

# New Feature Sets for Summarization by Sentence Extraction

Hans van Halteren, *University of Nijmegen*

A possible starting strategy for automatic text document summarization,<sup>1-3</sup> a natural language processing task, is sentence extraction. An extraction system tries to select the sentences with the most important information from the documents. These sentences can simply be presented in full to a user as an indicative summary, or—because

they don't necessarily provide a coherent account—be used as a basis for further processing. Ideally, the document would be thoroughly analyzed using linguistic and world knowledge to determine which sentences are appropriate for the extract. In practice, the necessary analysis is still too immature or too computationally intensive to yield sufficient results. Many existing systems extract sentences on the basis of a limited set of mundane features. Still, the features that are most often used tend to be based on notions about document structure (for example, sentence position within the document, sentence length, repetition of words from the title or headings, selected cue words or phrases) or information content (for example, presence of high-frequency content words). The sidebar “Automatic Summarization” provides more details.

I wanted to investigate whether I could enhance the feature-based extraction strategy by including some features that weren't primarily information retrieval oriented. The features I wanted to try were originally developed to recognize writing style and so were somewhat geared toward reducing the influence of subject-specific (IR) elements. The underlying idea is that, when you try to summarize an article through sentence extraction, you're assuming that the most important information is concentrated in specific sentences. If this is indeed true, the article's author, knowing which sentences are the most important, might have consciously or subconsciously written these sentences in a different *style* (measurable, recurring patterns in the usage of vocabulary, grammar, text structure, and so on) from the rest of the arti-

cle. If true, this supposition would likely allow valuable additions to the sentence extraction toolbox.

People study automatic writing-style recognition primarily in the context of authorship attribution. In this task, you examine a certain text and try to determine which out of a given group of authors wrote it.<sup>4</sup> The decision is based on information about several *style markers*, or features, such as vocabulary size or the distribution of a small set of specific vocabulary items. You generally learn about the markers from inspecting other texts by the same authors. One main focus of authorship attribution research is to create an inventory of useful style markers.<sup>5</sup> Another important focus is to develop techniques for using these style markers to provide reliable-enough probability estimates for each potential author.<sup>6</sup>

I developed a new technique for estimating probabilities as well as several sets of style markers that complement this technique. I wanted to find out if this technique and these features could also be used to locate extractable sentences in a document.

## Style recognition features

Numerous features can be used for style recognition. In most cases, you select a limited set of features because the classification systems that use the features can only handle so many. Certain researchers tend to specialize in specific feature sets, which are partly inspired by the kinds of texts they study. Because it was unclear which features would work best for sentence extraction when I started the pilot experiment, I tried several different sets. Each focused on a specific aspect of the text. The features

*Machine learning features developed for authorship attribution can also be used for sentence extraction. The author's pilot extraction system outperformed a hard-to-beat positional baseline system.*

## Automatic Summarization

The reason for the interest in automatic summarization is obvious: there is too much information for humans to process without support tools. An automatic summary of a document (or even of a set of documents about the same subject) can help in deciding which documents are worth reading. Once summarization technology is advanced enough, the summary could even substitute for the underlying document.

Summarization falls naturally into two subtasks: *extraction*, which identifies the information that should be placed in the summary, and *abstraction*, which casts the information into a user-friendly form. Research on the latter is underway but still in its early stages. Extraction has a slightly longer history<sup>1-4</sup> but is still not fully developed. The state of the art is such that systems that select sentences on the basis of various surface clues are still competitive, as recent summarization contests show.<sup>5,6</sup>

If surface clues are relatively sufficient, a logical next step is to look to machine learning techniques to efficiently process these clues. The machine learning paradigm has shown its value in many areas in NLP, such as speech recognition, information retrieval, and even machine translation. Julian Kupiec, Jan Pedersen, and Francine Chen introduced machine learning for summarization;<sup>7</sup> they used a Bayesian classifier with access to features such as sentence length, sentence position, and presence of specific indicator phrases, thematic words, or proper names. Since that time, people have investigated various other features and machine learning techniques.<sup>5,6,8-11</sup>

