Concept-Based Extractive Text Summarization Using Graph Modelling and Weighted Iterative Ranking

S. Chitrakala, N. Moratanch, B. Ramya, C.G. Revanth Raaj and B. Divya

Abstract Text summarization is a process of reducing the whole text document into a summary by retaining the most important information and to present it to the end user. Wikipedia which is a human-generated knowledge base is used to identify the key sentences using weighted iterative ranking algorithm which is the variation of HITS algorithm. The pre-processed input document is used in the construction of bipartite graph which maps the input sentences to the Wikipedia concepts. The bipartite graph captures the nested level of relationship between the sentences and concepts to ensure the highest level of efficiency in the extractive output summary. Weighted iterative ranking algorithm is used to retrieve top ranked sentences. The system produce summaries with good coverage, high coherency and low redundancy. The system can be deployed to summarize news articles, producing abstracts from documents, summarize web pages. The new article summarization would be helpful to mobile users.

Keywords Natural language · Wikipedia · Text summarization
Weighted iterative ranking · Bipartite graph · Concept based

S. Chitrakala (✉) · N. Moratanch · B. Ramya · C.G. Revanth Raaj · B. Divya
Department of CSE, College of Engineering, Anna University, Chennai, India
e-mail: au.chitras@gmail.com

N. Moratanch
e-mail: tancyanbil@gmail.com

B. Ramya
e-mail: ryaria18@gmail.com

C.G. Revanth Raaj
e-mail: revanthrajcgr@gmail.com

B. Divya
e-mail: divyabalamurugan95@gmail.com

N.R. Shetty et al. (eds.), *Emerging Research in Computing, Information, Communication and Applications*, https://doi.org/10.1007/978-981-10-4741-1_14

1 Introduction

Summarization generally focuses on generating a condensed and crisp version of a document that covers the document's main topic. In recent times, text summarization plays a prominent role in providing the most important and precise information to the users. There are two approaches to automatic summarization, extractive and abstractive methods of summarization. Extractive method of summarization selects a subset of existing words, phrases or sentences in the original text to form the summary. Abstractive method of summarization build an internal semantic representation and then use natural language generation techniques to create a summary that is closer to what a human might generate.

Summaries are composed by number of sentences. So, the basic idea of arriving at the summary is to include the sentences that serve more meaning to the summary and the sentences should be present in the same order as it is given in the original document. In other words, every selected sentence is expected to be both salient and novel.

The main contribution of this work is to cast the Wikipedia-based summarization problem into a general sentence-concept *bipartite framework*, and weighted *iterative ranking algorithm* for selecting summary sentences. The summary of sentences is produced to the users. Also, the system provides incremental summarization. The one-third of the original document is produced as the summary to the end users which holds all the main idea and important sentences needed to be delivered through the document.

This paper brings the approach for extractive text summarization using weighted iterative ranking algorithm.

The paper on "Text Summarization using Wikipedia" [1] captures relationship between sentences with the help of wiki concepts modelling it as a bipartite graph. The variation proposed in CSUMMIT is that, the system considers nested level of relationship between sentences and concepts to improve the efficiency of the generated summary to an extent.

The rest of the paper is organized as follows. Section 2 details the related work in the summarization domain followed by Sect. 3 which elaborates the system architecture in detail.

2 Related Work

More recently, summarization has become a successful task and many studies have been taken on that. The approaches are majorly classified as supervised and unsupervised learning approach whereas the latter is focused more in the new summarization algorithms.

A graph-based approach LexRank [2], where the salience of the sentence is determined by the concept of Eigenvector centrality. The sentences in the document are represented as a graph and the edges between the sentences represent weighted cosine similarity values. The sentences are clustered into groups based on their similarity measures and then the sentences are ranked based on their LexRank scores similar to PageRank algorithm [3] except that the similarity graph is undirected in LexRank method. The method outperforms earlier versions of lead and centroid based approaches.

In paper [4–6] fuzzy logic approach is used for automatic text summarization which is based on the feature selection and feature extraction. The sentences are ranked based on the fuzzy logic scoring which is obtained by applying fuzzy rule based. The summary is generated by ordering the ranked sentences in the order they occur in the original document to maintain coherency.

