Language Resources for Background Gathering

Horacio Saggion and Robert Gaizauskas

Department of Computer Science
University of Sheffield
211 Portobello Street - Sheffield, England, UK
Tel: +44-114-222-1947
Fax: +44-114-222-1810
{saggion,robertg}@dcs.shef.ac.uk

Abstract

We describe the Cubreporter information access system which allows access to news archives through the use of natural language technology. The system includes advanced text search, question answering, summarization, and entity profiling capabilities. It has been designed taking into account the characteristics of the background gathering task.

1. Introduction

Background gathering for journalists is the task of collecting information from a digital archive that will help to contextualise and support a breaking news event. Background information is used by journalists in a variety of tasks, including preparing for interviews, adding detail to stories, and, most explicitly, when journalists writing for newswires are asked to write a "backgrounder" to a breaking news story in order to support other journalists using the newswire. These backgrounders are coherent documents that can be read on their own, out of their production context. Essential components of such background texts are: accounts of similar events in the past (e.g. if the news event is a political resignation, then previous similar resignations might be relevant), profiles of role players in the events (e.g. a chronology of the career of the person leaving office), factual information (e.g. when and where the person was born, who the person is married to, etc.). Information retrieval is at present the default tool to support background gathering in the news room. However, the characteristics of the task also make natural language processing technologies such as question answering, information extraction, and summarization relevant in this context. We have designed and implemented the Cubreporter[1] prototype to support background gathering in a digital news archive. The system incorporates a standard information retrieval engine, as well as a question answering system and document summarization technology. Information extraction technology is also used to extract structured representations of events which are in turn used to populate a database to support similar event search.

This paper briefly describes the natural language processing technology used in the Cubreporter prototype including text processing components, semantic interpretation, summarization, profiling, and question answering.

2. The News Archive

We are currently working with a 11-year text archive (about 8.5 million stories from 1994 to 2004) provided by the Press Association (PA), the major UK domestic news wire service. Stories in the PA archive are classified into a number of topics or news categories from a controlled vocabulary representing the subject matter of the story (e.g., *Courts*, *Politics*). Within the same topic, stories are further identified by a number of free-text keywords that the journalists would assign which are called *catch-lines*. When a "news event" occurs, a reporter writes a *snap*, a line of text summarising the news and "moves" it to the wire. From that point on, stories follow an installment pattern where each installment carries an updated account of the story.

The archive has been encoded in XML following a DTD which captures not only the metadata delivered by the PA (e.g., date, news category, story topic, keywords) but also the document structure (e.g., headline, paragraph). In a given day, stories have been automatically grouped using the installment patterns and the groups displayed in the user interface when one of the stories is retrieved. A text index for the textual elements (story, headline, keywords, etc.) in the documents is created to allow on-line access via a keyword search facility. Furthermore, a paragraph index has been created to support passage retrieval in our question answering system. The indexing and the text search facilities are implemented using the Lucene Java library[2].

3. Resource Creation

Documents are analysed using GATE tools (http://gate.ac.uk) for sentence boundary identification, tokenisation, named entity recognition, part-of-speech tagging, etc. Individual stories are summarised using an in-house summarization toolkit, which produces sentence extracts (Saggion, 2002). It scores sentences based on a number of numeric-value summarization features and combines them to produce a sentence score used to rank and select sentences for the summary. The features used for the PA stories are: the sentence position, the similarity of the sentence to the catch-line, and the similarity of the sentence to the document headline. Sets of related documents (as identified by the grouping process) are multidocument summarised using a centroid-based summarization system (Saggion and Gaizauskas, 2004b). The metadata,

[1] The Cubreporter URL is http://nlp.shef.ac.uk/cubreporter.

[2] http://jakarta.apache.org/lucene

Input paragraph: The head of Australia's biggest bank resigned today after a multi-million dollar foreign currency trading scandal rocked the institution and sent its shares plummeting.

SUPPLE's output: head(e2), name(e4,'Australia'), country(e4), of(e3,e4), bank(e3), adj(e3,biggest), of(e2,e3), resign(e1), lsubj(e1,e2)

Figure 1: PA story paragraph and its SUPPLE's interpretation.

