Highlighting Diverse Concepts in Documents

Kun Liu Evimaria Terzi Tyrone Grandison*

Abstract

We show the underpinnings of a method for summarizing documents: it ingests a document and automatically highlights a small set of sentences that are expected to cover the different aspects of the document. The sentences are picked using simple coverage and orthogonality criteria. We describe a novel combinatorial formulation that captures exactly the document-summarization problem, and we develop simple and efficient algorithms for solving it. We compare our algorithms with many popular document-summarization techniques via a broad set of experiments on real data. The results demonstrate that our algorithms work well in practice and give high-quality summaries.

1 Introduction

The task of automatic identification of important information themes within a text document is often referred to as Automatic Document Summarization. Although research on document summarization dates back in the 1950's [13], the emergence of Web 2.0 applications has revealed new angles of the problem. For example, when one is reading online comments and discussions following blogs, videos and news articles, it is desirable to have a summary that highlights different aspects of these comments because each one of them often has a different focus, and there usually does not exist a single central idea like in traditional text documents.

In this paper, we focus on single-document summarization by reusing a subset of the sentences of the original document. This is often referred to as *summarization by extraction*. Our goal is given an input document to construct a summary that highlights different aspects of the document. We require the following two properties to be exhibited by the constructed summaries:

Coverage: The summary should consist of sentences that span a large portion of the spectrum of aspects discussed in the document.

Orthogonality: Each sentence in the summary should capture different aspects of the document. That is, the sentences in the summary should be as *orthogonal* to each other as possible.

*IBM Almaden Research Center, San Jose, CA, USA. Email: {kun, eterzi, tyroneg}@us.ibm.com.

Note that the notion of *coverage* is different from *relevance* used in query-oriented summarization systems (e.g., [2]), where important sentences should be relevant to a user-generated query or question. It is also different from *centrality* used in centroid-based summarization systems (e.g., [16]), where important sentences should be close to the centroid of the document(s). Instead, *coverage* encourages the selection of sentences such that each of them represents a different aspects of the document. The notion of *orthogonality* captures the same concept of *(anti)-redundancy* in many other document summarization systems (e.g., [2]); the sentences in the summary are not redundant if they are not very similar to each other.

Contribution: Our main contribution is in the novel and clean combinatorial formulation for the Document Summarization problem, which captures both coverage and orthogonality requirements. As a by-product of this definition we also can demonstrate a natural connection between Document Summarization and several known combinatorial problems such as Maximum Coverage and Unique Coverage [3].

Moreover, we develop simple and efficient algorithms for solving the problem. These algorithms are unsupervised, and do not need domain-specific information (e.g., cue words, sentence's position). We compare our algorithms with many popular document-summarization techniques via a broad set of experiments on real data. Finally, we develop a new way of automatic evaluation of summaries produced by sentence-extraction methods. We do so by creating mixed documents that contain sentences from different domains and evaluating the coverage and orthogonality of the summaries with respect to this ground truth.

Overall, despite the fact that document summarization as an extraction problem has been studied many years in information-retrieval and text-mining communities, we are not aware of any prior work that abstracts the problem in the way we present here.

Roadmap: The rest of the paper is organized as follows. Section 2 gives an indicative overview of the related work. Section 3 gives the necessary notation

and the formal problem definitions. Our algorithm for the DOCUMENT SUMMARIZATION problem are given in Section 4. Alternative algorithmic approaches to the same problem are discussed in Section 5. We present experimental results in Section 6 and conclude in Section 7.

2 Related work

In this section, we give an indicative overview of this topic with emphasis on summarization that is based on sentence extraction from a single document. We note that this review is far from complete, and we refer the interested readers to `http://www.summarization.com/` and `http://duc.nist.gov/` for a comprehensive collection of resources favored by the community.

Traditional document summarization approaches (see [14] for an excellent survey) have assumed that the salient parts of a text document can be determined by features like cue words, frequency of terms, position of terms or sentences in the document etc. Most of the traditional approaches (see for example [6]) have focused on defining score functions for the sentences; a score is assigned to every sentence using features like the ones mentioned above. The score of a sentence is *independent* of other sentences and the output summary simply contains highly-scored sentences. Since the score of every sentence is not affected by the other sentences, two overlapping sentences, both with high scores, may be picked together to be in the summary. In our approach, we overcome this problem by assigning scores to sets of sentences rather than sentences independently. Therefore, our objective function does not encourage overlapping sentences to co-exist in the constructed summaries.

At a high level, approaches like MMR [2] and Mead [16] also try to alleviate the problem of constructing summaries with overlapping sentences by defining a global objective function. Despite this high-level similarity, the objective function of [2] and [16] is significantly different from ours. For example, their objective function does not take into account at all our requirement for high coverage. Moreover, their optimization criterion, contrary to ours, is somewhat ad-hoc and does not have an intuitive interpretation.

Another line of work has focused on developing machine-learning approaches for summarization: a set of sentences is labelled by human evaluators as important for summarization and then a machine-learning algorithm is trained to extract summaries using the labelled sentences as training data, See [5, 11] for some examples of such methods. Document summarizers that use linguistic knowledge have also been developed, e.g., [17]. Our approach does not belong to either of these categories because it is unsupervised and it makes use of statistical information to generate summaries.

The proliferation of the Web has also inspired the text mining community to create hybrid systems for generating snippets in web search [19], for summarizing web sites [1], for identifying important blocks of web pages [18], and for extracting opinions from product reviews [10]. There is only a high-level similarity between these and ours since our methodology has no technical overlap with the solutions they proposed.

3 Problem definition

In this section we provide a formal definition of the DOCUMENT SUMMARIZATION problem that exactly captures the two requirements of coverage and orthogonality defined in the Introduction. We start this section by giving the necessary notation in Section 3.1. In Section 3.2 we explore the connection between different combinatorial problems and document summarization. We show that despite the high-level connection, there are certain obstacles that need to be overcome in order to find a clean combinatorial definition of the DOCUMENT SUMMARIZATION problem that captures successfully all our requirements. We given this definition in Section 3.3.

