

Description of the UAM system for generating very short summaries at DUC-2004*

Enrique Alfonseca	José María Guirao	Antonio Moreno-Sandoval
Computer Science Department	Computer Science Department	Department of Linguistics
Universidad Autónoma de Madrid	Universidad de Granada	Universidad Autónoma de Madrid
28049 Madrid (Spain)	18071 Granada (Spain)	28049 Madrid (Spain)
Enrique.Alfonseca@ii.uam.es	jmguirao@ugr.es	sandoval@maria.111f.uam.es

Abstract

This paper describes the techniques used for producing very short summaries (around 75 bytes) of single documents. As in last year's version, the processing has been divided into two separate steps: firstly, a sentence extractor selects the most relevant sentences from the document; and, next, portions of those sentences are put together in order to produce the final headline. The main novelty is the way in which the text chunks have been weighted, and completed with keywords and noun phrases obtained from the documents. Our runs are ranked in the 12th and 13th positions with respect to unigram recall (ROUGE-1), but failed to identify bigrams, trigrams and four-grams as accurately as most of the other systems.

1 Introduction

The UAM system has only participated in the first task in DUC 2003: the generation of very short summaries (less than 75 bytes), a problem also called *headline generation*. Our procedure uses extraction techniques: selecting and putting together several portions of the original documents for building the summary. In our approach, there are two separate extraction steps:

- The identification of the most relevant sentences.
- The extraction of relevant words and phrases from those sentences, while keeping the coherence of the output as much as possible.

1.1 Related work

Many previous systems for *headline generation* follow a two-level approach, in which the most relevant sentences are first selected, and then they are processed, removing chunks from them, until the total length does not exceed the limit. The fact that the most relevant ideas are usually expressed in a few sentences allows for this simplification, and most of the document can be discarded at the beginning. Sentence-extraction procedures are already well-studied and provide good results [Mani, 2001], so we shall centre on the differences in the compaction step. Some approaches are:

- Deletion of subordinate clauses
- Deletion of stopwords (determiners, auxiliary verbs, etc.) [Angheluta et al., 2003, Zhou and Hovy, 2003].

*This work has been sponsored by CICYT, project number TIC2001-0685-C02-01.

- Extracting fragments of the sentences using syntactic information, e.g. the verb and some kind of arguments, such as subject, objects or negative particles [Fuentes et al., 2003, Alfonseca and Rodríguez, 2003a, Lăcătușu et al., 2003, Dorr et al., 2003].
- Using pre-defined templates [Daumé III et al., 2003].

Other different approach to the problem consists in extracting, from the whole document, a list of topical keywords and multiword expressions (either collocations or base Noun Phrases) [Angheluta et al., 2003, Bergler et al., 2003, Kraaij et al., 2003]. This usually does not produce a grammatical or coherent summary, but it may provide a useful topical description.

1.2 Structure of the paper

This paper is structured as follows: Section 2 describes the method for extracting a few sentences from the document; Sections 3 and 4 describe the way in which we choose relevant chunks from those sentences and generate the final headlines; finally, Section 5 briefly summarises the approach followed and discusses the conclusions obtained.

2 Sentence extraction

In last year's approach [Alfonseca and Rodríguez, 2003a,b], we described a procedure, based on genetic algorithms, for finding the choice of sentences from a document that maximises some particular fitness function. The use of genetic algorithms for summarization had been used previously [Jaoua and Hamadou, 2003]. If that function is designed for scoring informativeness, then the most informative sentences can be identified in a rapid way. The design of the genetic algorithm is repeated here for convenience of the reader.

