Keyphrase Extraction for Summarization Purposes: The LAKE System at DUC-2004

Ernesto D'Avanzo* davanzo@itc.it

Bernardo Magnini† magnini@itc.it

Alessandro Vallin‡ vallin@itc.it

ABSTRACT

We report on ITC-irst participation at Task 1 (very short document summaries) at DUC-2004. We propose to exploit a keyphrase extraction methodology in order to identify relevant terms in the document. 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 multiwords and named entities recognition, which are not usually exploited.

Keywords

Linguistic Processing, Machine Learning, Keyphrase Extraction, Summarization

1. INTRODUCTION

As our first participation in DUC, we devoleped LAKE, a system that exploits the role of Keyphrase Extraction (hereafter KE) as a useful approximation to summarization, evaluating its performance in Task 1 (*very short single document summaries*. 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.

Keywords, or keyphrases[1], provide semantic metadata that characterize documents, producing an overview of the subject matter and contents of a document. Keyword 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 [12] and [5]).

There are two major tasks exploiting keyphrases: keyphrase assignment and keyphrase extraction (see [11]). 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 [10]. 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 [10, 4]. 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.

The paper is organized as follows. In Section 2 we report on the general architecture of our system, which combines a machine learning approach with a linguistic processing of the document. Section 3 shows the results obtained by the system and discusses the evaluation carried out with ROUGE. We conclude suggesting possible future improvements.

2. THE LAKE SYSTEM

In this section we describe LAKE (*Learning Algorithm for Keyphrase Extraction*), a keyphrase extraction approach developed at ITC-irst based on a supervised learning approach that makes use of a linguistic processing of the documents. The system works in two phases (see Figure 1): first it considers a number of linguistic features to extract a list of well motivated candidate keyphrases from a given document; then it uses a machine learning framework to select significant keyphrases for that document.

*Department of Information and Telecomunnications, University of Trento, and ITC-irst, Trento, Italy

†ITC-irst, Trento, Italy

‡ITC-irst, Trento, Italy

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

Document Understanding Workshop, HLT/NAACL Annual Meeting 2004 Boston Park Plaza Hotel and Towers, Boston, USA

More in detail, *candidate phrases* consist of words or sequences of words that match a set of previously manually defined linguistic patterns (see section 2.2). Then (see section 2.3), a supervised learning algorithm is used to score the head of each phrase, according to features (e.g. TF/IDF and the position within a document) that signal the relevance of the head in the whole document collection. The motivation for considering the heads of the candidate phrases instead of the phrase itself is that phrases do not appear frequently enough in the collection. Finally, the score of the head is assigned to the whole candidate phrase, and the best scored phrases that fill the 75 bytes required by the task are given as output.

Both the linguistic patterns and the features used in the classification phase have been defined considering the DUC-2003 material.

In the following sections we describe each of these components.

2.1 Linguistic Pre-Processing

The document collection provided by DUC was preprocessed according to the following three steps: (i) part of speech tagging, (ii) multiwords recognition, (iii) named entities recognition.

2.1.0.1 *Part of Speech Tagging.*

We use the Tree tagger [9] POS tagger developed at the University of Stuttgart. The tagged text is given to a module that recognizes multiword expressions.

2.1.0.2 *Multiwords Recognition.*

Sequences of words that are considered as single lexical units are detected in the input document according to their presence in WordNet [3]. For instance, the sequence "Christmas trees" is transformed into the single token "christmas_tree" and assigned to the part of speech found in WordNet.

2.1.0.3 *Named Entities Recognition.*

As for Named Entities recognition we used NERD [7], a multilingual Named Entity Recognizer for Italian and English. The system has been designed for the identification and the categorization of entity names (persons, locations and organizations), temporal expressions (dates and times) and certain types of numerical expressions (measures, monetary values, and percentages) in written texts. NERD relies on the combination of a set of language-dependent rules (approximately 350 for English and 400 for Italian) with a set of language-independent predicates, defined on the WordNet hierarchy, for the identification of both proper nouns and *trigger words*. The system, integrated into the DIOGENE Question Answering architecture, has been successfully used for the ITC-irst participation in the last two editions of the TREC QA main task, and in the first edition of the CLEF multiple language QA track.