3.1 Preliminaries We consider every document as a collection of sentences $\mathcal{D} = \{s_1, \ldots, s_m\}$, and use $\mathcal{T}$ to denote the set of unique terms that appear in the document. These are the set of terms that appear in the sentences of $\mathcal{D}$ after we remove stopwords and eliminate duplicates by performing stemming. Note that we have avoided adding refinements such as term-weighting, synonyms, N-grams (all of which generally result in performance improvements) in order to better evaluate the utility of our basic framework. No matter what the building blocks of sentences, the underlying assumption throughout this paper is that the elements in $\mathcal{T}$ are important parts of the document and thus capture its main themes. In the following presentation, we only consider binary associations of terms with sentences; that is, we consider sentences as simple sets of terms. The cardinality of $\mathcal{T}$ is $|\mathcal{T}| = n$.

3.2 Combinatorial aspects of the DOCUMENT SUMMARIZATION problem In this section, our goal is to demonstrate how certain combinatorial definitions of the DOCUMENT SUMMARIZATION problem have certain shortcomings. In principle we are seeking a problem definition that captures the requirements of coverage and orthogonality. We start by giving the definition of a *cover* of a set of sentences.

Definition 1. (Cover of a set of sentences) *Given a set of sentences $S \subseteq \mathcal{D}$, we define the* cover *of the set of sentences S to be the union of the terms appearing in the sentences in S. That is,*

$$C(S) = \bigcup_{s \in S} \bigcup_{t \in s} t.$$

Constructing summaries with large $C(S)$ satisfies the coverage requirement. But how about orthogonality? One could explicitly capture the requirement of orthogonality; for example, by asking for summary $S \subseteq \mathcal{D}$ such that for every $s_i, s_j \in S$ it holds that $s_i \cap s_j = \emptyset$ and $C(S)$ is maximized. The corresponding problem definition is the following.

Problem 1. (Pairwise Independence) *Given a document consisting of m sentences $\mathcal{D} = \{s_1, \ldots, s_m\}$ and an integer k, find a set of sentences $S \subseteq \mathcal{D}$ consisting of at most k sentences ($|S| \leq k$) such that $C(S)$ is maximized and for every $s, s' \in S$ $s \cap s' = \emptyset$.*

In this case, the sentences in the summary are forced to be strictly orthogonal. In practice, summaries with absolute orthogonality and large cover are expected to rarely exist. A more reasonable requirement would be to allow for some overlap between the sentences in the summary S. Problem 2 captures exactly this need.

Problem 2. (Bounded Cover) *Given a document consisting of m sentences $\mathcal{D} = \{s_1, \ldots, s_m\}$ and an integers k and ℓ, find a set of sentences $S \subseteq \mathcal{D}$ consisting of at most k sentences ($|S| \leq k$) such that $C(S)$ is maximized and every term in the original document $\mathcal{D}$ belongs to at most ℓ sentences in S.*

Though less strict this problem definition comes with its own disadvantages. For example, what is the right value of ℓ? Or even more, should the value of ℓ be the same for every element in $\mathcal{T}$?

From the computational complexity point of view, both Problems 1 and 2 are NP-hard problems, since they can be either be mapped to known NP-hard problems, or they contain an NP-hard problem as a special case: Problem 1 is identical to the k-Unique Coverage problem (see [3]) which is known to be NP-hard. Finally, Problem 2 is also NP-hard since it is a generalization of Problem 1; Problem 2 becomes identical to Problem 1 if the value of ℓ is set to $\ell = 1$.

3.3 On a proper definition of the Document Summarization problem Overall, the solutions to Problems 1 and 2 capture to a certain extent the original requirements for summaries that exhibit high coverage and high orthogonality. However, as the preceding discussion suggested, none of these definitions leaves us completely satisfied. The question is can we do better than Problems, 1 and 2? Or is there a more subtle and general way to capture the properties of cover and orthogonality in a simple problem definition? In this section, we address this question in a positive way.

We start by expressing the orthogonality requirement in a more subtle form. Consider a summary $S \subseteq \mathcal{D}$ consisting of k sentences. We define function $f : \{0, 1, \ldots, k\} \to \mathbb{R}$, such that $f(x)$ defines the benefit we have by covering a term *exactly* x times. Intuitively, we do not have any benefit for terms that are not covered and therefore $f(0) = 0$. Moreover, orthogonality suggests that the more sentences in S a term appears in, the less important this term is and the less benefit the summary has by covering it. Formally, for all $x_1, x_2 \in \{1, \ldots, k\}$ such that $x_1 \leq x_2$, it holds that $f(x_1) \geq f(x_2)$.

Observe that function f abstracts and somehow softens the hard constraints of Problems 1 and 2. Using this abstraction we can easily define the Document Summarization problem in a much cleaner way. We start with the definition of the *gain* of a summary $S \subseteq \mathcal{D}$.

Definition 2. (Gain of a summary) *Consider a document $\mathcal{D}$ with unique terms $\mathcal{T}$ and a summary $S \subseteq \mathcal{D}$. Let S partition the elements in $\mathcal{T}$ into $\mathcal{P}(\mathcal{T}, S) = \{\mathcal{T}_0, \ldots, \mathcal{T}_{|S|}\}$, where every term $t \in \mathcal{T}_x$ appears in exactly x sentences in S. We then define the* gain *of the summary S to be*

$$G_f(S) = \sum_{x=0}^{|S|} \sum_{t \in \mathcal{T}_x} f(x) = \sum_{x=1}^{|S|} \sum_{t \in \mathcal{T}_x} f(x). \tag{3.1}$$

Note that the definition of gain depends on the f function, which can be used as a tuning mechanism towards summaries that give priority to coverage, orthogonality or both.

Given function f we can then define the Document Summarization problem as follows.

Problem 3. (Document Summarization) *Given a document described by the set of sentences $\mathcal{D}$, function f, and integer k, find $S \subseteq \mathcal{D}$, with $|S| \leq k$ such that $G_f(S)$ is maximized.*

In this paper we use two different forms of the function f.

Uniform f_u: In the case of uniform f we set that $f(0) = 0$ and $f(x) = 1$ for every $x > 0$. The **uniform** function implicitly gives higher weight to coverage than orthogonality.

