Procesamiento del Lenguaje Natural, Revista nº 55, septiembre de 2015, pp 203-206 recibido 27-03-2015 revisado 25-04-2015 aceptado 21-05-2015

Summarization and Information Extraction in your Tablet[*]

Resumen y extracción de información en tu Tablet

Francesco Barbieri
Universitat Pompeu Fabra
C/Tànger 122 - Barcelona
francesco.barbieri@upf.edu

Francesco Ronzano
Universitat Pompeu Fabra
C/Tànger 122 - Barcelona
francesco.ronzano@upf.edu

Horacio Saggion
Universitat Pompeu Fabra
C/Tànger 122 - Barcelona
horacio.saggion@upf.edu

Resumen: En este artículo describimos la demonstración de una serie de aplicaciones de resumen automático y extracción de informaciones integradas en una tableta. Se presentan funcionalidades para resumir las últimas noticias publicadas en la Web, extraer información sobre eventos concretos, y resumir textos en ingés y español ingresados por el usuario. La aplicación está disponible en un Web-browser y una tableta con sistema operativo Android.
Palabras clave: Resumen automático, extracción de informaciones, aplicaciones Web

Abstract: In this article we present a Web-based demonstration of on-line text summarization and information extraction technology. News summarization in Spanish has been implemented in a system that monitors a news provider and summarizes the latest published news. The possibility to generate summaries from user's provided text is also available for English and Spanish. The demonstrator also features event extraction functionalities since it identifies the relevant concepts that characterize several types of events by mining English textual contents.
Keywords: Text summarization, Information Extraction, Web-based Applications

1 Introduction

Two Natural Language Processing (NLP) technologies which can help people assess content relevance or quickly skim textual content are text summarization (Lloret and Palomar, 2012; Saggion and Poibeau, 2013) and information extraction (Piskorski and Yangarber, 2013). We have integrated our summarization and information extraction technology into the Web-based application whose architecture is outlined in Figure 1. Our application allows a user to monitor the latest published news by having access to different types of summaries automatically generated; it also features a functionality to extract information about specific events from textual sources such as Wikipedia articles. NLP demonstrators integrated in mobile applications or Web browsers can be particularly effective to introduce NLP related topics such as classification, summarization, and information extraction to students. In particular, the applications and demonstrator we are showcasing here constitute the core platform of the hands-on practice of a NLP-related course at Universitat Pompeu Fabra where students learn and adapt summarization technology to different languages and input documents. Besides, all the components demonstrated here are made freely available for research and teaching[1].

2 Tools

In order to develop our demonstrator we use available NLP tools: The SUMMA library which is an easy-to-customize summarization software distributed free of charge[2] for research purposes (Saggion, 2008) and the GATE system (Cunningham et al., 2002), a Java library and open source NLP development environment which is freely available. We also rely on the ROME[3] set of Really Simple Syndication and Atom Utilities for Java to access the latest news from one or more news providers.

[*] We acknowledge support from the Spanish research project SKATER-UPF-TALN TIN2012-38584-C06-03, the EU project Dr. Inventor FP7-ICT-2013.8.1 611383, and UPF projects PlaQUID 65 2013-2014 and PlaQUID 47 2011-2012.

[1] http://taln.upf.edu/content/resources/699
[2] http://www.taln.upf.edu/pages/summa.upf/
[3] http://rometools.github.io/rome/

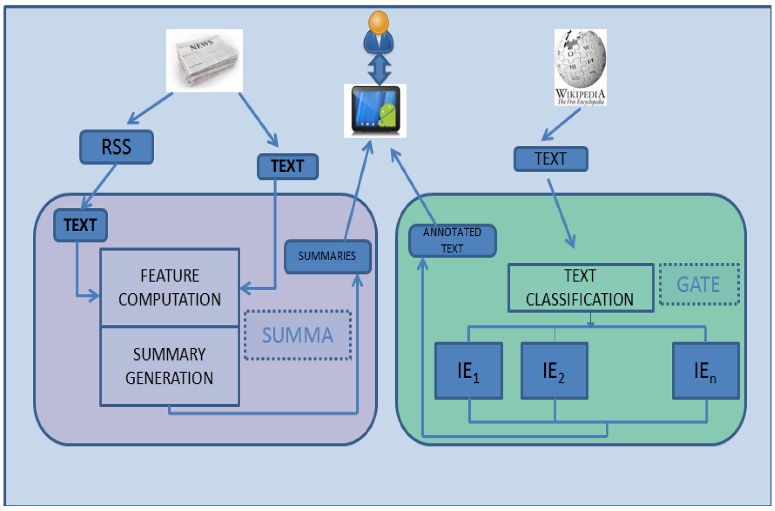


Figure 1: Overview of the Demo's Architecture

2.1 Summarization Application

Two summarization pipelines one for English and one for Spanish are implemented as GATE applications (i.e. gapp files) and integrated in the demonstrator. They are made up of the following SUMMA components (refer to Saggion (2014) for a description of these modules):

1. a term weighting computation algorithm based on term frequency and inverted document frequency (i.e. inverted document frequency table),
2. a vector computation module which represents each document sentence as a vector of terms and weigths,
3. a sentence position scorer,
4. a term frequency scorer based on term frequency * inverted document frequency,
5. a document vector computation module to represent the whole document as a vector,
6. a sentence-document similarity scorer,
7. a first-sentence similarity scorer, and
8. various sentence rankers to provide different summarization functionalities to the users.

