A correction was made in table 2. The original was published in the Proceedings of the ACL2004 Workshop *Text Summarization Branches Out*, Barcelona, Spain, July 25-26, 2004.

Hybrid Text Summarization: Combining external relevance measures with Structural Analysis

Gian Lorenzo Thione, Martin van den Berg, Livia Polanyi, Chris Culy
FX Palo Alto Laboratory
3400 Hillview Ave, Bldg. 4
Palo Alto, CA 94304
{thione|vdberg|polanyi|culy}@fxpal.com

Abstract

In this paper, a novel linguistically advanced text summarization system is described for reducing the minimum size of highly readable variable-sized summaries of digitized text documents produced by text summarization methods that use discourse analysis to rank sentences for inclusion in the final summary. The basic algorithm used in FXPAL's PALSUMM text summarization system combines text structure methods that preserve readability and correct reference resolution with statistical methods to reduce overall summary length while promoting the inclusion of important material.

1 Introduction

In this paper, we present algorithms to address the shortcomings of both purely structural and purely statistical methods of sentence extraction summarization. We present the PALSUMM hybrid summarization algorithms that use structural methods based on discourse parsing to construct a representation of the text, apply conventional statistical methods to identify salient information (See discussion and references in Marcu 2003) and then construct a partial discourse tree that includes the information identified as most salient along with the text at all nodes dominating that salient information. Optionally, sentence compression techniques are applied to the resulting summary to further compress text length. (Grefenstette, 1998; Knight and Marcu, 2002).

The novelty of our approach lies in combining text structural methods with sentence extraction methods which evaluate relevance on the basis of external factors such as lexical frequency or lexical field information in the specific document, in related or documents in general or, alternatively by matching lexical items in a query against lexical items in a document. The sentences selected by the external oracle are then provided context for anaphora resolution and reference interpretation through inclusion of hierarchically superordinate information from the structural tree.

2 The PALSUMM System

PALSUMM summarization algorithms operate on data structures generated by FX Palo Alto's Linguistic Discourse Analysis System (LIDAS). LIDAS is a computational discourse parser implementing the Unified Linguistic Discourse Model (U-LDM). A description of the LIDAS system and the U-LDM as well as a summary of an article from the *New Yorker* are described in earlier work (Polanyi et al, 2004a, b, Thione 2004). Due to space limitations we can only sketch the main points of the system here.

The LIDAS parser itself is purely symbolic. It parses a text discourse segment by discourse segment to construct a tree that captures discourse continuity and accessibility relations between the segments. The tree identifies what discourse constituents are available for further development and what information given by discourse constituents is available to be referred to. We use the fact that the resulting tree encodes (semantic) accessibility relations between the segments, and not rhetorical relations, to guarantee that the pruning algorithm used to summarize preserve antecedents for anaphors thus fostering readability.

The basic units of this theory (Basic Discourse Unit or BDUs) are the syntactic reflexes of linguistically realized minimal semantic unit of meaning[1] or functions, interpreted relative to the context given by the preceding discourse. To identify the BDUs in a text, LIDAS relies on the Xerox Linguistic Environment to parse sentences from a text (Maxwell and Kaplan, 1989). After sentential parsing is complete, the XLE sentence parse trees are segmented into BDUs using a set of robust sentence and discourse level rules described in detail in Polanyi et al 2004a, b. After parsing, BDUs (which need not be contiguous) are recombined into one or more discourse trees corresponding to (parts of) the sentence, called BDU-trees.

For each BDU-tree, one BDU, normally the main clause of a sentence or a compound unit of discourse directly derived from it, is designated as the Main-BDU (M-BDU) and is represented by the root node of the BDU-tree. The entire BDU-tree is attached as a unit to the emerging Open Right Tree representation of the structure of the discourse by relating syntactic, semantic and lexical information in the M-BDU (and preposed adverbial modifiers, clauses and "cue" words) to information available in nodes along the right edge of the tree using formal linguistic discourse attachment rules involving relationships among semantic, syntactic and lexical information to compute both the site of attachment and the attachment relation.

Although a full discussion of these rules lies beyond the scope of this paper, Table 1 sketches some simple principles which are both language and domain independent.[2]

These rules are weighted and ordered in application, and multiple rules may "vote" for the same or different attachment points and discourse relations. The precise relationships among the rules remains a subject for future research.

[1] We understand a minimum unit of *Meaning* to communicate information about not more than one "event" or state of affairs in a "possible world" of some type (roughly event-type predicates); while a minimal *Functional* unit encodes information about how previously occurring (or possibly subsequent) utterances relate structurally, semantically, interactionally or rhetorically to other units in the discourse or context in which the discourse takes place (Greetings, discourse PUSH/POP markers, connectives etc. are all Functional segments).