Our particular implementation is strongly influenced by the so-called Edmundsonian paradigm [Edmundson, 1969], which consists in weighting the sentences with a linear combination of several variables. The following are the set of features that were considered useful for last year's approach, and which we have applied again:

- Summaries that contain long sentences are better summaries than summaries that contain short sentences [Marcu and Gerber, 2001]. A numerical function can be defined as the sum of the lengths of all the sentences in the extract (measured in number of words):

$$L(\mathcal{S}) = \sum_{i=0}^N length(s_i).$$

- Summaries that contain sentences that occur in the beginning of a paragraph in the original documents are better summaries [Lin and Hovy, 1997, Hovy and Lin, 1999, Mani, 2001]. Similarly, the position of the paragraph inside the document can also be encoded as a feature.

$$W(\mathcal{S}) = \sum_{i=0}^N 1/(\alpha + position_in_paragraph(s_i))$$

$$T(\mathcal{S}) = \sum_{i=0}^N 1/(\beta + paragraph_position_in_document(s_i))$$

- Summaries that contain the sentences in the same order than in the original documents are better than otherwise [Marcu, 2001]. This function can be directly implemented by forcing the sentences to be always ordered.

- Summaries that contain sentences from all the paragraphs are better than summaries that focus only on a few paragraphs [Marcu, 2001]:

$$C(\mathcal{S}) = |\{p : paragraph(p) \wedge (\exists s \in \mathcal{S} : sep)\}|.$$

- Summaries that only contain complete sentences (with subject and verb) are better than summaries that contain any incomplete sentence:

$$V(\mathcal{S}) = |\{s : s \in \mathcal{S} \wedge has_subject(s) \wedge has_verb(s)\}|$$

- Questions are usually low-informative sentences:

$$I(\mathcal{S}) = -|\{s : s \in \mathcal{S} \wedge is_a_question(s)\}|$$

0	0	0	0	1	1	0	1	0	1	0	1	1
0	0	1	0	0	1	0	0	0	1	0	1	1
1	0	0	0	0	1	0	1	0	0	0	0	1
0	0	1	0	0	0	1	0	0	1	0	1	0
0	0	1	1	0	0	1	0	0	1	0	0	1

Figure 1: Initial population of summaries. Each line is the genotype of the summary of a document with 13 sentences. For instance, the first summary consists of the sentences at the positions 5, 6, 8, 10, 12 and 13.

- Sentences with a high degree of word overlapping probably convey the same information, so the summary is redundant. The word overlapping can be measured, for instance, using the cosine similarity [Otterbacher et al., 2002]. We calculate it as the number of pairs of sentences from the summary that have a number of common words higher than a threshold θ :

$$O(\mathcal{S}) = -|\{< s, t >: s, t \in \mathcal{S} \wedge \text{word_overlapping}(t, s) > \theta\}|$$

- Sentences whose words have a high topical weight should be more informative about the topic of the document [Lin and Hovy, 2000]. We have used the χ^2 weights:

$$\chi(\mathcal{S}) = \sum_{s_i \text{ in } \mathcal{S}} \sum_{w_j \text{ in } s_i} \text{weight}(w_j)$$

If all the weight functions just depend on the sentence we are currently weighting, then the easiest way to find the best summary is to weight each sentence separately and choose the best ones. However, for some applications we might want to take into account that fact that one sentence has been selected in order to choose another. This is our case, because weight function $O(\mathcal{S})$ has a different value depending on the combination of sentences that has been chosen. Thus, it is necessary to weight all the possible complete summaries in order to find the best one. However, as Marcu [2001] explains, when scoring summaries for generating an extract, current search procedures are slow if the weight of a sentence depends on whether other sentences have been selected or not. Our proposal consisted in using genetic algorithms for finding the summaries with high scores in a very short time.

