A Keyphrase-Based Approach to Summarization: the LAKE System at DUC-2005

Ernesto D'Avanzo
ITC-irst
and
DIT - University of Trento
38050, Povo (TN), Italy
davanzo@itc.it

Bernardo Magnini
ITC-irst
38050, Povo (TN), Italy
magnini@itc.it

Abstract

The paper reports on LAKE participation at DUC-2005. We propose to exploit a keyphrase extraction methodology in order to identify relevant terms in the document. Afterward, a score mechanism is used to score the best sentences for each cluster of documents. At its heart, the LAKE algorithm first considers a number of linguistic features to extract a list of well motivated candidate keyphrases, then uses a machine learning framework to select significant keyphrases for a document. With respect to other approaches to keyphrase extraction, LAKE makes use of linguistic processors such as named entities recognition, which are not usually exploited. We discuss results and comment on both human assessment (Linguistic Quality and Responsiveness of the summaries), the ROUGE based evaluation, and the Pyramid evaluation.

1 Introduction

LAKE participated in the DUC-2004 evaluation exercise (D'Avanzo et al., 2004), task 1 (*very short single document summaries*, limited to 75 bytes). The system was based on the idea of Keyphrase Extraction (hereafter KE) as a useful approximation to summarization. Our decision to participate was mainly motivated by the fact that some features of Task 1, i.e. the length limit of the output summaries and the fact that summaries could be returned as lists of disjointed items, seemed to fit well in a KE approach. In further experiments LAKE has been tested as a useful device in text mining application suitable for small devices as well (D'Avanzo and Kuflik, 2005). Still, in (Bordoni and D'Avanzo, 2002) is discussed the usefulness of KE for knowledge management purposes.

As this year the task was to provide a 250 words summary for a cluster of about 50 documents, we have enriched the LAKE system with a number of functionalities. The system still extracts an ordered (according to their position in the document) list of relevant keyphrases from each document of the cluster. Then we compare the keyphrase lists for each document and we estimate both the relevance and the coverage of each list. Finally, the keyphrase list which maximizes the two parameters is selected as the most representative of the cluster and each keyphrase is substituted with the whole sentence in which it appears, until a 250 word summary is built. In this paper we discuss results obtained at DUC-2005 and comment on both human assessment (Linguistic Quality and Responsiveness of the summaries) and the ROUGE based evaluation. LAKE scored very well (first position) as far as the Linguistic Quality was concerned, confirming the hypothesis that an ordered list of relevant keywords is a good representation of the document content.

The paper is organized as follows. Section 2 provides the background on the use of keyphrases in a number of Information access tasks. In Section 3 we report on the general architecture of our system, which combines a machine learning approach with a linguistic processing of the document. Section 4 shows the extensions we have introduced in order to adapt LAKE for the multi-document summarization scenario at DUC-2005. Section 5 shows the results obtained by the system and discusses the evaluation carried out, including the new Pyramid based approach. We conclude suggesting possible future improvements.

2 Keyphrase Extraction

Keywords, or *keyphrases*[1], provide semantic metadata that characterize documents, producing an overview

[1]Throughout this document we use the latter term to subsume the former.

of the subject matter and contents of a document. Keyphrase extraction is a relevant technique for a number of text-mining related tasks, including document retrieval, Web page retrieval, document clustering and summarization, Human and Machine Readable Indexing and Interactive Query Refinement (see (Turney, 2000) and (Gutwin et al., 1998)).

There are two major tasks exploiting keyphrases: keyphrase assignment and keyphrase extraction (see (Turney, 1999)). In a keyphrase assignment task there is a predefined list of keyphrases (i.e, a *controlled vocabulary* or *controlled index terms*). These keyphrases are treated as classes, and techniques from *text categorization* are used to learn models for assigning a class to a given document. A document is converted to a vector of features and machine learning techniques are used to induce a *mapping* from the feature space to the set of keyphrases (i.e. labels). The features are based on the presence or absence of various words or phrases in the input documents. Usually a document may belong to different classes.

