.4).
5. Prepare separate evaluation set for every identified sentence community of source / model summary by uniquely mapping the sentences from target / machine generated summary (see sec 2.5).
6. Use all Evaluation sets and apply evaluation scheme to generate the final score (see sec-2.6).

4 Evaluation

At this section, we present the evaluation score of (1) All-peers and (2) No Model case. For each automatic metric submitted to the AESOP task, NIST calculated Pearson's, Spearman's, and Kendall's correlations with Pyramid and Overall Responsiveness, as well as the discriminative power of the automatic metric in comparison with these two manual metrics. The results of our system are given in Table-1 and Table-2.

Table-1: Correlations for Our System; initial and update summaries, all peers.

	Initial Summaries				Update Summaries			
	Pyramid		Responsiveness		Pyramid		Responsiveness	
	value	P(95% CI)	value	P(95% CI)	value	P(95% CI)	value	P(95% CI)
Pearson	0.975	0.000 (0.956 - 0.985)	0.977	0.000 (0.959 - 0.987)	0.926	0.000 (0.873 - 0.957)	0.928	0.000 (0.876 - 0.958)
Spearman	0.965	0.000	0.959	0.000	0.940	0.000	0.905	0.000
Kendall	0.859	0.000	0.853	0.000	0.821	0.000	0.752	0.000

Table-2: Correlations for Our System; initial and update summaries, no models.

	Initial Summaries				Update Summaries			
	Pyramid		Responsiveness		Pyramid		Responsiveness	
	value	P(95% CI)	value	P(95% CI)	value	P(95% CI)	value	P(95% CI)
Pearson	0.939	0.000 (0.889 - 0.967)	0.920	0.000 (0.856 - 0.956)	0.900	0.000 (0.821 - 0.945)	0.884	0.000 (0.795 - 0.936)
Spearman	0.947	0.000	0.937	0.000	0.903	0.000	0.849	0.000
Kendall	0.829	0.000	0.822	0.000	0.767	0.000	0.682	0.000

Part-2: GUIDED SUMMARIZATION - Task

Introduction: The guided summarization task of TAC-2010 includes two sub-tasks:

1. The guided summarization task was to write a 100-word summary of a set of 10 newswire articles for a given topic, where the topic falls into a predefined category. Participants were given a list of aspects for each category, and a summary must include all aspects found for its category.
2. Additionally, an "update" component of the guided summarization task was to write a 100-word "update" summary of a subsequent 10 newswire articles for the topic, under the assumption that the user had already read the earlier articles.

We believe that output of a good generic system may be able to answer the most of the general queries, which are used in guided summarization task. So we prepare a generic multi-document summarization system for above discussed task. The devised system does not require any (1) learning or training, or (2) External vocabulary, thesaurus, any other knowledge base support or (3) other corpus support or (4) any type of fine tuning. The proposed system is highly unsupervised in nature and even does not require any human intervention in entire execution of the system i.e. highly automatic.