Let us suppose that the original document has N sentences: then, each summary is represented as a vector of N bits, where a 0 in the i^{th} position means that the i^{th} sentence will not be extracted, and a 1 means otherwise. Initially, the algorithm starts with a random set of vectors of bits, each of which is considered an individual in a population. For instance, let us suppose that a document contains 13 sentences. Figure 1 shows a possible initial population of five random summaries of that document. Next, at every generation, the fitness of each summary is calculated, using a linear combination of the functions described above:

$$F(\mathcal{S}) = w_L \cdot L(\mathcal{S}) + w_W \cdot W(\mathcal{S}) + w_T \cdot T(\mathcal{S}) + w_C \cdot C(\mathcal{S}) + w_V \cdot V(\mathcal{S}) + w_I \cdot I(\mathcal{S}) + w_O \cdot O(\mathcal{S}) + w_\chi \cdot \chi(\mathcal{S}) - (\text{summary_length} - \text{target_length})^2 \quad (1)$$

At each generation, the less adapted summaries disappear, and the others remain and replicate. Mutation and crossover allows for variations in the population, and ideally the best fitted summary is found in a few hundreds of generations.

The implementation of the genetic algorithm that we have used is the PGAPack library [Levine, 1996]. The method is very easy to program, and the performance is good considering its complexity. Using a Pentium III 900MHz with Linux RedHat 9, it takes in average 20 seconds to obtain the extracts of a document collection.

2.1 Training the model weights

As in last year, the weights for each function were obtained by training yet another genetic algorithm on top of the sentence extractor, which evolved the possible weights.

By running an additional genetic algorithm on top of the previous one, it was possible to discover automatically the weights that produce the best results. The procedure chosen was the following:

Year	w_L	w_W	α	w_T	β	w_C	w_V	w_I	w_O	θ	w_χ
2003	0.0125	0.00625	8.25	0.01875	2	0	0.5	15.9375	0.5	1	-
2004	4.0	0.625	-3.5625	4.125	0.5	-	0	-	0.0625	1	0

Table 1: Weights obtained automatically with a genetic algorithm.

1. Initialise the population as a vector of weights, where each weight is an 8-bit fixed point numbers with 4 bits for the integer part.
2. The fitness function for each vector of weights is the unigram recall of the extract generator using those weights (excluding closed-class words).
3. A genetic algorithm finds the best weights.

The first row in Table 1 shows the weights that we obtained, using this procedure, and training with a handful of hand-written summaries, in 2003. The second row displays the weights obtained in 2004 training with all the data set from the DUC-2003 competition. Last year we did not use χ^2 for training, so the value of that weight was not calculated. This year we have disregarded w_C , as its weight was 0, and w_I , whose weight has been directly set to the maximum, as questions have proved to be much less informative than assertions.

The first thing to note is the large disparity of the weights between both experiments:

- The length of the sentence appears to be now much more relevant. This is reasonable, as longer sentences are expected to attain a higher recall.
- The position of the sentence in the paragraph, unexpectedly, now receives a negative value for the first four sentences in each paragraph. This is surprising, as the first sentence in the document is usually expected to be more informative than the rest.
- The position of the paragraph in the document now receives a large weight, so sentences from the first paragraph will be more likely to be selected.
- The sentence overlapping metric again receives a positive weight, but much lower than before.
- Unexpectedly as well, the χ^2 values and the incomplete sentences (w_V) have received a weight zero, so they do not seem to be correlated with the informativeness of the sentences.

Before attempting to draw conclusions from these results, it must be noted that we repeated the experiment once more, using only for training the documents from the collection D100 from DUC-2003. And, in this last experiment, the weights obtained were also very different from those shown in Table 1. By examining the sentences one by one, it can be observed that most of them have a similar number of unigrams in common with the judges' summaries. This implies that there are many possible ways of generating summaries that maximise the fitness function used. Therefore, the weights shown above should not be taken as definitive, as more work is probably necessary in order to find a better fitness function that discriminates between sentences with the same unigram recall.

3 Generating the very short summaries

The sentence extraction algorithm was executed on the original DUC documents so we obtained three or four sentences from each. As 4(are) y'6(As)-3agrap92(pr)9(eac)27(h1(g,)-361(w)28(e)-309(c)-n(lu)1(s)-1(i)1(28(ed)8376(t

- A stemmer based on the LaSIE stemmer [Gaizauskas et al., 1995].
- Three chunkers written in C++ and Java; the first one detects Complex Quantifiers; the second one detects base Noun Phrases, and the third one detects complex verbs [Manandhar and Alfonseca, 2000].
- A subject-verb and verb-object detector, written in Java *ad hoc* with hand-crafted rules.

