Opinosis: A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions

Kavita Ganesan and **ChengXiang Zhai** and **Jiawei Han**
Department of Computer Science
University of Illinois at Urbana-Champaign
{kganes2,czhai,hanj}@cs.uiuc.edu

Abstract

We present a novel graph-based summarization framework (Opinosis) that generates concise abstractive summaries of highly redundant opinions. Evaluation results on summarizing user reviews show that Opinosis summaries have better agreement with human summaries compared to the baseline extractive method. The summaries are readable, reasonably well-formed and are informative enough to convey the major opinions.

1 Introduction

Summarization is critically needed to help users better digest the large amounts of opinions expressed on the web. Most existing work in Opinion Summarization focus on predicting sentiment orientation on an entity (Pang et al., 2002) (Pang and Lee, 2004) or attempt to generate aspect-based ratings for that entity (Snyder and Barzilay, 2007) (Lu et al., 2009)(Lerman et al., 2009)(Titov and Mcdonald, 2008). Such summaries are very informative, but it is still hard for a user to understand why an aspect received a particular rating, forcing a user to read many, often highly redundant sentences about each aspect. To help users further digest the opinions in each aspect, it is thus desirable to generate a concise textual summary of such redundant opinions.

Indeed, in many scenarios, we will face the problem of summarizing a large number of highly redundant opinions; other examples include summarizing the 'tweets' on Twitter or comments made about a blog or news article. Due to the subtle variations of redundant opinions, typical extractive methods are often inadequate for summarizing such opinions. Consider the following sentences:

1. *The iPhone's $\underline{\text{battery lasts long}}$, only had to charge it once $\overline{\text{every few days}}$.*
2. *iPhone's $\underline{\text{battery is bulky}}$ but it is $\underline{\text{cheap}}$..*
3. *iPhone's $\underline{\text{battery is bulky}}$ but it $\underline{\text{lasts long}}$!*

With extractive summarization, no matter which single sentence of the three is chosen as a summary, the generated summary would be biased. In such a case, an abstractive summary such as '*iPhone's battery is cheap, lasts long but is bulky*' is a more complete summary, conveying all the necessary information. Extractive methods also tend to be verbose and this is especially problematic when the summaries need to be viewed on smaller screens like on a PDA. Thus, an informative and concise abstractive summary would be a better solution.

Unfortunately, abstractive summarization is known to be difficult. Existing work in abstractive summarization has been quite limited and can be categorized into two categories: (1) approaches using prior knowledge (Radev and McKeown, 1998) (Finley and Harabagiu, 2002) (DeJong, 1982) and (2) approaches using Natural Language Generation (NLG) systems (Saggion and Lapalme, 2002) (Jing and McKeown, 2000). The first line of work requires considerable amount of manual effort to define schemas such as frames and templates that can be filled with the use of information extraction techniques. These systems were mainly used to summarize news articles. The second category of work uses deeper NLP analysis with special techniques for text regeneration. Both approaches either heavily rely on manual effort or are domain dependent.

In this paper, we propose a novel flexible summarization framework, Opinosis, that uses graphs to produce abstractive summaries of highly redundant opinions. In contrast with the previous work, Opinosis assumes no domain knowledge and uses shallow NLP, leveraging mostly the word order in the existing text and its inherent redundancies to generate informative abstractive summaries. The key idea of Opinosis is to first construct a textual graph that represents the text to be summarized. Then, three unique properties of this graph are used to explore and score various subpaths that help in generating candidate abstractive summaries.

Evaluation results on a set of user reviews show that Opinosis summaries have reasonable agreement with human summaries. Also, the generated summaries are readable, concise and fairly well-formed. Since Opinosis assumes no domain knowledge and is highly flexible, it can be potentially used to summarize any highly redundant content and could even be ported to other languages. (All materials related to this work including the dataset and demo software can be found at `http://timan.cs.uiuc.edu/downloads.html`.)

2 Opinosis-Graph

Our key idea is to use a graph data structure (called Opinosis-Graph) to represent natural language text and cast this abstractive summarization problem as one of finding appropriate paths in the graph. Graphs have been commonly used for extractive summarization (e.g., LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004)), but in these works the graph is often *undirected* with *sentences as nodes* and similarity as edges. Our graph data structure is different in that each node represents a *word unit* with *directed edges* representing the structure of sentences. Moreover, we also attach positional information to nodes as will be discussed later.

Algorithm 1 (A1): $OpinosisGraph(Z)$

```
1:  Input: Topic related sentences to be summarized: $Z = \{z_i\}_{i=1}^n$
2:  Output: $G = (V, E)$
3:  for $i = 1$ to $n$ do
4:    $w \leftarrow Tokenize(z_i)$
5:    $sent\_size \leftarrow SizeOf(w)$
6:    for $j = 1$ to $sent\_size$ do
7:      $LABEL \leftarrow w_j$
8:      $PID \leftarrow j$
9:      $SID \leftarrow i$
10:     if $ExistsNode(G, LABEL)$ then
11:       $v_j \leftarrow GetExistingNode(G, LABEL)$
12:       $PRI_{v_j} \leftarrow PRI_{v_j} \cup (SID, PID)$
13:     else
14:       $v_j \leftarrow CreateNewNode(G, LABEL)$
15:       $PRI_{v_j} \leftarrow (SID, PID)$
16:     end if
17:     if not $ExistsEdge(v_{j-1} \rightarrow v_j, G)$ then
18:       $AddEdge(v_{j-1} \rightarrow v_j, G)$
19:     end if
20:   end for
21: end for
```

Our graph representation is closer to that used by Barzilay and Lee (Barzilay and Lee, 2003) for the task of paraphrasing, wherein each node in the graph represents a unique word. However, in their work, such a graph is used to identify regions of commonality and variability amongst similar sentences. Thus, the positional information is not required nor is it maintained. In contrast, we maintain positional information at each node as this is critical for the selection of candidate paths.

