Summarization Through Submodularity and Dispersion

Anirban Dasgupta
Yahoo! Labs
Sunnyvale, CA 95054
anirban@yahoo-inc.com

Ravi Kumar
Google
Mountain View, CA 94043
tintin@google.com

Sujith Ravi
Google
Mountain View, CA 94043
sravi@gooogle.com

Abstract

We propose a new optimization framework for summarization by generalizing the submodular framework of (Lin and Bilmes, 2011). In our framework the summarization desideratum is expressed as a sum of a submodular function and a non-submodular function, which we call *dispersion*; the latter uses inter-sentence dissimilarities in different ways in order to ensure non-redundancy of the summary. We consider three natural dispersion functions and show that a greedy algorithm can obtain an approximately optimal summary in all three cases. We conduct experiments on two corpora—DUC 2004 and user comments on news articles—and show that the performance of our algorithm outperforms those that rely only on submodularity.

1 Introduction

Summarization is a classic text processing problem. Broadly speaking, given one or more documents, the goal is to obtain a concise piece of text that contains the most salient points in the given document(s). Thanks to the omnipresent information overload facing all of us, the importance of summarization is gaining; semi-automatically summarized content is increasingly becoming user-facing: many newspapers equip editors with automated tools to aid them in choosing a subset of user comments to show. Summarization has been studied for the past in various settings—a large single document, multiple documents on the same topic, and user-generated content.

Each domain throws up its own set of idiosyncrasies and challenges for the summarization task. On one hand, in the multi-document case (say, different news reports on the same event), the text is often very long and detailed. The precision/recall requirements are higher in this domain and a semantic representation of the text might be needed to avoid redundancy. On the other hand, in the case of user-generated content (say, comments on a news article), even though the text is short, one is faced with a different set of problems: volume (popular articles generate more than 10,000 comments), noise (most comments are vacuous, linguistically deficient, and tangential to the article), and redundancy (similar views are expressed by multiple commenters). In both cases, there is a delicate balance between choosing the salient, relevant, popular, and diverse points (e.g., sentences) versus minimizing syntactic and semantic redundancy.

While there have been many approaches to automatic summarization (see Section 2), our work is directly inspired by the recent elegant framework of (Lin and Bilmes, 2011). They employed the powerful theory of submodular functions for summarization: submodularity embodies the "diminishing returns" property and hence is a natural vocabulary to express the summarization desiderata. In this framework, each of the constraints (relevance, redundancy, etc.) is captured as a submodular function and the objective is to maximize their sum. A simple greedy algorithm is guaranteed to produce an approximately optimal summary. They used this framework to obtain the best results on the DUC 2004 corpus.

Even though the submodularity framework is quite general, it has limitations in its expressivity. In particular, it cannot capture redundancy constraints that depend on pairwise dissimilarities between sentences. For example, a natural constraint on the summary is that the sum or the minimum of pairwise dissimilarities between sentences chosen in the summary should be maximized; this, unfortunately, is not a submodular function. We call functions that depend on inter-sentence pair-

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 1014–1022,
Sofia, Bulgaria, August 4-9 2013.

wise dissimilarities in the summary as *dispersion* functions. Our focus in this work is on significantly furthering the submodularity-based summarization framework to incorporate such dispersion functions.

We propose a very general graph-based summarization framework that combines a submodular function with a non-submodular dispersion function. We consider three natural dispersion functions on the sentences in a summary: sum of all-pair sentence dissimilarities, the weight of the minimum spanning tree on the sentences, and the minimum of all-pair sentence dissimilarities. These three functions represent three different ways of using the sentence dissimilarities. We then show that a greedy algorithm can obtain approximately optimal summary in each of the three cases; the proof exploits some nice combinatorial properties satisfied by the three dispersion functions. We then conduct experiments on two corpora: the DUC 2004 corpus and a corpus of user comments on news articles. On DUC 2004, we obtain performance that matches (Lin and Bilmes, 2011), without any serious parameter tuning; note that their framework does not have the dispersion function. On the comment corpus, we outperform their method, demonstrating that value of dispersion functions. As part of our methodology, we also use a new structured representation for summaries.

2 Related Work

Automatic summarization is a well-studied problem in the literature. Several methods have been proposed for single- and multi-document summarization (Carbonell and Goldstein, 1998; Conroy and O'Leary, 2001; Takamura and Okumura, 2009; Shen and Li, 2010).

Related concepts have also been used in several other scenarios such as query-focused summarization in information retrieval (Daumé and Marcu, 2006), microblog summarization (Sharifi et al., 2010), event summarization (Filatova, 2004), and others (Riedhammer et al., 2010; Qazvinian et al., 2010; Yatani et al., 2011).

