SUMMaR: Combining Linguistics and Statistics for Text Summarization

Manfred Stede and **Heike Bieler** and **Stefanie Dipper** and **Arthit Suriyawongkul**[1]

Abstract. We describe a text summarization system that moves beyond standard approaches by using a hybrid approach of linguistic and statistical analysis and by employing text-sort-specific knowledge of document structure and phrases indicating importance. The system is highly modular and entirely XML-based so that different components can be combined easily.

1 INTRODUCTION

Most text summarization systems nowadays take the approach of *sentence extraction*: Following the determination of most relevant terms in the document, sentences are selected that contain such relevant terms, and the sequence of these sentences is deemed the summary. Term relevance is traditionally measured by relative frequencies in different corpora; various extensions of this basic method are being used, such as the position of sentences in documents (e.g., prefer sentences at the beginning of the document or at the end).

In contrast to generic summarization systems that emphasize robustness (produce *some* summary for *any* text), our project trades breadth against quality and argues that good summarization should be *text-sort-specific*. The idea is that the notion of 'importance' is relative to the sort of text; for example, in a news story, the most important information is the event stated right at the beginning, whereas for an op-ed piece, the most important information is the topic plus the opinion the author conveys. Our approach is to use a declarative representation of text-sort knowledge that supplements the standard extraction technique in creating the summary. With SUMMaR[2], we created a highly modular, XML-based architecture that allows for plugging in various analysis tools and combining their results flexibly. A GUI displays interim analysis results as well as the final summary, thus facilitating further fine-tuning. SUMMaR is an implemented prototype with some components still under improvement. Among the text sorts we are working with (news, commentary, scientific papers, reviews), reviews are currently most prominent; in particular, here we use *movie reviews* to illustrate our approach.

2 OVERVIEW OF SYSTEM ARCHITECTURE

For text analysis in any application, flexible module combination is an important desire. We achieve this by consistently using *standoff annotation*: For each information layer, a single XML representation is created, which points either to the tokenized source document, or to another layer (e.g., PoS layer points to source text, whereas syntax-chunk layer points to PoS layer). Details of the XML format can be found in [Dipper 2005]. The analysis modules can then inspect layers individually or create combinations of them by merging them to a standard inline XML representation; for this purpose, we use the approach of [Witt et al. 2005].

SUMMaR processes documents in plain text, XML, and (to some extent) HTML. The first step is to map these input formats to a common XML format representing the basic layout of the text, i.e., the structure of headlines and paragraphs (see Section 3). Then, tokenization identifies sentence and token boundaries. This is input to a syntactic parser followed by co-reference analysis; these steps are taken to identify cohesion problems in the resulting text extracts, which will later be resolved by partial re-generation.

For developing the system, it has proven useful to employ a GUI that displays the source text and, on demand, the various annotations as produced by analysis modules. This could to some extent be achieved by configuring a workbench such as GATE (http://gate.ac.uk), but we found it more comfortable to write our own web-based GUI geared specifically to the needs of summarization, so that, e.g., the most relevant sentences of the text can be highlighted dynamically when changing relevance thresholds, etc.

We claim that for quality summarization, the system needs to know the *text sort* of the document under consideration: It makes a difference for the summarization process whether, for example, a news report or a company white paper or a review article is being handled. The key is to encode these differences in a declarative knowledge source so that the processing modules can in fact stay the same. One piece of text-sort-specific information is the *target template* for the summary: For movie reviews, it should contain the source of the review, movie title, the assigned overall rating (if any), indication of the story, and indication of more specific judgements of the reviewer. Such a task-oriented summarization is in effect a mixture of traditional information extraction ("find the title; find the overall rating") and traditional summarization ("condense the description of the story, and the evaluative portions, if any").

Our module for *content structure* identification employs text-sort knowledge in order to assign labels to paragraphs similar to the *zones* in [Teufel and Moens 1997]. When zones have been identified, their contents are either directly transferred to the target template (for movie reviews: title and overall rating), or they are subject to *sentence relevance* calculation, as described in Section 4.

3 DOCUMENT STRUCTURE KNOWLEDGE

The basis for our text analysis modules is an XML format that marks paragraphs, headings, and emphasized text portions (e.g. italics, bold face). It represents the logical document structure in a sense similar to a Latex source, albeit much simpler. Considerably less simple is the procedure to derive this logical structure from different types of

[1] University of Potsdam, Germany, email: stede|bieler|dipper|art@ling.uni-potsdam.de

[2] Project funded by Bundesministerium für Bildung und Forschung, grant 03WKH22. The authors are responsible for the content of this paper.

GOOD BYE, LENIN! (2003)
***1/2 (out of four)

starring Daniel Brühl, Kathrin Sass, Chulpan Khamatova, Maria Simon
screenplay by Bernd Lichtenberg and Wolfgang Becker
directed by Wolfgang Becker

Good bye, Lenin! is that rarest of beasts, a popular film that's actually about something. Detailing a former East German's mixed emotions at the demise of communism, it's precise in its modelling of a historical turning point without either trivializing or preaching. One doesn't have to pick out the plums of insight from a thin pudding of plot: the elements of analysis and narrative fuse so seamlessly that they carry you along, making a happy medium that is supremely satisfying. One wishes that Hollywood could turn out a film such as this, which, for all its movie-movie gusto, deals with complex issues real people have to deal with, making its huge success back home a heartening sign in this age of *Amélie* and cultural amnesia.