Algorithm **A1** outlines the steps involved in building an Opinosis-Graph. We start with a set of sentences relevant to a specific topic, which can be obtained in different ways depending on the application. For example, they may be all sentences related to the *battery life* of the *iPod Nano*. We denote these sentences as $Z = \{z_i\}_{i=1}^n$ where each z_i is a sentence containing part-of-speech (POS) annotations. (A1:4) Each $z_i \in Z$ is split into a set of *word units*, where each unit, w_j consists of a word and its corresponding POS annotation (e.g. "*service:nn*", "*good:adj*"). (A1:7-9) Each unique w_j will form a node, v_j, in the Opinosis-Graph, with w_j being the label. Also, since we only have one node per unique word unit, each node keeps track of all sentences that it is a part of using a *sentence identifier* (SID) along with its *position of occurrence* in that sentence (PID). (A1:10-16) Each node will thus carry a *Positional Reference Information* (PRI) which is a list of {SID:PID} pairs representing the node's membership in a sentence. (A1:17-19) The original structure of a sentence is recorded with the use of directed edges. Figure 1 shows a resulting Opinosis-Graph based on four sentences.

The Opinosis-Graph has some unique properties that are crucial in generating abstractive summaries. We highlight some of the core properties by drawing examples from Figure 1:

Property 1. *(**Redundancy Capture**). Highly redundant discussions are naturally captured by subgraphs.*

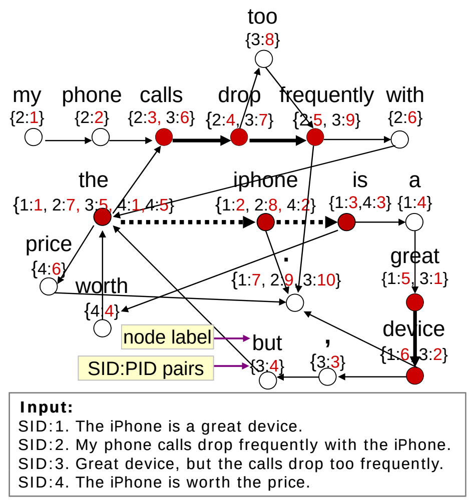
Figure 1 shows that although the phrase '*great device*' was mentioned in different parts of sentences (1) and (3), this phrase forms a relatively heavy sub-path in the resulting graph. This is a good indication of salience.

Property 2. *(**Gapped Subsequence Capture**). Existing sentence structures introduce **lexical links** that facilitate the discovery of new sentences or reinforce existing ones.*

The main point conveyed by sentences (2) and (3) in Figure 1 is that *calls drop frequently*. However, this is expressed in slightly different ways and is reflected in the resulting subgraph. Since sentence (2) introduces a *lexical link* between '*drop*' and '*frequently*', the word '*too*' can be ignored for sentence (3) as the same amount of information is retained. This is analogous to capturing a *repetitive gapped subsequence* where similar sequences with minor variations are captured. With this, the subgraph *calls drop frequently* can be considered redundant.

Property 3. *(**Collapsible Structures**). Nodes that resemble hubs are possibly collapsible.*

In Figure 1 we see that the subgraph '*the iPhone is*', is fairly heavy and the 'is' node acts like a



```
Input:
SID:1. The iPhone is a great device.
SID:2. My phone calls drop frequently with the iPhone.
SID:3. Great device, but the calls drop too frequently.
SID:4. The iPhone is worth the price.
```

Figure 1: Sample *Opinosis-Graph*. Thick edges indicate salient paths.

'*hub*' where it connects to various other nodes. Such a structure is naturally captured by the Opinosis-Graph and is a good candidate for compression to generate a summary such as '*The iPhone is a great device and is worth the price*'. Also, certain word POS (e.g. linking verbs like 'is' and 'are') often carry hub-like properties that can be used in place of the outlink information.

3 Opinosis Summarization Framework

In this section, we describe a general framework for generating abstractive summaries using the Opinosis-Graph. We also describe our implementation of the components in this framework.

At a high level, we generate an abstractive summary by repeatedly searching the Opinosis graph for appropriate subgraphs that both encode a valid sentence (thus meaningful sentences) and have high redundancy scores (thus representative of the major opinions). The sentences encoded by these subgraphs would then form an abstractive summary.