In concept-based approach, the concepts are extracted for a piece of text from external knowledge base such HowNet [7] and Wikipedia [8]. In the methodology proposed in [7], the importance of sentences are calculated based on the concepts retrieved from HowNet instead of words. A conceptual vector model is built to obtain a rough summarization and similarity measures are calculated between the sentences to reduce redundancy in the final summary.

An algebraic-statistical method Latent Semantic Analysis (LSA) [9, 10] is used where hidden semantic structures of words and sentences and popularly used in text summarization task are extracted. It is an unsupervised approach that does not need any sort of training or external knowledge. LSA captures the context of the input document and extracts information such as words that frequently occur together and words that are commonly seen in different sentences. A high number of common words amongst the sentences indicate that the sentences are semantically related.

Dharmendra Hingu, Deep Shah and Sandeep S. Udmale proposed an extractive approach [11] for summarizing the Wikipedia articles by identifying the text features and scoring the sentences accordingly incorporating neural network model [12]. The preprocessed passage is sent to the feature extraction steps, which is based on multiple features of sentences and words. The scores obtained after the feature extraction are fed to the neural network, which produces a single value as output score, signifying the importance of the sentences. Usage of the words and sentences is not considered while assigning the weights which results in less accuracy.

Conditional random fields (CRF) [13] are used to identify and extract correct features to determine the important sentence of the given text. CRF segmentation assigns a label sequence to each token based on the training set. The goal of the proposed approach is to classify the sentences based on the patterns to segments. The main advantage of the method is that it is able to identify correct features and provides better representation of sentences and groups terms appropriately into its segments.

3 System Architecture

The system focuses to build a summarization system using graph based approach which employs Wikipedia concepts to determine the key sentences using weighted iterative ranking algorithm based on variation of HITS algorithm. Generalized bipartite graph framework with inclusion of concepts ensures coverage, the use of nested level relationship between sentence and concepts aids in better capturing of information and weighted iterative ranking algorithm promotes coherency.

A. *Pre-processing*

The original source document is given as input to the system. The input document is divided into meaningful units. The sentences are tokenized and are produced as output. The tokenized sentences are further processed for removal of stop words. The words with low semantic content are termed as stop words. These words do not contribute in identifying the important sentences in a text for example prepositions, articles, etc. These noisy terms are very common within a text and can be removed are stemmed. Stemming is a process of reducing the words with the same root or stem to a common form. This removal can be done with the help of maintaining a database of stop words. Further the sentences are stemmed. Stemming is a process of reducing the words with the same root or stem to a common form. This is done by eliminating the variable suffixes. The preprocessing is followed by other steps as shown in Fig. 1 which shows the work flow of the proposed system.