[2] One reviewer remarked, quite correctly: "how a sentence is attached to the emerging representation of the structure of the discourse … is the heart of the algorithm". This issue is discussed in detail in Polanyi et al., 2004a,b ; Thione et al. 2004.

Evidence attachment is a subordination
Syntactic promotion: If the *subject* of an M-BDU co-refers with the *object* of the AP.
Sub-cases: If the subject of the M-BDU refers to a sub-case of the subject of the AP. Sub-cases include subsets (*all children*/*some children*), sub-types *(people*/*children*), etc.
Verbal properties: If the tense, aspect, modality or genericity of the verbs are different.

Evidence attachment is a coordination
Narrative: If the verbs express events.
Lists: If the subjects are synonyms/antonyms and/or the syntactic structures of M-BDU and AP are sufficiently similar.

Table 1. Some simple examples of discourse principles.

The U-LDM is similar in form to RST, but its primitives are rather different. Whereas RST takes rhetorical relations as primitives, the LDM takes its primitives from syntactic structure. The ontology of LDM relations has three top relations: coordination, subordination and n-ary. **Coordinations** express a symmetric relationship between the children, including: lists, narratives, etc. **Subordinations** express an asymmetric relationship between children, including: elaborations, interruptions, etc. Finally, **n-aries** include a number of cases where the structure is defined by specific language constructions. Note that these constructions are not arbitrary, and often follow from (sentence) syntactic constructions. Examples include scope setting operators and units (*when john comes, he will be happy*), and more or less fixed forms like greetings and question-answer pairs, etc. It is the practice to also consider genre-specific structures (e.g. "a paper consists of a title, an abstract, an introduction, some sections, a conclusion, and references") to be n-aries.

Because to characterize the large structure of the discourse we only need to refer to coordinations, subordinations and n-aries, it is often claimed that the number of relations in the LDM is much smaller than in RST, even though strictly speaking this depends on which versions of LDM and RST one compares. The real difference between the two theories lies in the rather different origins of the rules.

All non-terminal nodes in U-LDM trees are first class citizens and contain, in addition to a node label, content and context information inherited from child nodes. Under RST only terminal nodes have content; non-terminal nodes that represent the relationships obtaining among spans of the text longer than one sentence are labeled for the relationship between daughter nodes only.

As in Summarist (Hovy and Lin, 1999), once the source text has been parsed and a discourse tree incrementally constructed, text summarization algorithms are applied to the resulting tree. However, the difference between constructing a semantic rather than a rhetorical representation of the text accounts for how PALSUMM summaries preserve readability and reference resolution: because the entire analysis involves matching semantically defined contextual units to the appropriate contexts available on the tree, nodes that structurally dominate other nodes necessarily contain the information needed to contextually interpret the dominated units.

3 Pruning PALSUMM Trees

Summarization methods based on discourse structure all rely on assigning a numeric value to all intermediate and leaf nodes encoding their **importance**, based on the labels at the nodes. The difference between different methods originates in the different ways this importance measure is calculated. Because RST and U-LDM trees differ, there are key differences between the simple pruning methods applied to U-LDM trees as opposed to RST trees.

Under the U-LDM theory of discourse, the asymmetric relationship expressed by subordinations implicitly encodes a notion of importance. The subordinated child elaborates or further qualifies the head, or temporarily interrupts the flow of discourse. Subordinated material is almost always less important to the main line of the text than subordinating material: the level of embedding thus gives a first rough measure of importance of a unit of discourse.

Our original summarization algorithm, SymTrim, used the level of embedding directly. It pruned the tree at a given level of embedding, and generated a summary based on the span of the remaining tree. The number of possible summary lengths, however, was restricted to the number of embedding levels, resulting in a discreet number of summaries of a fixed length, often ones longer than desired. This led to a need for more subtle pruning algorithms.

3.1 Solving the SymTrim restriction

There are two theoretic problems that underlie the practical problems of SymTrim. First, across the board pruning at a fixed level is of limited utility. If two sections of a document differ significantly in size, the larger section will have more space for deeper sub-trees. Consequently, units of equal importance may occur at deeper levels of larger sub-trees.

Secondly, no method that relies solely on purely structural information can determine what parts of the document contain important information. For this an approximation the meaning of the units is needed. A description of the relationships among them does not suffice.

We address the first issue by not trimming the tree at an absolute level, but at a level relative to the depth of the sub-branch in which a node is found. We address the second issue by skewing the pruning level using statistical methods[3] as an oracle to indicate relative importance.