5 System Description

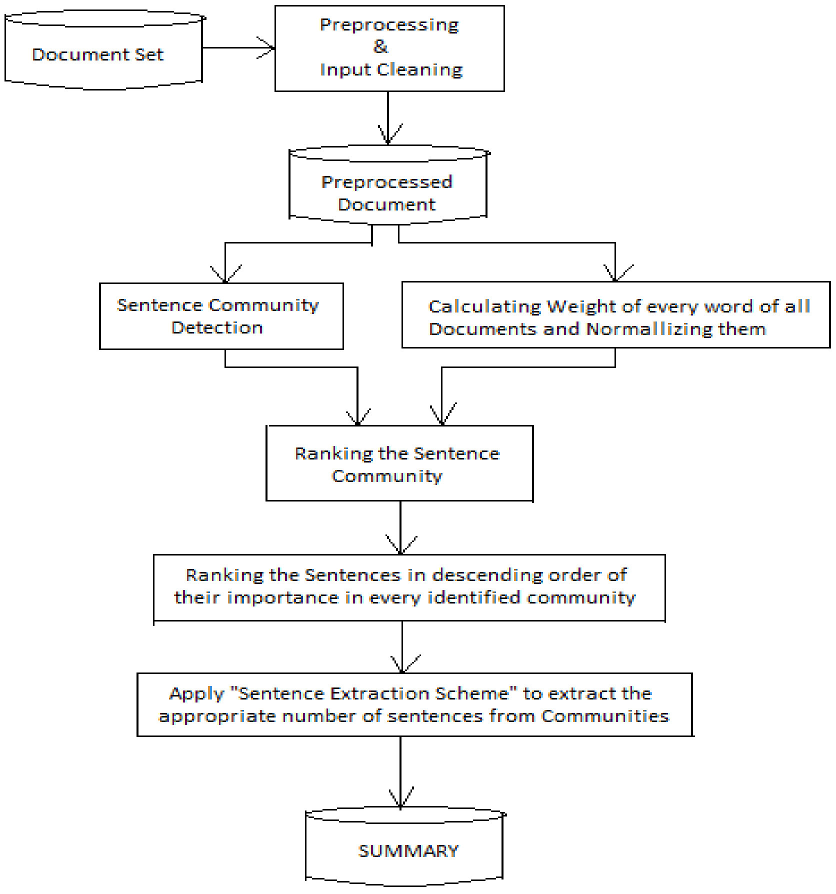
Document Set
Preprocessing & Input Cleaning
Preprocessed Document
Sentence Community Detection
Calculating Weight of every word of all Documents and Normallizing them
Ranking the Sentence Community
Ranking the Sentences in descending order of their importance in every identified community
Apply "Sentence Extraction Scheme" to extract the appropriate number of sentences from Communities
SUMMARY

Figure 1: System Description for Guided Summarization

5.1 Input cleaning and preprocessing

This step includes (1) removal of unnecessary symbols, (2) Sentence filtration and (3) Stemming. For stemming, we use porter stemming algorithm.

5.2 Calculating weight of all words of every document

A word may be common to several documents. But, we believe that it may have different importance in different documents, in which exist. So we calculated the weight or weighted importance of every word of all documents. In entire weight calculation scheme, we treat every document separately and exploit same strategies as given in Part-1, (see section 2.2). To handle the common words, which might be common to more than one documents, we use two level of indexing (i.e. in addition to word index, we used the document index to represent such words). Finally, we normalize the calculated weight of all words.

5.3 Sentence community detection

We take the set of pre-processed documents apply sentence filtration. Next, we prepare the graph of sentences by treating every sentence (i.e. every sentence of given documents set) as node of the graph. An undirected graph of sentences is created, in which the connection between any two nodes or sentences, depends upon the occurrence of common words between them. The rest of the process is similar to Section 2.3 of Part-1.

By using this process, we prepare the sentence communities of all sentences of documents of given document set.

5.4 Ranking of sentence communities

The ranking of sentence communities means, arranging the sentence communities in descending order of their information content. For this, we calculate the sum of weight of all words in every identified community (this is similar to eq-7, Part-1, Section 2.4). Finally we rank all the communities in descending order of the calculated weight.

5.5 Ranking the sentences in descending order of their importance in every identified sentence community

At this phase we, calculate the weight of every sentence in given community. Based on the calculated weight, we rank the sentences in descending order of their weight. The entire weight calculation scheme depends upon following facts.

Average Weight of sentence: The average weight of sentence is the ratio of weight of sentence and total weight of all sentences in given community. It can be given as:

$$W_1(S) = \frac{W_S}{W_T} \quad \text{-- (12)}$$

Where,

$W_1(S)$= average weight of sentence.

W_S = sum of weight of all words of given sentence ‘S’. See section –2.2 (for calculation of weight of words).

W_T = sum weight of all words in given community.

Calculating the average weight of Common words: In this scheme we prepare the list of those words of sentence ‘S’, which are common to any other sentence(s) in the same given community ‘C’. Next, we add the weight of those words (see sec 2.2, to calculate the weight of word). The average weight of common words of any sentence

‘S’, are the ratio of sum weight of common words of ‘S’ and sum of weight of all words in same sentence community ‘C’. It can be given as:

$$W_2(S)=\frac{W(CW,S)}{W_T} \qquad \text{--- (13)}$$

Where,

$W_2(S)$= average weight of common words of given sentence ‘S’.
$W(CW,S)$= sum of weight of common words of sentence ‘S’.
W_T = sum weight of all words in given community ‘C’.

