# Linguistic summaries on small screens

E. D'Avanzo[1] & T. Kuflik[2]
[1]*ITC-irst and University of Trento, Italy*
[2]*Management Information Systems Department, University of Haifa, Israel*

## Abstract

Nowadays computers are becoming ever smaller and cheaper, and as a result they have become available everywhere. One major drawback of the small computing devices is their small screen (e.g., PDA's and smartphones) that makes it harder to view large amounts of information. The problem is even worse given the flood of information today's users are facing in their daily tasks. This paper presents LAKE, a flexible summarization system consisting of different independent linguistic modules that can easily be configured and parameterized. The system can be used to reduce the amount of information viewed by users of small-screen devices while maintaining its essence. The system works as follows: a set of linguistically motivated candidate phrases is identified. A learning device then chooses the best phrases. Finally, phrases at the top of the ranking are merged to form a summary. LAKE is tested within the framework of eSMEs, a joint Project of Italdata S.p.A. Siemens Business Services Group and Multiplanning s.r.l. companies.

*Keywords: text mining, machine learning, linguistic knowledge, keyphrase extraction, summarization, small devices.*

## 1 Introduction

Nowadays users are facing problems while viewing information displayed on small screen devices (e.g. PDAs and Smartphones). Such devices require presentation of the most essential information only, in order to allow their users to use them effectively. *Text summarization* is a possible solution to such a problem. Hahn and Mani [1] defined summarization as 'the art of abstracting key content from one or more information sources'. In fact, the importance of

summarization tools is constantly increasing as demonstrated by the increasing research in this area.

This work suggests that keyphrases can be used for summarization of information. Such a solution may provide the means to reduce *information overload* [2] that is most important for small devices.

## 2 Background and related work

Text summarization is as a process that takes a document as input, and outputs a shorter, surrogate document, consisting of its most important content [3].

Summaries may be *extract* based or *abstract* based. The former type of summary is constructed by choosing the most relevant pieces of text, while the latter consists of a *gloss* that describes the contents of a document without necessarily featuring any of that content.

The most popular way to construct a summary of a document is to detect fragments within the document and then combine them into an *extract*. Many systems have been developed using sentence selection techniques, following this idea. Most of these systems employ a mixture of linguistic knowledge and statistical methods for summarization. An example of such an approach is Kupiec's *Trainable Document Summarizer* [4], which uses a set of discrete features, such as *sentence length*, *fixed phrase*, *paragraph*, *thematic word* feature, and *upper case word* feature, for scoring and selecting sentences. However, sentence selection strategy could produce disjointed sentences that do not read well. To achieve greater coherence, an alternate approach to summarization can be used, assuming that a text contains a small number of "best" paragraphs, which can represent the whole document. An example is *Salomon*, a system developed by Moens [5] that first models the structure of documents to be summarized, starting with an analysis of the "typical form of discourse" of such materials. The result is a *text grammar*. Text examples are then tagged for further analysis by applying a "partial parser" to identify commonly occurring word patterns. A knowledge engineering effort is required to construct these patterns manually.

Another approach to summarization is based on *keyphrase extraction*. Keyphrases often accompany journal articles, where a list of keyphrases - single words or phrases - is supplied. Turney [6] supports the usefulness of keyphrases usage for a variety of tasks, including summarizing, indexing, labeling, categorizing, clustering, browsing, etc. He designed *Extractor* (and its variant *GenEx*), a system using a supervised learning approach to extract keyphrases from documents. Given a text, a pre-processing module removes stopwords (non content words like *a*, *the*, *again*, etc) from the text. A set of candidate keyphrases are then identified by combining all remaining words in uni-grams, bi-grams and tri-grams. Finally the candidates are matched with human generated keyphrases. A *decision tree* is built, learning a model that will be used in the keyphrases identification later on. A similar approach to Turney's is demonstrated by *Kea* [7], a learning system developed by Witten at New Zealand Digital Library. *Kea* also uses a supervised learning approach, treating each candidate phrase as a

keyphrase or a non-keyphrase depending on whether it matches the human generated one. Unlike Extractor, *Kea* uses a Naïve Bayes classifier as learning device and only two features: *TF*IDF* and *first occurrence*. Kea demonstrated the wide range of applications where keyphrases turned out to be suitable, from browsing, back of book indices, and skimming, finally to summarizing Internet searches on PDA's [8].