Furthermore, we have used a multiword chunker, that locates and marks words that tend to appear together in the whole collection (e.g. *Czech Republic* or *New York*), so they are treated as single tokens.

After that processing is done, all the verb phrases are extracted from the document. Most of them contain pointers to their subject, arguments and agent (if in passive). Furthermore, for all the words in each document we calculated their χ^2 weight using the rest of the documents as contrast set. For this, the multiword expressions were considered as single units.

In order to rank the verb phrases, the following algorithm was used:

1. Take each verb phrase and its arguments (subject, direct object and agent).
2. Discard a verb phrase if any of the following holds:
 - If the parser has not been able to identify any argument of the verb phrase.
 - If the verb, in WordNet, is a hyponym of *communicate*, *utter*, or *judge*, because these verbs usually are not relevant enough to appear in the headline.
 - If the subject is in first or second person.
 - If the verb is in passive voice, and the parser did not find its subject nor its agent.
 - If the sum of the χ^2 weights of all the words in its arguments is zero.
3. If the previous filters have removed all the verb phrases from a document, then consider them all and apply no filters, because we do not want to get an empty summary.
4. For each of the remaining verb phrases, calculate the sum of the χ^2 weights of all the words in the arguments.
5. Rank them according to that metric.

Figure 2 shows a 4-sentence sample extract, and Table 2 shows the verb phrases that were extracted. As can be seen, two of them were discarded because the main verb was *to say*, and other three were left out because the words in their arguments had a zero χ^2 weight associated. Table 3 shows all the summaries that can be obtained, with different lengths, by choosing a different number of verb phrases at each time. As can be seen, the last summary is very complete and contains all the relevant information from the document. In our case, in which the summary is limited to a length of 75 bytes, we have to stop with only one verb phrase, although this headline misses much of the relevant information (that the document is about schizophrenia, and the patients improved with the treatment).

Finally, we included the prepositional complements of the arguments of a verb when their χ^2 weight was above a threshold (that we set at 100 by trial and error). This helped choose phrases such as “*its site is a favourite for the sale*” or “*selling used books on eBay*”.

4 Generating the summaries

As said before, the verb phrases are chosen in decreasing order of weight until the total length of the generated summary is higher than 75 bytes. However, it can be seen from the example in Table 3 that the length limit sometimes leaves out important information and, at the same time, there is still plenty of space to complete the generated extracts. In the example, the summary has 34 bytes, so there is still place for 41 additional bytes. Therefore, we decided to complete them as described below.

Schizophrenia patients whose medication could not stop the imaginary voices in their heads gained some relief after researchers repeatedly sent a magnetic field into a small area of their brains.

About half of 12 patients studied said their hallucinations became much less severe after the treatment , which feels like having a woodpecker knock on your head ” once a second for up to 16 minutes , said researcher Dr. Ralph Hoffman.

The voices stopped completely in three of these patients.

The effect lasted for up to a few days for most participants , and one man reported that it lasted seven weeks after being treated daily for four days.

Figure 2: Example extract from the first document in collection d100a.

Phrase	Filtered	Weight
Schizophrenia patients could not stop the imaginary voices	not	702.8202
Schizophrenia patients gained some relief	not	403.1114
researchers sent a magnetic field	not	1565.7462
12 patients studied	not	404.54895
About half of 12 patients studied said their hallucinations	[say]	-
their hallucinations became much less severe	[no weight]	-
the treatment feels	[no weight]	-
having a woodpecker	[no weight]	-
... said researcher Dr. Ralph Hoffman	[say]	-
The voices stopped	not	249.70877

Table 2: Verb phrases extracted from the extract in Figure 2.