In keyphrase extraction (KE), keyphrases are selected from the body of the input document, without a predefined list. When authors assign keyphrases without a controlled vocabulary (*free text keywords* or *free index terms*), typically about 70% to 80% of their keyphrases appear somewhere in the body of their documents (Turney, 1997). This suggests the possibility of using author-assigned free-text keyphrases to train a KE system. In this approach, a document is treated as a set of candidate phrases and the task is to classify each candidate phrases as either a keyphrase or non-keyphrase (Turney, 1997; Frank et al., 1999). A feature vector is calculated for each candidate phrase and machine learning techniques are used to learn a model which classifies each candidate phrase as a keyphrase or non-keyphrase.

3 LAKE

LAKE (Linguistic Analysis based Keyphrase Extractor) is a keyphrase extraction system based on a supervised learning approach which makes use of linguistic processing of documents. The system uses WAye Hayes as the learning algorithm and $TF \times IDF$ term weighting with the *position* of a phrase as features. Unlike other keyphrase exctraction systems, like 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. Extraction is driven by a set of ”patterns” which are stored in a pattern database; once there, the main work is done by the learner device. The linguistic database makes LAKE unique in its category.

LAKE is based on three main components: the Linguistic Pre-Processor, the candidate Phrase Extractor and the Candidate Phrase Scorer.

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 built 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 (Schmid, 1994). 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 (adjective); 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 (Fellbaum, 1998). 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.

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 *Washington* refers to a city and not a person. For Named Entities recognition we used LingPipe[2], a suite of Java tools designed to perform linguistic analysis on natural language data. 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.

[2]LingPipe is free, available at http://www.alias-i.com/lingpipe/index.html

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, the focus was on uni-grams and bi-grams (for instance Named Entity, noun, and sequences of adjective+noun, etc.), while in the latter have been 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. Patterns containing punctuation have been eliminated. Manually have been 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.

As an example, let consider a document belonging to the DUC corpus[3] 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 allowing the selection of the most appropriate phrases as representative of the original text. The score is based on a combination of $TF \times IDF$ (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 first 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. It has been decided to estimate the values of the $TF \times IDF$ using the head of the candidate phrase, instead of the phrase itself. According to the principle of headedness (Arampatzis et al., 2000), 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, it has been used the Naïve Bayes Classifier provided by the WEKA package (Witten and Frank, 1999). The classifier was trained in the following way on a corpus with the available keyphrases. 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. $TF \times IDF$ 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 LAKE at DUC-2005

At DUC-2005 participants, given a *user profile*, a DUC *topic*, and a cluster of documents relevant to the DUC topic, were asked to create from the documents a brief, well-organized, fluent summary addressing the need for information expressed in the topic, at the level of granularity specified in the user profile. The summary should not be longer than 250 words (whitespace-delimited tokens) and should include (in some form or other) all the information in the documents that contributes to meeting the information need. Each group was allowed to submit one set of results, i.e., one summary for each topic/cluster. A number of extensions, described in the rest of this Section, were necessary in order to adapt the LAKE system to the new task.

As a first step, we continued to use keyphrases as a document surrogate. In other words, we exploited the LAKE core system abilities to extract from each document j of a cluster an ordered list of keyphrases kl_j. Two options has been added with respect to last year system. First, it is possible to set the number of keyphrases that the system extracts from each document. Second, it is it is possible to set the maximum number of words composing a keypkrase. In short, for a given document j the system is able to extract a keyphrase list kl_j, as long as we like and with the possibility to choose the number of words (i.e. up to four words) contained in each keyphrase of the extracted list.

[3] http://www-nlpir.nist.gov/projects/duc/data.html

Table 1: Examples of types of phrases and their patterns

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

Then we compare the keyphrase lists for each document and we estimate two measures which we think are crucial for selecting the most representative kl_j among those produced for a certain cluster, both the relevance and the coverage of each list. Given a kl for a document d of a cluster C_j, the next step is to look for a score mechanism able to select the best kl and a as consequence the document that better represents the whole cluster.