# 3 LAKE system

Linguistic Analysis based Knowledge Extractor ("LAKE") is a keyphrase extraction summarization system. LAKE is based on a supervised learning approach that makes use of a linguistic processing of the summarized documents. Like Kea, it uses a Naïve Bayes as a learning algorithm, and *TF*IDF* term weighting and the *position* of a phrase as features. Unlike Kea and Extractor, LAKE chooses the candidate phrases using linguistic knowledge. The candidate phrases generated by LAKE are sequences of Part of Speech containing Multiword expressions and Named Entities. We define such elements as "patterns" and store them in a patterns database; once there, the main work is done by the learner device. The linguistic database makes LAKE unique in its category. Another difference which sets LAKE apart from Extractor and Kea is that of the task for which it has been designed, i.e. summarization for different devices. Lake participated in DUC-2004 (Document Understanding Conference) [9]. Participating groups at DUC-2004 were allowed to submit up to three runs. Submissions were automatically evaluated using ROUGE program [10]. In general, LAKE placed in the middle of the ranking (19 out of 39 systems).

"LAKE" has three main components: Linguistic Pre-Processor, Candidate Phrase Extractor and Candidate Phrase Scorer. Figure 1 presents Lake Architecture.

## 3.1 Linguistic Pre-Processor

Every document is analyzed by the Linguistic Pre-Processor in the following three consecutive steps: Part of speech analysis, Multiword recognition and Named Entity Recognition

### 3.1.1 Part of Speech tagger

The Part of Speech (POS) tagger builds upon a tokenizer and sentence delimiter, labeling each word in a sentence with its appropriate tag. It decides if a given word is a noun, verb, adjective, etc. The POS tagger adopted by "LAKE" is the TreeTagger, developed at the University of Stuttgart [11]. The TreeTagger uses a decision tree to obtain reliable estimates of transition probabilities. It determines the appropriate size of the context (number of words) which is used to estimate the transition probabilities. For example, if we have to find the probability of a noun appearing after a determiner followed by an adjective we find out whether the previous tag is ADJ; if yes, then we go into the "yes" branch and check if the tag previous to this was a determiner; if "yes" then we get to a probability of this occurrence.

### 3.1.2 Multiwords Recognition

Sequences of words that are considered as single lexical units are detected in the input document according to their presence in WordNet [12]. For instance, the sequence "Christmas trees" is transformed into the single token "Christmas_tree" and the PoS tag found in WordNet is assigned to it.

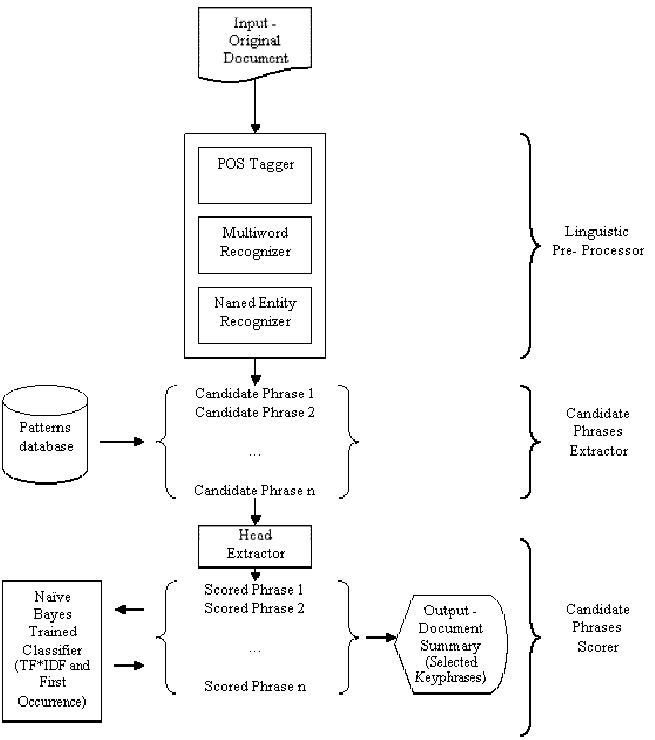


Figure 1: LAKE system architecture.

### 3.1.3 Named Entities Recognition

The task of Named Entity Recognition (NER) requires a program to process a text and identify expressions that refer to people, places, companies, organization, products, and so forth. Thus the program should not merely identify the boundaries of a naming expression, but also classify the expression, e.g., so that one knows that "Rome" refers to a city and not a person. For Named Entities recognition we used LingPipe, a suite of Java tools designed to perform linguistic analysis on natural language data (LingPipe is free, available at http://www.alias-i.com/lingpipe/index.html). The tool includes a statistical named-entity detector, a heuristic sentence boundary detector, and a heuristic within-document co reference resolution engine. Named entity extraction models are included for English news and can be trained for other languages and genres.