Objects						
Document	Entity	Word	WordSense	Type	Start	End
20040202_HSA6142	e2	head	head#4	object	0	8
20040202_HSA6142	e3	bank	bank#1	object	12	36
20040202_HSA6142	e4	Australia	_	country	12	21
Events						
Document	Entity	Word	WordSense	Type	Start	End
20040202_HSA6142	e1	resign	resign#2	event	37	51

Figure 2: Entries for nouns and verbs in the database.

story content, installments, and summaries are stored in a mySQL database for on-line access.

3.1. Mapping Sentences into Semantics

In order to create semantic representations of each text we rely on SUPPLE, a freely-available, open source natural language parser (Gaizauskas et al., 2005). In the representation created during parsing, the predicates are, for the most part, either the unary predicates formed from the morphological roots of nominal or verbal forms in the text or binary predicates from a closed set of grammatical relations (e.g. `lobj` for the verb logical object, `lsubj` for the verb logical subject) or of prepositions (e.g. `in`, `after`) or the special binary predicate `name` to identify the name of a named entity. These predicate-argument structures, or simplified quasi-logical forms (SQLF), are also mapped into the mySQL database for on-line access during background search. The database contains tables to store entities such as objects, events, and relations between these elements such as the logical subject and object of the events. Because of the time required to fully analyse each story, we parse only the leading paragraph of each text, only considering stories in particular categories ("Home News", "London News", etc.).

Consider the text fragment shown in Figure 1 and the output produced by the parser. From this output records in the database are created for each noun (e2, e4, and e3) and verb (e1) and their relations: logical subject <e1,e2>, logical object, qualifications, apposition, and preposition (<e2,e3> and <e3,e4>) (See Figure 2). The records contain: the document identifier, an entity identifier produced by the parser, the noun or verb root, the entity type (either object or a named entity type such as "person"), the start and end offsets of the entity in the paragraph, and the word WordNet sense. In order to associate WordNet word senses to each noun and verb in the archive we rely on: (i) the availability of centroids of topic signatures for each word sense (Agirre and Lopez de Lacalle, 2003), and (ii) a similarity metric which, given a word form and its word context, computes the semantic proximity of the word to each topic signature and decides the closest word sense for the word instance. Once word senses are computed for each event representation in the database, then similar events can be selected by means of word senses and lexical relations. For example, for an event such "head of bank resigns", similar resignation events can be found not only by selecting from the database those records where the form "head" has been interpreted as the WordNet word-sense number 4 of "head" but also by selecting synonyms (from "head" to "chief"), or hyponyms (from "head" to "executive"), or hyperonyms (from "head" to "leader").

3.2. Profiles

The PA archive contains pre-compiled person profiles which we have automatically identified and stored in database tables for quick access. Usually these profiles will have a catch-line matching the pattern `KEYWORD PERSON Profile` where `KEYWORD` is an uppercase keyword such as "POLITICS" or "SHOWBIZ," and `PERSON` is usually the person's surname (e.g. "Blair" for "Tony Blair"). Given one such story, we carry out named entity recognition and coreference resolution and extract coreference chains for each named person in the story. If the person name given in the catch-line matches one of the names in a coreference chains, then all names in that chain are considered aliases for the person, and records are created in the database containing each alias, the catch-line keyword, and the document identifier. As an example story HSA1816 from the PA archive has as catch-line "DEFENCE Harding Profile" the analysis of this story gives a coreference chain with the following names: "Harding", "Sir Peter Harding", "Sir Peter", "Sir Peter Robin Harding" which correspond with the alias "Harding" in the catch-line. Records are created for all names in the chain, in this way the names "Harding" or "Sir Peter" will provide access to the profile of "Sir Peter Harding".

For persons whose profiles are not in the PA archive and for other entity types, profiles are created automatically using

question answering and summarization technology. The entities to be profiled are persons, organizations, and locations, identified during text analysis by a named entity recognition process. We follow a method we proposed for answering definition questions in TREC QA 2004. The approach involves three steps: (i) web-based knowledge acquisition for the target entity – terms are gathered which help in the process of identification of definitional passages, (ii) document retrieval for each target, and (iii) "nugget" identification and filtering from the returned set of documents, where a nugget is a piece of information deem relevant for the target.