Exponential f_e: In the case of exponential f we set that $f(0) = 0$ and $f(x) = \frac{1}{2^{x-1}}$ for $x > 0$. The **exponential** function imposes a relatively stronger requirement for orthogonal summaries.[1]

It is obvious that when f is the **uniform** function, then Problem 3 is identical to the k-MAXIMUM COVERAGE problem (see [8]), which is known to be NP-hard. Thus, Problem 3 is also NP-hard, since it is a generalization of an NP-hard problem. In fact, by using appropriate f functions, we can instantiate Problem 3 to Problems 1 and 2 as well. For example, if $f(0) = 0$, $f(1) = 1$ and $f(x) = 0$ for every $x > 1$, then Problem 3 becomes identical to Problem 1. Similarly for f such that $f(0) = 0$, $f(x) = 1$ for every $1 \leq x \leq \ell$ and $f(x) = 0$ for $x > \ell$ Problem 3 becomes identical to Problem 2.

Notice that the definition of Problem 3 with function f_u is at a high-level related to (in fact is a generalization of) the problem addressed in [7]. However, this connection remains only at a high-level since there was no formal problem definition provided in [7].

Weighted Document Summarization: Note that the gain function given in Equation (3.1) can be enhanced to incorporate a weight $w(t)$ for every term t. Then, the gain is alternatively computed as follows:

$$\begin{aligned} G_f^w(S) &= \sum_{x=0}^{|S|} \sum_{t \in \mathcal{T}_x} w(t) \times f(x) \\ &= \sum_{x=1}^{|S|} \sum_{t \in \mathcal{T}_x} w(t) \times f(x). \end{aligned} \tag{3.2}$$

This gain function not only depends on the function f, but also on the weighting scheme w. As before, function f is a tuning mechanism towards summaries that give priority to coverage, orthogonality or both. The weighting scheme can bias preferences towards covering some terms. For example, terms related to major concepts can be assigned higher weight in order to be included in a summary before terms associated with less important concepts. In fact, gain function G_f^w given in Equation (3.2) is a generalization of the gain function G_f given in Equation (3.1); G_f is the instance of G_f^w where the weights for all terms are set to 1, e.g., for all $t \in \mathcal{T}$, $w(t) = 1$.

[1] Constant 2 in the denominator can in principle be replaced by any other constant. In practice we have observed that for constants in the range $[2, 4]$ the results appear to be identical.

As long as the weighting scheme assigns constant weight to every term, e.g, the weights are term specific, then the above generalization does not affect any of the results presented in this paper. For simplicity of exposition we present our formulations, algorithms and results for the simple version of the gain function first, where unit weights are associated with all terms. The different weighting schemes do not seem to *significantly* affect the experimental results. This observation echoes similar previous findings, e.g., [7].

4 Algorithms

In this section we give a simple and efficient algorithm for solving Problem 3. Notice that the algorithm is as general as the problem that it solves. The algorithm is a simple greedy procedure that takes as input a document $\mathcal{D}$, integer k and function f and outputs a set $S \subseteq \mathcal{D}$ of at most k sentences. The pseudocode of this generic greedy algorithm is given in Algorithm 1.

The algorithm operates in rounds and it greedily picks sentences from the set $\mathcal{D}$. That is, it starts by picking the sentence $s_1 \in \mathcal{D}$ such that $G_f(s_1)$ is maximized. Then, among all the sentences in $\mathcal{D} \setminus s_1$, it proceeds by picking sentence $s_2 \in \{\mathcal{D} \setminus s_1\}$ such that the marginal gain by adding this sentence to the existing summary is maximized. This process continues until k sentences are picked or if no sentence with a positive marginal gain remains. We refer to this generic greedy algorithm as `Greedy`. As we have already discussed, we focus our attention on two specific f functions: the **uniform** f_u, and the **exponential** f_e. We refer to the instantiation of the greedy algorithm that uses function f_u in the evaluation of the marginal gain (Algorithm 1, line 5) as the `GreedyUniform` algorithm. Similarly, when function f_e is used instead we refer to the corresponding instantiation of the `Greedy` as `GreedyExp`.

Algorithm 1 The `Greedy` algorithm.

1: **Input:** Document $\mathcal{D}$, integer k and function f.
2: **Output:** A set of sentences $S \subseteq \mathcal{D}$ with $|S| \leq k$.
3: $C = \mathcal{D}$, $S = \emptyset$, $W = \bigcup_{s \in \mathcal{D}} s$, `MaxImpr` $= \infty$, $\ell = 0$
4: **while** $\ell \leq k$ and `MaxImpr` > 0 **do**
5: pick $s \in C$ such that $s = \arg\max_{s' \in C} G_f(S \cup s') - G_f(S)$
6: `MaxImpr` $= G_f(S \cup s) - G_f(S)$
7: **if** `MaxImpr` > 0 **then**
8: $C = C \setminus \{s\}$
9: $\ell = \ell + 1$
10: $S = S \cup \{s\}$
11: **end if**
12: **end while**

A classical result from combinatorial optimization ([15]), allows us to make the following statement:

OBSERVATION 1. *The* `GreedyUniform` *algorithm is an* $\left(1-\frac{1}{e}\right) \approx 0.67$*-approximation algorithm for Problem 3, when the gain is computed using the* **uniform** *function* f_u.

Unfortunately, we cannot make an analogous statement for the `GreedyExp` algorithm. However, we have observed that not only does `GreedyExp` give meaningful summaries in practice, but it also performs better in terms of the objective function G_{f_e}, when compared to other combinatorial algorithms with provable approximation guarantees for this specific function. For an extensive discussion on these algorithms see [3]. We omit their presentation due to lack of space and because we believe that `GreedyExp` is much simpler to use in practice. In fact, our experiments (not reported here) showed that `GreedyExp` performs better than the combinatorial algorithms in [3] in terms of the objective function G_{f_e}. Moreover, `GreedyExp` algorithm is better in terms of running time.

Algorithms for the Weighted Document Summarization: The generic `Greedy` algorithm can be easily modified to handle the weighted gain function G_f^w given in Equation (3.2). In fact, the only change that needs to be made is to use function G_f^w instead of G_f in the evaluation of the marginal gain of a sentence (Algorithm 1, line 5). We call this instance of the `Greedy` algorithm the `WeightedGreedy` algorithm. As before, depending on the function f being used we have the `WeightedGreedyUniform` and the `WeightedGreedyExp` algorithms for f_u and f_e functions respectively.

Running time: The running time of the `Greedy` algorithm is $O(kmn)$; every iteration of the `while` loop (line 4 of Algorithm 1) requires going through the remaining candidate sentences and the actual terms these sentences contain. Therefore, every iteration requires time $O(nm)$. Since the loop is repeated for at most k times, the total running time is $O(knm)$.