3.2 Score Adjustment and Percolation

We assign every node a relative depth $T(l)$, based on the local and global structure of the tree branch to which it belongs, calculated as follows: (1) establish the absolute depth $D(l)$ of each node, (2) calculate an **embedding branch weight** $W(l)$ by percolating the value of $D(l)$ up from the leaves according to the percolation algorithm outlined in Figure 1, (3) assign each node a relative depth $T(l) = 1 - (D(l) - 1) / W(l)$.[4]

We also compute a statistical score that approximates the "semantic importance" of every node. To do so, we begin by seeding every leaf node l with a statistical seed $S(l)$ using the MEAD statistical summarizer. Each segment is scored by MEAD in the context of the full document, with a score that mirrors its judgment of the relevance of that segment for a summary.

[3] We use the publicly available MEAD (Radev et al. 2003). Adopting a sentence extraction approach, it is capable of assigning scores to each and every sentence. PALSUMM does its own discourse segmentation and sends the segments to MEAD as if they were sentences. This allows us to assign independent scores to discourse segments, thus enabling sub-sentential summarization (segment-extraction vs. sentence extraction) and yielding more compressed yet still highly readable summaries.

[4] The expression for $T(l)$ was chosen to assign the top node relative depth 1.

```
1. For seeding $V$, each leaf node $l$ is assigned an a priori score $V(l)$.
2. Repeat for each node $c_0$ with children $c_1 \ldots c_n$, and relation type $R$ until no values change:
    2.1 Percolate or maintain highest score: $V(c_0) := \max_i (V(c_i))$ , $0 \le i \le n$
    2.2 Percolate highest score downwards into non-subordinated nodes:
        if $R$ is subordination and $c_i$ is the head of $n$: $V(c_i) := V(c_0)$ if $V(c_i) < V(c_0)$
        if $R$ is coordination or n-aries: for all $i \le n$, $V(c_i) := V(c_0)$ if $V(c_i) < V(c_0)$,
```

Figure 1. General percolation algorithm. Both statistical seeds (V=S) and structural seeds (V=W) are percolated according to this algorithm, resulting in values $S(n)$ and $T(n)=1 - (D(n) - 1)/W(n)$ for nodes n.

MEAD's metrics include: TF/IDF cosine similarity between a segment and the document – optionally skewed towards a query entered by the user, the relative position of a segment within the document, an adverse score against segments deemed as too similar to the current summary, and our own implementation of a feature concerning the presence of certain cue words (Hirschberg, 1993). After scoring, the values are percolated up through the tree, as before. During percolation of both structurally and statistically obtained scores, the new value of a node that receives a higher score from a child node is percolated downwards through all non-subordinated children. Children of coordinations and n-aries are considered equally relevant and scored equally, whereas subordinated children are less relevant then subordinating ones.[5] After percolation we normalize the statistical scores, dividing by the maximum occurring value.

Different summarization algorithms result from the choice of seeding algorithms and methods of combining scores. Note that the percolation algorithm in Figure 1 respects structural embedding by always assigning lower or equal scores to subordinated nodes.

3.3 Pruning Algorithms

In order for summaries to maintain textual coherence and readability, constituents that contain contextual or referential information necessary to interpreting other constituents selected for the summary must be marked for inclusion. For any node, this information is available in nodes that are siblings of the same coordination or n-ary, and in nodes that dominate it through subordination-type relations. As long as the score assigned to nodes respects subordinations as in Figure 1, any pruning of the tree that excludes constituents whose final relevance score is smaller than a chosen value is guaranteed to preserve the antecedents for the anaphora in the text, preserving well-formedness of the resulting tree and the readability of the summary it yields.

In Table 2 we list four different final score assignments, based on the embedding level of the nodes (L), their percolated statistical score (S) and the percolated relative depth score (T).

SymTrim	$F = {}^{1}/_{L}$
SymTrim-R	$F = T$
HybReduce	$F = {}^{1}/_{L} * S$
HybReduce-R	$F = T * S$

Table 2. Different scoring algorithms.

After scores are calculated and combined, a relative threshold is computed by sorting the set of constituent by final score and identifying the cutoff value that more closely approximates the request of the user in terms of desired summary length. Note that the root node will always have normalized score 1 and will therefore always be included in a full summary.[6]

4 Evaluating PALSUMM

The PALSUMM corpus contains over 300 FXPAL Technical Reports in a wide range of domains. The Reports vary in size from 10 to 30 pages. To evaluate the readability of summaries and create a baseline for evaluating the SymTrim-R and HybRduce-R algorithms, we conducted a small pilot study on five documents selected from the corpus. The documents were hand-annotated with their U-LDM discourse structures. The SymTrim-R and HybReduce-R variants were then automatically applied to these

[5] In a modified percolation scheme, downward percolation is restricted to preceding siblings in discourse-level coordination nodes. This is a result of the fact that contextual information necessary to preserve readability and referential integrity must appear before access.