During the web-based knowledge acquisition phase (see (Saggion and Gaizauskas, 2004a)) relevant terms associated with each target are identified by extracting terms that co-occur with the target in definition bearing sentences found on Web pages, Wikipedia pages, and BBC News pages. The result of the process is a list of terms for each target. Each term t has an associated weight which is the number of times t and the target co-occur in the sources examined. The process produces a mined list of terms which includes all morphological variants and nominalisations of terms co-occurring with the target. Each term is recorded with its associated frequency of occurrence.

The nugget identification step looks for evidence to classify a sentence as 'definition/profile bearing'. The following sources of evidence are used: presence of the target or target alias in the sentence; presence of relevant terms found in external sources; and presence of definition patterns in the sentence. Two types of patterns are used: manually created lexical patterns ("X who is", "X whose", etc.) and part-of-speech and named entity patterns.

Inducing POS definition patterns Because lexical patterns are far too constrained to match naturally occurring sentences, we induce definition/profile patterns from data. The induced patterns are sequences of four elements (which must include the target entity), where each element in the pattern is a part-of-speech tag, a date, the target entity, or a title.

Each pattern has a weight which represents its relative "importance". Patterns are induced and used separately depending on the type of target (person or other entity). The weighted definition patterns are induced in the following way:

- for each target and nugget from a human created corpus of targets and nuggets, a collection of sentences from a text collection is constructed. The target and the text nugget are used as a query submitted to an information retrieval engine and the top 10 passages are retrieved (ranks 1 to 10).
- each sentence retrieved is automatically marked with the target (TARGET), POS information (NNP, VB, etc.), dates, and titles.
- a coreference algorithm, provided in GATE, is run to identify references to the target and those corefering expressions are marked as TARGET as well.
- sequences of four elements are collected, the score associated to the sequence is $1/sentence\_rank$. So patterns found in the most 'relevant' sentences get score 1 and those found in the least relevant sentence get score 0.1.
- scores for each pattern are summed for each instance
- patterns are translated into JAPE grammar rules for use in a GATE pattern recogniser.

Nugget identification and filtering Documents are retrieved from the archive using the target as a query. Each of the documents retrieved from the collection is analysed and the following scores computed for each sentence in the returned documents:

- main entity score: is 1 if the sentence contains the target (i.e., "Franz Kafka"), 0.5 if the sentence contains a target alias (i.e., "Kafka"), and 0 otherwise;
- related terms score: is the sum of the frequency of the related terms occurring in the sentence;
- definition pattern score: is 1 if the sentence matches a definition pattern and 0 otherwise
- POS pattern score: is the sum of the scores of the POS-patterns matching the sentence.

Each sentence is sorted in descending order by (in order) the main entity, the related terms score, the lexical definition patterns, and the POS-patterns. So, no sentence under consideration is in principle rejected. However this sorting is expected to rank relevant sentences higher than irrelevant ones. Sentences are output in rank order until a maximum number of characters is reached. Sentences are not included in the profile if they are regarded as too similar to a previously included sentences. Two sentences are considered too similar if they have 50% or more of their tokens in common.

4. Question Answering System

Our QA component is composed of a paragraph retrieval component followed by an answer extraction (AE) component (for a full description the reader is referred to (Gaizauskas et al., 2003)). The AE component receives as input a set of candidate passages and returns a set of answers. Paragraph are retrieved using the question as a query. The system first analyses the passages returned by the search engine and the question using SUPPLE. In this step, the expected answer type (EAT) is determined using a general purpose question answering grammar and depending on the question, a special attribute is created which indicates the attribute whose value to be output for the answer entity.

Given the sentence level "semantic" representations (SQLF) of candidate answer-bearing passages and of the question, a discourse interpretation step then creates a discourse model of each retrieved passage by running a coreference algorithm against the semantic representation of successive sentences in the passage, in order to unify them with the discourse model built for the passage so far. This results in multiple references to the same entity across the

passage being merged into a single unified instance. Next, coreference is computed again between the SQLF of the question and the discourse model of the passage, in order to unify common references.