5 Other approaches to DOCUMENT SUMMARIZATION

In this section we describe some alternative solutions to the document-summarization problem. These solutions consider either one or both of the coverage and orthogonality requirements. We use them as a baseline for evaluating the performance of our methods in the experimental section.

Clustering: Each sentence s_i corresponds to an n-dimensional vector over the space of terms. Picking k sentences that are representative of the document and diverse of each other can also be considered as a *clustering* problem. The sentences can be clustered in k clusters and the centroid of each cluster can be considered as the representative of the sentences in the cluster. In that way k-representatives are picked. In our implementation we use standard `k-median` algorithm combined with the cosine similarity for measuring the similarity between sentences. The classical `k-median` algorithm is iterative and its running time is $O\left(I\left(kmn\right)\right)$, where I is the number of iterations of the algorithm, and the factor of n comes from the fact that the distance computation between two sentences takes $O(n)$ time.

SVD: The requirement of orthogonality between the sentences in the summary suggests that a row basis found by doing a Singular Value Decomposition (SVD) on the terms-sentence matrix D, could be a valid algorithm for solving the document summarization problem. We have implemented this approach in our system as follows: First we find an orthonormal orthogonal basis for the columns of the matrix using classical SVD algorithm. Then we use the k basis vectors that correspond to the k largest singular values as indicative representative sentences. The problem with these k vectors is that they do not correspond to any of the original sentences in the document. Thus, we map them to their closest sentence from the original document using a simple greedy heuristic. We refer to this two-step process as the `SVD` algorithm. The time requirement for computing the singular value decomposition of the $m \times n$ document matrix is $O\left(mn\left(m+n\right)\right)$. In addition to that, `SVD` needs $O(knm)$ for the greedy step; in this step it computes the distance between all the m sentences from the k picked singular vectors. Since the computation of the distance takes $O(n)$ time.[2].

Furthest: This is a simple heuristic that is motivated by the *furthest-first traversal* algorithm, for which Hochbaum and Shmoys [9] showed that it achieves a 2-approximation for the clustering formulation of p-centers. `Furthest` works as follows: it starts with a random sentence of the document and add it to the summary. Then, it picks the sentence that is the most dissimilar to the ones that has already picked. One minus the cosine similarity is used as the measure of dissimilarity between sentences. Also, the distance between a single sentence and a set of sentences is defined as the sum of the distances between the former and every element in the set. The running time of `Furthest`

[2]The running time of this algorithm can be improved by using approximate algorithms for the SVD matrix decomposition. See `www.cs.rpi.edu/~drinep/svd_exper_PCI.pdf`

is $O(k^2nm)$ since it runs in k iterations and in every iteration it computes the distances between at most m remaining sentences with at most k picked ones.

Mead: Mead [16] is a publicly available toolkit for multi-lingual summarization and evaluation [3]. The core Mead system consists of two modules: *classifier* and *reranker*. A classifier assigns a score to each sentence by considering a list of features (e.g., distance of a sentence to the centroid of the documents, position or length of the sentence, etc.). The reranker modifies the score to remove redundancy in the summary. In our experiments, we used Mead's *default-classifier* and *default-reranker*. We refer to this version of Mead as `Mead(Default)`.

For comparison purposes, we also tested Mead's *default-classifier* and the *(MMR)-reranker*, [4] which we call `Mead(MMR)`. MMR [2] is a method mostly suited for query-based and multi-document summarization. In MMR, sentences are assigned a score that is a linear combination of their relevance to a query (or for generic summaries, the centroid of the documents) and their redundancy with the sentences that have already been extracted. The weight of every component of is controlled by parameter λ; when $\lambda = 0$, MMR computes a maximal diversity ranking among the sentences; when $\lambda = 1$, it computes incrementally the standard relevance-ranked list. For our experiments we used $\lambda = 0.3$, as this was the value suggested by the MMR original paper [2].

6 Experiments

The goal of our experimental evaluation is to show that our algorithms give summaries that satisfy our original requirements of coverage and orthogonality. We show that the summaries produced by our methods exhibit much higher coverage and orthogonality than summaries produced by other methods (e.g., those described in Section 5) or even human-generated summaries. Moreover, we show that our algorithms produce summaries that have comparable scores to human and other computer-generated summaries when the evaluation is done using ROUGE's [12] precision and recall measures. Note that this is despite the fact that optimizing these measures was not our primary goal when designing our framework.

6.1 Data description For the rest of the experimental evaluation we use text data from the 2002 Document Understanding Conference (DUC) [4]. We refer to this dataset as the **DUC** dataset. The **DUC** data consists of news articles about different events ranging from natural disasters, everyday pieces of news and biographical information about individuals. At a high level the *base documents* (or simply documents) of the dataset are organized into *document clusters* (or simply clusters). The dataset consists of 60 such clusters (3 of which being disregarded due to logistic issues) and every cluster contains 5–15 documents (on average there are 10 documents per cluster). Each one of the clusters belongs in one of the following four thematic categories:

$\mathbf{C_1}$: All the documents in a cluster belonging in $\mathbf{C_1}$ refer to a single natural disaster event.

$\mathbf{C_2}$: All the documents in a cluster of category $\mathbf{C_2}$ refer to a single event (other than natural disaster event) that appeared in the news.

$\mathbf{C_3}$: All the documents in a cluster of category $\mathbf{C_3}$ refer to different distinct events of a certain type.

$\mathbf{C_4}$: Finally, all the documents in a cluster of category $\mathbf{C_4}$ may present biographical information about a single individual.

All the documents of a cluster belong in the same category, namely the category of the cluster. All four thematic categories are associated with approximately the same number of clusters (and thus approximately also the same number of documents).

For the majority of our experiments, and unless mentioned otherwise, we used the **DUC** dataset as follows: we used the 57 clusters as 57 distinct documents. We concatenated all the documents of each cluster into a single large document which we then used as input to the summarization algorithms. In this way, we obtained 57 distinct documents that are used as input to our summarization algorithms. We refer to the **DUC** dataset processed in this way as the **ClusteredDUC** dataset.

In addition to the documents and the clusters, the **DUC** dataset also contains two sets of human-generated summaries for every cluster. Each set contains two summaries of 200 and 400 terms long. This is approximately equivalent to summaries consisting of 10 and 20 sentences respectively. Each such summary was generated by humans that picked subset of the sentences in each cluster to summarize the whole cluster. We refer to these human-generated summaries as **Human1** and **Human2**. We compare those summaries with the computer-generated ones produced by our algorithms.