2.2 Candidate Phrase Extraction

The selection of relevant phrases has been strongly task-oriented. Within the framework of the DUC summarization task, the concept of relevance was heuristically defined: we considered significant the syntactic patterns that described either a precise and well defined entity, or concise events/situations. In the former case we focussed on uni-grams and bi-grams (for instance *Named Entity*, *noun*, *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*). Despite the important role heuristics play in our approach, the choice of relevant patterns was also linguistically motivated: sequences like *noun+adjective*, that are not allowed in English, were not taken into consideration.

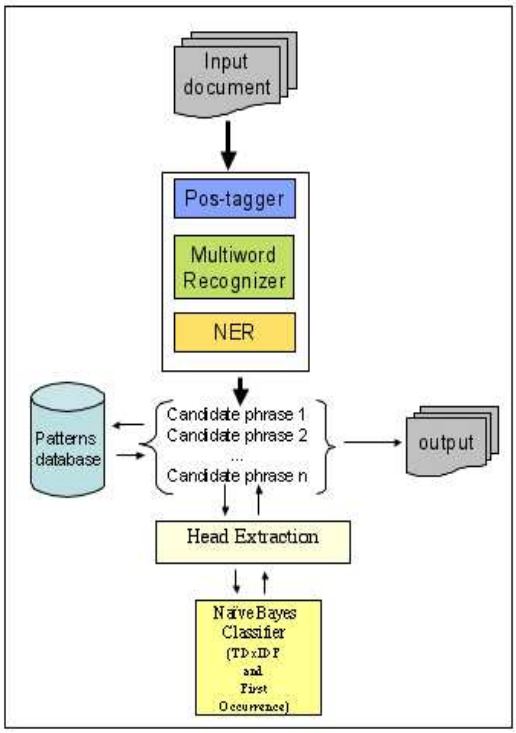

Figure 1: Architecture of the LAKE system.

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. In order to outline three different settings (i.e. runs), we wrote three versions of our PoS-pattern filter: one containing 121 patterns (without any verbal form), one containing 223 patterns (with a few verbal constructions) and the largest one that consisted of 654 patterns (with verbs, adverbs and prepositions). Once we had extracted all the uni-grams, bi-grams, tri-grams, and four-grams from the PoS-tagged output of NERD, we filtered them with the patterns we previously defined.

Table 1 shows some of the candidate phrases that our largest filter accepted as candidates from the document that reports on the possible extradiction of Pinochet from London (where he is recovering from an operation) to Spain (document APW19981023.1166). We use the following abbreviations (borrowed from Penn Treebank Project[2] (see [8])): NE stands for Named Entity, JJ for Adjective, NN for a singular noun, CC for a conjunction, VDB for a verb at the past tense, MD for a modal auxiliary, VBN for a verb

[2]http://www.cis.upenn.edu/ treebank/home.html

	Pattern	Example
Uni-grams	NE	*London*
	NE	*1973*
Bi-grams	JJ+NN	*Chilean dictator*
	JJ+NN	*Spanish magistrate*
	JJ+NN	*urinary infection*
Tri-grams	NN+CC+NN	*genocide and terrorism*
	NN+VBD+NE	*newspaper reported Friday*
	NN+VBD+NN	*room lacked television*
Four-grams	NE+MD+VB+VBN	*Augusto Pinochet would be extradicted*
	VBN+IN+JJ+NNS	*detained by British police*
	NN+TO+VB+NN	*extradition to stand trial*
	NN+VBD+JJ+NN	*dictatorship caused great suffering*
	VBN+IN+DT+NE	*detained in the London*
	NN+VBD+VBN+NN	*family had asked police*

Table 1: A sample of candidate phrases extraction.

at the past participle, IN for a preposition, NNS for a plural noun, TO for the preposition TO, VB for a verb in its base form, DT for a determiner.