A summary for a cluster C is represented by sentences of the document d_j belonging to C_j, which best represents fact reported in C. To estimate the representativeness of a document d in a cluster C we use two measures: the relevance of the document in C and the coverage of the document in C. Since documents are represented as list of relevant keyphrases, the two measures are computed over such keyphrase list.

The relevance of a keyphrase list kl_j with respect to a cluster C_j is computed considering the frequency of the keyphrases composing the list. The intuition is that keyphrases with higher frequency bring the more relevant information in the cluster. Relevance is calculated according to the following formula:

$$relevance(kl_j) = \frac{\sum_{w=1}^{n} freq(w, kl_j)}{freq(w, C_j)} \quad (1)$$

where $freq(w, kl_j)$ is the count of a word w in a certain document and $freq(w, C_j)$ is the count of w in all the document in cluster C_j.

The coverage of a keyphrase list kl_j is an indication of the amount of information that the keyphrase list contain with respect to the total amount of information included in a cluster of documents. Coverage is calculated according to the following formula:

$$coverage(kl_j, C) = \frac{length(kl_j)}{maxlength(kl_j, C)} \quad (2)$$

where $length(kl_j)$ is the number of keyphrases extracted from document j and $maxlength(kl_j, C)$ id the length of the longest keyphrase list extracted from a document belonging to cluster C_j. The intuition underlying being that the longer the keyphrase list, the more is its coverage for a certain cluster.

Finally, relevance and coverage are combined according to the following formula:

$$rep(kl_j) = relevance(kl_j, C) \times coverage(kl_j, C) \quad (3)$$

which gives an overall measure of the representativeness of a keyphrase list for a certain document with respect to a cluster.

Finally, the keyphrase list which maximize the two parameters is selected as the most representative of the cluster and each keyphrase is substituted with the whole sentence in which it appears, until a 250 word summary is built.

5 Results and Discussion

5.1 Linguistic Quality and Responsiveness

Summaries at DUC-2005 have been evaluated by human assessors according to both their Linguistic Quality and to their Responsiveness. Linguistic quality assess how readable and fluent the summaries are, and measure the qualities of the summary without comparing it with a model summary or DUC topic. Five *Quality Questions* were used:

1. Grammaticality

Table 2: Results of the LAKE system at DUC 2005 ..

	Average score	Relative position
Linguistic Quality	3.968	1/31
Responsiveness (Scaled)	16.7	19/31
ROUGE-2	0.056270211	20/31
ROUGE-SU4	0.1106907611	20/31

2. Non-redundancy
3. Referential clarity
4. Focus
5. Structure and Coherence

All linguistic quality questions were assessed on a five-point scale from "1" (very poor) to "5" (very good). As Table 2 shows LAKE, in average, obtained very good results in this sense.

As for responsiveness the evaluation assesses how well each summary responds to the topic. After having read the topic statement and all the associated summaries, assessors grade each summary according to how responsive it is to the topic. The score was an integer between 1 and 5, with 1 being least responsive and 5 being most responsive. For a given topic, some summary was required to receive each of the five possible scores, but no distribution was specified for how many summaries had to receive each score. The number of human summaries per topic also varied. Therefore, raw responsiveness scores cannot be directly compared across topics. The result LAKE obtained for *scaled responsiveness* is reported in Table 2. As can be seen LAKE scored 19 out of 31 systems participating.

5.2 ROUGE Based Evaluation

A second evaluation was conducted running ROUGE-1.5.5 with the main goal of computing recall scores (i.e., ROUGE-2 and ROUGE-SU4), even though other scores are computed by the system. Table 2 reports the results of these two scores. For both the evaluation LAKE scored 20 out of 31 participating systems.

5.3 Pyramid Based Evaluation

ROUGE provides an automatic method to evaluate systems, however, Nenkova *et al.* (Nenkova and Passoneau, 2004) showed that ROUGE measure cannot be used as an absolute measure of the system's performance. To fill up this gap they proposed the *Pyramid* approach, that is a manual method for summarization evaluation, developed in an attempt to address the fact that humans choose different words when write a summary.

Table 3: Results for the Pyramid metric.