Graph-based methods have been used for summarization (Ganesan et al., 2010), but in a different context—using paths in graphs to produce very short abstractive summaries. For a detailed survey on existing automatic summarization techniques and other related topics, see (Kim et al., 2011; Nenkova and McKeown, 2012).

3 Framework

In this section we present the summarization framework. We start by describing a generic objective function that can be widely applied to several summarization scenarios. This objective function is the sum of a monotone submodular *coverage* function and a non-submodular *dispersion* function. We then describe a simple greedy algorithm for optimizing this objective function with provable approximation guarantees for three natural dispersion functions.

3.1 Preliminaries

Let C be a collection of texts. Depending on the summarization application, C can refer to the set of documents (e.g., newswire) related to a particular topic as in standard summarization; in other scenarios (e.g., user-generated content), it is a collection of comments associated with a news article or a blog post, etc. For each document $c \in C$, let $S(c)$ denote the set of sentences in c. Let $U = \cup_{c \in C} S(c)$ be the universe of all sentences; without loss of generality, we assume each sentence is unique to a document. For a sentence $u \in U$, let $C(u)$ be the document corresponding to u.

Each $u \in U$ is associated with a *weight* $w(u)$, which might indicate, for instance, how similar u is to the main article (and/or the query, in query-dependent settings). Each pair $u, v \in U$ is associated with a *similarity* $s(u, v) \in [0, 1]$. This similarity can then be used to define an inter-sentence distance $d(\cdot, \cdot)$ as follows: let $d'(u, v) = 1 - s(u, v)$ and define $d(u, v)$ to be the shortest path distance from u to v in the graph where the weight of each edge (u, v) is $d'(u, v)$. Note that $d(\cdot, \cdot)$ is a metric unlike $d'(\cdot, \cdot)$, which may not be a metric. (In addition to being intuitive, $d(\cdot, \cdot)$ being a metric helps us obtain guarantees on the algorithm's output.) For a set S, and a point $u \notin S$, define $d(u, S) = \min_{v \in S} d(u, v)$.

Let $k > 0$ be fixed. A *summary* of U is a subset $S \subseteq U, |S| = k$. Our aim is to find a summary that maximizes

$$f(S) = g(S) + \delta h(S), \tag{1}$$

where $g(S)$ is the *coverage* function that is non-negative, monotone, and submodular[1], $h(S)$ is a

[1] A function $f : U \to \Re$ is *submodular* if for every

dispersion function, and $\delta \geq 0$ is a parameter that can be used to scale the range of $h(\cdot)$ to be comparable to that of $g(\cdot)$.

For two sets S and T, let P be the set of unordered pairs $\{u, v\}$ where $u \in S$ and $v \in T$. Our focus is on the following dispersion functions: the *sum* measure $h_{\mathrm{s}}(S,T) = \sum_{\{u,v\}\in P} d(u,v)$, the *spanning tree* measure $h_{\mathrm{t}}(S,T)$ given by the cost of the minimum spanning tree of the set $S \cup T$, and the *min* measure $h_{\mathrm{m}}(S,T) = \min_{\{u,v\}\in P} d(u,v)$. Note that these functions span from considering the entire set of distances in S to considering only the minimum distance in S; also it is easy to construct examples to show that none of these functions is submodular. Define $h_\star(u,S) = h_\star(\{u\},S)$ and $h_\star(S) = h_\star(S,S)$.

Let O be the optimal solution of the function f. A summary $\tilde{S}$ is a γ-*approximation* if $f(\tilde{S}) \geq \gamma f(O)$.

3.2 Algorithm

Maximizing (1) is NP-hard even if $\delta = 0$ or if $g(\cdot) = 0$ (Chandra and Halldórsson, 2001). For the special case $\delta = 0$, since $g(\cdot)$ is submodular, a classical greedy algorithm obtains a $(1 - 1/e)$-approximation (Nemhauser et al., 1978). But if $\delta > 0$, since the dispersion function $h(\cdot)$ is not submodular, the combined objective $f(\cdot)$ is not submodular as well. Despite this, we show that a simple greedy algorithm achieves a provable approximation factor for (1). This is possible due to some nice structural properties of the dispersion functions we consider.

Algorithm 2 Greedy algorithm, parametrized by the dispersion function h; here, U, k, g, δ are fixed.

$S_0 \leftarrow \emptyset$; $i \leftarrow 0$
for $i = 0, \ldots, k-1$ **do**
  $v \leftarrow \arg\max_{u \in U \setminus S_i} g(S_i + u) + \delta h(S_i + u)$
  $S_{i+1} \leftarrow S_i \cup \{v\}$
end for

3.3 Analysis

In this section we obtain a provable approximation for the greedy algorithm. First, we show that a greedy choice is well-behaved with respect to the dispersion function $h_\cdot(\cdot)$.