In this model, possible answer entities are identified and scored as follows. First each sentence in each passage is given a score based on counting matches of entity types (unary predicates) between the sentence SQLF and the question SQLF. Next each entity from a passage not so matched with an entity in the question (and hence remaining a possible answer) gets a preliminary score according to (1) its semantic proximity to the EAT using WordNet and (2) whether or not it stands in a relation R to some other entity in the sentence in which it occurs which is itself matched with an entity in the question which stands in relation R to the sought entity (e.g. an entity in a candidate answer passage which is the subject of a verb that matches a verb in the question whose subject is the sought entity will have its score boosted). An overall score is computed for each entity as a function of its preliminary score and the score of the sentence in which it occurs.

Finally, the ranked entity list is post-processed to merge and boost the scores of multiple occurrences of the same answer found in multiple passages and the top scoring hypotheses are then proposed as answers.

5. User Interface

The user interacts with the system through a web client which communicates with a web server. Both the text index and the relational database are accessed by the server as needed during on-line processing and web content is dynamically created for return to the client. The user interface currently supports:

- access to full documents using a keyword search facility and an advanced search facility where structured searches over four different fields can be performed: story topic, headline, story content, and news category. Both text search interfaces allow the user to specify dates to narrow the search.
- question answering access where a factoid type of question is entered and a ranked list of answers and answer contexts (where the answer is highlighted) is returned to the user together with links to the documents where the answers were found. In the document, the passage where the answer was found is also highlighted for answer verification.
- access to profiles by entering a person's name.

Result pages show for each matched document: the document identifier, catch-line, headline, summary, and leading paragraph. The user can access the full document following a link where the installment pattern can be also accessed. Copies of the documents can be saved locally or recorded for future reference.

6. System Evaluation

Some of the technologies we use have been intrinsically evaluated: for example our question answering technology has been participated in TREC/QA 2003-2005 (see e.g. (Gaizauskas et al., 2003)) and our summarization technology has participated in DUC 2004 (see e.g. (Saggion and Gaizauskas, 2004b)) and DUC 2005. However, one of the main objectives of this project is to carry out extrinsic evaluation of the background writing task following a methodology proposed in our previous work (Barker and Gaizauskas, 2005).

7. Conclusions and Future Work

Cubreporter contributes to the creation and use of NLP technology to support the task of background gathering. However, the approaches we describe can be used to support information access to large text collections in any application domain. Our work to date has focussed on development/adaptation of NLP technology and integration in a user interface. Much of our current work is concentrated on deployment and testing of the similar event search facility. Our future work will focus on the extrinsic evaluation of the system to verify how advanced NLP techniques can contribute to the background gathering task, in comparison with conventional search engine technology.

Acknowledgements

The authors would like to acknowledge the support of the UK Engineering and Physical Sciences Research Council, research grant: R91465.

8. References

E. Agirre and O. Lopez de Lacalle. 2003. Clustering WordNet Word Senses. In *Proceedings of RANLP 2003*, pages 121–130.

E.J. Barker and R. Gaizauskas. 2005. Evaluating Cub Reporter: proposals for extrinsic evaluation of journalists using language technologies to access a news archive in background research. In *Proceedings of the COLIS 2005 Workshop on Evaluating User Studies in Information Access*.

R. Gaizauskas, M.A. Greenwood, M. Hepple, I. Roberts, H. Saggion, and M. Sargaison. 2003. The University of Sheffield's TREC 2003 Q&A Experiments. In *Proceedings of the 12th Text REtrieval Conference*.

R. Gaizauskas, M. Hepple, H. Saggion, and M. Greenwood. 2005. SUPPLE: A Practical Parser for Natural Language Engineering Applications. In *International Workshop on Parsing Technologies*.

H. Saggion and R. Gaizauskas. 2004a. Mining on-line sources for definition knowledge. In *Proceedings of FLAIRS 2004*, Florida, USA. AAAI.

H. Saggion and R. Gaizauskas. 2004b. Multi-document summarization by cluster/profile relevance and redundancy removal. In *Proceedings of Document Understanding Conference*, Boston, MA, May 6-7. NIST.

H. Saggion. 2002. Shallow-based Robust Summarization. In *Automatic Summarization: Solutions and Perspectives*, ATALA, December, 14.