Description: This score calculate the commonness of information of sentence ‘S’, w.r.t. other sentences of given community ‘C’.

Average sentence Length: This is a ratio of length of given sentence ‘S’ (i.e. count of number of words in sentence ‘S’) and total number of words in all sentences in given community ‘C’. The scheme can be given as:

$$W_3(S)=\frac{\#words(S)}{\#words(C)} \qquad \text{--- (14)}$$

Where,
$W_3(S)$= average length of sentence ‘S’.
$\#words(S)$= count of number of words in sentence ‘S’.
$\#words(C)$= count of total number of words in given sentence community ‘C’.

Final Weight calculation: The final weight of sentence ‘S’ is the combination of all the above discussed features. It can be given as:

$$W(S)=W_1(S)\times W_2(S)\times W_3(S) \qquad \text{--- (15)}$$

Ranking of sentences in sentence community: Rank all the sentences in descending order of the weight of sentences for given community ‘C’ (see Eq-15, for calculation of final weight).

5.6 Applying Sentence Extraction scheme

We select the single top sentence from every identified community and arrange them according to the importance/rank of the native community. Later we select, the top ‘K’ sentences from set of extracted sentences to reach the word count, nearly equal to the given word limit.

5.7 Pseudo Code

I. **Generating 100 word summary for document set ‘A’**: for this, we use the following steps (see figure-1):

```
Step 1. Apply input cleaning and preprocessing for every document of given
        set ‘A’ (See section 5.1).
Step 2. Calculate the weight of all words of every document of given set (see
        section 5.2).
```

Step 3. Identify sentence communities by using all sentences of given set of documents (see section 5.3).

Step 4. Rank all the identified sentence communities in descending order of their weight (see section 5.4).

Step 5. Rank the sentences in descending order of their importance in every identified sentence community (see section 5.5).

Step 6. Apply the sentence extraction scheme (see section 5.6).

II. **Generating the update summary**: For this we make a simple change in the above discussed system for guided summarization. We use the documents from both sets (i.e. set 'A' and set 'B' both) till "step 5". Thus our devised system at "step 5" contains sentences from both document sets (i.e. Set A and B) in identified sentence communities. Here, we rank the sentences in every sentence community, according to descending order of their weight (see Eq-15, and "step 5" of above discussed algorithm).

"Maintaining sentences from both sets, up to Step 5" is just because, we want to capture the story duration related feature. It is a well known fact that, the important or famous stories lasts for long time and gets better coverage.

Due to this fact, a famous story can be captured by a high rank sentence community if, after removal of sentences which may be related to stories of definite time range and earlier document set (i.e. set A), the community will still contain a minimum number of sentences from different document set (i.e. from set 'B') but related to same story. So as a next step, we remove all the sentences related to set 'A' from every community without changing the order of rest of the sentences. We also maintain the ranking of all sentence communities (i.e. do not change the rank of any community after removal of sentences related to set – 'A').

After removal of sentence(s) related to set-'A' from sentence community, if any sentence community contains very few like 2-3 sentences and is in top rank than we discard that community. As, the presence of very few sentences in such community after removal of sentences from set 'A', shows that after given period of time the related story has lost their focus.

Finally, we apply the same step, as discussed in step -6 and collect the required number of sentences.

5.8 Evaluation

Eight NIST assessors selected and wrote summaries for the 46 topics in the TAC 2010 guided summarization task. Each topic had 2 docsets (A,B), and NIST assessors wrote 4 model summaries for each docset. The NIST human summarizer IDs are A-H.

NIST received 41 runs from 23 participants for the guided summarization task. The participants each submitted up to two runs, and their summarizer IDs are 3-43.

In addition, two baseline runs were included in the evaluation, and their summarizer IDs are 1-2:

Baseline 1 (summarizer ID = 1): returns all the leading sentences (up to 100 words) in the most recent document. Baseline 1 provides a lower bound on what can be achieved with a simple fully automatic extractive summarizer.