Lemma 1. *Let O be any set with $|O| = k$. If S is such that $|S| = \ell < k$, then*
(i) $\sum_{u\in O\setminus S} h_{\mathrm{s}}(u,S) \geq |O \setminus S| \frac{\ell h_{\mathrm{s}}(O)}{k(k-1)}$;
(ii) $\sum_{u\in O\setminus S} d(u,S) \geq \frac{1}{2} h_{\mathrm{t}}(O) - h_{\mathrm{t}}(S)$*; and*
(iii) there exists $u \in O \setminus S$ such that $h_{\mathrm{m}}(u,S) \geq h_{\mathrm{m}}(O)/2$.

Proof. The proof for (i) follows directly from Lemma 1 in (Borodin et al., 2012).

To prove (ii) let T be the tree obtained by adding all points of $O \setminus S$ directly to their respective closest points on the minimum spanning tree of S. T is a spanning tree, and hence a Steiner tree, for the points in set $S \cup O$. Hence, $\mathrm{cost}(T) = h_{\mathrm{t}}(S) + \sum_{u\in O\setminus S} d(u,S)$. Let $\mathrm{smt}(S)$ denote the cost of a minimum Steiner tree of S. Thus, $\mathrm{cost}(T) \geq \mathrm{smt}(O \cup S)$. Since a Steiner tree of $O \cup S$ is also a Steiner tree of O, $\mathrm{smt}(O \cup S) \geq \mathrm{smt}(O)$. Since this is a metric space, $\mathrm{smt}(O) \geq \frac{1}{2} h_{\mathrm{t}}(O)$ (see, for example, (Cieslik, 2001)). Thus,

$$h_{\mathrm{t}}(S) + \sum_{u\in O\setminus S} d(u,S) \geq \frac{1}{2} h_{\mathrm{t}}(O)$$

$$\Rightarrow \sum_{u\in O\setminus S} d(u,S) \geq \frac{1}{2} h_{\mathrm{t}}(O) - h_{\mathrm{t}}(S).$$

To prove (iii), let $O = \{u_1, \ldots, u_k\}$. By definition, for every $i \neq j$, $d(u_i, u_j) \geq h_{\mathrm{m}}(O)$. Consider the (open) ball B_i of radius $h_{\mathrm{m}}(O)/2$ around each element u_i. By construction for each i, $B_i \cap O = \{u_i\}$ and for each pair $i \neq j$, $B_i \cap B_j = \emptyset$. Since $|S| < k$, and there are k balls B_i, there exists $k - \ell$ balls B_i such that $S \cap B_i = \emptyset$, proving (iii). □

We next show that the tree created by the greedy algorithm for $h = h_{\mathrm{t}}$ is not far from the optimum.

Lemma 2. *Let $u_1, \ldots, u_k$ be a sequence of points and let $S_i = \{u_j, j \leq i\}$. Then, $h_{\mathrm{t}}(S_k) \geq {}^1/_{\log k} \sum_{2\leq j\leq k} d(u_j, S_{j-1})$.*

Proof. The proof follows by noting that we get a spanning tree by connecting each u_i to its closest point in S_{i-1}. The cost of this spanning tree is $\sum_{2\leq j\leq k} d(u_j, S_{j-1})$ and this tree is also the result of the greedy algorithm run in an online fashion on the input sequence $\{u_1, \ldots, u_k\}$. Using the result of (Imase and Waxman, 1991), the competitive ratio of this algorithm is $\log k$, and hence the proof. □

We now state and prove the main result about the quality of approximation of the greedy algorithm.

$A, B \subseteq U$, we have $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.

Theorem 3. *For $k > 1$, there is a polynomial-time algorithm that obtains a γ-approximation to $f(S)$, where $\gamma = 1/2$ for $h = h_{\mathrm{s}}$, $\gamma = 1/4$ for $h = h_{\mathrm{m}}$, and $\gamma = 1/3 \log k$ for $h = h_{\mathrm{t}}$.*

Proof. For h_{s} and h_{t}, we run Algorithm 1 using a new dispersion function h', which is a slightly modified version of h. In particular, for $h = h_{\mathrm{s}}$, we use $h'(S) = 2h_{\mathrm{s}}(S)$. For $h = h_{\mathrm{t}}$, we abuse notation and define h' to be a function over an ordered set $S = \{u_1, \ldots, u_k\}$ as follows: $h'(S) = \sum_{j \leq |S|} d(u_j, S_{j-1})$, where $S_{j-1} = \{u_1, \ldots, u_{j-1}\}$. Let $f'(S) = g(S) + \delta h'(S)$.