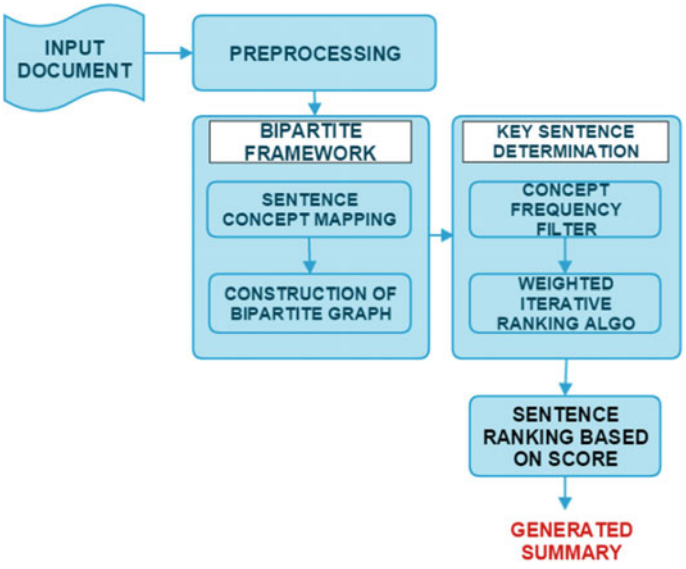

Fig. 1 Work flow of the proposed system

B. *Sentence-Concept mapping*

The pre-processed text document is given as query to the mapper. The mapper will map these queries with the corresponding Wikipedia concepts. Wikipedia article titles (concepts) are extracted from the results to the query ("hits" in Lucene terminology). The entire Wikipedia corpus is indexed using the Lucene engine. The Wikipedia dump is pre-processed to remove XML tags and other unnecessary information such as talk pages, comment section and edit history information. The cleaned Wikipedia dump is indexed using Lucene engine. The text and title fields of Wikipedia dump are indexed to make query search faster. The concepts (Wikipedia title name) is extracted for each pre-processed sentence, i.e. "query" and top concepts are retained for each query. The relationship between sentences is captured by higher degree of overlap of concepts among sentences. In most news articles, there exists overlap in concepts between sentences which conveys the relationship between two different sentences. For example, if two sentences are mapped to average number of same concepts, then they are more related to each other. The relationship can be captured with the help of bipartite graph data structure which consists of two disjoint set of nodes namely sentence node and concept nodes.

C. *Construction of Bipartite graph*

The sentence–concept mapping is represented as weighted bipartite graph- two distinct set of nodes representing sentences and concepts respectively. The bipartite graph consists of two sets of nodes (i) sentence node (ii) concept node and weighted edge defines the relationship between sentence and concept with the weight indicating the degree of correlation (The weight mentioned here refers to "Lucene hit score"). An edge exists from a sentence node to concept node if the concept is one amongst the extracted concept from Wikipedia. The graph is modelled as many-many mapping since one sentence can map to "n" number of concepts and a concept node can map to "m" number of sentences, which shows there is possibly overlap in concepts among sentences which helps in measuring the most related sentences to maintain coherency. The model also captures nested level of relations, i.e. when two sentences are related (Two sentences are related when they are mapped to higher number of same set of concepts), an edge can be simulated between a concept and a sentence to which it is not mapped initially based on transitive relationship between the corresponding related sentences and concepts that they are mapped to. This transitive dependence captures the second level of relationship between the sentence and concepts. The graph-based modelling ensures that the relationship among sentences is well captured and application of ranking algorithms to graph models are more efficient Fig. 2.

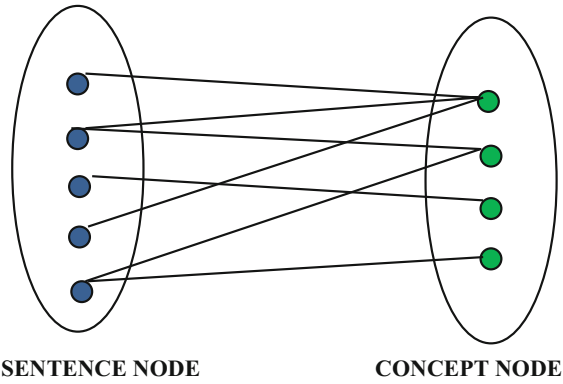

Fig. 2 Example of bipartite graph consisting of two sets of disjoint nodes U, a set of sentence nodes and V, a set of concept nodes

Algorithm1 Construction of bi-partite graph

```
Input: Sentence-Concept mapping C
Output: Bi-partite graph G
for each sentence s_i in source document
        add a node s_i to sentence set N
end for
for each concept c_j in concept class
        add a node c_j to concept set M
 end for
 for each mapping in C from s_i to c_j
        add an undirected edge between s_i and c_j
        nodes in graph
        edge_i.weight= Lucene_hit_score;
 end for
```

D. *Key Sentence Determination*

The main task of summary generation is to select key summary sentences that form a part of the summary. The sentences are selected based on the scores associated with the sentence. An iterative ranking algorithm is proposed to calculate sentence-concept score and rank the sentence based on their score which is helpful in identifying summary sentences.

1. *Concept frequency filter-Sentence Filtering*

A simple heuristic to filter important sentences is to rank the concepts in descending order based on their frequency. The concept-frequency score is used only to eliminate those sentences that do not contribute to the summary by any means and it acts as a filtering technique rather than a ranking method. The frequency here refers to the number of sentences that maps on to a particular concept. It is considered that more sentences the concepts maps to, it becomes important and core concept of the article since may sentences correspond to that concept. The sentences that maps on to the highest ranked concepts may contribute to the final summary. This simple heuristic however does not distinguish between summary and non-summary

sentences. The heuristic is used only to eliminate those sentences that maps only to a low ranked concept or no concept. The importance of a concept cannot be determined only with its frequency. The bipartite graph is updated after elimination of sentences that cannot contribute to the final summary. The importance of sentence has to be incorporated along with the concept to find summary sentences. Thus, an iterative ranking algorithm is proposed to mutually calculate sentence-concept scores which aid in selecting summary sentences. The pseudo code is presented in Algorithm 2.