Summary	Bytes
researchers sent a magnetic field	34
Schizophrenia patients could not stop the imaginary voices; after researchers sent a magnetic field	100
Schizophrenia patients could not stop the imaginary voices; Schizophrenia patients gained some relief, after researchers sent a magnetic field	144
Schizophrenia patients could not stop the imaginary voices; Schizophrenia patients gained some relief, after researchers sent a magnetic field; 12 patients studied	165
Schizophrenia patients could not stop the imaginary voices; Schizophrenia patients gained some relief, after researchers sent a magnetic field; 12 patients studied; The voices stopped	185

Table 3: Summaries with different levels of compression that can be obtained by concatenating the verb phrases with the highest scores.

4.1 Keyword extraction

For the headlines for which there was still space left, we added the highest-frequency words from the document and from its collections. Needless to say, a keyword is only added if it does not appear already in the summary. The summary of the previous example, after this step, becomes the following:

study, patient, schizophrenia, treatment, researchers sent a magnetic field.

It is possible to guess the meaning of the document from the summary: a study about schizophrenia patients, in which researchers sent a magnetic field as treatment. Although the use of separate keywords has diminished greatly the grammaticality and readability of the texts, we have recovered some of the information that the 75-byte limit had forced us to lose.

4.2 Noun phrase extraction

After the addition of the keywords, for some of the documents there was still free space left until the 75 bytes. In the second run submitted, we have extended the summaries with noun phrases chosen from the documents.

The criteria used to weight a noun phrase are the following:

- Its length (in number of words)
- The sum of the χ^2 weights of its words.
- Whether it is headed by a proper noun or not (a function that can take as values 1 or 0).
- The number of proper nouns in the noun phrase.
- The number of multiword expressions.
- The frequency of the sequence of PoS tags of this Noun Phrase among the hand-written summaries from DUC-2003.

The total weight assigned to a Noun Phrase is calculated as a linear combination of the six functions above. As in the case of the sentence selector, the weights for the linear combination were obtained by training a genetic algorithm to find the particular weights for which the noun phrases selected maximised the unigram recall on the DUC-2003 data.

Finally, the summaries with less than 75 bytes were extended with the top scoring Noun Phrases until their length exceeded the limit. The following are a few example headlines from the DUC 2004 data set in which there is a noun phrase added at the end.

module, shuttle, station, zarya, Cabana fired Endeavour 's thrusters, the Russian-made Zarya control module.

indonesia, to decide the fate of East Timor; to establish the real will, Indonesian-controlled East Timor.

official, the objective to disrupt the training , organization and infrastructure of the Bin network.

5 Conclusions

We have described an approach for automatic generation of very short summaries (no more than 75 bytes) from articles. It starts by selecting the most relevant sentences, using a genetic algorithm and a combination of well-known heuristics as the fitness function. The weights for each heuristic were obtained with another genetic algorithm built on top of them. Secondly, the verb phrases and their arguments were extracted from those sentences. Some of them were discarded, and the rest were ranked with respect to the χ^2 score of their constitutive words. For generating the summaries, they were connected with prepositions and conjunctions whenever possible. Finally, the extracts that still had space were completed with the top-frequency words from the documents and collections, and with noun phrases selected by means of a combination of criteria.

With the length limit of 75 bytes, and the evaluation metric proposed (ROUGE), a keyword extraction procedure probably produces better unigram results than choosing verb phrases. The fact that we scored much better on unigrams than on multi-grams indicates that a large part of our score is due to the keywords added at the beginning of the summary. The mixed approach implemented keeps some grammaticality, but at the same time includes the most informative keywords at the beginning.

In our experiments, by setting the length limit to 150 or 200 we would obtain summaries that were both grammatical and informative; however, the length limit forced us to leave out relevant verb phrases, and thence it was necessary to complete them with keywords.