Consider the ith iteration of the algorithm. By the submodularity of $g(\cdot)$,

$$\begin{aligned} &\sum_{u \in O \setminus S_i} g(S_i \cup \{u\}) - g(S_i) \qquad (2) \\ &\geq \quad g(O \cup S_i) - g(S_i) \geq g(O) - g(S_k), \end{aligned}$$

where we use monotonicity of $g(\cdot)$ to infer $g(O \cup S_i) \geq g(O)$ and $g(S_i) \leq g(S_k)$.

For $h = h_{\mathrm{s}}$, the proof follows by Lemma 1(i) and by Theorem 1 in (Borodin et al., 2012).

For h_{t}, using the above argument of submodularity and monotonicity of g, and the result from Lemma 1(ii), we have

$$\begin{aligned} &\sum_{u \in O \setminus S_i} g(S_i \cup u) - g(S_i) + \delta d(u, S_i) \\ &\geq \quad g(O) - g(S_i) + \delta(h_{\mathrm{t}}(O)/2 - h_{\mathrm{t}}(S_i)) \\ &\geq \quad (g(O) + \delta h_{\mathrm{t}}(O)/2) - (g(S_i) + \delta h_{\mathrm{t}}(S_i)) \\ &\geq \quad f(O)/2 - (g(S_i) + \delta h_{\mathrm{t}}(S_i)). \end{aligned}$$

Also, $h_{\mathrm{t}}(S_i) \leq 2\ \mathrm{smt}(S_i)$ since this is a metric space. Using the monotonicity of the Steiner tree cost, $\mathrm{smt}(S_i) \leq \mathrm{smt}(S_k) \leq h_{\mathrm{t}}(S_k)$. Hence, $h_{\mathrm{t}}(S_i) \leq 2h_{\mathrm{t}}(S_k)$. Thus,

$$\begin{aligned} &\sum_{u \in O \setminus S_i} g(S_i \cup u) - g(S_i) + \delta d(u, S_i) \\ &\geq \quad f(O)/2 - (g(S_i) + \delta h_{\mathrm{t}}(S_i)) \\ &\geq \quad f(O)/2 - (g(S_k) + 2\delta h_{\mathrm{t}}(S_k)) \\ &\geq \quad f(O)/2 - 2f(S_k). \qquad (3) \end{aligned}$$

By the greedy choice of u_{i+1},

$$\begin{aligned} &f'(S_i \cup u_{i+1}) - f'(S_i) \\ &= \quad g(S_i \cup u_{i+1}) - g(S_i) + \delta d(u_{i+1}, S_i) \\ &\geq \quad (f(O)/2 - 2f(S_k))/|O \setminus S_i| \\ &\geq \quad \frac{1}{k}(f(O)/2 - 2f(S_k)). \end{aligned}$$

Summing over all $i \in [1, k-1]$,

$$f'(S_k) \geq (k-1)/k (f(O)/2 - 2f(S_k)). \qquad (4)$$

Using Lemma 2 we obtain

$$\begin{aligned} f(S_k) = g(S_k) + \delta h_{\mathrm{t}}(S_k) &\geq \frac{f'(S_k)}{\log k} \\ &\geq \quad \frac{1 - 1/k}{\log k}(f(O)/2 - 2f(S_k)). \end{aligned}$$

By simplifying, we obtain $f(S_k) \geq f(O)/3 \log k$.

Finally for h_{m}, we run Algorithm 1 twice: once with g as given and $h \equiv 0$, and the second time with $g \equiv 0$ and $h \equiv h_{\mathrm{m}}$. Let S_g and S_h be the solutions in the two cases. Let O_g and O_h be the corresponding optimal solutions. By the submodularity and monotonicity of $g(\cdot)$, $g(S_g) \geq (1 - 1/e) g(O_g) \geq g(O_g)/2$. Similarly, using Lemma 1(iii), $h_{\mathrm{m}}(S_h) \geq h_{\mathrm{m}}(O_h)/2$ since in any iteration $i < k$ we can choose an element u_{i+1} such that $h_{\mathrm{m}}(u_{i+1}, S_i) \geq h_{\mathrm{m}}(O_h)/2$. Let $S = \arg\max_{X \in \{S_g, S_h\}} f(X)$. Using an averaging argument, since g and h_{m} are both non-negative,

$$f(X) \geq (f(S_g) + f(S_h))/2 \geq (g(O_g) + \delta h_{\mathrm{m}}(O_h))/4.$$

Since by definition $g(O_g) \geq g(O)$ and $h_{\mathrm{m}}(O_h) \geq h_{\mathrm{m}}(O)$, we have a $1/4$-approximation. □

3.4 A universal constant-factor approximation

Using the above algorithm that we used for h_{m}, it is possible to give a universal algorithm that gives a constant-factor approximation to each of the above objectives. By running the Algorithm 1 once for $g \equiv 0$ and next for $h \equiv 0$ and taking the best of the two solutions, we can argue that the resulting set gives a constant factor approximation to f. We do not use this algorithm in our experiments, as it is oblivious of the actual dispersion functions used.