Going strictly by the definition of true abstraction (Radev et al., 2002), our problem formulation is still more extractive than abstractive because the generated summary can only contain words that occur in the text to be summarized; our problem definition may be regarded as a word-level (finer granularity) extractive summarization. However, compared to the conventional sentence-level extractive summarization, our formulation has flavors of abstractive summarization wherein we have elements of *fusion* (combining extracted portions) and *compression* (squeezing out unimportant material from a sentence). Hence, the sentences in the generated summary are generally not the same as any original sentence. Such a "shallow" abstractive summarization problem is more tractable, enabling us to develop a general solution to the problem. We now describe each component in such a summarization framework.

3.1 Valid Path

A valid path intuitively refers to a path that corresponds to a meaningful sentence.

Definition 1. *(**Valid Start Node - VSN**). A node v_q is a valid start node if it is a natural starting point of a sentence.*

We use the positional information of a node to determine if it is a VSN. Specifically, we check if $Average(PID_{v_q}) \leq \sigma_{vsn}$, where σ_{vsn} is a parameter to be empirically set. With this, we only qualify nodes that tend to occur early on in a sentence.

Definition 2. *(**Valid End Node - VEN**). A node v_s is a valid end point if it completes a sentence.*

We use the natural ending points in the text to be summarized as hints to which node may be a valid end point of a path (i.e., a sentence). Specifically, a node is a *valid end node* if (1) the node is a punctuation such as *period* and *comma* or (2) the node is any coordinating conjunction (e.g., *'but'* and *'yet'*).

Definition 3. *(**Valid Path**). A path $W = \{v_q...v_s\}$ is valid if it is connected by a set of directed edges such that (1) v_q is a **VSN**, (2) v_s is a **VEN**, and (3) W satisfies a set of well-formedness POS constraints.*

Since not every path starting with a VSN and ending at a VEN encodes a meaningful sentence, we further require a valid path to satisfy the following POS constraints (expressed in regular-expression) to ensure that a valid path encodes a well-formed sentence:

1. $.*(/nn)+.*(/vb)+.*(/jj)+.*$
2. $.*(/jj)+.*(/to)+.*(/vb).*$
3. $.*(/rb)*.*(/jj)+.*(/nn)+.*$
4. $.*(/rb)+.*(/in)+.*(/nn)+.*$

This also provides a way (if needed) for the application to generate only specific type of sentences like *comparative sentences* or *strictly opinionated sentences*. These rules are thus application specific.

3.2 Path Scoring

Intuitively, to generate an abstractive summary, we should select a valid path that can represent most of the redundant opinions well. We would thus favor a valid path with a high redundancy score.

Definition 4. *(**Path Redundancy**). Let $W = \{v_q...v_s\}$ be a path from an Opinosis-Graph. The path redundancy of W, $r(q, s)$, is the number of overlapping sentences covered by this path, i.e.,*

$$r(q,s) = n_q \bar{\cap} n_{q+1} ... \bar{\cap} n_s,$$

where $n_i = PRI_{v_i}$ and $\bar{\cap}$ is the intersection between two sets of SIDs such that the difference between the corresponding PIDs is no greater than σ_{gap}, and $\sigma_{gap} > 0$ is a parameter.

Path redundancies provide good indication of how many sentences discuss something similar at each point in the path. The σ_{gap} parameter controls the maximum allowed gaps in discovering these redundancies. Thus, a common sentence X between nodes v_q and v_r, will be considered a valid intersect if $(PID_{v_{rx}} - PID_{v_{q_x}}) \leq \sigma_{gap}$.

Based on path redundancy, we propose several ways to score a path for the purpose of selecting a good path to include in the summary:

1. $S_{basic}(W) = \frac{1}{|W|} \sum_{k=i+1,i}^{s} r(i,k)$
2. $S_{wt\_len}(W) = \frac{1}{|W|} \sum_{k=i+1,i}^{s} |v_i, v_k| * r(i,k)$
3. $S_{wt\_loglen}(W) = \frac{1}{|W|}(r(i,i+1) + \sum_{k=i+2,i+1}^{s} log_2|v_i, v_k| * r(i,k))$

v_i is the first node in the path being scored and v_s is the last node. $|v_i, v_k|$ is the length from node v_i to v_k. $|W|$ is the length of the entire path being scored. The S_{basic} scoring function scores a path purely based on the level of redundancy. One could also argue that high redundancy on a longer path is intuitively more valuable than high redundancy on a shorter path as the former would provide better coverage than the latter. This intuition is factored in by the S_{wt_len} and S_{wt_loglen} scoring functions where the level of *redundancy* is *weighted* by the *path length*. S_{wt_loglen} is similar to S_{wt_len} only that it scales down the path length so that it does not entirely dominate.

3.3 Collapsed paths

In some cases, paths in the Opinosis-Graph may be collapsible (as explained in Section 2). In such a case, the collapse operation is performed and then the path scores are computed. We will now explain a few concepts related to collapsible structures. Let $\widehat{W} = \{v_i...v_k\}$ be a path from the Opinosis-Graph.

Definition 5. ***(Collapsible Node)****. Node v_k is a candidate for collapse if its POS is a* verb.

We only attempt to collapse nodes that are *verbs* due to the heavy usage of verbs in opinion text and the ease with which the structures can be combined to form a new sentence. However, as mentioned earlier other properties like the outlink information can be used to determine if a node is collapsible.