6 In other applications of the algorithms described here, where the purpose is not that of retrieving a full summary of a document but rather that of building the necessary minimal context for interpreting a certain selected discourse constituent, percolation is only limited to the immediate surrounding context, where certain relations (usually ad-hoc binaries) constitute a barrier to further percolation towards upwards constituent.

discourse structures, and the summaries submitted to a panel of 12 non-experts. The panelists were asked to judge the summaries on a 6-point scale for readability by answering a set of questions including "How readable is this summary?" and "Did you get confused at any point in the summary?" The initial results suggest that the discourse algorithms produced readable summaries and that the relative effectiveness of the discourse algorithms varies according to some still to be determined property of the documents.

5 Conclusion

Structural sentence extraction systems including Summarist and PALSUMM that create summaries by choosing sentences or parts of sentences corresponding to nodes at a given level of depth of a tree structured representation of the structure of the text produce excellent summaries that preserve the style and "flavor" of the original text. However, the summaries constructed may be longer than needed, including information that could be omitted without serious loss of informativity[7]. The excessive length results from the top-down nature of standard structural extraction algorithms which start by choosing the top context and then includes every possible sub-context down to a certain level.

In this paper, we have proposed hybrid algorithms which capitalize on the strengths of these methods while compensating for their limitations by proposing additional manipulations on the base trees. In our view, the value of the summarization methods described here, is the ability to compress a summary further without substantial loss of informativity. For summaries, especially those designed for display on various sized devices, the work presented here constitutes an advance in the state of the art.

6 Acknowledgements

The authors would like to thank Dr. Sara Bly (Sara Bly Consulting) who designed and carried out the evaluation and Dr. Candace Kamm of FXPAL who provided help and guidance in the design and organization of the study.

[7] Some of this excessive length can be addressed through compressing less relevant aspects of constituent sentences as in Grefenstette, 1998; Knight and Marcu, 2002.

7 References

Gregory Grefenstette. 1998. Producing intelligent text reduction to provide an audio screening service for the blind. Working Notes of AAAI Spring Symposium on Intelligent Text Summarization. 111–118. Stanford.

Julia Hirschberg and Diane Litman. Empirical Studies on the Disambiguation of Cue Phrases. 1993. Computational Linguistics, 19-3:501-530.

Ed Hovy and C-Y. Lin. 1999. *Automated Text Summarization in SUMMARIST. In I. Mani and M.* Maybury (eds), Advances in Automated Text Summarization. Cambridge: MIT Press, pp. 81–94.

Kevin Knight and Daniel Marcu. 2000. Statistics Based Summarization — Step One: Sentence Compression. In: AAAI-2000 Proceedings. Pp. 703-710. Austin TX.

Daniel Marcu. 2000. The Theory and Practice of Discourse Parsing and Summarization. The MIT Press. Cambridge, MA.

Daniel Marcu. 2003. Automatic Abstracting, Encyclopedia of Library and Information Science, pp.245-256, 2003.

John Maxwell and Ronald M. Kaplan. 1989. An overview of disjunctive constraint satisfaction. In Proceedings of the International Workshop on Parsing Technologies, Pittsburgh, PA.

Livia Polanyi, Martin van den Berg, Chris Culy, Gian Lorenzo Thione, David Ahn. 2004a. A Rule Based Approach to Discourse Parsing. 5th SigDial Workshop.

Livia Polanyi, Martin van den Berg, Chris Culy, Gian Lorenzo Thione 2004b. Sentential structure and discourse parsing. Discourse Annotation Workshop, ACL04.

Dragomir Radev, Timothy Allison, Sasha Blair-Goldensohn, and John Blitzer, Arda Çelebi, Elliott Drabek, Wai Lam, Danyu Liu, Hong Qi, Horacio Saggion, Simone Teufel, Michael Topper, Adam Winkel. 2003. "The MEAD Multidocument Summarizer". `http://www.summarization.com/mead/`

Radu Soricut and Daniel Marcu. 2003. Sentence Level Discourse Parsing using Syntactic and Lexical Information. Proceedings of HLT/NAACL, May 27-June 1, Edmonton, Canada.

Gian Lorenzo Thione, Martin van den Berg, Livia Polanyi, Chris Culy. 2004. "LiveTree: An Integrated Workbench for Discourse Processing". Discourse Annotation Workshop, ACL04.