4 Using the Framework

Next, we describe how the framework described in Section 3 can be applied to our tasks of interest, i.e., summarizing documents or user-generated content (in our case, comments). First, we represent the elements of interest (i.e., sentences within comments) in a structured manner by using dependency trees. We then use this representation to

generate a graph and instantiate our summarization objective function with specific components that capture the desiderata of a given summarization task.

4.1 Structured representation for sentences

In order to instantiate the summarization graph (nodes and edges), we first need to model each sentence (in multi-document summarization) or comment (i.e., set of sentences) as nodes in the graph. Sentences have been typically modeled using standard ngrams (unigrams or bigrams) in previous summarization work. Instead, we model sentences using a structured representation, i.e., its syntax structure using dependency parse trees. We first use a dependency parser (de Marneffe et al., 2006) to parse each sentence and extract the set of dependency relations associated with the sentence. For example, the sentence "I adore tennis" is represented by the dependency relations (nsubj: adore, I) and (dobj: adore, tennis).

Each sentence represents a single node u in the graph (unless otherwise specified) and is comprised of a set of dependency relations (or ngrams) present in the sentence. Furthermore, the edge weights $s(u, v)$ represent pairwise similarity between sentences or comments (e.g., similarity between views expressed in different comments). The edge weights are then used to define the inter-sentence distance metric $d(u, v)$ for the different dispersion functions. We identify similar views/opinions by computing semantic similarity rather than using standard similarity measures (such as cosine similarity based on exact lexical matches between different nodes in the graph). For each pair of nodes (u, v) in the graph, we compute the semantic similarity score (using WordNet) between every pair of dependency relation (rel: a, b) in u and v as:

$$s(u,v) = \sum_{\substack{\mathrm{rel}_i \in u, \mathrm{rel}_j \in v \\ \mathrm{rel}_i = \mathrm{rel}_j}} \mathrm{WN}(a_i, a_j) \times \mathrm{WN}(b_i, b_j),$$

where rel is a relation type (e.g., nsubj) and a, b are the two arguments present in the dependency relation (b does not exist for some relations). $\mathrm{WN}(w_i, w_j)$ is defined as the WordNet similarity score between words w_i and w_j.[2] The edge weights are then normalized across all edges in the graph.

This allows us to perform approximate matching of syntactic treelets obtained from the dependency parses using semantic (WordNet) similarity. For example, the sentences "I adore tennis" and "Everyone likes tennis" convey the same view and should be assigned a higher similarity score as opposed to "I hate tennis". Using the syntactic structure along with semantic similarity helps us identify useful (valid) nuggets of information within comments (or documents), avoid redundancies, and identify similar views in a semantic space.

4.2 Components of the coverage function

Our coverage function is a linear combination of the following.

(i) *Popularity.* One of the requirements for a good summary (especially, for user-generated content) is that it should include (or rather not miss) the popular views or opinions expressed by several users across multiple documents or comments. We model this property in our objective function as follows.

For each node u, we define $w(u)$ as the number of documents $|C_{u_{\mathrm{rel}}} \subseteq C|$ from the collection such that at least one of the dependency relations $\mathrm{rel} \in u$ appeared in a sentence within some document $c \in C_{u_{\mathrm{rel}}}$. The popularity scores are then normalized across all nodes in the graph. We then add this component to our objective function as $w(S) = \sum_{u \in S} w(u)$.

(ii) *Cluster contribution.* This term captures the fact that we do not intend to include multiple sentences from the same comment (or document). Define $\mathcal{B}$ to be the clustering induced by the sentence to comment relation, i.e., two sentences in the same comment belong to the same cluster. The corresponding contribution to the objective function is $\sum_{B \in \mathcal{B}} |S \cap B|^{1/2}$.

(iii) *Content contribution.* This term promotes the diversification of content. We look at the graph of sentences where the weight of each edge is $s(u, v)$. This graph is then partitioned based on a local random walk based method to give us clusters $\mathcal{D} = \{D_1, \ldots, D_n\}$. The corresponding contribution to the objective function is $\sum_{D \in \mathcal{D}} |S \cap D|^{1/2}$.