Algorithm 2 Concept-frequency filter

```
Input: Bipartite-graph G
Output: Updated Bi-partite graph G after elimination of
                    certain sentences
for each concept node c_j in G do
      c_j.frequency_score ← c_j .no_of_edges
end for
Sort frequency score of concept in descending order
 for each sentence node s_j in G do
      if s_i .edgelist ϵ {∅} ∨ s_i .edgelist ϵ {c_l } then
        Delete sentence node s_i from graph G
      end if
  end for
```

2. *Weighted iterative ranking algorithm*

The main goal of Sentence Ranker is to rank sentence nodes in bipartite graph G in descending order of their importance. The importance of a sentence is tied to the concept and vice versa. Thus, the sentence-concept score is mutually calculated. The basic idea of algorithm is based on HITS algorithm [14] which works iteratively in mutually reinforcing manner to rank web pages based on authority-hub scores. A score is associated to each concept and sentence node in graph G iteratively. The iterative update is done for K times where K is determined based on convergence property. The sentence-scores are normalized after each iterative update to prevent them from exceeding without bound. From observation and analysis, it is noted that sentence scores are steady within 5–10 iterations. The ranking is done only once for all sentence nodes in graph G and global ranking is saved permanently. The values of g_{ij} and h_{ij} in Algorithm 3 refers to the Lucene hit scores for forward mapping (Sentence to concept) and backward mapping (Concept to Sentence) respectively. Refer Algorithm 3 for pseudo code. The following equations are used in Algorithm 3.

$$s_j^{(k+1)} = \sum_{i \in Nj} g_{ij} c_i^{(k)}, \forall j \in M \tag{1}$$

$$c_i^{(k+1)} = \sum_{j \in Mi} h_{ij} s_j^{(k)}, \forall i \in N \tag{2}$$

where $s_i^{(k)}$ represents sentence score and $y_j^{(k)}$ represents the sentence score and concept score after the kth update and concept score after kth update and $x_i^{(0)}$ is

initialized to $1/\sqrt{n}$. Normalise the sentence score after each iteration, so that scores doesnt exceed the bounds as given in Eq. (3)

$$\sum_{i \in N} \left(x_i^{(k)} \right)^2 = 1 \tag{3}$$

The sentence are ranked in descending order in descending order of $s_j^{(K)}$ as given in Eq. (4)

$$r = \arg\left(\text{descend}\left(x_1^{(k)} x_2^{(k)} \ldots \ldots x_n^{(k)} \right) \right) \tag{4}$$

The sentence score ranked in descending order is used to generate the final summary.

E. *Generation of Summary*

The summary is generated by selecting d leading sentences according to their rank. Let r denote set of sentences {s_{1},s_{2},s_{3}….s_{n}} where the indices denotes the rank of the sentence and d < n as in Eq. (4). The value of d is selected such that the summary generated is approximately nearer to one-fourth of the document. In practice, word-based summaries are required where word-size is set default to 50-word or 100-word summary. Since, the sentences are ranked using iterative sentence ranker, word-based summaries are produced by approximating the summary to the nearest sentence delimiter. Thus, both sentence-based and word-based summary can be generated where d can either be set as default or calculated dynamically based on the number of sentences in the article. The short summary covering major concepts in the article is presented to the user.

4 Experimental Results and Analysis

The system has been tested against the standard DUC 2002 dataset provided by National Institute of Standards and Technology (NIST) [15]. The DUC 2002 dataset consist of about 567 English NEWS articles. DUC is the most commonly and frequently used dataset for summarization task.