References

- E. Alfonseca. Wraetlic user guide version 1.0, 2003.
- E. Alfonseca and P. Rodríguez. Description of the UAM system for generating very short summaries at DUC-2003. In *Proceedings of the Document Understanding Conference-2003*, 2003a.
- E. Alfonseca and P. Rodríguez. Generating extracts with genetic algorithms. In *Advances in Information Retrieval*, volume 2633 of *Lecture Notes in Computer Science*, pages 511–519. Springer-Verlag, 2003b.
- R. Angheluta, M.-F. Moens, and R. De Busser. K. u. leuven summarization system - DUC 2003. In *Proceedings of DUC-2003*, 2003.
- S. Bergler, R. Witte, M. Khalife, Z. Li, and F. Rudzickz. Using knowledge-poor coreference resolution for text summarization. In *Proceedings of DUC-2003*, 2003.
- T. Brants. Tnt – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, WA, U.S.A, 2000.
- H. Daumé III, A. Echihiabi, D. Marcu, D.S. Munteanu, and R. Soricut. GLEANS: A generator of logical extracts and abstracts for nice summaries. In *Proceedings of DUC-2003*, 2003.
- B. Dorr, D. Zajic, and R. Schwartz. Hedge Trimmer: A parse-and-trim approach to headline generation. In *Proceedings of Workshop on Automatic Summarization*, Edmonton, 2003.
- H. P. Edmundson. New methods in automatic abstracting. *Journal of the Association for Computational Machinery*, 16(2):264–286, 1969.
- M. Fuentes, M. Massot, H. Rodríguez, and L. Alonso. Mixed approach to headline extraction for DUC 2003. In *Proceedings of DUC-2003*, 2003.
- R. Gaizauskas, T. Wakao, K. Humphreys, H. Cunningham, and Y. Wilks. University of sheffield: Description of the lasie system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 207–220. Morgan Kaufmann, 1995.
- E. Hovy and C.-Y. Lin. Automated text summarization in summarist. In *I. Mani and M. T. Maybury (eds.) Advances in Automatic Text Summarization*, pages 81–94. MIT Press, Cambridge, Massachusetts, 1999.
- M. Jaoua and A. Ben Hamadou. Automatic text summarization of scientific articles based on classification of extract’s population. In *Proceedings of CICLING-2003*, 2003.
- W. Kraaij, M. Spitters, and A. Hulth. Headline extraction based on a combination of uni- and multidocument summarization techniques. In *Proceedings of DUC-2003*, 2003.
- V. F. Lăcățusu, P. Parker, and S. M. Harabagiu. LITE-GISTEXTER: Generating very short summaries with minimal resources. In *Proceedings of DUC-2003*, 2003.
- D. Levine. User guide to the pgapack parallel genetic algorithm library, 1996.

- C.-Y. Lin and E. Hovy. Identifying topics by position. In *Proceedings of the 5th Applied Natural Language Processing Conference*, pages 283–290, New Brunswick, New Jersey, 1997.
- C.-Y. Lin and E. Hovy. The automated acquisition of topic signatures for text summarization. In *Proceedings of the COLING conference*, 2000.
- S. Manandhar and E. Alfonseca. Noun phrase chunking with APL2. In *Proceedings of the APL-Berlin-2000 Conference, Berlin. Also published as E. Alfonseca and S. Manandhar, Noun Phrase chunking with APL2, APL Quote Quad (ACM SIGAPL), Vol. 30:4, p. 135-143*, 2000.
- I. Mani. *Automatic Summarization*. John Benjamins Publishing Company, 2001.
- D. Marcu. Discourse-based summarization in DUC-2001. In *Proceedings of Document Understanding Conference, DUC-2001*, 2001.
- D. Marcu and L. Gerber. An inquiry into the nature of multidocument abstract. In *Proceedings of the NAACL'01 workshop on text summarisation*, Pittsburgh, PA, 2001.
- J. Otterbacher, A. J. Winkel, and D. R. Radev. The michigan single and multi-document summarizer for duc-2002. In *Document Understanding Conference, DUC-2002*, 2002.
- L. Zhou and E. Hovy. Headline summarization at ISI. In *Proceedings of DUC-2003*, 2003.