### References

1. P.B. Baxendale, "Man-Made Index for Technical Literature—An Experiment," *IBM J. Research and Development*, vol. 2, no. 4, Oct. 1958, pp. 354–361.
2. H.P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM J. Research and Development*, vol. 2, no. 2, 1958, pp. 159–165; reprinted in I. Mani and M.T. Maybury, *Advances in Automatic Text Summarization*, MIT Press, 1999, pp. 15–21.
3. H.P. Edmundson, "New Methods in Automatic Abstracting," *Comm. ACM*, vol. 16, no. 2, Apr. 1969, pp. 264–285.
4. C.D. Paice, "Constructing Literature Abstracts by Computer: Techniques and Prospects," *Information Processing & Management*, vol. 26, no. 1, 1990, pp. 171–186.
5. D. Harman and D. Marcu, eds., *Proc. Document Understanding Conferences 2001 (DUC 01)*, Nat'l Inst. of Standards and Technology, 2001; [www-nlpir.nist.gov/projects/duc/pubs.html/#2001](http://www-nlpir.nist.gov/projects/duc/pubs.html/#2001).
6. U. Hahn and D. Harman, eds., *Proc. Document Understanding Conferences 2002 (DUC 02)*, Nat'l Inst. of Standards and Technology, 2002; [www-nlpir.nist.gov/projects/duc/pubs.html/#2002](http://www-nlpir.nist.gov/projects/duc/pubs.html/#2002).
7. J. Kupiec, J. Pedersen, and F. Chen, "A Trainable Document Summarizer," *Proc. 18th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR 1995)*, ACM Press, pp. 68–73; reprinted in I. Mani and M.T. Maybury, *Advances in Automatic Text Summarization*, MIT Press, 1999, pp. 55–60.
8. C. Aone et al., "A Trainable Summarizer with Knowledge Acquired from Robust NLP Techniques," *Advances in Automatic Text Summarization*, I. Mani and M.T. Maybury, eds., MIT Press, 1999, pp. 71–80.
9. A. Berger and V.O. Mittal, "Query-Relevant Summarization Using FAQs," *Proc. 38th Ann. Meeting Assoc. Computational Linguistics (ACL 99)*, Assoc. Computational Linguistics, 1999, pp. 294–301.
10. C.Y. Lin, "Training a Selection Function for Extraction," *Proc. 18th Ann. Int'l ACM Conf. Information and Knowledge Management (CIKM 99)*, ACM Press, 1999, pp. 55–62.
11. C.Y. Lin and E. Hovy, "Identifying Topics by Position," *5th Conf. Applied Natural Language Processing (ANLP 97)*, Assoc. Computational Linguistics, 1997, pp. 283–290.

in each set refer to properties of individual tokens. You select a single token—a word or punctuation mark—at a specific position in the text, and the features express properties of this token and its context at this position. For example, take the second occurrence of "the" in "The vice president of the company had to resign last month." You could assign properties to this token such as *current* = "the", *previous* = "of", and *part-of-speech* = *article*.

### Trigrams set

The first feature set, *trigrams*, is the simplest. It focuses on lexical context combined with information on position in the sentence. Linguistic analysis and access to context outside the sentence are unnecessary, and you can use the features even on very small fragments of text. The features are

- *Current token*. This is the token in its full

form, as it occurs in the text, including capitalization and diacritics. Punctuation marks are considered individual tokens.

- *Previous token*. This is the token to the immediate left of the current token or a special marker if the current token is first in the sentence.
- *Next token*. This is the token to the immediate right of the current token or a special marker if the current token is last in the sentence.
- *Sentence length*. This is the length, in tokens, of the sentence that includes the current token. The length is mapped onto one of seven possible values: 1, 2, 3, 4, 5–10, 11–20, and 21+. (I usually map numerical values to a few ranges for learnability and model size reduction. Generally, I choose the ranges intuitively instead of on the basis of statistical analysis.)
- *Position within the sentence*. This feature

can have three possible values. In sentences of length higher than six, I assign the first three tokens the value *Start*, the last three the value *End*, and the rest the value *Middle*. In shorter sentences, I only use *Start* and *End* with a mid-sentence dividing point.

### Tags set

This feature set requires a bit more knowledge about the tokens—frequency information and wordclass tags. Wordclass tags indicate part of speech and a few other morphosyntactic properties. Here are the features:

- *Current token*.
- *Current token's wordclass tag*. An automatic tagger<sup>7</sup> provides the tag, which is based on written texts from the British National Corpus Sampler CD-ROM ([www.hcu.ox.ac.uk/BNC/getting/sampler.html](http://www.hcu.ox.ac.uk/BNC/getting/sampler.html)).