The most commonly used evaluation metrics in summarization domain are ROUGE metric, precision, recall and f-measure scores. The ROUGE evaluation approach depends on n-gram co-occurrence between the reference summary (i.e. ideal summary) and the machine generated extractive summary. ROUGE-N is computed as follows

Algorithm 3 Weighted Iterative Sentence Ranker

```
Input: Bipartite-graph G
Output: Sentence rank
Initialize si .score=1/√n for k=0
for k ϵ {0,...,K-1} do
    normalise ← 0
    for each concept cj in G do
        cj.score ← 0
        for each sentence si in cj.edgelist do
            cj.score=cj.score*gij +si.score
        end for
    end for
    for each sentence si in G do
        si.score ← 0
        for each concept cj in si .edgelist do
            si.score=si.score*hij + cj.score
        end for
        normalise+=square(si)
    end for
    normalise=sqrt(normalise)
    for each sentence si in G do
        si.score=si.score/normalise
    end for
end for
Rank sentences in descending order of their sentence rank
```

$$\textbf{ROUGE} - \textbf{N} = \frac{\sum_{S \in reference\_summaries} \sum_{N\text{-}grams} Count_{match}(N\text{-}gram)}{\sum_{S \in reference\_summaries} \sum_{N\text{-}grams} Count\ (N\text{-}gram)}$$

where *n* stands for the length of the n-gram, $\textbf{Count}_{\textbf{match}}$ **(N-gram)** is the maximum number of n-grams co-occurring in a machine generated summary and the ideal summary, **Count (N-gram)** is the number of N-grams in the ideal summary. Since ROUGE -1 scores are not sufficient enough to distinguish different summarizers extended set of evaluation metrics such as precision, recall, f-measure are calculated.

$$\text{Recall} = \frac{|S_{ref} \cap S_{cand}}{|S_{ref}|}$$

$$\text{Precision} = \frac{|S_{ref} \cap S_{cand}|}{|S_{cand}|}$$

Where, $S_{ref} \cap S_{cand}$ indicates the number of sentences that co-occur in both reference and candidate summaries.

The CSUMMIT summarizer system is compared against MS WORD summarizer which shows greater ROUGE- 2 scores compared to MS summarizer and the corresponding results are shown in Table 1. The visual interpretation of the result is shown in Figs. 3, 4 and 5 shows the implementation snapshot of CSUMMIT summarizer.

Table 1 Evaluation results

Summarizer	Rouge-1	Rouge-2	Precision	Recall	F measure
CSUMMIT	0.47	0.25	0.60	0.52	0.55
MS WORD	0.47	0.16	0.36	0.39	0.37

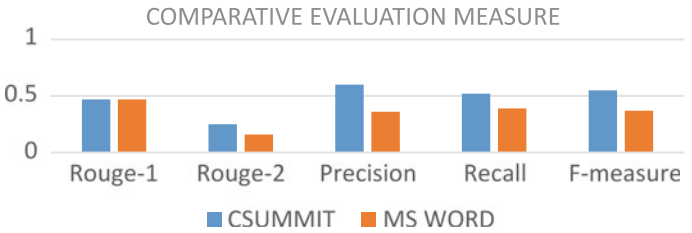

Fig. 3 Comparative analysis of evaluation metrics

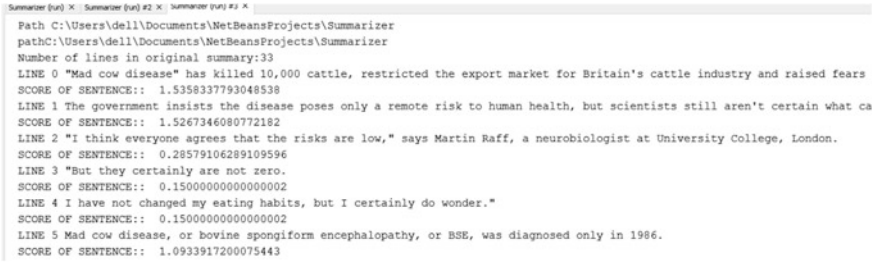

Fig. 4 Example snapshot indicating sentences along with its scores after applying weighted iterative ranking algorithm