Alex (Daniel Bruhl) is a young man who has watched the DDR crumble before his very eyes. One day he's clashing with Stasi thugs over civil rights, the next his country is a memory and he has to learn a new set of

Figure 1. Excerpt from movie review webpage

source documents. We process different kinds of XML formats (from our partner projects), plain text, and to some extent HTML. The current implementation of our logical structure extractor is written in Python, with optional help from an XSL Transformation engine. For the case of plain text documents, the only layout information is the use of spacing, vertically and horizontally, and some creative typography like using asterisks around characters for **emphasis**. We built a set of heuristic rules that identify paragraph breaks on the basis of average line length and presence of single and double line breaks in the document; similarly, headings are identified on the basis of length, cue words (like numbering), and surrounding line breaks.

Having derived the logical structure, we enrich it with information on content zones. The inventory of zones has been determined by a corpus study. For movie reviews, it includes labels such as Title, Rating, DescStory, CommentOnStory, CommentOnActors, Credits, LegalNotice. Our corpus includes 50 reviews from 10 different sources, and so the documents differ widely, both in length and in structure. There are regularities, however: Title is at the beginning; legal notice, if any, at the end; overall rating either towards beginning or towards end; story is usually told continuously (no intervening non-story paragaraphs), and so on. We have formalised the regularities not as a document grammar, but as a set of local rules, and the process of assigning each paragraph of the logical structure a content-label is one of constraint satisfaction with optimization. These steps of inferring logical and content structure are described in detail in [Stede and Suriyawongkul, to appear].

4 SUMMARY PRODUCTION

As mentioned earlier, the first items of the summary template (title and rating) are determined by simple information extraction rules from the content structure representation. For the indicative summary of the story, only those paragraphs that have received a 'Story' label are submitted to our sentence relevance calculation, which is based on computing term weights with the $tf * idf$ method, where the relation of the frequency of the term in the specific document and the number of documents in which this terms occur, plays a role. As to the notion of 'term', we are experimenting with wordforms (easy to detect, but with inflection problem), stems as determined by a Porter-stemmer (resolves inflection problems, but overgenerates), $n$grams (all combinations of n adjacent characters in the text; a very robust method that can cope with spelling errors), and lemmas (for applications where a lemmatizer for the language is available). The $tf * idf$ measure counts how often a term occurs in the document (term frequency tf), and in how many documents of our corpus for the specific text sort (document frequency df). Terms with high term frequency and a low document frequency are the most indicative terms, because they reflect the specific topic (in this case, of the movie).

For a text like that of Figure 1, the method would find terms such as *East German's, Lenin, amnesia* – but it would not find terms such as *popular, seamlessly, satisfying*. The latter are typical for any opinion-conveying text and thus will be frequent in a corpus of reviews of any kind, including movies. In order to determine the reviewer's opinion (which should be reflected in a good summary), we look for sentences including such evaluative expressions (currently from a hand-collected list, but this is being replaced by a trained statistical classifier), and use these sentences as the basis for the final entry of the summary template, 'Evaluation'. This is an important difference to other approaches to sentence extraction: text-sort-specific terms (here: evaluative ones) are used to determine *important* sentences, whereas the $tf * idf$ measure is used to determine sentences that are *indicative* of the content. The list of evaluative phrases is thus part of the text-sort knowledge (albeit in this case shared between movie reviews and all kinds of other product reviews), which collectively is represented as a distinct XML document. It contains the information on necessary and possible zones for the text sort, and terms acting as indicators for zones, such as the above-mentioned evaluative terms, or a number of asterisks for the 'Rating' Zone (e.g., in Figure 1: "***1/2 (our of four)". In the end, the filled output template for the text in Figure 1 would look like this:

`Source:` Film Freak Central, Feb 22, 2006
`Title:` Good Bye, Lenin (2003)
`Rating:` ***1/2 (out of four)
`Story:` <*extract from story description*>
`Evaluation:` <*extract from evaluative portions*>

5 CONCLUSION

Document analysis is to a large extent a matter of statistical relevance calculations, but it should also be driven by information on document structure. We have illustrated this for the case of text summarization: Given loosely-structured documents consisting of a fairly predictable set of content zones (but not in a fixed order; otherwise it is a highly-structured document), we propose to first identify this content structure as a useful step of preprocessing. For summarization, this helps to make sure that portions of all relevant zones are actually part of the result.

REFERENCES

[Dipper 2005] S. Dipper. "XML-based Stand-off Representation and Exploitation of Multi-Level Linguistic Annotation." In: Proc. of Berliner XML Tage 2005, pp. 39-50, Berlin, Germany.

[Stede and Suriyawongkul, to appear] M. Stede, A. Suriyawongkul. "Identifying Logical Structure and Content Structure in Loosely-Structured Documents." To Appear.

[Teufel and Moens 1997] S. Teufel, M. Moens. "Sentence extraction as a classification task." In: Mani and Maybury, eds.: *Advances in Automatic Text Summarization*, MIT Press, 1997.

[Witt et al. 2005] A. Witt, H. Lüngen, F. Sasaki, D. Göcke. "Unification of XML Documents with Concurrent Markup." *Literary and Linguistic Computing* 20(1), 103-116, 2005.