We would like to note that the human-generated summaries for this dataset are prepared specially for the tasks defined in DUC. It does not guarantee that these summaries satisfy our coverage and orthogonality criteria. Some experimental results presented in the following subsections also validate this claim. Therefore,

[3] `http://www.summarization.com/mead/`

[4] `http://www.summarization.com/mead/addons/scripts/mmr-reranker.perl`

experiments involving the comparison of our algorithms and human-generated summaries should only be viewed as an indicator of the general quality of our algorithms. We still pick this dataset since it is considered as a standard, publicly available, dataset for general document-summarization. Moreover, since the measures of coverage and orthogonality are new, there does not exist a standard dataset along with human-generated summaries that are specially designed for these measures.

6.2 Coverage and orthogonality of the summaries The goal of this experiment is to evaluate the summaries produced by the different algorithms: `GreedyExp`, `GreedyUniform`, `SVD`, `k-median`, `Furthest`, `Mead(Default)` and `Mead(MMR)` with respect to our original requirements of *coverage* and *orthogonality*. We use the **ClusteredDUC** dataset. The result of the experiment shows that our algorithms `GreedyExp` and `GreedyUniform` produce summaries that demonstrate both high coverage and high orthogonality.

Evaluation metrics: For this evaluation we report the following two quantities that naturally capture the notions of coverage and orthogonality.

For a summary S consisting of a number of sentences we evaluate its *coverage* as the percentage of terms from $\mathcal{T}$ that appear in the sentences in S. That is,

$$\text{Coverage}(S) = \frac{\left|\bigcup_{s \in S} s\right|}{|\mathcal{T}|}.$$

For any summary S, we have that $0 \leq \text{Coverage}(S) \leq 1$. The closer the coverage value is to 1 the better the coverage of a summary.

Similarly, we evaluate the *orthogonality* of a summary S as the average Jaccard distance between the distinct pairs of sentences that appear in the summary. Recall that the Jaccard distance between two sentences (that is, sets of terms from $\mathcal{T}$) s, s' is defined as $\text{JD}(s, s') = 1 - |s \cap s'| / |s \cup s'|$. We evaluate the orthogonality of the summary S to be

$$\text{Orthogonality}(S) = \frac{2}{|S|(|S|-1)} \sum_{s,s' \in S s \neq s'} \text{JD}(s, s').$$

For two sentences s, s', the value of $\text{JD}(s, s')$ takes values between 0 and 1, and so does $\text{Orthogonality}(S)$. The closer the value of $\text{Orthogonality}(S)$ is to 1, the more orthogonal the summary.

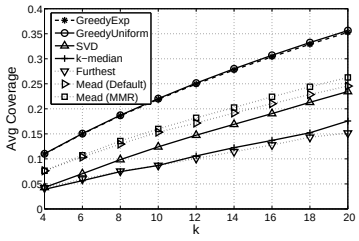
Results: Figure 1 reports the average coverage (Figure 1(a)) and the average orthogonality (Figure 1(b)) of the summaries produced by algorithms `GreedyExp`, `GreedyUniform`, `SVD`, `k-median`, `Furthest`, `Mead(Default)` and `Mead(MMR)` for $k = \{4, 6, 8, \ldots, 20\}$. The average is taken over the 57 different documents in the **ClusteredDUC** dataset that were used as input to the summarization algorithms.

With respect to Coverage (Figure 1(a)) `GreedyExp` and `GreedyUniform` produce summaries with noticeably larger Coverage than the rest of the algorithms. In fact, these two algorithms produce summaries that are approximately equivalent with respect to Coverage, and this Coverage is approximately 50% larger than the the Coverage achieved by the second best algorithms, namely `Mead(Default)`, `Mead(MMR)` and `SVD`. Finally, `k-median` and `Furthest` produce significantly lower-coverage results than the other methods.

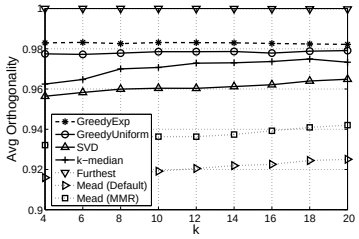
When it comes to Orthogonality (Figure 1(b)), we observe that all five algorithms produce summaries with high Orthogonality values. `GreedyExp` has slightly higher average Orthogonality value than `GreedyUniform` for all values of k; such an observation is expected given that the f function used in `GreedyExp` penalizes harsher non-orthogonal solutions. Notice that the widely used `Mead(Default)` and `Mead(MMR)` algorithms appear to have slightly smaller Orthogonality values compared to all other algorithms. Notice that `Furthest` algorithm produces summaries with Orthogonality value equal to 1 for all values of k. Although this could be considered as a good feature of the `Furthest` algorithm, when we look at the combination of the Coverage and Orthogonality values of its summaries we observe that despite their high Orthogonality value, they have significantly low Coverage. This implies that in fact `Furthest` selects sentences with small number of terms and thus does not necessarily give good summaries. This is actually true when one looks at the actual summaries produced by `Furthest`. The takeaway message from the above observation is that a summarization algorithm cannot be judged only by the Coverage or only by the Orthogonality value of the summaries it produces. The two metrics should always be considered together. This is similar to the measures of precision and recall used in the classical evaluation of information-retrieval and classification algorithms.

The results for `WeightedGreedyExp` and `WeightedGreedyUniform` are almost the same as those for their unweighted counterparts; and the variances of the values Figure 1(a) and Figure 1(b) are quite small, so for clarity of presentation, we exclude them from the plots.

6.3 Computer-generated vs. human-generated summaries The goal of this experiment is to compare

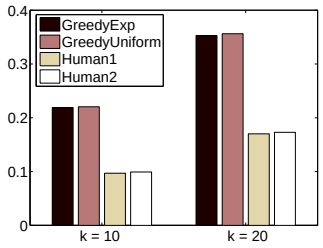


(a) Average Coverage

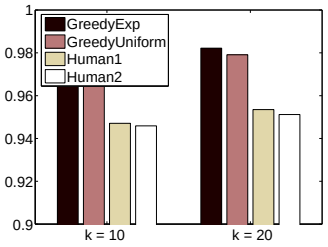


(b) Average Orthogonality

Figure 1: **ClusteredDUC** dataset; Average Coverage (Figure 1(a)) and Average Orthogonality (Figure 1(b)) of the summaries produced by the different algorithms for values of $k \in \{4, 6, 8, \ldots, 20\}$.