- *Previous token's wordclass tag.* A special marker takes the tag's place if the current token is first in the sentence.
- *Next token's wordclass tag.* A special marker takes the tag's place if the current token is last in the sentence.
- *Token's frequency in the document.* I map the frequency onto one of five possible values: 1, 2–5, 6–10, 11–20, and 21+. Because of the mapping, no further mechanism is used to normalize the frequency on the basis of the document length.
- *Sentence length.*
- *Position within the sentence.*

**Distribution set**

This feature set ignores the local context and focuses on token distribution throughout the document. The features include

- *Current token.*
- *Current token's wordclass tag.*
- *Token's frequency in the document.*
- *Token length,* measured in ASCII characters. I map the length onto one of seven possible values: 1, 2, 3, 4, 5–10, 1–20, and 21+.
- *Token's distribution in the document.* I split the document into seven equal (in number of tokens), consecutive parts. I count the blocks the current token is in and then map that number onto one of four possible values: 1, 2–3, 4–6, and 7.
- *Distance to the token's previous occurrence,* measured in sentences and mapped onto one of seven possible values: none (meaning the current occurrence is the first), 0 (meaning the previous occurrence is in the same sentence), 1, 2–3, 4–7, 8–15, and 16+. All instances of the token must have the exact same form (for example, same capitalization).
- *Distance to the token's next occurrence,* also measured in sentences and mapped onto one of the seven values just listed.

**Classification software**

I built my classification software around the Weighted Probability Distribution Voting machine learning system.<sup>8</sup> WPDV is a supervised learning approach to automatically classifying items. The *case*, or set of information elements about the item being classified, is represented as a set of feature-value pairs (I use *set* and *subset* in their mathematical sense.). Here is a sample set from the authorship attribution task using the trigrams set on the sentence example mentioned earlier:

$$F_{\text{case}} = \{f_{\text{cur}} = \text{"the"}, f_{\text{prev}} = \text{"of"}, f_{\text{next}} = \text{"company"}, f_{\text{len}} = 5-10, f_{\text{pos}} = \text{middle}\}$$

I always treat the values as symbolic and atomic—not numerical or structured—and taken from a finite (although possibly large) set of possible values. I then estimate the probability of a specific class for the case in question on the basis of the number of times I saw that class with the same feature-value pair in the training data. To be exact, the probability  $P$  that class  $C$  should be assigned to  $F_{\text{case}}$  is estimated as a weighted sum over all possible subsets  $F_{\text{sub}}$  of  $F_{\text{case}}$ :

$$P(C) = N(C) \sum_{F_{\text{sub}} \subset F_{\text{case}}} W_{F_{\text{sub}}} \frac{\text{freq}(C|F_{\text{sub}})}{\text{freq}(F_{\text{sub}})}$$

For the controlled pilot experiment to determine the features' and the sentence extraction system's usefulness, manually annotated data distinguishing extract and nonextract sentences was vital.

with the frequencies (freq) measured on training data, and  $N(C)$  a normalizing factor such that

$$\sum_C P(C) = 1.$$

You can assign the weight factors  $W_{F_{\text{sub}}}$  in many different ways, but training material in the current sentence extraction task was so limited that I based the weight factors on the number of elements in the subset:

$$W_{F_{\text{sub}}} = B^{|F_{\text{sub}}|}.$$

I selected a weight base  $B$  of 0.8 after initial experiments indicated that it tended to lead to good results.