### 3.2 Candidate phrase extractor

Syntactic patterns that described either a precise and well defined entity or concise events/situations were selected as "candidate phrases" (e.g. phrases that may be selected as document reorientations). In the former case, we focused on uni-grams and bi-grams (for instance Named Entity, noun, and sequences of adjective+noun, etc.), while in the latter we considered longer sequences of parts of speech, often containing verbal forms (for instance noun+verb+adjective+noun). Sequences such as *noun+adjective* that are not allowed in English were not taken into consideration .We also eliminated patterns containing punctuation. We manually selected a restricted number of PoS sequences that could have been significant in order to describe the setting, the protagonists and the main events of a newspaper article. To this end, particular emphasis was given to named entities, proper and common names. Once all the uni-grams, bi-grams, tri-grams, and four-grams were extracted from the *linguistic pre-processor*, they were filtered with the patterns defined above.

Table 1: Examples of types of phrases and their patterns.

| Type of Phrase | Pattern | Example |
|---|---|---|
| Uni-Gram | NE<br>NE | London<br>1973 |
| Bi-Gram | JJ+NN<br>JJ+NN<br>JJ+NN | Chilean dictator<br>Spanish magistrate<br>urinary infection |
| Tri-Gram | NN+CC+NN<br>NN+VBD+NE<br>NN+VBD+NN | genocide and terrorism<br>newspaper reported Friday<br>room locked television |
| Four-Gram | NE+MD+VB+VBN<br>VBN+IN+JJ+NNS<br>NN+TO+VB+NN<br>NN+VBD+JJ+NN | Augusto Pinochet would be extradited<br>detained by British police<br>extradition to stand trial<br>dictatorship caused great suffering |

As an example, we can consider a document belonging to the DUC corpus (http://www-nlpir.nist.gov/projects/duc/data.html) that reports on the possible extradition of Pinochet from London to Spain. Table 1 shows some of the candidate phrases that our largest filter accepted as candidates from this document.

### 3.3 Candidate phrases scorer

In this phase a score is assigned to each candidate phrase in order to rank it and to allow the selection of the most appropriate phrases as representative of the original text. The size of the output is set to 75 bytes, as required by DUC [9]. The score is based on a combination of *TF*IDF* (i.e. the product of the frequency of a candidate phrase in a certain document and the inverse frequency of the phrase in all documents) and f*irst occurrence*, i.e. the distance of the candidate phrase from the beginning of the document in which it appears. (These features are commonly used keyphrase-related features.) However, since

the frequency of a candidate phrase in the whole collection is not significant, candidate phrases do not appear frequently enough in the collection. We decided to estimate the values of the TF*IDF using the head of the candidate phrase, instead of the phrase itself. According to the principle of headedness [13], any phrase has a single word as head. The head is the main verb in the case of verb phrases, and a noun (last noun before any post-modifiers) in noun phrases.

As learning algorithm, we used the Naive Bayes Classifier provided by the WEKA package [14]. The classifier was trained on the DUC-2003 material in the following way. From the document collection we extracted all the nouns and the verbs. Each of them was marked as a positive example of a relevant keyphrase for a certain document if it was present in the assessor's judgment of that document; otherwise it was marked as a negative example. Then the two features (i.e. TF*IDF and first occurrence) were calculated for each word. The classifier was trained upon this material and a ranked word list was returned (e.g., *dictator*, *magistrate*, *infection*, etc. – see Table 1). The system automatically looks in the candidate phrases for those phrases containing these words. In our case *Chilean dictator*, *Spanish magistrate*, *urinary infection,* etc. The top candidate phrases matching the word output of the classifier are kept. The model obtained is reused in the subsequent steps. When a new document or corpus is ready we use the pre-processor module to prepare the candidate phrases. The model we got in the training is then used to score the phrases obtained. In this case the pre-processing part is the same. So, using the model we got in the training, we extract nouns and verbs from documents, and then we keep the candidate phrases containing them.