Because the summarizers are based mainly on superficial and statistical features, they are *quasi* language-independent. The inverted document frequency table constitutes the sole language-specific component which is different between the two applications. Each application computes different types of summaries based on content relevance critera and at different compression rates. One type of summary just considers the relevance of sentences according to their position in the document, another type of summary considers the relevance of the sentence according to the distribution of words it contains. A third type of summary is based on scores given to sentences based on their similarity to a sentence-centroid computed out of all document's sentences.

2.2 Event Extraction Application

Based on the availability of the annotated CONCISUS corpus of event summaries in Spanish and English, we have developed a multi-domain information extraction application which targets three domains: airplane accidents, train accidents, and earthquakes (Saggion and Szasz, 2012). The event extraction application identifies different concept types that characterize the events. For example, for an airplaine accident some target concepts are the date of the accident, the place of the accident, the airline, the flight number, etc. The extraction procedure is implemented as follows: after a linguistic analysis of the input document for which we use ANNIE (Cunningham et al., 2002), a text classification algorithm is applied to the document to identify the target domain. Then, given the identified domain, an appropriate domain-specific information extraction component is invoked. The whole process is implemented as a conditional pipeline to properly control the execution of each information extraction component. The text classification algorithm to identify one of the 3 target domains is a Support Vector Machines classifier (Joachims, 1998) trained over unigrams (lemmas). During development the classifier achieved perfect F1 (training on 83 documents and testing on 29 unseen documents). The demonstrator includes a classifier trained on the whole CONCISUS corpus of 112 articles.

Since each domain requires the extraction of different concepts, three information extraction systems are implemented using a Support Vector Machines token classification approach with a context window of size five. All IE systems use exactly the same features for concept identification computed by ANNIE: word identity, lemmas, gazetteer list information, named entities, and POS tags. Cross-validation performance of the system varies from domain to domain with an F1 of 0.63 for extracting 28 aviation accident concepts, an F1 of 0.61 for extracting 20 train accident concepts, and an F1 of 0.40 for the extraction of 30 earquake-related concepts. The number of concepts to be learnt, the small size of the corpus, and the skewed distribution of concepts in the corpus partially explain the results.

3 Deployment

The Web application has been built relying on widespread Web technologies and libraries. The core component is made up of its Server Module that enables clients to invoke the summarization and information extraction services by accessing on-line REST endpoints that deliver their output in JSON format[4]. The Server Module is implemented as Java Web Application that exposes RESTful Web Services by relying on Jersey, a popular open source Java framework[5]. Jersey implements the Java API for RESTful Web Services specifications[6], thus supporting a modular and versatile development of RESTful Web Services in Java. The following set of functionalities can be invoked by querying the application's REST endpoints:

- retrieve and summarize on-line RSS feeds, like the feeds exposed by on-line newspapers (see Figure 2 for the automatic summaries about a recent aviation accident);
- summarize a pasted text, like for instance a news article;
- extract useful information from a textual excerpt (i.e., the concepts associated to the identified news event) – see Figure 3 for information extracted from the recent GermanWigs airplane crash described in Wikipedia.

When a REST endpoint of the Server Module is queried, a properly configured instance of SUMMA is exploited in order to process one or more texts, thus generating the results that are serialized as JSON and sent back to the client. The Server Module implements a thread-based service of RSS feeds retrieval: this service is responsible for maintaining a in-memory copy of the contents of the RSS feeds that can be processed by the application. In particular the service periodically performs asynchronous downloads of the contents of one or more RSS feed URLs in order to refresh a local copy of such information.

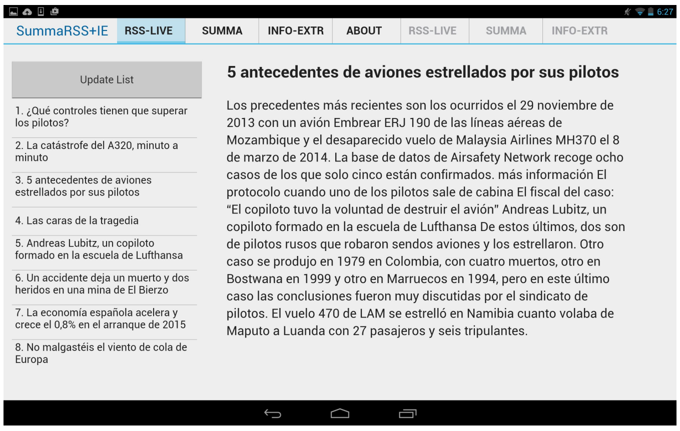
The Client Module is implemented by relying on HTML and Javascript. In particular, the Javascript framework JQuery[7] is exploited to implement the client (browser) logic. The REST endpoints exposed by the Server Module are queried by AJAX calls issued by the Client Module in order to retrieve and process data in response to user interactions. Our application can be deployed on any Java Web Application Container; in our current deployment we use the version 7 of Apache Tomcat[8].