The WPDV system, then, estimates the probability  $P_{\text{token}}$  that a given token in a given context is in a given style. In a two-way authorship attribution task, you would use the styles of authors  $A$  and  $B$ ; in the sentence extraction task, you would use the extract

style and nonextract style. For an overall decision, you combine the individual tokens' probabilities. In the authorship attribution task, you sum the probabilities per token over all the tokens in the text sample; in the extraction task, you sum them over all tokens in a sentence. In both cases, this yields overall sentence scores  $T_{\text{overall}}$  for each style  $X$ :

$$T_{\text{overall}}(X) = \sum_{\text{token}} \text{If } P_{\text{token}}(X) > 0.5 \text{ Then } (P_{\text{token}}(X) - 0.5)^D \text{ Else } 0$$

in which you can use the decisiveness factor  $D$  to give more weight to more decisive local scores. The system projects the overall scores to the range  $[0 \dots 1]$  by dividing the sentence score by the sum of all overall scores:

$$P_{\text{overall}}(X) = T_{\text{overall}}(X) / \sum_{X_i} T_{\text{overall}}(X_i).$$

**The experiment**

For the controlled pilot experiment to determine the features' and the sentence extraction system's usefulness, manually annotated data distinguishing extract and nonextract sentences was vital. I used a set of single-document extracts—an average of 400 words long—that were based on per-document summaries of 147 documents (from 29 document sets) from the Document Understanding Conference (DUC) 2001 data.<sup>9</sup>

In the experiment, I focused on the precision and recall with respect to the extract sentences in the manually annotated data. However, because I wanted to consider all possible extract sizes rather than using one or more predefined sizes, I couldn't just measure the precision and recall for extracts with a predefined size. Using precision-recall curves was a good alternative.

Also, it's better to express precision and recall in terms of information content rather than the number of sentences. As there is no easy way to measure each sentence's actual information content, I roughly approximated it by the number of words in the sentence. For example, a 23-word sentence found in both the system and model outputs counted 23 points toward precision and recall, rather than just one point for the whole sentence.

I judged the relative quality of two precision-recall curves both visually and, more objectively, in terms of a measurement of the surface below the curve.

For each DUC 2001 document set in the data, I trained WPDV models for the various

feature sets on sentences from all other document sets. I used these models to select extract sentences from the document set under consideration.

I evaluated the models' relative performance on the basis of mutual comparison and on comparison with two baseline systems. The first baseline randomly decided whether a sentence was an extract sentence, and so this baseline could be considered an absolute lower limit on performance quality. The corresponding precision-recall curve is labeled "random" in Figure 1 and has a below-curve surface of 0.16. A more competitive baseline gave preference to sentences nearer to the start of the document—one of the most informative features known for the extraction task. Its curve is labeled "positional" in Figure 1 and has a below-curve surface of 0.39.

Figure 1 shows the selected extract sentences' precision as a function of the recall of extract sentences in all 29 sets. The graphs represent the scores for the optimum settings of D—2 for the trigrams and tags feature sets and 1 for the distribution feature set.

The trigrams and tags sets both performed better than chance, with surfaces of 0.23 and 0.24, but the positional system clearly outperformed them. The distribution feature set did much better, with a below-curve surface of 0.36—almost as good as the positional system. Because all three's performances were well above chance, they could contribute to a feature-based extraction system, but none appeared to be strong enough to use by itself.

## The WPDV-XTR extraction system

The pilot experiment showed that parts of the style recognition feature sets could be useful additions to the extraction system's feature sets. To test a system's potential using these features, I participated in the DUC 2002 multidocument extraction competition (<http://duc.nist.gov>). I used the WPDV-XTR system. My main decisions in constructing WPDV-XTR were

- Which of the features mentioned earlier should I select?
- How should I combine them with each other and possibly with other features?

Because I didn't have time for extensive experimentation before the DUC 2002 deadline, I chose a feature cocktail on the basis of