## 4 eSMEs Project

The eSMEs Project (eBusiness for Small and Medium Enterprises) is a joint research project of Italdata S.p.A. Siemens Business Services Group and Multiplanning s.r.l. Consulting firm. The project goal is to develop a prototypical Web-based multi-channel platform able to supply E-procurement and Advanced Marketing services in A.S.P. (Application Service Providing) modality. In addition, a Web Service Layer (WSL) will allow the integration of currently existing systems (in a company) into the project, and enable integration of new ones, as they become available (Figure 2 presents project architecture).

The research of the eSMEs Project focuses on studying, designing and experimenting with a multichannel A.S.P. platform (Internet, GPRS, UMTS, xDSL, Wi-Fi, Satellite, etc), carried out by means of:

- Web Service Layer (WSL): allowing integration of pre-existing systems
- Advanced Marketing System: allowing improved marketing activity by means of Customer Profiling and Text Mining
- E-procurement systems supporting SMEs in their supply services. In particular, it aims at cataloging, and personalizes products. Moreover, its goal is to optimize enterprise processes by categorizing and organizing information based on user requests.

The project is funded by the Regione Campania.

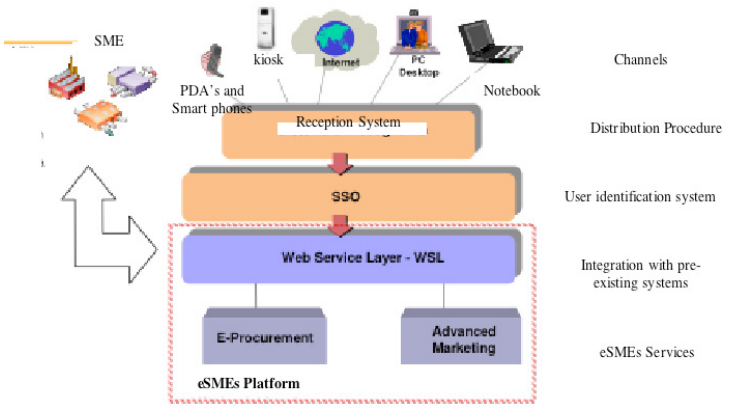


Figure 2: eSMEs project.

## 5 LAKE's role in the eSMEs

The project contemplates the use of small devices used by Multiplanning s.r.l professionals. Multiplanning s.r.l is a management and banking consulting firm. The staff is composed of fifty people with different background (lawyers, bankers, tax consultants, business consultants, accountants). Professionals, when out of office (often), need immediate access to information using small devices. eSMEs' goal is to provide these users with succinct yet useful representations of the information (e.g., e-mail, on-line news, intranet documents) that allow them to continue working in a seamless manner outside the office.

### 5.1 Experiments

An on-line demo of LAKE has been created. The demo allows the user to enter any URL or to paste a given text (in the case that they want to experiment with an e-mail or own document). In both cases LAKE extracted five keyphrases that were presented to the user on a small device emulator and also on a web form next to the device emulator. The users were asked to evaluate the keyphrases and give a rating of: "satisfied" or "not satisfied". They were asked to do it twice: for the keyphrases displayed on the device and again for the keyphrases displayed on the Web form. This setting was important in order to evaluate the impact of the keyphrases and device together.

### 5.2 Discussion

Other attempts at summarizing Web content on PDA's using keyphrases have been made. Jones et al. [8], reports on a users' study that compared how accurately users categorized documents presented on small screens when the document surrogates consisted of either keyphrases or document titles only. The

authors found encouraging results in the sense that there are no significant performance differences between the two conditions. We agree with the authors concerning the benefit of being able to extract keyphrases when no other document metadata can be identified. However, they do not seem to consider using author titles as *gold standard* to be misleading. In fact, sometimes the choice of title may be decided by the fantasy of an author, and bear no relation to the content of the document. In fact, users may be faced with titles which are completely unrelated to a certain category, but which rather contain some words that muddle him.

Table 2: Experimental results.

| | | |
|---|---|---|
| Number of voters | 40 | |
| Number of documents | 196 | |
| Number of keyphrases | 980 | |
| Maximum documents per voters | 5 | |
| Satisfied | 706 | 72 % |
| Not-satisfied | 274 | 18 % |

In our evaluation, we preferred to investigate the acceptability of the extracted keyphrases to a human reader with respect to the original document. In this sense, our evaluation approach is similar to that used by Turney [15]. In his evaluation framework, Turney gave three possible responses: "good", "bad", and "no opinion". In his experimentation, Turney obtained results that 81.9 % of the keyphrases extracted were *acceptable* (*good* + *no opinion*). In the Turney experimental framework, acceptable means *not bad*. Table 2 synthesizes the user judgment about the keyphrases in our experiments without taking into account the keyphrases displayed on the device emulator (this setting is close to Turney's). 72% of keyphrases extracted were rated as satisfying. The result is quite similar to that of Turney [15] (80%), especially if we consider that his result was obtained counting "good" and "no opinion" together. With respect to Turney's approach, we preferred two possibilities to avoid user disorientation. Besides, we do not know how *not bad* keyphrases should be treated.