(iv) *Cover contribution.* We also measure the cover of the set S as follows: for each element s in U first define cover of an element u by a set S' as $\mathrm{cov}(u, S') = \sum_{v \in S'} s(u, v)$. Then, the

[2] There exists various semantic relatedness measures based on WordNet (Patwardhan and Pedersen, 2006). In our experiments, for WN we pick one that is based on the path length between the two words in the WordNet graph.

cover value of the set S is defined as $\text{cov}(S) = \sum_{u \in S} \min(\text{cov}(u, S), 0.25\text{cov}(u, U))$.[3]

Thus, the final coverage function is: $g(S) = w(S) + \alpha \sum_{B \in \mathcal{B}} |S \cap B|^{1/2} + \beta \sum_{D \in \mathcal{D}} |S \cap D|^{1/2} + \lambda \text{cov}(S)$, where α, β, λ are non-negative constants. By using the monotone submodularity of each of the component functions, and the fact that addition preserves submodularity, the following is immediate.

Fact 4. *$g(S)$ is a monotone, non-negative, submodular function.*

We then apply Algorithm 1 to optimize (1).

5 Experiments

5.1 Data

Multi-document summarization. We use the DUC 2004 corpus[4] that comprises 50 clusters (i.e., 50 different summarization tasks) with 10 documents per cluster on average. Each document contains multiple sentences and the goal is to produce a summary of all the documents for a given cluster.

Comments summarization. We extracted a set of news articles and corresponding user comments from Yahoo! News website. Our corpus contains a set of 34 articles and each article is associated with anywhere from 100–500 comments. Each comment contains more than three sentences and 36 words per sentence on average.

5.2 Evaluation

For each summarization task, we compare the system output (i.e., summaries automatically produced by the algorithm) against the human-generated summaries and evaluate the performance in terms of ROUGE score (Lin, 2004), a standard recall-based evaluation measure used in summarization. A system that produces higher ROUGE scores generates better quality summary and vice versa.

We use the following evaluation settings in our experiments for each summarization task:

(1) For multi-document summarization, we compute the ROUGE-1[5] scores that was the main evaluation criterion for DUC 2004 evaluations.

(2) For comment summarization, the collection of user comments associated with a given article is typically much larger. Additionally, individual comments are noisy, wordy, diverse, and informally written. Hence for this task, we use a slightly different evaluation criterion that is inspired from the DUC 2005-2007 summarization evaluation tasks.

We represent the content within each comment c (i.e., all sentences $S(c)$ comprising the comment) as a single node in the graph. We then run our summarization algorithm on the instantiated graph to produce a summary for each news article. In addition, each news article and corresponding set of comments were presented to three human annotators. They were asked to select a subset of comments (at most 20 comments) that best represented a summary capturing the most popular as well as diverse set of views and opinions expressed by different users that are relevant to the given news article. We then compare the automatically generated comment summaries against the human-generated summaries and compute the ROUGE-1 and ROUGE-2 scores.[6]

This summarization task is particularly hard for even human annotators since user-generated comments are typically noisy and there are several hundreds of comments per article. Similar to existing work in the literature (Sekine and Nobata, 2003), we computed inter-annotator agreement for the humans by comparing their summaries against each other on a small held-out set of articles. The average ROUGE-1 F-scores observed for humans was much higher (59.7) than that of automatic systems measured against the human-generated summaries (our best system achieved a score of 28.9 ROUGE-1 on the same dataset). This shows that even though this is a new type of summarization task, humans tend to generate more consistent summaries and hence their annotations are reliable for evaluation purposes as in multi-document summarization.

5.3 Results

Multi-document summarization. **(1)** Table 1 compares the performance of our system with the previous best reported system that participated in the DUC 2004 competition. We also include for comparison another baseline—a version

[3]The choice of the value 0.25 in the cover component is inspired by the observations made by (Lin and Bilmes, 2011) for the α value used in their cover function.

[4]http://duc.nist.gov/duc2004/tasks.html

[5]ROUGE v1.5.5 with options: -a -c 95 -b 665 -m -n 4 -w 1.2

[6]ROUGE v1.5.5 with options: -a -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d -l 150

of our system that approximates the submodular objective function proposed by (Lin and Bilmes, 2011).[7] As shown in the results, our best system[8] which uses the h_{s} dispersion function achieves a better ROUGE-1 F-score than all other systems.

(2) We observe that the h_{m} and h_{t} dispersion functions produce slightly lower scores than h_{s}, which may be a characteristic of this particular summarization task. We believe that the empirical results achieved by different dispersion functions depend on the nature of the summarization tasks and there are task settings under which h_{m} or h_{t} perform better than h_{s}. For example, we show later how using the h_{t} dispersion function yields the best performance on the comments summarization task. Regardless, the theoretical guarantees presented in this paper cover all these cases.