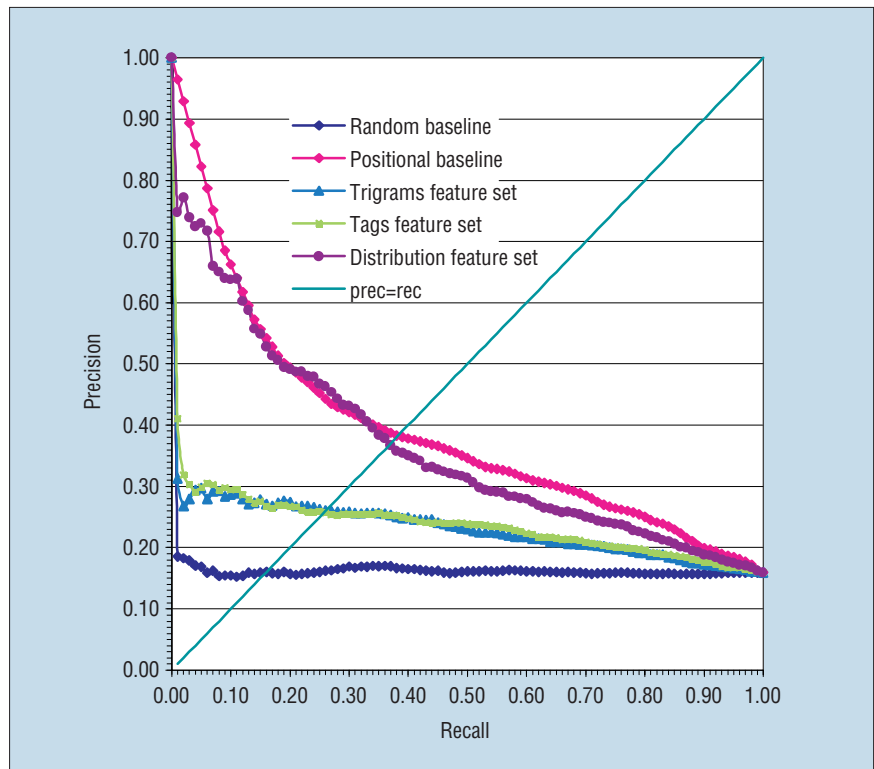


Figure 1. Precision-recall curves for two baseline systems and three experimental systems.

my observations from the pilot experiment and intuition about various information sources. Also, I didn't take any special measures to adapt WPDV-XTR to multidocument extraction. The choices I made weren't optimal, as you will see later.

### Selected features

I chose seven features for WPDV-XTR.

**Current token.** I used this feature exactly as defined earlier. This feature is probably most useful with function words, which are more likely to be influenced by style than content words. I used all the tokens—not just a selected collection of clear indicators (*cue phrases*) as some other systems do.<sup>2,10</sup>

**Distance to previous occurrence.** I also used this feature as explained earlier. A closer examination of the pilot experiment results showed that this feature was easily the most informative of all those used so far, so I kept it as is.

**Sentence length.** Although this feature didn't appear to do that well in the pilot experiment, I kept it intact because the extraction literature described it as useful.<sup>11,12</sup>

**Tag context.** The wordclass tags of tokens in the direct context had proven useful but not useful enough to warrant several separate features. I combined them into a single feature, with the sequence of three tags either ending at the current position or starting at the current position. I used both forms for each token position but in the same feature position, using feature overloading. (The WPDV system lets the same feature occur more than once in a single case, with different values. Both feature-value pairs are used in combinations but not combined with each other.)

**Token distribution in and across documents.** This feature proved useful but needed a more compact form. I modified the feature into a concatenation of three pieces of information:

- Token's frequency in the document.
- Token's distribution in the document.
- Token's *document bias*. This shows if the token occurs in many or few documents. To calculate a token's document bias, you divide its frequency (per 1,000 words) in the documents containing the token by its frequency in the whole document collection and take the log of the result (and for