Definition 6. ***(Collapsed Candidates, Anchor)****. Let v_k be a collapsible node. The collapsed candidates of v_k (denoted by $CC = \{cc_i\}_{i=1}^{m}$) are the remaining paths after v_k in all the valid paths going through $v_i...v_k$. The prefix $v_i...v_k$ is called the anchor, denoted as $C_{anchor} = \{v_i...v_k\}$. Each path $\{v_i...v_n\}$, where v_n is the last node in each $cc_i \in CC$, is an individually valid path.*

C_{anchor}	CC	Connector
a. the sound quality is	cc_1 : really good cc_2 : clear	and
b. the iphone is	cc_1 : great cc_2 : expensive	but

Table 1: Example of anchors, collapsed candidates and suitable connectors

Table 1 shows a simplistic example of anchors and corresponding collapsed candidates. Once the anchor and collapsed candidates have been identified, the task is then to combine all of these to form a new sentence.

Definition 7. ***(Stitched Sentence)*** *A stitched sentence is one that combines C_{anchor} and CC to form a combined, logical sentence.*

We will now describe the stitching procedure that we use, by drawing examples from Table 1. Since we are dealing with verbs, C_{anchor} can be combined with the corresponding CC with commas to separate each $cc_i \in CC$ with one exception - the correct sentence connector has to be used for the last cc_i. For C_{anchor_a}, the phrases *really good* and *clear* can be connected by 'and' due to the same sentiment orientation. For C_{anchor_b}, the collapsed candidate phrases are well connected by the word 'but'. We use the existing Opinosis-Graph to determine the most appropriate connector. We do this by looking at all *coordinating conjunction* (e.g. 'but', 'yet') nodes (v_{cconj}) that are connected to the first node of the last collapsed candidate, cc_m. This would be the node labeled '*clear*' for C_{anchor_a} and '*expensive*' for C_{anchor_b}. We denote these nodes as v_{0,cc_m}. The v_{cconj}, with the highest *path redundancy* with v_{0,cc_m}, will be selected as the connector.

Definition 8. ***(Collapsed Path Score)*** *The final path score after the entire collapse operation is the average across path scores computed from v_i to the last node in each $cc_i \in CC$.*

The collapsed path score essentially involves computing the *path scores* of the individual sentences assuming that they are not collapsed and then averaging them.

3.4 Generation of summary

Once we can score all the valid paths as well as all the collapsed paths, the generation of an abstractive summary can be done in two steps: First, we rank all the paths (including the collapsed paths) in descending order of their scores. Second, we eliminate duplicated (or extremely similar) paths by using a similarity measure (in our experiments, we used Jaccard). We then take the top few remaining paths as the generated summary, with the number of paths to be chosen controlled by a parameter σ_{ss}, which represents summary size.

Although conceptually we enumerate all the valid paths, in reality we can use a redundancy score threshold, σ_r to prune many non-promising paths. This is reasonable because we are only interested in paths with high redundancy scores.

4 Summarization Algorithm

Algorithms **A2** and **A3** describe the steps involved in Opinosis Summarization. A2 is the starting point of the Opinosis Summarization and A3 is a subroutine where path finding takes place, invoked from within A2.

Algorithm 2 (A2): $OpinosisSummarization(Z)$

```
1:  Input: Topic related sentences to be summarized: $Z = \{z_i\}_{i=1}^{n}$
2:  Output: $\mathcal{O}$ ={Opinosis Summaries}
3:  $g \leftarrow OpinosisGraph(Z)$
4:  $node\_size \leftarrow SizeOf(g)$
5:  for $j = 1$ to $node\_size$ do
6:    if $VSN(v_j)$ then
7:      $pathLen \leftarrow 1$
8:      $score \leftarrow 0$
9:      $cList \leftarrow CreateNewList()$
10:     Traverse$(cList, v_j, score, PRI_{v_j}, label_{v_j}, pathLen)$
11:     $candidates \leftarrow \{candidates \cup cList\}$
12:   end if
13: end for
14: $\mathcal{C} \leftarrow EliminateDuplicates(candidates)$
15: $\mathcal{C} \leftarrow SortByPathScore(\mathcal{C})$
16: for $i = 1$ to $\sigma_{ss}$ do
17:   $\mathcal{O} = \{\mathcal{O} \cup PickNextBestCandidate(\mathcal{C})\}$
18: end for
```

(A2:3) Opinosis Summarization starts with the construction of the Opinosis-Graph, described in detail in Section 2. This is followed by the *depth first* traversal of this graph to locate *valid paths* that become *candidate summaries*. (A2:6-12) To achieve this, each node v_j in the Opinosis-Graph is examined to determine if it is a *VSN* and, if it is, path finding will start from this node by invoking subroutine A3. A3 takes the following as input: *list* - a list to hold candidate summaries; v_i - the node to continue traversal from; *score* - the accumulated path score; $PRI_{overlap}$ - the intersect between PRIs of all nodes visited so far (see Definition 4); *sentence* - the summary sentence formed so far; *len* - the current path length. (A2:7-10) Before invoking A3 from A2, the path length is set to '1', path score is set to '0' and a new list is created to store candidate summaries generated from node v_j. (A2:11) All candidate summaries generated from v_j will be stored in a common pool of candidate summaries.