Peer id id	Average Score	Rank Score
14	0.2477	1
17	0.2398	2
10	0.2340	3
15	0.2322	4
7	0.2307	5
4	0.2197	6
16	0.2170	7
32	0.2134	8
6	0.2110	9
19	0.2089	10
12	**0.2086**	**11**
11	0.2085	12
21	0.2063	13
26	0.1970	14
28	0.1944	15
3	0.1894	16
13	0.1855	17
25	0.1691	18
1	0.1666	19
27	0.1631	20
31	0.1587	21
24	0.1491	22
20	0.1446	23
30	0.1376	24
23	0.1216	25

In short, the method seeks to match content units in peer summaries (i.e., produced automatically by the systems) with similar content units found in a pool of human summaries. A good peer summary is one where its contents units are observed across many human summaries.

Table 3 and Table 4 show the results obtained. LAKE obtained competitive results scoring 11^{th} and 10^{th}, respectively for *score* (also named *original score*) and for *modified score*. The *original score* uses as X the same number as units appearing in the peer (i.e., it is precision oriented), while the *modified score* uses as X the average number of units found in the human (model) summaries (i.e., it is recall oriented).

6 Conclusion and Future Work

In this paper we reported on ITC-irst participation at DUC-2005. We have described the LAKE system, which exploits keyphrases extraction for summarization. The system couples a rather sophisticated lin-

Table 4: Results for the Pyramid metric.

Peer id	Average Modified Score	Rank Score
10	0.2000	1
17	0.1972	2
14	0.1874	3
7	0.1840	4
15	0.1793	5
4	0.1722	6
16	0.1706	7
11	0.1691	8
19	0.1672	9
12	**0.1645**	**10**
6	0.1639	11
32	0.1607	12
21	0.1589	13
3	0.1459	14
26	0.1413	15
13	0.1412	16
28	0.1400	17
25	0.1395	18
27	0.1306	19
1	0.1258	20
31	0.1215	21
24	0.1140	22
30	0.1131	23
20	0.0937	24
23	0.0609	25

guistic analysis of the documents, used for candidate phrases extraction, with a simple binary classifier, used for assigning a score to candidate phrases. The linguistic processing includes both multiwords and named entities recognition, while the classifier uses as feature both $TF \times IDF$ and the position in the document. Summaries are generated considering both the relevance and the coverage of keyphrases for a certain topic. Results show an high linguistic quality of the summaries and an average responsiveness. Moreover, results obtained in Pyramid metric seem very competitive.

7 Acknowledgments

We would like to thank Roberto Zanoli and Alberto Lavelli for their contribution in these two years.

References

A. Arampatzis, T. van der Weide, C. Koster, and P. van Bommel. 2000. An evaluation of linguistically-motivated indexing schemes. In *In Proceedings of the BCSIRSG '2000*.

Luciana Bordoni and Ernesto D'Avanzo. 2002. Prospects for integrating text mining and knowledge management. *The IPTS Report*, 68:21–25.

Ernesto D'Avanzo and Tsvi Kuflik. 2005. Linguistic summaries on small screens. In *Data Mining VI*, pages 195–204. WIT Press.

Ernesto D'Avanzo, Bernardo Magnini, and Alessandro Vallin. 2004. Keyphrase extraction for summarization purposes: The lake system at duc-2004. In *HLT/EMNLP. Human Language Technology Conference. Conference on Empirical Methods in Natural Language Processing*.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *IJCAI*, pages 668–673.

C. Gutwin, G. Paynter, I. Witten, C. NevillManning, and E. Frank. 1998. Improving browsing in digital libraries with keyphrase indexes. Technical report, Department of Computer Science, University of Saskatchewan, Canada.

A. Nenkova and R. Passoneau. 2004. Evaluating content selection in summarization. In *Proceedings of the HLT-NAACL conference*.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, Manchester, UK.

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

P.D. Turney. 1999. Learning to extract keyphrases from text. Technical Report ERB-1057. (NRC #41622), National Research Council, Institute for Information Technology.

P.D. Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2 (4):303–336.

Ian H. Witten and Eibe Frank. 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.