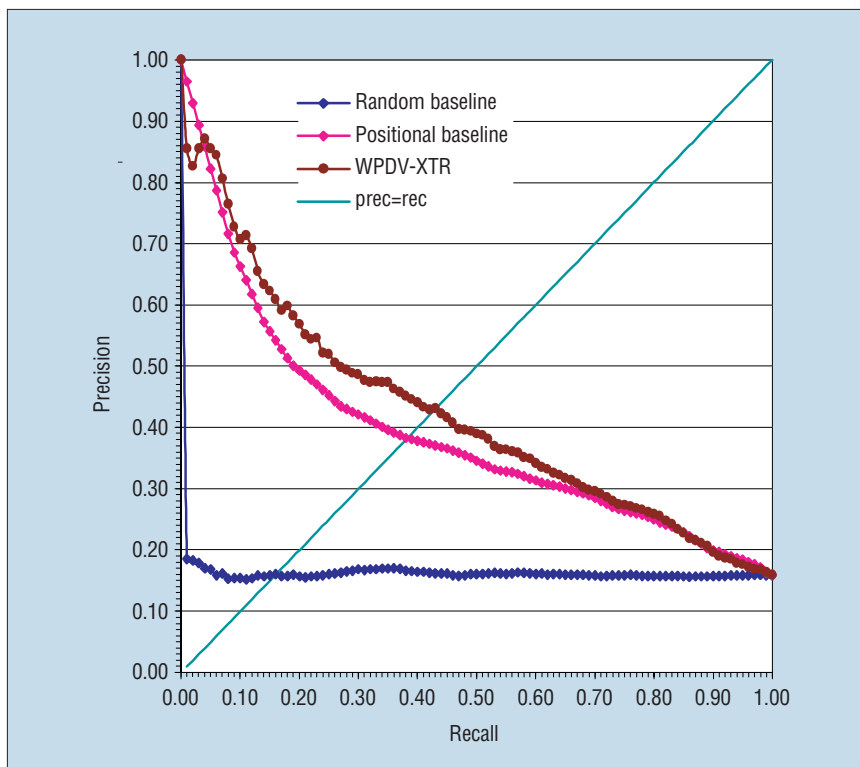


Figure 2. Precision-recall curves for WPDV-XTR and two baseline systems.

this task, round down to 0, 1, 2, 3, or 4). A low document bias indicates the token is mainly a function word, and a high document bias shows that it is mainly a content word.

**Relative token frequency.** This feature gives information about the current token's importance in the document. To calculate it, you divide its frequency (per 1,000 words) in the current document by its frequency in those documents from the document collection containing the token. Generally, the tokens central to the document's subject will have the highest relative frequency. I mapped the relative frequency onto one of six possible range values: 0–1/2 (meaning less than 0.5 times expected), 1/2–3/2, 3/2–2, 2–4, 4–8, and 8+.

**Sentence position.** The final feature in the cocktail, the sentence position in the document, was one of the best features for the task. Because most of the documents in the sources for the DUC 2002 data had no information about paragraph borders, I could use only the absolute position in the document. I mapped it onto one of six possible values: 1, 2–3, 4–7, 8–15, 16–31, and 32+.

### Results for training data

On the training data, WPDV-XTR (with the decisiveness factor  $D$  set to 4) performed better than the positional baseline on the chosen criterion, with a below-curve surface of 0.42. However, as Figure 2 shows, it performed worse than the baseline at the high-precision-low-recall end of the curve. This could mean that WPDV-XTR might yet perform worse than the positional baseline when having to produce small extracts.

### Sentence selection for predefined size extracts

In the pilot experiments, I looked only at the relation between sentence scores and a sentence's appropriateness for extraction. In an actual extraction system, however, you have a maximum number of words that you can include in the extract. The basic technique is obvious: start including sentences from the highest-scoring ones down and stop when you reach the size limit. However, there are a few details to work out.

First, the DUC 2002 competition required multidocument extraction—I had to summarize many different documents about the same subject in a single extract. It's far from clear that sentence scores from different doc-

uments should be comparable. However, because I lacked a solid theoretical basis for normalization, I decided just to use the sentence scores calculated on the basis of the individual documents for multidocument extract selections.

Then the possibility of repeated information exists, especially in multidocument summarization, because basic information could be repeated in several documents. Before including a sentence, you should check if the information in it is already present in a previous sentence in the extract. Given my rudimentary handling of content, all I know to do is to count how many words the sentences have in common. The question is, how much overlapping information should two sentences have for me to consider them repetitive? And which of two overlapping sentences should I include? Because I wasn't able to experiment and lacked a good basis for decisions, I ignored the problem for the time being. A later check of WPDV-XTR extracts showed that only about 1.5 percent of the space in 200-word extracts and about 2 percent of the space in 400-word extracts was wasted because of repeated information.

I also considered how to use the allowed space optimally. If 180 of the allowed 200 words are already taken and the next scoring sentence is 40 words long, it can't be included. However, the next scoring sentence after that might be short enough. The WPDV-XTR includes sufficiently short sentences until it has rejected 10 sentences because of their length.

Although it's unlikely, two sentences could have the exact same score. The WPDV-XTR first sorts the sentences and then selects top down, so that the (system-determined) secondary sort key actually solves ties. In the current implementation, this means that the system prefers the tying sentence that is nearer the beginning of the document, and in case of multidocument extraction, prefers sentences with earlier time stamps.