(a) Average Coverage



(b) Average Orthogonality

Figure 2: **ClusteredDUC** dataset; Average Coverage (Figure 2(a)) and Average Orthogonality (Figure 2(b)) of the summaries produced by `GreedyExp`, `GreedyUniform` and **Human1** and **Human2** for $k = 10$ and $k = 20$.

the computer-generated summaries to the pre-existing human-generated summaries **Human1** and **Human2**. We again use the **ClusteredDUC** dataset. The results of the experiment demonstrate that 1) summaries produced by `GreedyExp` and `GreedyUniform` have higher coverage and orthogonality than human-generated summaries; 2) summaries produced by `GreedyExp` and `GreedyUniform` are comparable to those generated by widely-used summarization systems `Mead(Default)` and `Mead(MMR)`, when ROUGE [12] is used as the evaluation metric.

Evaluation metrics 1: For this evaluation, we compare the coverage and orthogonality of computer-generated summary and human-generated summary using the same metrics defined in the previous section.

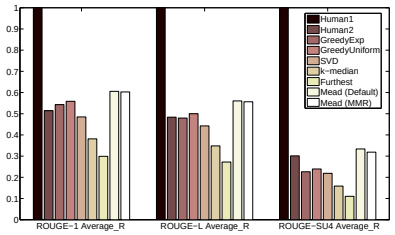
Results: Figure 2 reports the average coverage and the average orthogonality of the summaries produced by `GreedyExp` and `GreedyUniform`, as well as human-generated summaries **Human1** and **Human2**. The average is taken over the 57 different documents in the **ClusteredDUC** dataset. We use values of $k = 10$ and 20 since these are the values of k for which we have human-generated summaries. From the figures we can see that `GreedyExp` and `GreedyUniform` produce summaries with noticeably larger coverage and orthogonality.

Evaluation metrics 2: For this evaluation, we compare the similarity of computer-generated summaries and human-generated summaries using ROUGE[5], a standard package adopted by Document Understanding Conference (DUC).[6] In particular, we consider the following three ROUGE metrics: 1) ROUGE-N (for $N = 1$), which measures the co-occurrence of N-grams in a computer-generated summary and a human-generated summary; 2) ROUGE-L, which measures the longest common subsequences in a computer-generated summary and a human-generated summary; 3) ROUGE-SU4, which measures the co-occurrence of skip-bigrams and unigrams with a maximum skip distance of 4. We refer interested readers to [12] for detailed mathematical definitions of these metrics.

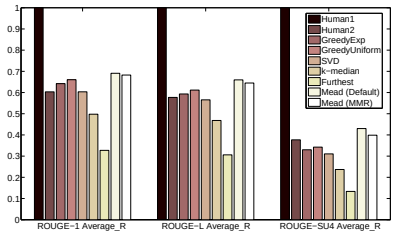
Results: We run ROUGE 1.5.5 with the following parameters `./ROUGE-1.5.5.pl -n 2 -2 4 -U -m -c 95`. Figures 3(a) and 3(b) show the average *recall* scores for the computer-generated summaries when **Human1** is used as the reference summary. Similarly, Figures 3(c) and 3(d) show the average *recall* scores for the computer-generated summaries when **Human2** is

[5] `http://berouge.com`

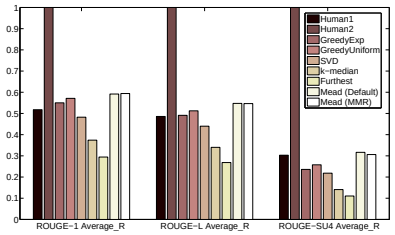
[6] `http://duc.nist.gov`



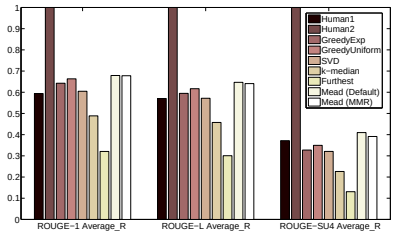
(a) $k = 10$, **Human1** used as the reference.



(b) $k = 20$, **Human1** used as the reference.

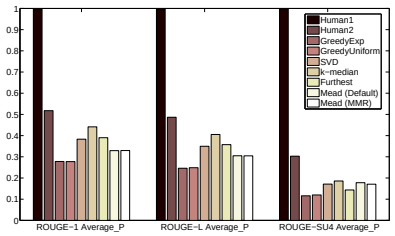


(c) $k = 10$, **Human2** used as the reference.

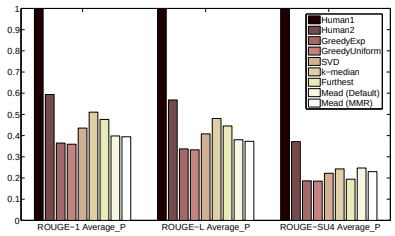


(d) $k = 20$, **Human2** used as the reference.

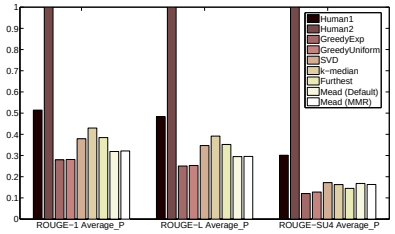
Figure 3: **ClusteredDUC** dataset; Average Recall Scores of ROUGE-1, ROUGE-L, ROUGE-SU4 when $k = 10$ and $k = 20$.



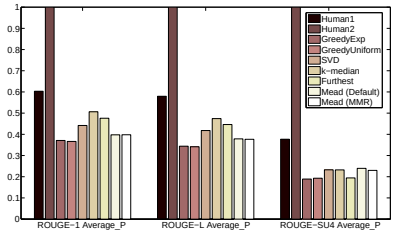
(a) $k = 10$, **Human1** used as the reference.



(b) $k = 20$, **Human1** used as the reference.



(c) $k = 10$, **Human2** used as the reference.



(d) $k = 20$, **Human2** used as the reference.

