Robust Generic and Query-based Summarisation

Horacio Saggion
Kalina Bontcheva
Hamish Cunningham
Department of Computer Science
University of Sheffield
211 Portobello Street - Sheffield - S1 4DP
England - United Kingdom
{saggion, kalina, hamish}@dcs.shef.ac.uk

Abstract

We present a robust summarisation system developed within the GATE architecture that makes use of robust components for semantic tagging and coreference resolution provided by GATE. Our system combines GATE components with well established statistical techniques developed for the purpose of text summarisation research. The system supports "generic" and query-based summarisation addressing the need for user adaptation.

1 Introduction

Two approaches are generally considered in automatic text summarisation research: the shallow sentence extraction approach and the deep, understand and generate approach (Mani, 2000). Sentence extraction methods are quite robust, but sentence extracts suffer from lack of cohesion and coherence. Methods that identify the essential information of the document by either information extraction or text understanding and that use the key information to produce a new text, lead to high-quality summarisation (Paice and Jones, 1993; Saggion and Lapalme, 2002) but suffer from the knowledge-bottleneck problem: adapting information extraction rules, templates, and generation grammars to new tasks or domains is time consuming. An alternative to these approaches is to use combination of robust techniques for semantic tagging together with statistical methods (Saggion, 2002).

Here, we present a summarisation system that makes use of robust components for semantic tagging and coreference resolution provided by GATE (Cunningham et al., 2002). Our system combines GATE components with well established statistical techniques developed for the purpose of text summarisation. The result is the sentence extraction system shown in Figure 1, the relevant sentences of the document are highlighted in the GATE user interface. The figure also shows semantic information identified within the document (e.g., named entities). All summarisation components developed as part of this research are made available as a Java Library for research purposes [1].

2 The Summariser

Our system is a pipeline of linguistic and statistical components. Some of them are based on ANNIE, a free IE system available as part of GATE[2]. A number of components have been developed for the purpose of this research and they make use of the information produced by ANNIE. These modules can be coupled and decoupled to produce different summarisation configurations. The system supports "generic" and query-based summarisation addressing the need for user adaptation.

The input to the process is a document, a compression rate, and a query (optional). The document is automatically transformed by a text structure analyser into a GATE document: a structure containing the "text" of the original input and a number of annotation sets. Each component in the pipeline adds new information to the document in the form of new

---

[1]The summarisation components can be obtained by contacting Horacio Saggion http://www.dcs.shef.ac.uk/~saggion.
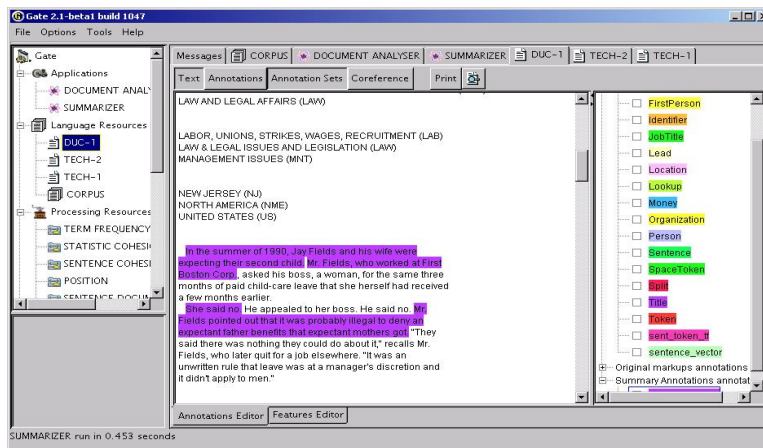
[2]http://gate.ac.uk/.

Figure 1: Summarisation results

annotations or document features. Some summarisation components compute numerical features for the purpose of sentence scoring. These features are linearly combined in order to produce the sentence final score.