When users judged keyphrases on the device emulator, we obtained the result that 68% of them (corresponding to the 27 users) were satisfied with the information they got on the device. In this setting people were allowed to choose the device that was closer to their own (the emulator environment contains both PDA's and smartphones). Moreover, 25% of users claimed that they were "not satisfied" only because the keyphrases were too long to be read on the device (we recall that LAKE can extract keyphrases up to four words long). This is encouraging because some tuning on the user preferences may improve the results.

## 6 Conclusions and future work

In this work we presented LAKE system, a text summarization system producing short document summaries that can be easily displayed on today's mobile devices and allow users to browse document summaries with relative ease.

The participation of LAKE in the eSMEs allowed us to evaluate the system by end-user in their daily work. Results are quite encouraging. LAKE's performance is similar to that of Turney's system in the human judgment of keyphrases extracted. In addition, we are quite satisfied with the evaluation of the users considering the impact of keyphrases on the emulator.

At the time of writing this paper we are re-implementing LAKE on a unique Java platform, which will allow us to improve its portability. We are planning this jointly with Italdata S.p.A.Siemens Group in the light of future experimentation, which will include also personalization.

## Acknowledgments

We would like to thank Multiplanning's staff who helped in performing the experimentation we described in last part of this work. Particular thanks go to the CEO of Multiplanning who gave us the opportunity to do this collaboration. We are also grateful to Italdata S.p.A. Siemens Business Service Group and their Project Managers for the overall discussion upon the eSMEs Project. Finally, we would thank Bernardo Magnini for his patience in the discussion.

## References

[1] Hahn, U. & Mani, I. The Challenges of Automatic Summarization. *Computer*, Volume 33, Issue 11 (November 2000). Pages: 29–36 .

[2] M. Song, M., Song, I.-Y. & Hu, X. KP Spotter: a flexible information gain-based keyphrase extraction system. *Proceedings of the 5th ACM International Workshop on Web Information and Data Management.* New Orleans, Louisiana, USA. 2003.

[3] Jackson, P. & Moulinier, I. *Natural Language Processing for Online Applications*. John Benjamins Publishing Company. Amsterdam/Philadelphia. 2002.

[4] Kupiec, J., Pedersen, J.O. & Chen, F. A Trainable Document Summarizer. Research and Development in Information Retrieval. 1995.

[5] Moens, M.-F. *Automatic Indexing and Abstracting of Document Texts*. Norwell, MA: Kluwer Academic. 2000.

[6] Turney, P.D. Extraction of keyphrases from text: Evaluation of four algorithms. Technical Report ERB-1051. (NRC \#41550), National Research Council, Institute for Information Technology. 1997.

[7] Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C. & Nevill-Manning, C.G. Domain-specific keyphrase extraction. *IJCAI 1999*, pp. 668-673. 1999.

[8] Jones, S., Jones, M. & Deo, S. Using keyphrases as search result surrogates on small screen devices. *Personal and Ubiquitous Computing*, Volume 8 Issue 1. February 2004.

[9] Document Understanding Conference. http://www-nlpir.nist.gov/projects/duc/intro.html

[10] Lin, Chin-Yew & Hovy, E.H. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In Proceedings of 2003 Language Technology Conference (HLT-NAACL 2003), Edmonton, Canada, May 27 - June 1, 2003.
[11] Schmid, H. Probabilistic part-of-speech tagging using decision trees. *Proc. of the Int. Conference on New Methods in Language Processing*, Manchester, UK. 1994.
[12] Fellbaum, C. WordNet: *An Electronic Lexical Database*. MIT Press.
[13] Arampatzis, A., van der Weide, T., Koster, C. & van Bommel, P. An evaluation of linguistically-motivated indexing schemes. *Proceedings of BCSIRSG* '2000. 2000.
[14] Witten, I.H. & Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann. 1999.
[15] Turney, P.D. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4): 303-336. 2000.