Baseline 2 (summarizer ID = 2): output of MEAD automatic summarizer with all default settings, set to producing 100-word summaries.

NIST evaluated all summaries manually for overall responsiveness and for content according to the Pyramid method. All summaries were also automatically evaluated using ROUGE/BE.

The results of our system are given in Table-3.

Table-3: TAC Evaluation results on given document sets

	CLUSTER-A	CLUSTER-B
ROUGE-2	0.0683	0.06062
ROUGE SU4	0.10436	0.10015
Avg.Pyramid	0.340	0.224
Responsiveness	2.783	2.304

6 Conclusion and Future Work

From the results for both task i.e. AESOP and Update summarization task, it is clear that our scheme of giving emphasis on (1) role & importance of words and sentences in document, (2) number & coverage strength of topics etc.; gives good result.

In both tasks we found that knowledge gap between the words (due to use of different writing strategies or selection of different word set to express same or similar things) is a major hurdle. It will be interesting to extend and test this system by using N-grams and Wikipedia based Information gap reduction method as used in [14]. A better document cleaning approach may be other future work for this system.

Acknowledgments I (Niraj kumar), would like to thank MHRD-India, for partial financial support for AESOP task.

References

1. Ani Nenkova, Rebecca Passonneau, and Kathleen McKeown. 2007. The pyramid method: Incorporating human content selection variation in summarization evaluation. ACM Trans. Speech Lang. Process., 4(2):4.
2. Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurance statistics. In Proceedings of HLT-NAACL 2003.
3. Teufel, S. and H. van Halteren. 2004. Evaluating Information Content by Factoid Analysis: Human Annotation and Stability. Proceedings of the NLP 2004 conference. Barcelona, Spain.

4. Nenkova, A. and R. Passonneau. 2004. Evaluating Content Selection in Summarization: The Pyramid Method. Proceedings of the HLT-NAACL 2004 conference.
5. Hovy, E.H., C.Y. Lin, and L. Zhou. 2005. Evaluating DUC 2005 using Basic Elements. Proceedings of DUC-2005 workshop.
6. Hovy, E.H., C.Y. Lin, L. Zhou, and J. Fukumoto.2006. Automated Summarization Evaluation with Basic Elements. Full paper. Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 06). Genoa, Italy.
7. Conroy, J.M. and H.Trang Dang. 2008. Mind the Gap: Dangers of Divorcing Evaluations of Summary Content from Linguistic Quality. Proceedings of the COLING conference. Manchester, UK.
8. Li, S., Wang, H., Yu, S. (2004) Research on Maximum Entropy Model for Keyword Indexing. Chinese Journal of Computers 27(9), 1192–1197.
9. Clauset, A., Newman, M. E. J., Moore, C. (2004). Finding community structure in verylarge networks. Physical Review E, 70:066111.
10. Newman, M. E. J., Girvan, M. (2004). Finding and evaluating community structure in networks. Physical review E, 69:026113, 2004.
11. Kumar, N., Srinathan, K. (2008). Automatic Keyphrase Extraction from Scientific Documents Using N-gram Filtration Technique. In the Proceedings of ACM DocEng 2008, ACM 978-1-60558-081-4/08/09.6.
12. Donaway R, Drummey K, Mather L.A Comparison of Rankings Produced by Summarization Evaluation Measures. In Proceeding of ANLP/NAACL Workshop on Automatic Summarization, pages 69-78, 2000.
13. Porter Stemming Algorithm for suffix stripping, web –link http://telemat.det.unifi.it/book/2001 /wchange/ download/ stem_porter.html.
14. Niraj Kumar;Venkata Vinay Babu Vemula;Kannan Srinathan;Vasudeva Varma;EXPLOITING N-GRAM IMPORTANCE AND ADDITIONAL KNOWEDGE BASED ON WIKIPEDIA FOR IMPROVEMENTS IN GAAC BASED DOCUMENT CLUSTERING;KDIR 2010.
15. Niraj Kumar, Kannan Srinathan and Vasudeva Varma; Evaluating Information Coverage in Machine Generated Summary and Variable Length Documents; COMAD-2010, Nagpur, India.