### Reevaluating system settings

Although WPDV-XTR scored better than the positional baseline in the pilot experiment (using my own evaluation measure), I didn't know how it compared to other state-of-the-art extraction systems or how well it performed tasks other than single-document extraction. So I participated in the DUC 2002 competition to find out.

The US National Institute of Standards

and Technology has held the annual DUC as part of DARPA's Tides (Translingual Information Detection, Extraction, and Summarization) program since 2000. They focus on creating and evaluating summarization systems. DUCs' goal is to gradually improve summarization by working on increasingly difficult summarization problems according to a published summarization road map ([www-nlpir.nist.gov/projects/duc/roadmapping.html](http://www-nlpir.nist.gov/projects/duc/roadmapping.html)). Toward this goal, in 2001, the NIST started this DUC competition in which several groups try to fulfill a number of NIST-defined summarization tasks. The NIST evaluates the summaries (abstracts or extracts) through various comparisons with summaries produced by human experts.

The DUC 2002 competition's multidocument extraction portion consisted of creating two generic sentence extracts of the document sets of approximately 10 single-subject newswire or newspaper articles. These extracts had to summarize the whole document set in approximately 200 and 400 (whitespace-delimited) tokens. The extracts had to consist of some subset of NIST-predefined "sentences" in a sentence-separated form of the document set, and systems had to use each predefined sentence in its entirety or not at all in constructing an extract. Ten groups entered multidocument extracts into the DUC 2002 competition.

## Evaluation

I again used precision and recall requirements to evaluate extraction systems. However, because the DUC 2002 task explicitly requested 200-word and 400-word extracts, I used these two sampling points instead of the overall precision-recall curve.

For these two points, I calculated precision and recall with regard to the extracts by human summarizers the DUC organizers provided (after the participants had sent in their contributions). In addition to the precision and recall, considering the number of words in each sentence ("W"), as used earlier, I also calculated precision and recall purely in terms of the number of sentences ("S"). The truth about information content should have been somewhere in the middle.

To derive a single measurement, I combined precision and recall into an F-value<sup>13</sup>

$$F = \frac{(\beta^2 + 1) * \textit{precision} * \textit{recall}}{\beta^2 * \textit{precision} + \textit{recall}}$$

with  $\beta$  equal to 1:

**Table 1. Results of 10 submissions for 200-word extracts.**

System	Sentence-based F-value	Word-based F-value
Manual	0.213	0.222
Position	0.191	0.215
WPDV-XTR	0.188	0.211
System 19	0.183	0.199
System 24	0.172	0.193
System 28	0.136	0.167
System 20	0.126	0.144
System 29	0.089	0.102
System 31	0.082	0.094
System 25	0.080	0.092
System 16	0.063	0.077
System 22	0.038	0.042
Random	0.030	0.032

$$F_{\beta=1} = \frac{2 * \textit{precision} * \textit{recall}}{\textit{precision} + \textit{recall}}$$

To get an overall quality assessment, I averaged the F-value over the measured extracts for the 57 document collections for which two human-made extracts were provided. Given the number of sentences under consideration and the values in question, the 95 percent confidence intervals for all F-values were about  $\pm 0.035$  for 200-word extracts and about  $\pm 0.030$  for 400-word extracts.

The availability of two human summarizers' extracts let me compare the results from automatic extraction with those from manual extraction. The average F-values for intersummarizer measurements of 0.213 (S) and 0.222 (W) for the 200-word extracts and 0.250 (S) and 0.270 (W) for the 400-word extracts were rather low. This means that I can't draw strong conclusions, because I don't know if two manually created extracts form a sufficient basis for a good benchmark.

In the following discussion, I refer to the Position system developed by Wessel Kraaij of TNO TPD, a Dutch research institute. This positional baseline system selected sentences that were closer to the start of the documents. It selected the first sentence of each document in the set, then each second sentence, and so on, until the extract's length was between 95 and 100 percent of the required length. Apart from the positional baseline, I also included measurements on a random baseline extractor and compared two manual extracts for each document collection.