2.3 Scoring Candidate Phrases

In this phase we assign a score the each candidate phrases in order to rank them and to make it easier their selection for the final 75 bytes output. The basic idea was to implement a binary classifier that, given a candidate phrase, would be able to classify it either as a relevant keyphrase for a given document or as not relevant for that document. The classifier was trained on two features: $TF \times IDF$ (i.e. the product between the frequency of a candidate phrase in a certain document and the inverse frequency of the phrase in all the documents belonging to the same cluster) and *First Occurrence*, i.e. the distance of the candidate phrase from the beginning of the document in which it appears. Two reasons led us to choose these features: they are very easy to calculate and many systems performing at the state-of-the-art make use of them.

We estimated the values of the two features using the head of the candidate phrase, instead of the phrase itself. According to the *principle of headedness* [1], any phrase has a single word as head. The head is the main verb in the case of verb phrases, and it is a noun (the last noun before any post-modifiers) in noun phrases.

As for the learning algorithm we used the Naive Bayes Classifier provided by the WEKA package [13] and publicly available at the University of Waikato Web site [3]. The classifier was trained on the DUC-2003 material in the following way. From the document collection we extracted all the nouns and all the verbs. Each of them was marked as a positive example of 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 and the learner was run.

As a result, the classifier prefers the candidate phrases whose head both maximizes its $TF \times IDF$ and tends to occur at the beginning of a document. For example, the head *dictator* receives an high $TF \times IDF$ relatively to the document APW19981023.1166, suggesting that a candidate phrase such as *Chilean dictator*, selected by a linguistically motivated pattern, is likely to be relevant for that document.

[3] http://www.cs.waikato.ac.nz/ml/weka/

3. RESULTS AND DISCUSSION

Participating groups at DUC-2004 were allowed to submit up to three runs. Submissions were ranked according to their priority, i.e. according to the confidence in the results. As mentioned above (section 2.2), we designed three different settings, corresponding to the three versions of linguistic patterns we used. We assigned the highest priority to the run that exploited all the 654 patterns we manually extracted, because it is able to mark as candidate phrases also complex sequences including verbs. The shortest filter, containing no verbs, was used in the run with priority 2, while the intermediate one was chosen as priority 3.

Submissions were automatically evaluated using the ROUGE program [6]. Table 3 shows the results obtained on the three runs, each with a different ROUGE score for uni-gram (Rouge-1), bi-gram (Rouge-2), tri-gram (Rouge-3), four-gram (Rouge-4), and the longest common substring (Rouge-L). The best result has been obtained by LAKE on the second run, using a set of 223 linguistic patterns.

Table 2 shows the rankings of our submissions for Task 1 among all of the submission. As we can see, LAKE is in the middle of the ranking.

Figure 2 shows the results obtained by LAKE on the three runs, denoted with peer codes 87, 88, and 89 respectively, each with a different ROUGE score for uni-gram (Rouge-1), bi-gram (Rouge-2), tri-gram (Rouge-3), four-gram (Rouge-4), and the longest common substring (Rouge-L). The best result has been obtained by LAKE on the second run (peer code 88), using a set of 121 linguistic patterns (i.e., the patterns with no verbs). This confirms that patterns containing verbal forms introduced noise.

3.1 Discussion on the Linguistic Patterns

We encountered two major difficulties in definining the linguistic patterns:

1. since the relevance of a keyphrase cannot be defined exclusively from a syntactic point of view, the number of relevant patterns was potentially too large to be described. Important information that a human would extract from a document could be contained in every