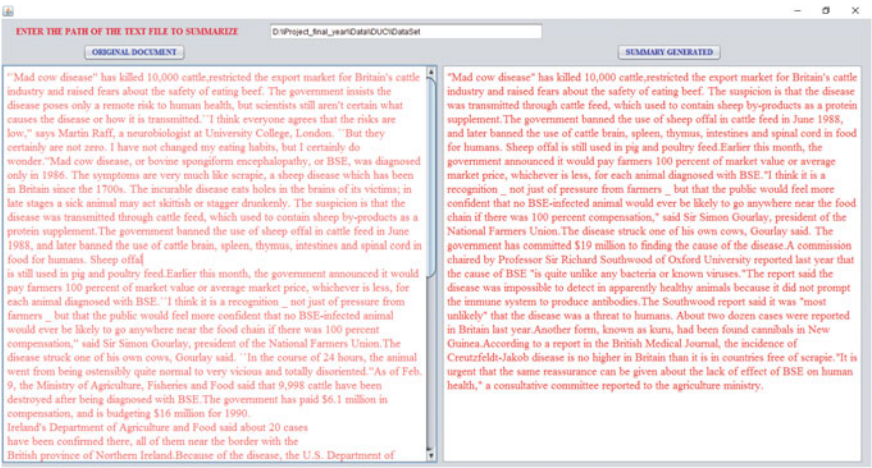

Fig. 5 Example snapshot of summarized text for the original document from DUC 2002 dataset

5 Conclusion

Wikipedia- knowledge base generated by human is employed to identify the important sentences in the given input document since it is used to identify the salient topics. This paper combines the bipartite graph framework with weighted iterative ranking algorithm to determine the key sentences. The bipartite graph is extended to capture the nested levels of relationship between sentences and concepts which ensures a higher level of efficiency in the generated summary. The weighted iterative ranking algorithm which uses weighted graph generates the extractive summary. CSUMMIT summarizer shows improved performance compared to the baseline summarizers.

The system can be extended for multiple documents and can be made domain specific to match the requirements of the user.

References

1. Sankarasubramaniam, Yogesh, Krishnan Ramanathan, and Subhankar Ghosh. "Text summarization using Wikipedia." *Information Processing & Management* 50.3 (2014): 443–461.
2. Erkan, Günes, and Dragomir R. Radev. "LexRank: Graph-based lexical centrality as salience in text summarization." *Journal of Artificial Intelligence Research* (2004): 457–479.
3. Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, Stanford, CA.
4. Farshad Kyoomarsi, Hamid Khosravi, Esfandiar Eslami and Pooya Khosravyan Dehkordy, "Optimizing Text Summarization Based on Fuzzy Logic", In proceedings of Seventh IEEE/ACIS International Conference on Computer and Information Science, IEEE, University of Shahid Bahonar Kerman, UK, 347–352, 2008.
5. Ladda Suanmali, Mohammed Salem, Binwahlan and Naomie Salim, "Sentence Features Fusion for Text summarization using Fuzzy Logic, IEEE, 142–145, 2009.
6. Ladda Suanmali, Naomie Salim and Mohammed Salem Binwahlan, "Fuzzy Logic Based Method for Improving Text Summarization", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 2, No.1, 2009.
7. Meng Wang, Xiaorong Wang and Chao Xu, "An Approach to Concept Oriented Text Summarization", in Proceedings of ISCIT'05, IEEE international conference, China, 1290–1293, 2005.
8. Ramanathan, Krishnan, et al. "Document summarization using Wikipedia."*Proceedings of the First International Conference on Intelligent Human Computer Interaction*. Springer India, 2009.
9. Ozsoy, Makbule Gulcin, Ferda Nur Alpaslan, and Ilyas Cicekli. "Text summarization using latent semantic analysis." *Journal of Information Science* 37.4 (2011): 405–417.
10. Mashechkin, I. V., et al. "Automatic text summarization using latent semantic analysis." *Programming and Computer Software* 37.6 (2011): 299–305.
11. Hingu, Dharmendra, Deep Shah, and Sandeep S. Udmale. "Automatic text summarization of Wikipedia articles." *Communication, Information & Computing Technology (ICCICT), 2015 International Conference on*. IEEE, 2015.

12. Khosrow Kaikhah, "Automatic Text Summarization with Neural Networks", in Proceedings of second international Conference on intelligent systems, IEEE, 40–44, Texas, USA, June 2004.
13. Batcha, Nowshath K., Normaziah A. Aziz, and Sharil I. Shafie. "CRF based feature extraction applied for supervised automatic text summarization."*Procedia Technology* 11 (2013): 426–436.
14. Kleinberg, Jon M. "Authoritative sources in a hyperlinked environment."*Journal of the ACM (JACM)* 46.5 (1999): 604–632.
15. Document Understanding Conference (DUC), 2002. http://tides.nist.gov/.