Algorithm 3 (A3): $Traverse(...)$

```
1:  Input: $list, v_k \subseteq V, score, PRI_{overlap}, sentence, len$
2:  Output: A set of candidate summaries
3:  $redundancy \leftarrow SizeOf(PRI_{overlap})$
4:  if $redundancy \geq \sigma_r$ then
5:    if $VEN(v_k)$ then
6:      if $ValidSentence(sentence)$ then
7:        $finalScore \leftarrow \frac{score}{len}$
8:        $AddCandidate(list, sentence, finalScore)$
9:      end if
10:   end if
11:   for $v_n \in Neighbors_{v_k}$ do
12:     $PRI_{new} \leftarrow PRI_{overlap} \; \bar{\cap} \; PRI_{v_n}$
13:     $redundancy \leftarrow SizeOf(PRI_{new})$
14:     $newSent \leftarrow Concat(sentence, label_{v_n})$
15:     $L \leftarrow len + 1$
16:     $newScore \leftarrow score + PathScore(redundancy, L)$
17:     if $Collapsible(v_n)$ then
18:       $C_{anchor} \leftarrow newSent$
19:       $tmp \leftarrow CreateNewList()$
20:       for $v_x \in Neighbors_{v_n}$ do
21:         Traverse$(tmp, v_x, 0, PRI_{new}, label_{v_x}, L)$
22:         $CC \leftarrow EliminateDuplicates(tmp)$
23:         $CCPathScore \leftarrow AveragePathScore(CC)$
24:         $finalScore \leftarrow newScore + CCPathScore$
25:         $stitchedSent \leftarrow Stitch(C_{anchor}, CC)$
26:         $AddCandidate(list, stitchedSent, finalScore)$
27:       end for
28:     else
29:       Traverse$(list, v_n, newScore, PRI_{new}, newSent, L)$
30:     end if
31:   end for
32: end if
```

(A3:3-4) Algorithm A3 starts with a check to ensure that the minimum path redundancy requirement is satisfied (see definition 4). For the very first node sent from A2, the path redundancy is the size of the raw PRI. (A3:5-10) If the redundancy requirement is satisfied, a few checks are done to determine if a *valid path* has been found. If it has, then the resulting sentence and its final score are added to the list of candidate summaries.

(A3:11-31) Traversal proceeds recursively through the exploration of all neighboring nodes of the current node, v_k. (A3:12-16) For every neighboring node, v_n the *PRI* overlap information, path length, summary sentence and path score are updated before the next recursion. (A3:29) If a v_n is not collapsible, then a regular traversal takes place. (A3:17-27) However, if v_n is collapsible, the updated sentence in A3:14, will now serve as an *anchor* in A3:18. (A3:21) A3 will then attempt to start a recursive traversal from all neighboring nodes of v_n in order to find corresponding collapsed candidates. (A3:22-26) After this, duplicates are eliminated from the *collapsed candidates* and the *collapsed path score* is computed. The resulting *stitched sentence* and its *final score* are then added to the original list of candidate summaries.

(A2:14-18) Once all paths have been explored for candidate generation, duplicate candidates are removed and the remaining are sorted in descending order of their path scores. The best σ_{ss} candidates are 'picked' as final Opinosis summaries.

5 Experimental Setup

We evaluate this abstractive summarization task using reviews of *hotels*, *cars* and various *products*[1]. Based on these reviews, 2 humans were asked to construct 'opinion seeking' queries which would consist of an *entity name* and a *topic of interest*. Example of such queries are: *Amazon Kindle:buttons*, *Holiday Inn, Chicago: staff*, and so on. We compiled a set of 51 such queries. We create one review document per query by collecting all review sentences that contain the query words for the given entity. Each review document thus consists of a set of unordered, redundant review sentences related to the query. There are approximately 100 sentences per review document.

We use ROUGE (Lin, 2004b) to quantitatively assess the agreement of Opinosis summaries with human composed summaries. ROUGE is based on an n-gram co-occurrence between machine summaries and human summaries and is a widely accepted standard for evaluation of summarization tasks. In our experiments, we use ROUGE-1, ROUGE-2 and ROUGE-SU4 measures. ROUGE-1 and ROUGE-2 have been shown to have most correlation with human summaries (Lin and Hovy, 2003) and higher order ROUGE-N scores ($N > 1$) estimate the fluency of summaries.

We use multiple reference (human) summaries in our evaluation since it can achieve better correlation with human judgment (LIN, 2004a). We leverage Amazon's Online Workforce[2] to get 5 different human workers to summarize each review document. The workers were asked to be concise and were asked to summarize the major opinions in the review document presented to them. We manually reviewed each set of reference summaries and dropped summaries that had little or no correlation with the majority. This left us with around 4 reference summaries for each review document.

To allow performance comparison between humans, Opinosis and the baseline method, we implemented a Jackknifing procedure where, given K references, the ROUGE score is computed over K sets of K-1 references. With this, average human performance is computed by treating each reference summary as a 'system' summary, computing ROUGE scores over the remaining K-1 reference summaries.