(3) We also analyze the contributions of individual components of the new objective function towards summarization performance by selectively setting certain parameters to 0. Table 2 illustrates these results. We clearly see that each component (popularity, cluster contribution, dispersion) individually yields a reasonable summarization performance but the best result is achieved by the combined system (row 5 in the table). We also contrast the performance of the full system with and without the dispersion component (row 4 versus row 5). The results show that optimizing for dispersion yields an improvement in summarization performance.

(4) To understand the effect of utilizing syntactic structure and semantic similarity for constructing the summarization graph, we ran the experiments using just the unigrams and bigrams; we obtained a ROUGE-1 F-score of 37.1. Thus, modeling the syntactic structure (using relations extracted from dependency parse tree) along with computing similarity in semantic spaces (using WordNet) clearly produces an improvement in the summarization quality (+1.4 improvement in ROUGE-1 F-score). However, while the structured representation is beneficial, we observed that dispersion (and other individual components) contribute similar performance gains even when using ngrams alone. So the improvements obtained from the structured representation and dispersion are complementary.

System	ROUGE-1 F
Best system in DUC 2004	37.9
(Lin and Bilmes, 2011), no tuning	37.4[7]
Our algorithm with $h = h_{\mathrm{m}}$	37.5
$h = h_{\mathrm{s}}$	**38.5**
$h = h_{\mathrm{t}}$	36.8

Table 1: Performance on DUC 2004.

Comments summarization. **(1)** Table 3 compares the performance of our system against a baseline system that is constructed by picking comments in order of decreasing length, i.e., we first pick the longest comment (comprising the most number of characters), then the next longest comment and so on, to create an ordered set of comments. The intuition behind this baseline is that longer comments contain more content and possibly cover more topics than short ones.

From the table, we observe that the new system (using either dispersion function) outperforms the baseline by a huge margin (+44% relative improvement in ROUGE-1 and much bigger improvements in ROUGE-2 scores). One reason behind the lower ROUGE-2 scores for the baseline might be that while long comments provide more content (in terms of size), they also add noise and irrelevant information to the generated summaries. Our system models sentences using the syntactic structure and semantics and jointly optimizes for multiple summarization criteria (including dispersion) which helps weed out the noise and identify relevant, useful information within the comments thereby producing better quality summaries. The 95% confidence interval scores for the best system on this task is [36.5–46.9].

(2) Unlike the multi-document summarization, here we observe that the h_{t} dispersion function yields the best empirical performance for this task. This observation supports our claim that the choice of the specific dispersion function depends

[7] Note that Lin & Bilmes (2011) report a slightly higher ROUGE-1 score (F-score 38.90) on DUC 2004. This is because their system was tuned for the particular summarization task using the DUC 2003 corpus. On the other hand, even without any parameter tuning our method yields good performance, as evidenced by results on the two different summarization tasks. However, since individual components within our objective function are parametrized it is easy to tune them for a specific task or genre.

[8] For the full system, we weight certain parameters pertaining to cluster contributions and dispersion higher ($\alpha = \beta = \delta = 5$) compared to the rest of the objective function ($\lambda = 1$). Lin & Bilmes (2011) also observed a similar finding (albeit via parameter tuning) where weighting the cluster contribution component higher yielded better performance. If the maximum number of sentences/comments chosen were k, we brought both h_{s} and h_{t} to the same approximate scale as h_{m} by dividing h_{s} by $k(k-1)/2$ and h_{t} by $k-1$.

Objective function components	ROUGE-1 F
$\alpha = \beta = \lambda = \delta = 0$	35.7
$w(S) = \beta = \lambda = \delta = 0$	35.1
$h = h_{\mathrm{s}}, w(S) = \alpha = \beta = \lambda = 0$	37.1
$\delta = 0$	37.4
$w(S), \alpha, \beta, \lambda, \delta > 0$	**38.5**

Table 2: Performance with different parameters (DUC).

on the summarization task and that the dispersion functions proposed in this paper have a wider variety of use cases.

(3) Results showing contributions from individual components of the new summarization objective function are listed in Table 4. We observe a similar pattern as with multi-document summarization. The full system using all components outperform all other parameter settings, achieving the best ROUGE-1 and ROUGE-2 scores. The table also shows that incorporating dispersion into the objective function yields an improvement in summarization quality (row 4 versus row 5).

System	ROUGE-1	ROUGE-2
Baseline (decreasing length)	28.9	2.9
Our algorithm with $h = h_{\mathrm{m}}$	39.2	13.2
$h = h_{\mathrm{s}}$	40.9	15.0
$h = h_{\mathrm{t}}$	**41.6**	**16.2**

Table 3: Performance on comments summarization.