2.1 General Purpose Components

These general purpose components are part of the ANNIE system, distributed with GATE:

- Unicode tokeniser: splits text into simple tokens, such as numbers, punctuation, symbols, and words of different types (e.g. with an initial capital, all upper case);

- Sentence splitter that identifies sentence boundaries;

- Gazetteer lookup that identifies and classifies key words related to particular entity types and help in the process of named entity recognition;

- Named entity recogniser that identifies and classifies more complex sequences of tokens in the source document. We use JAPE (Java Annotation Pattern Engine), a pattern-matching engine implemented in Java, to identify entities of type person, location, organisation, money, date, percentage, and address. For other semantic categories in particular domains, specific grammar rules can be developed.

- Part-of-speech tagging is done with an implementation of the "independence and commitment" learning approach to POS tagging;

- Morphological analyser (this module is not part of the GATE distribution), is a rule-based lemmatiser that produces an affix and root for each noun and verb in the input text.

- Coreference resolution, it is a light-weight, corpus-based approach for the resolution of named entities anaphora in text.

2.2 Summarisation Components and Scoring

These modules have been developed for the purpose of summarisation research and are made available as a library of Java classes and configuration file (i.e. creole in GATE terminology):

- Corpus statistics: token statistics including token frequency and lemma (or root) frequency are computed in this step.

- Vector space model (Salton, 1988): is used to create a vector representation of different text units. Each vector contains the tokens of the text unit and the value *token frequency * inverted document frequency*. Inverted document frequencies (i.e., distribution of tokens in a big collection) for English is computed using the British National Corpus (this information is a parameter of the summariser making possible to experiment with frequencies from different

corpora). Vector representations are produced for : (a) the whole document, (b) the lead-part of the document (the n% initial tokens of the document, where n is given as a parameter), and (c) each sentence.

- Term frequency: this module computes the value $\sum tf * idf$ for each sentence in the document. The sum is taken over the sentence tokens and normalised by the maximum term frequency over all sentences.

- Content-based analysis: this module computes the similarity between two text units by computing the *cosine* between their vector representations (other similarity metrics will be incorporated in the future). We perform the following computations:

  similarity between each sentence and the whole document;

  similarity between each sentence and the lead-part of the document;

  similarity between each sentence and its previous sentence (similarity forward);

  similarity between each sentence and its following sentence (similarity backwards);

  The similarities forward and backward are combined in a single numeric value representing how "cohesive" the sentence is to the previous and following text. We identify sentences that: (a) begin segments (they are dissimilar with the previous sentence but similar to the following sentence); (b) are in the middle of a segment (are similar to both previous and following sentences); (c) close segments (they are similar to the previous sentence but not to the following sentence); or (d) have no relation with previous or following sentences.

- Named entity statistics module: based on the output of the coreference module we compute coreference classes grouping together all mentions of the same named entity (e.g., "Bill Clinton" and "Mr. Clinton" belong to the same class). For each coreference class we identify its size and frequency (ne_freq), the sentence containing the first mention of an ele-

ment in the coreference class, and the *inverted NE frequency* (or i_ne_freq) (e.g., the ratio of the number of sentences / the number of sentences containing an element of the coreference class).

- Named entity scorer. This module performs the following computations:

  first mention of a named entity: sentences containing the first mention of a class with more than one instance receive a bonus;

  named entity density: is the ratio of the number of coreference classes in the sentences to the number of coreference classes in the text;

  in a way similar to the content based analysis of sentences, we measure the cohesiveness of sentences; based on the links named entities have in the text (e.g., forward and backward links);

  in a way similar to the term distribution scorer, we compute a composite value representing the distribution of the coreference classes in the sentence ($\sum ne\_freq * i\_ne\_freq$), this value is normalised by the maximum value obtained for all sentences.