Due to the limited work in abstractive summarization, no natural baseline could be used for comparison. The existing work in this area is mostly domain dependent and requires too much manual effort (explained in Section 1). The next best baseline is to use a state of the art extractive method. Thus, we use MEAD (Radev et al., 2000) as our baseline. MEAD is an extractive summarizer based on cluster centroids. It uses a collection of the most important words from the whole cluster to select the best sentences for summarization. By default, the scoring of sentences in MEAD is based on 3 parameters - *minimum sentence length*, *centroid*, and *position in text*. MEAD was ideal for our task because a good summary in our case would be one that could capture the most essential information. This is exactly what centroid-based summarization aims to achieve. Also, since the *position in text* parameter is irrelevant in our case, we could easily turn this off with MEAD.

We introduce a **readability test** to understand if Opinosis summaries are in fact readable. Suppose we have N sentences from a system-generated summary and M sentences from corresponding human summaries. We mix all these sentences and then ask a human assessor to pick at most N sentences that are *least readable* as the prediction of system summary.

$$readability(\mathcal{O}) = 1 - \frac{\#CorrectPick}{N}$$

If the human assessor often picks out system generated summaries as being least readable, then the readability of system summaries is poor. If not, then the system generated summaries are no different from human summaries.

6 Results

The baseline method (MEAD) selects 2 most representative sentences as summaries. To give a fair comparison, we fix the Opinosis summary size, $\sigma_{ss} = 2$. We also fix $\sigma_{vsn} = 15$. The best Opinosis configuration with $\sigma_{ss} = 2$ and $\sigma_{vsn} = 15$ is called Opinosis$_{best}$ ($\sigma_{gap} = 4, \sigma_r = 2$, S_{wt_loglen}). ROUGE scores reported are with the use of *stemming* and *stopword removal*.

Performance comparison between humans, Opinosis and baseline. Table 2 shows the performance comparison between humans, Opinosis$_{best}$ and the baseline method. First, we see that the baseline method has very high recall scores compared to Opinosis. This is because extractive methods that just 'select' sentences tend to be much longer resulting in higher recall. However, these summaries tend to carry information that may not be significant and is clearly reflected by the poor

[1]Reviews collected from Tripadvisor, Amazon, Edmunds
[2]https://www.mturk.com

Recall				
	ROUGE-1	ROUGE-2	ROUGE-SU4	Avg # Words
Human	0.3184	**0.1106**	0.1293	17
Opinosis	0.2831	0.0853	0.0851	15
Baseline	**0.4932**	0.1058	**0.2316**	75

Precision				
	ROUGE-1	ROUGE-2	ROUGE-SU4	Avg # Words
Human	0.3434	0.1210	0.1596	17
Opinosis	**0.4482**	**0.1416**	**0.2261**	15
Baseline	0.0916	0.0184	0.0102	75

F-score				
	ROUGE-1	ROUGE-2	ROUGE-SU4	Avg # Words
Human	0.3088	**0.1069**	**0.1142**	17
Opinosis	**0.3271**	0.0998	0.1027	15
Baseline	0.1515	0.0308	0.0189	75

Table 2: Performance comparison between Humans, Opinosis$_{best}$ and Baseline.

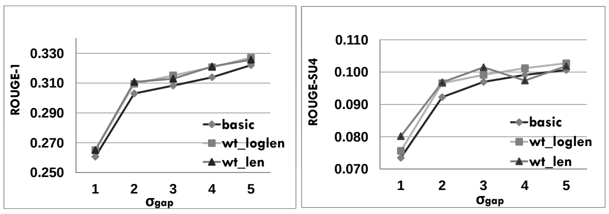


Figure 2: ROUGE scores (f-measure) at different levels of σ_{gap}, $\sigma_r = 2$.

precision scores.

Next, we see that humans have reasonable agreement amongst themselves given that these are independently composed summaries. This agreement is especially clear with the ROUGE-2 recall score where the recall is better than Opinosis but comparable to the baseline even though the summaries are much shorter. It is also clear that Opinosis is closer in performance to humans than to the baseline method. The recall scores of Opinosis summaries are slightly lower than that achieved by humans, while the precision scores are higher (Wilcoxon test shows that the increase in precision is statistically more significant than the decrease in recall). In terms of f-scores, Opinosis has the best ROUGE-1 score and its ROUGE-2 and ROUGE-SU4 scores are comparable with human performance. The baseline method has the lowest f-scores. The difference between the f-scores of Opinosis and that of humans is statistically insignificant.