Figure 4: **ClusteredDUC** dataset; Average Precision Scores of ROUGE-1, ROUGE-L, ROUGE-SU4 when $k = 10$ and $k = 20$.

used as reference. Among the computer-generated summaries the following trend is obvious: the summaries produced by `Mead(Default)` and `Mead(MMR)` have the highest recall scores; after all `Mead(Default)` and `Mead(MMR)` were generated to optimize those scores. In most of the cases though, the summaries produced by `GreedyExp`, `GreedyUniform` have scores very close to those generated by `Mead(Default)` and `Mead(MMR)`. On

the other hand, the summaries of SVD, k-median and Furthest are relatively more distant from the human-generated summaries. At this point, another indicative observation is in place: the similarity between the two human-generated summaries is not significantly larger than (in fact in many cases is smaller than) the similarity between computer-generated and human-generated summaries. This observation illustrates the overall difficulty of document summarization and points out the subjectiveness involved in the task.

Figures 4(a) and 4(b) show the average *precision* scores for the computer-generated summaries when **Human1** is used as the reference summary. Similarly, Figures 4(c) and 4(d) show the average *precision* scores for the computer-generated summaries when **Human2** is used as reference. It is interesting to observe that, among the computer-generated summaries, those produced by GreedyExp, GreedyUniform have very low precision scores. Contrast this with the high-precision scores attained by k-median and Furthest. This finding contradicts our intuition that Furthest produces mostly low-quality summaries that consist of (rather incoherent) small sentences. Recall though, that the definition of precision in ROUGE is simply the number of entities (e.g., N-grams) of the computer-generated summary that are matched by the reference (e.g., human) summary, divided by the number of entities in the computer-generated summary. Using this definition, summaries with small number of words are going to exhibit very high precision scores. As an extreme example, consider a summary that simply contains the top-k most frequent words in the document. In most of the cases, such a summary will have a precision score close to 1. Therefore, high precision scores alone do not mean high-quality summaries. The performance of each algorithm should be judged collectively taking into account all evaluation metrics.

To sum up, the high coverage and orthogonality scores of GreedyExp and GreedyUniform, as well as their low ROUGE precision scores show that these algorithms can produce summaries that capture many different aspects of a document while other algorithms cannot. The high ROUGE recall scores of GreedyExp and GreedyUniform indicate that they can still produce summaries comparable to those generated by widely-used summarization systems such as Mead(Default) and Mead(MMR) when ROUGE is used as the evaluation metric.

6.4 Topic identification In the following two experiments, we want to explore the ability of different algorithms to create summaries that pinpoint different aspects of the input document. In the unweighted topic coverage, we are only interested on whether a specific theme touched in the document appears in the summary. In the weighted case, we are also interested in the proportions (in terms of number of sentences) in which different themes appear in the documents and the summaries.

6.4.1 Unweighted topic identification The main result of this experiment is that our algorithms, GreedyExp and GreedyUniform are the ones that successfully cover all possible aspects of a document, even when they are restricted to construct summaries with small number of sentences.

Dataset: For this experiment we use the base documents of the **DUC** datasets to compose *synthetic* documents, which we then summarize. Every synthetic document is a concatenation of four base documents from the original **DUC** dataset; the i-th base document that participates in this synthesis (with $i = \{1, 2, 3, 4\}$), is selected uniformly at random amongst the base documents of the i-th category. In this way, every one of the four categories is represented in every synthetic document. We generate 100 such synthetic documents by repeating the above process 100 times. We refer to the dataset we generate in such a way as the **Mixed-DUC** dataset. The size of the synthetic documents in the **MixedDUC** dataset are between 15Kb and 60Kb, and each such document has approximately 120–500 sentences.

Metrics: A synthetic document generated as above can be represented by a 4-dimensional vector $D = (1, 1, 1, 1)$, where $D(i) = 1$ denotes the fact that a document from category 1 has been used in the synthesis of D. A summary S of D can also be represented by a 4-dimensional vector S such that $S(i) = 1$ if a sentence from category $\mathbf{C}_i$ is included in the summary. Otherwise $S(i) = 0$. We define the *topical coverage* of summary S with respect to document D to be the normalized dot product of these two vectors.

$$\textsc{Tc}\,(S|D) \quad = \quad \frac{1}{4} S \cdot D.$$

The higher the value of the topic coverage of a summary (the closer it is to 1), the better this summary is in capturing the different aspects of the input document.

Results: Figure 5 shows the average topic coverage of the summaries obtained by GreedyExp, GreedyUniform, SVD, k-median, Furthest, Mead(Default) and Mead(MMR) algorithms for values of $k \in \{4, \ldots, 20\}$. The average is taken over all the 100 documents of the

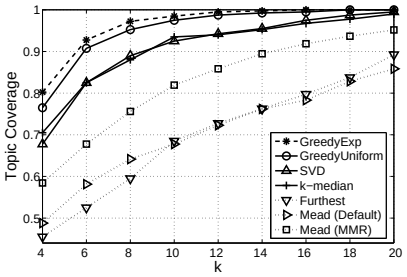


Figure 5: **MixedDUC** dataset; topic coverage of `GreedyExp`, `GreedyUniform`, `SVD`, `k-median`, `Mead(Default)`, `Mead(MMR)` and `Furthest` algorithms for summaries consisting of $k = \{4, \ldots, 20\}$ sentences.

MixedDUC dataset. The variances are quite small for all algorithms so we do not report them. From the figure we can see that the summaries produced by `GreedyExp` and `GreedyUniform` algorithms exhibit almost identical topic coverage, which is also higher than the topic coverage exhibited by the summaries of the other algorithms. The second best pair of algorithms with respect to the TC measure are the `SVD` and `k-median`, while `Mead(MMR)` is slightly worse than the latter two. `Mead(Default)` and `Furthest` produce significantly low topic-coverage summaries. Observe that `GreedyExp` and `GreedyUniform` algorithms can cover almost all 4 categories with summaries consisting just of 6 sentences. Notice that the results of the `WeightedGreedyExp` and `WeightedGreedyUniform` were almost identical to the results for `GreedyExp` and `GreedyUniform`. Thus, these algorithms were excluded from the plots for the sake of clarity.

6.4.2 Weighted topic identification In this last experiment we are primarily interested in comparing the performance of the different `Greedy` algorithms with their `WeightedGreedy` counterparts. For the `WeightedGreedy` algorithms we have used a frequency-based weighing scheme. That is, every term in $\mathcal{T}$ was assigned weight equal to its frequency in the document.