References

Cunningham, H., D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02).*

Joachims, T. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Claire Nédellec and Céline Rouveirol, editors,

[4]http://json.org/
[5]https://jersey.java.net/
[6]JAX-RS, https://jax-rs-spec.java.net/
[7]https://jquery.com/
[8]http://tomcat.apache.org/

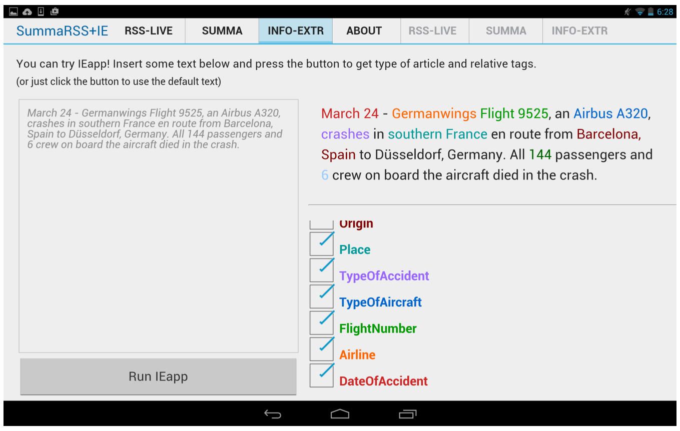


1. ¿Qué controles tienen que superar los pilotos?
2. La catástrofe del A320, minuto a minuto
3. 5 antecedentes de aviones estrellados por sus pilotos
4. Las caras de la tragedia
5. Andreas Lubitz, un copiloto formado en la escuela de Lufthansa
6. Un accidente deja un muerto y dos heridos en una mina de El Bierzo
7. La economía española acelera y crece el 0,8% en el arranque de 2015
8. No malgastéis el viento de cola de Europa

5 antecedentes de aviones estrellados por sus pilotos

Los precedentes más recientes son los ocurridos el 29 noviembre de 2013 con un avión Embrear ERJ 190 de las líneas aéreas de Mozambique y el desaparecido vuelo de Malaysia Airlines MH370 el 8 de marzo de 2014. La base de datos de Airsafety Network recoge ocho casos de los que solo cinco están confirmados. más información El protocolo cuando uno de los pilotos sale de cabina El fiscal del caso: "El copiloto tuvo la voluntad de destruir el avión" Andreas Lubitz, un copiloto formado en la escuela de Lufthansa De estos últimos, dos son de pilotos rusos que robaron sendos aviones y los estrellaron. Otro caso se produjo en 1979 en Colombia, con cuatro muertos, otro en Bostwana en 1999 y otro en Marruecos en 1994, pero en este último caso las conclusiones fueron muy discutidas por el sindicato de pilotos. El vuelo 470 de LAM se estrelló en Namibia cuanto volaba de Maputo a Luanda con 27 pasajeros y seis tripulantes.

Figure 2: RSS Summarization Application



You can try IEapp! Insert some text below and press the button to get type of article and relative tags.
(or just click the button to use the default text)

March 24 - Germanwings Flight 9525, an Airbus A320, crashes in southern France en route from Barcelona, Spain to Düsseldorf, Germany. All 144 passengers and 6 crew on board the aircraft died in the crash.

March 24 - Germanwings Flight 9525, an Airbus A320, crashes in southern France en route from Barcelona, Spain to Düsseldorf, Germany. All 144 passengers and 6 crew on board the aircraft died in the crash.

☐ Origin
☑ Place
☑ TypeOfAccident
☑ TypeOfAircraft
☑ FlightNumber
☑ Airline
☑ DateOfAccident

Figure 3: Event Identification in a Wikipedia Article

Proceedings of ECML-98, 10th European Conference on Machine Learning, number 1398 in Lecture Notes in Computer Science, pages 137–142, Chemnitz, Germany. Springer Verlag, Heidelberg.

Lloret, E. and M. Palomar. 2012. Text summarisation in progress: a literature review. *Artif. Intell. Rev.*, 37(1):1–41.

Piskorski, J. and R. Yangarber. 2013. Information extraction: Past, present, and future. In T. Poibeau, H. Saggion, J. Piskorski, and R. Yangarber, editors, *Multi-source, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing. Springer.

Saggion, H. 2008. SUMMA: A Robust and Adaptable Summarization Tool. *Traitement Automatique des Langues*, 49(2):103–125.

Saggion, H. 2014. Creating summarization systems with SUMMA. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland, May 26-31, 2014.*, pages 4157–4163.

Saggion, H. and T. Poibeau. 2013. Automatic text summarization: Past, present, and future. In T. Poibeau, H. Saggion, J. Piskorski, and R. Yangarber, editors, *Multi-source, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing. Springer.

Saggion, H. and S. Szasz. 2012. The CONCISUS corpus of event summaries. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012), Istanbul, Turkey, May 23-25, 2012*, pages 2031–2037.