Comparison of scoring functions. Next, we look into the performance of the three scoring functions, S_{basic}, S_{wt_len} and S_{wt_loglen} described in Section 3. Figure 2 shows ROUGE scores of these scoring methods at varying levels of σ_{gap}. First,

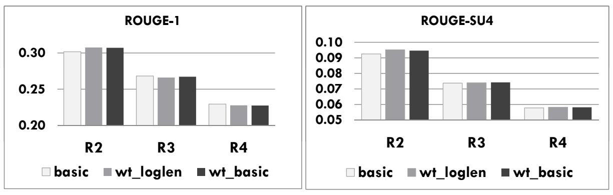


Figure 3: ROUGE scores (f-measure) at different levels of σ_r averaged across $\sigma_{gap} \in [1, 5]$

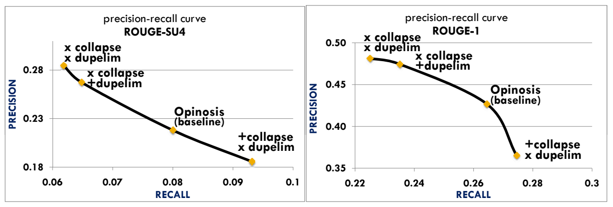
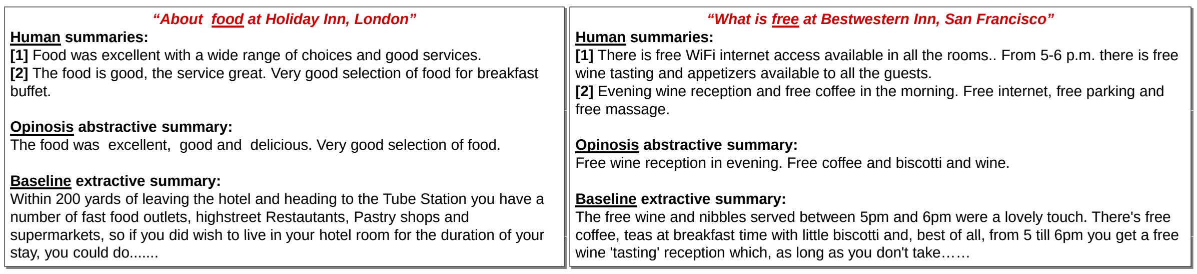


Figure 4: Precision-Recall comparison with different Opinosis features turned off.

it can be observed that S_{wt_basic} which does not use *path length* information, performs the worst. This is due to the effect of heavily favoring redundant paths over longer but reasonably redundant ones that can provide more coverage. We also see that S_{wt_len} and S_{wt_loglen} are similar in performance with S_{wt_loglen} marginally outperforming S_{wt_len} when $\sigma_{gap} > 2$. Since S_{wt_len} uses the raw *path length* in its scoring function, it may be inflating the path scores of long but insignificant paths. S_{wt_loglen} scales down the path length, thus providing a reasonable tradeoff between redundancy and the length of the selected path. The three scoring functions are not influenced by different levels of σ_r as shown in Figure 3.

Effect of gap setting (σ_{gap}). Now, we will examine the effect of σ_{gap} on the generated summaries. Based on Figure 2, we see that setting σ_{gap}=1 yields in relatively low performance. This is because σ_{gap}=1 implies immediate adjacency between the PIDs of two nodes and such strict adjacency enforcements prevent redundancies from being discovered. When σ_{gap} is increased to 2, there is a big jump in performance, after which improvements are observed in smaller amounts. A very large gap setting could increase the possibility of generating ill-formed sentences, thus we recommend that σ_{gap} is set between 2-5.

Effect of redundancy requirement (σ_r) . Figure 3 shows the ROUGE scores at different levels of σ_r. It is clear that when $\sigma_r > 2$, the quality of summaries is negatively impacted. Since we only have about 100 sentences per review document, $\sigma_r > 2$ severely restricts the number of paths that can be explored, yielding in lower ROUGE scores. Since the scoring function can account for the level of redundancy, σ_r should be set according to the size of the input data. For our dataset, $\sigma_r = 2$ was ideal.


"About food at Holiday Inn, London"

Human summaries:
[1] Food was excellent with a wide range of choices and good services.
[2] The food is good, the service great. Very good selection of food for breakfast buffet.

Opinosis abstractive summary:
The food was excellent, good and delicious. Very good selection of food.

Baseline extractive summary:
Within 200 yards of leaving the hotel and heading to the Tube Station you have a number of fast food outlets, highstreet Restautants, Pastry shops and supermarkets, so if you did wish to live in your hotel room for the duration of your stay, you could do.......

"What is free at Bestwestern Inn, San Francisco"

Human summaries:
[1] There is free WiFi internet access available in all the rooms.. From 5-6 p.m. there is free wine tasting and appetizers available to all the guests.
[2] Evening wine reception and free coffee in the morning. Free internet, free parking and free massage.

Opinosis abstractive summary:
Free wine reception in evening. Free coffee and biscotti and wine.

Baseline extractive summary:
The free wine and nibbles served between 5pm and 6pm were a lovely touch. There's free coffee, teas at breakfast time with little biscotti and, best of all, from 5 till 6pm you get a free wine 'tasting' reception which, as long as you don't take……

Figure 5: Sample results comparing Opinosis summaries with human and baseline summaries.

Effect of collapsed structures and duplicate elimination. So far, it has been assumed that all features used in Opinosis are required to generate reasonable summaries. To test this hypothesis, we use Opinosis$_{best}$ as a baseline and then we turn off different features of Opinosis. We turn off the *duplicate elimination feature*, then the *collapsible structure feature*, and finally *both*. Figure 4 shows the resulting precision-recall curve. From this graph, we see that without duplicate elimination and when collapsing is turned off, the precision is highest but recall is lowest. No collapsing implies shorter sentences and thus lower recall, which is clearly reflected in Figure 4. On top of this, if duplicates are allowed, the overall information coverage is low, further affecting the recall. Notice that the presence of duplicates with the collapse feature turned on results in very high recall (even higher than the baseline). This is caused by the presence of similar phrases that were not eliminated from the collapsed candidates, resulting in long sentences that artificially boost recall. The Opinosis baseline which uses duplicate elimination and the collapsible structure feature, offers a reasonable tradeoff between precision and recall.