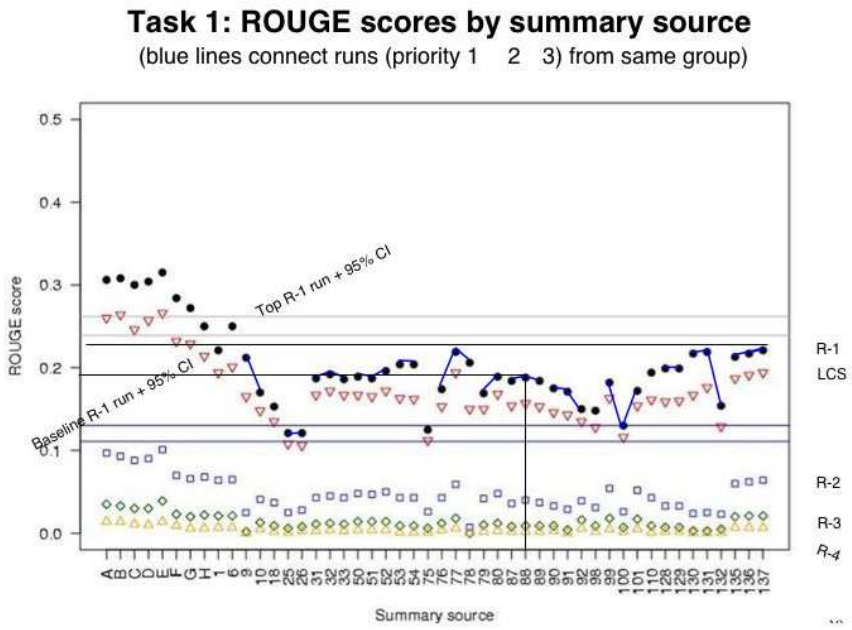

Figure 2: Rouge measure score.

peer code	Task	Priority	R-1	R-2	R-3	R-4	R-L	R-W	Total submissions
87	1	1	23	24	28	28	24	24	39
88	1	2	19	21	20	23	23	23	39
89	1	3	24	23	24	25	27	25	39

Table 2: Official rankings of our submissions for Tasks 1 with respect to ROUGE metrics

pattern (independently from its part of speech) or in several, non consecutive patterns;

2. it was particularly difficult to determine a priori the syntactic role of the sequences containing verbal forms: we aimed at extracting mainly noun phrases, but we often got truncated and irrelevant sentences.

To sum up, PoS-tagging information proved to be far from exhaustive, introducing a lot of noise. Some candidate phrases turned out to be useless pieces of longer sentences (*newspaper reported Friday*), or irrelevant, though syntactically complete, noun phrases (*urinary infection*, *family had asked police*). The shorter filter, containing no verbs, proved to be the most reliable one, as the automatic evaluation using the ROUGE software showed.

3.2 Discussion on the Features

The two features we used to train a classifier for candidate phrase scoring revealed as effective as we thought. However, it might improve the performance of the classifier to consider features able to capture some semantic properties of keyphrases. In particular, the fact that relevant keyphrases in a document tend to be related to the main topic of the document could be captured computing the lexical chains [2] of the document, and then considering the membership to such chains as a feature of the candidate phrase.

	Run 87	Run 88	Run 89
Rouge-1	0.18448	0.18817	0.18362
Rouge-2	0.03638	0.04001	0.03667
Rouge-3	0.00786	0.00933	0.00892
Rouge-4	0.00171	0.00207	0.00189
Rouge-L	0.15412	0.15656	0.15294

Table 3: Results of the LAKE system for three runs.

4. ACKNOWLEDGMENTS

We would like to thank people at TCC division at ITC-irst. In particular, Matteo Negri for the NERD system, Claudio Giuliano and Alfio Gliozzo, for the suggestions on the machine learning apparatus, Bonaventura Coppola for his help shortly before the deadline, and Alberto Lavelli for discussing the overall approach.

5. REFERENCES

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

[2] R. Barzilay and M. Elhadad. Using lexical chains for

text summarization. In *Intelligent Scalable Text Summarization Workshop, ACL*, 1997.

[3] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[4] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *IJCAI*, pages 668–673, 1999.

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

[6] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of 2003 Language Technology Conference*, 2003.

[7] B. Magnini, M. Negri, H. Tanev, and R. Prevete. A WordNet-Based Approach to Named Entities Recognition. In *Proceedings of the SemaNet'02 workshop on Building and Using Semantic Networks*, Taipei, Taiwan, 2002.

[8] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1994.

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

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

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

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

[13] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.