Objective function components	ROUGE-1	ROUGE-2
$\alpha = \beta = \lambda = \delta = 0$	36.1	9.4
$w(S) = \beta = \lambda = \delta = 0$	32.1	4.9
$h = h_{\mathrm{t}}, w(S) = \alpha = \beta = \lambda = 0$	37.8	11.2
$\delta = 0$	38.0	11.6
$w(S), \alpha, \beta, \lambda, \delta > 0$	**41.6**	**16.2**

Table 4: Performance with different parameters (comments).

6 Conclusions

We introduced a new general-purpose graph-based summarization framework that combines a submodular coverage function with a non-submodular dispersion function. We presented three natural dispersion functions that represent three different ways of ensuring non-redundancy (using sentence dissimilarities) for summarization and proved that a simple greedy algorithm can obtain an approximately optimal summary in all these cases. Experiments on two different summarization tasks show that our algorithm outperforms algorithms that rely only on submodularity. Finally, we demonstrated that using a structured representation to model sentences in the graph improves summarization quality.

For future work, it would be interesting to investigate other related developments in this area and perhaps combine them with our approach to see if further improvements are possible. Firstly, it would interesting to see if dispersion offers similar improvements over a tuned version of the submodular framework of Lin and Bilmes (2011). In a very recent work, Lin and Bilmes (2012) demonstrate a further improvement in performance for document summarization by using mixtures of submodular shells. This is an interesting extension of their previous submodular framework and while the new formulation permits more complex functions, the resulting function is still submodular and hence can be combined with the dispersion measures proposed in this paper. A different body of work uses determinantal point processes (DPP) to model subset selection problems and adapt it for document summarization (Kulesza and Taskar, 2011). Note that DPPs use similarity kernels for performing inference whereas our measures are combinatorial and not kernel-representable. While approximation guarantees for DPPs are open, it would be interesting to investigate the empirical gains by combining DPPs with dispersion-like functions.

Acknowledgments

We thank the anonymous reviewers for their many useful comments.

References

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proc. PODS*, pages 155–166.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. SIGIR*, pages 335–336.

Barun Chandra and Magnús Halldórsson. 2001. Facility dispersion and remote subgraphs. *J. Algorithms*, 38(2):438–465.

Dietmar Cieslik. 2001. *The Steiner Ratio*. Springer.

John M. Conroy and Dianne P. O'Leary. 2001. Text summarization via hidden Markov models. In *Proc. SIGIR*, pages 406–407.

Hal Daumé, III and Daniel Marcu. 2006. Bayesian query-focused summarization. In *Proc. COLING/ACL*, pages 305–312.

Marie-Catherine de Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. LREC*, pages 449–454.

Elena Filatova. 2004. Event-based extractive summarization. In *Proc. ACL Workshop on Summarization*, pages 104–111.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proc. COLING*.

Makoto Imase and Bernard M. Waxman. 1991. Dynamic Steiner tree problem. *SIAM J. Discrete Mathematics*, 4(3):369–384.

Hyun Duk Kim, Kavita Ganesan, Parikshit Sondhi, and ChengXiang Zhai. 2011. Comprehensive review of opinion summarization. Technical report, University of Illinois at Urbana-Champaign.

Alex Kulesza and Ben Taskar. 2011. Learning determinantal point processes. In *Proc. UAI*, pages 419–427.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proc. ACL*, pages 510–520.

Hui Lin and Jeff Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Proc. UAI*, pages 479–490.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out: Proc. ACL Workshop*, pages 74–81.

G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14(1):265–294.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 43–76. Springer.

Siddharth Patwardhan and Ted Pedersen. 2006. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. In *Proc. EACL Workshop on Making Sense of Sense: Bringing Computational Linguistics and Psycholinguistics Together*, pages 1–8.

Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation summarization through keyphrase extraction. In *Proc. COLING*, pages 895–903.

Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tür. 2010. Long story short—Global unsupervised models for keyphrase based meeting summarization. *Speech Commun.*, 52(10):801–815.

Satoshi Sekine and Chikashi Nobata. 2003. A survey for multi-document summarization. In *Proc. HLT-NAACL Workshop on Text Summarization*, pages 65–72.

Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. 2010. Summarizing microblogs automatically. In *Proc. HLT/NAACL*, pages 685–688.

Chao Shen and Tao Li. 2010. Multi-document summarization via the minimum dominating set. In *Proc. COLING*, pages 984–992.

Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on maximum coverage problem and its variant. In *Proc. EACL*, pages 781–789.

Koji Yatani, Michael Novati, Andrew Trusty, and Khai N. Truong. 2011. Review spotlight: A user interface for summarizing user-generated reviews using adjective-noun word pairs. In *Proc. CHI*, pages 1541–1550.