Readability of Summaries. To test the readability of Opinosis summaries, we conducted a *readability test* (described in Section 5) using summaries generated from Opinosis$_{best}$. A human assessor picked the 2 least readable sentences from each of the 51 test sets (based on 51 summaries). Collectively, there were 565 sentences out of which 102 were Opinosis generated. Out of these, the human assessor picked only 34 of the sentences as being least readable, resulting in an average readability score of 0.67. This shows that more than 60% of the generated sentences are indistinguishable from human composed sentences. Of the 34 sentences with problems, 11 contained *no information* or were *incomprehensible*, 12 were *incomplete* possibly due to false positives when the sentence validity check was done, and 8 had *conflicting information* such as '*the hotel room is clean and dirty*'. This happens due to mixed feelings about the same topic and can be resolved using sentiment analysis. The remaining 3 sentences were found to contain *poor grammar*, possibly caused by the gaps allowed in finding redundant paths.

Sample Summaries. Finally, in Figure 5 we show two sample summaries on two different topics. Notice that the Opinosis summaries are concise, fairly well-formed and have closer resemblance to human summaries than to the baseline summaries.

7 Conclusion

In this paper, we described a novel summarization framework (Opinosis) that uses textual graphs to generate abstractive summaries of highly redundant opinions. Evaluation results on a set of review documents show that Opinosis summaries have better agreement with human summaries compared to the baseline extractive method. The Opinosis summaries are concise, reasonably well-formed and communicate essential information. Our readability test shows that more than 60% of the generated sentences are no different from human composed sentences.

Opinosis is a flexible framework in that many of its modules can be easily improved or replaced with other suitable implementation. Also, since Opinosis is domain independent and relies on minimal external resources, it can be used with any corpus containing high amounts of redundancies.

Our graph representation naturally ensures the coherence of a summary, but such a graph emphasizes too much on the surface order of words. As a result, it cannot group sentences at a deep semantic level. To address this limitation, we can use a similar idea to overlay parse trees and this would be a very interesting future research.

8 Acknowledgments

We thank the anonymous reviewers for their useful comments. This paper is based upon work supported in part by an IBM Faculty Award, an Alfred P. Sloan Research Fellowship, an AFOSR MURI Grant FA9550-08-1-0265, and by the National Science Foundation under grants IIS-0347933, IIS-0713581, IIS-0713571, and CNS-0834709.

References

[Barzilay and Lee2003] Barzilay, Regina and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 16–23, Morristown, NJ, USA. Association for Computational Linguistics.

[DeJong1982] DeJong, Gerald F. 1982. An overview of the FRUMP system. In Lehnert, Wendy G. and Martin H. Ringle, editors, *Strategies for Natural Language Processing*, pages 149–176. Lawrence Erlbaum, Hillsdale, NJ.

[Erkan and Radev2004] Erkan, Günes and Dragomir R. Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

[Finley and Harabagiu2002] Finley, Sanda Harabagiu and Sanda M. Harabagiu. 2002. Generating single and multi-document summaries with gistexter. In *Proceedings of the workshop on automatic summarization*, pages 30–38.

[Jing and McKeown2000] Jing, Hongyan and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Lerman et al.2009] Lerman, Kevin, Sasha Blair-Goldensohn, and Ryan Mcdonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09)*.

[Lin and Hovy2003] Lin, Chin-Yew and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proc. HLT-NAACL*, page 8 pages.

[LIN2004a] LIN, Chin-Yew. 2004a. Looking for a few good metrics : Rouge and its evaluation. *proc. of the 4th NTCIR Workshops, 2004*.

[Lin2004b] Lin, Chin-Yew. 2004b. Rouge: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain*.

[Lu et al.2009] Lu, Yue, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *18th International World Wide Web Conference (WWW2009)*, April.

[Mihalcea and Tarau2004] Mihalcea, R. and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*, July.

[Pang and Lee2004] Pang, Bo and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278.

[Pang et al.2002] Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86.

[Radev and McKeown1998] Radev, DR and K. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500.

[Radev et al.2000] Radev, Dragomir, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *In ANLP/NAACL Workshop on Summarization*, pages 21–29.

[Radev et al.2002] Radev, Dragomir R., Eduard Hovy, and Kathleen McKeown. 2002. Introduction to the special issue on summarization.

[Saggion and Lapalme2002] Saggion, Horacio and Guy Lapalme. 2002. Generating indicative-informative summaries with sumum. *Computational Linguistics*, 28(4):497–526.

[Snyder and Barzilay2007] Snyder, Benjamin and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL*, pages 300–307.

[Titov and Mcdonald2008] Titov, Ivan and Ryan Mcdonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL-08: HLT*, pages 308–316, Columbus, Ohio, June. Association for Computational Linguistics.