Recall that `WeightedGreedy` algorithms with frequency-based weighting schemes are expected to be more sensitive to the proportion of sentences from different themes that appear in the document. Therefore, we tailor our data-generation and result-evaluation process to be able to evaluate this sensitivity. Our conclusion is that `WeightedGreedyExp` and `WeightedGreedyUniform` performs slightly better than their unweighted counterparts when k is small; when k gets larger, the four algorithms perform almost equally well.

Datasets: We form synthetic documents from base documents as before. However, here every synthetic document D is a concatenation of X documents from category $\mathbf{C}_i$ and 1 document from all other categories $\mathbf{C}_j$ with $i \neq j$ and $1 \leq i, j \leq 4$. We set X to take values in $\{1, 2, \ldots, 10\}$ and for any given value of X we generate 10 synthetic documents. That is we generate a total of 100 synthetic documents. For a fixed value of S, we refer to the dataset we generate in such a way as the **MixedDUC**-X dataset. The size of the synthetic documents in the **MixedDUC**-X dataset are between 15Kb and 64Kb, and each such document has approximately 120–700 sentences.

Metrics: A synthetic document D can again be represented by a 4-dimensional vector. But now, the i-th element of the vector takes a real value that is the percentage of sentences in D that come from a document in category $\mathbf{C}_i$. Summaries also have the same vector representation as in Section 6.4.1 but here the value of $S(i)$ gives the percentage of the sentences in S that come from a document in category $\mathbf{C}_i$. We define the *weighted topical coverage* of summary S with respect to document D to be the cosine similarity between vectors D and S. That is,

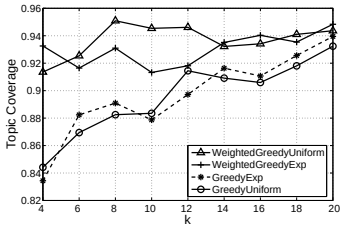
$$\text{WTC}(S|D) \quad = \quad \frac{S \cdot D}{|S||D|}.$$

The higher the value of the weighted topic coverage of a summary, the better this summary is in capturing the different aspects of the input document.

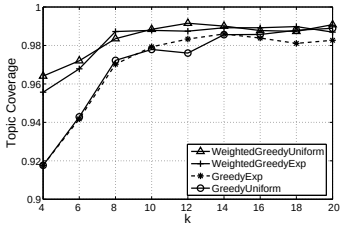
Results: Figure 6 shows the average topic coverage of the summaries obtained by `WeightedGreedyUniform`, `WeightedGreedyExp`, `GreedyExp` and `GreedyUniform` algorithms for values of $k \in \{4, \ldots, 20\}$. The average is taken over all the 10 documents of the **MixedDUC**-X dataset. We show here the results for $X = 2$ and $X = 8$, but the results for the other values of X look similar. The variances for all the cases are quite small, so we do not report them. From the figure we can see that the summaries produced by `WeightedGreedyExp` and `WeightedGreedyUniform` algorithms exhibit slightly better weighted topic coverage when k is small; as k increases, both weighted and unweighted algorithms perform almost equally well.

7 Conclusions

In this paper we provided a new combinatorial flavor to the document summarization problem. Given a document we defined the problem of summarizing it as the problem of picking k sentences from the original docu-



(a) **MixedDUC**-2 dataset; weighted topic coverage



(b) **MixedDUC**-8 dataset; weighted topic coverage

Figure 6: **MixedDUC**-X datasets, for $X = 2, 8$; weighted topic coverage of `WeightedGreedyExp`, `WeightedGreedyUniform`, `GreedyExp` and `GreedyUniform` algorithms for summaries consisting of $k = \{4, \ldots, 20\}$ sentences.

ment, such that the constructed summary exhibits two key properties: coverage and orthogonality. We captured these two requirements in a simple combinatorial formulation of the problem and presented simple and efficient algorithms for solving it. In a wide set of experiments on real document data we showed that our methods, despite their simplicity, work extremely well in practice.

The takeaway message from this paper, is that simple combinatorial definitions and algorithms for information-retrieval tasks seem to work well in practice and should not be overlooked. In the future we plan to combine our approach with other domain-specific knowledge in order to further improve our results. Moreover, an interesting direction of future research is in the definition of other text-mining and information-retrieval tasks as simple combinatorial problems.

References

[1] E. Amitay and C. Paris. Automatically summarising web sites - is there a way around it? In *CIKM*, pages 173–179, 2000.

[2] J. G. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336, 1998.

[3] E. D. Demaine, M. T. Hajiaghayi, U. Feige, and M. R. Salavatipour. Combination can be hard: approximability of the unique coverage problem. In *SODA*, pages 162–171, 2006.

[4] Document Understanding Conference (DUC). www-nlpir.nist.gov/projects/duc/guidelines/2002.html, 2002.

[5] J. Goldstein, G. M. Ciany, and J. G. Carbonell. Genre identification and goal-focused summarization. In *CIKM*, pages 889–892, 2007.

[6] J. Goldstein, M. Kantrowitz, V. O. Mittal, and J. G. Carbonell. Summarizing text documents: Sentence selection and evaluation metrics. In *SIGIR*, pages 121–128, 1999.

[7] Y. Gong and X. Liu. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR*, pages 19–25, 2001.

[8] D. Hochbaum, editor. *Approximation algorithms for NP-hard problems*. PWS Publishing Company, 1997.

[9] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

[10] M. Hu and B. Liu. Opinion extraction and summarization on the web. In *AAAI*, 2006.

[11] J. Kupiec, J. O. Pedersen, and F. Chen. A trainable document summarizer. In *SIGIR*, pages 68–73, 1995.

[12] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004*, Barcelona, Spain, 2004.

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

[14] I. Mani and M. T. Maybury. *Advances in Automatic Text Summarization*. The MIT Press, 1999.

[15] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[16] D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhang. MEAD - a platform for multidocument multilingual text summarization. In *LREC 2004*, Lisbon, Portugal, May 2004.

[17] B. Schiffman, I. Mani, and K. J. Concepcion. Producing biographical summaries: combining linguistic knowledge with corpus statistics. In *ACL*, pages 458–465, 2001.

[18] R. Song, H. Liu, J.-R. Wen, and W.-Y. Ma. Learning block importance models for web pages. In *WWW*, pages 203–211, 2004.

[19] A. Turpin, Y. Tsegay, D. Hawking, and H. E. Williams. Fast generation of result snippets in web search. In *SIGIR*, pages 127–134, 2007.