- Sentence position: for each sentence two values are computed. Absolute position: sentence i receives the value i^{-1}. Relative position: if the sentence is at the beginning of a paragraph, this value is set to *initial*, if the sentence is at the end of the paragraph (for paragraphs with more than one sentence), this value is set to *final*, if the sentence is in the middle of the paragraph (for paragraphs with more than two sentences), this value is set to *middle*. These three values are parameters of the sentence position scorer.

- Query-based scorer: a query (e.g., string) can be specified as parameter to the summarisation process in order to boost the value of sentences which 'content' is close to the query 'content'. The query is analysed and a vector representation is produced for it. A similarity value is computed between each sentence and the query.

The final score for a sentence is computed using the following formula:

$$\sum_{i=1}^{n} value(feature_i) * weight_i$$

where the weights are obtained experimentally and constitute parameters of the summarisation process (the summariser comes with pre established weights that can be modified by the user). The scores are used to produce a ranked list of sentences. Sentences on the ranked list are included in the summary until the compression rate is reached. A module is also available that allows the user to specify "text units", section headings for example, that should be excluded from the ranked list. The annotations can be used to produce a stand-alone version of the summary.

3 Evaluation

Evaluation is an essential step of any natural language processing task. However, many research projects make use of in-house evaluation, making it difficult to replicate experiments, to compare results, or to use evaluation data for training purposes. When text summarisation systems are evaluated by comparing extracted sentences to a set of "correct" extracted sentences, then co-selection is measured by precision, recall and F-score. Gate's AnnotationDiff tool enables two sets of annotations on a document to be quantitative compared (i,e. two summaries produced by two summarisation configurations). We are making use of human annotated corpus (source documents and sets of extracts) (Saggion et al., 2002b) in order to evaluate different system configurations and to identify experimentally the best feature combination. Processing resources for content-based evaluation have already been integrated in the system (Pastra and Saggion, 2003). Future work will include the use of document-summary (non extractive) pairs (from the Document Understanding Conferences Corpus as well as from the HKNews Corpus (Saggion et al., 2002a)) and machine learning algorithms to obtain the best combination of the summarisation features, where 'extracts' will be learn based on the automatic alignment between the non-extractive summaries and their source documents. The summarisation

system presented here provides a framework for experimentation in text summarisation research. The summariser combines two orthogonal approaches in a simple way taking advantage of robust techniques for semantic tagging, coreference resolution, and statistical analysis. Our work in progress is also looking at the automatic acquisition of 'cue phases' from corpora in order to implement the indicator phrases method. Future versions of this system will contain multi-document and multi-lingual summarisation components.

References

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *ACL 2002*.

Mani, I. (2000). *Automatic Text Summarization*. John Benjamins Publishing Company.

Paice, C. D. and Jones, P. A. (1993). The Identification of Important Concepts in Highly Structured Technical Papers. In Korfhage, R., Rasmussen, E., and Willett, P., editors, *Proc. of the 16th ACM-SIGIR Conference*, pages 69–78.

Pastra, K. and Saggion, H. (2003). Colouring summaries Bleu. In *Proceedings of Evaluation Initiatives in Natural Language Processing*, Budapest, Hungary. EACL.

Saggion, H. (2002). Shallow-based Robust Summarization. In *Automatic Summarization: Solutions and Perspectives*, ATALA.

Saggion, H. and Lapalme, G. (2002). Generating Indicative-Informative Summaries with SumUM. *Computational Linguistics*.

Saggion, H., Radev, D., Teufel, S., and Lam, W. (2002a). Meta-evaluation of Summaries in a Cross-lingual Environment using Content-based Metrics. In *Proceedings of COLING 2002*, pages 849–855, Taipei, taiwan.

Saggion, H., Radev, D., Teufel, S., Wai, L., and Strassel, S. (2002b). Developing Infrastructure for the Evaluation of Single and Multi-document Summarization Systems in a Cross-lingual Environment. In *LREC 2002*, pages 747–754, Las Palmas, Gran Canaria, Spain.

Salton, G. (1988). *Automatic Text Processing*. Addison-Wesley Publishing Company.