**Table 2. Results of 10 submissions for 400-word extracts.**

System	Sentence-based F-value	Word-based F-value
Position	0.265	0.294
WPDV-XTR	0.258	0.290
Manual	0.250	0.270
System 19	0.223	0.240
System 24	0.222	0.249
System 28	0.197	0.241
System 20	0.172	0.191
System 29	0.156	0.179
System 31	0.153	0.172
System 25	0.148	0.165
System 16	0.128	0.156
System 22	0.084	0.097
Random	0.070	0.078

## DUC 2002 competition results

Table 1 shows the various systems' F-values and the positional baselines for 200-word extracts. Given the confidence intervals, WPDV-XTR, system 19, and system 24 lead the group—with minor differences—and are significantly better than the rest. Their extracts aren't significantly worse than the manual ones, except for system 24 at the sentence level.

Table 2 shows the results for 400-word extracts. In this case, WPDV-XTR performed significantly better than the other participating systems (according to the given measuring method) and even better than the human summarizers, but still worse than the positional baseline. Another method the DUC uses, vocabulary overlap measurement, yields a similar overall result, but places WPDV-XTR slightly ahead of the positional baseline and also puts manual extracts at the top for the 400-word case.

I must stress that these measurements are greatly influenced by the document types used in DUC 2002. Newswire and newspaper texts, especially for factual news, tend to put the most important information close to the start of the text; this explains the overwhelming success of the positional baseline system.

## Reevaluation

After the competition, I wondered if the WPDV-XTR system couldn't perform better than the positional baseline for this task in principle or if I just wasn't using the system

**Table 3. Individual features' effectiveness for 200-word extracts.**

Feature	Single (no. of sentences/ no. of words)	Contribution (no. of sentences/ no. of words)
Sentence position	0.202/0.227	+0.054/+0.062
Distance to previous	0.159/0.180	+0.010/+0.012
Current token	0.058/0.066	-0.008/-0.007
Relative token frequency	0.051/0.059	-0.015/-0.015
Token distribution	0.048/0.054	-0.002/-0.001
Tag context	0.049/0.052	-0.005/-0.004
Sentence length	0.031/0.036	-0.023/-0.019

**Table 4. Individual features' effectiveness for 400-word extracts.**

Feature	Single (no. of sentences/ no. of words)	Contribution (no. of sentences/ no. of words)
Sentence position	0.273/0.300	+0.068/+0.076
Distance to previous	0.246/0.277	+0.016/+0.026
Relative token frequency	0.108/0.129	+0.005/+0.015
Token distribution	0.110/0.127	+0.002/+0.012
Current token	0.110/0.126	+0.006/+0.018
Tag context	0.093/0.102	-0.011/-0.000
Sentence length	0.077/0.102	-0.012/-0.001

optimally. To investigate, I ran additional experiments varying the system's use. I tried changing the parameter settings, changing feature selection, and switching the training data to the DUC 2002 extracts, gradually increasing the degree of variations. My experience and hindsight helped me make optimal choices regarding the test data, which benefited the results. Therefore, you should see these experiments as an investigation of the WPDV-XTR system's ceiling for this task.

**Parameter settings**

First I ran the system with different parameter settings, the parameters being the weight-base factor B used in WPDV feature combination and the decisiveness factor D. In DUC 2002, I set WPDV-XTR's parameters at 0.8 and 4, respectively—without thorough experimentation, however, because the competition's deadline was rapidly nearing. The new experiments showed that these settings weren't optimal, but they also weren't significantly worse than the observed optimal settings.

For the 200-word extracts, the F-values for the various settings ranged from 0.179 (S)

and 0.198 (W) to 0.205 (S) and 0.224 (W), none significantly worse or better than the participating system's 0.188 (S) and 0.211 (W). The highest values appeared to be found for lower weight bases and higher decisiveness factors, with 0.205 and 0.224 at B = 2, D = 7. This would indicate that, for the 200-word extraction task, individual clues are more important than combinations of clues (because low B did best), and you should take only the most decisive clues into account (because high D did best). A possible explanation for this is that the task at hand differs from the single-document extraction task. In the 200-word multidocument extraction task, it isn't so much the sentences in a document that are competing for a place in the extract but rather the high-scoring sentences from different documents. The new high values were also higher than the Position system's—0.198 (S) and 0.219 (W).

For the 400-word extracts, the F-values for the various settings ranged from 0.245 (S) and 0.272 (W) to 0.262 (S) and 0.294 (W), none significantly worse or better than the participating system's 0.258 (S) and 0.290 (W). Here, the highest scores were much less local-

ized. They were found both for lower B with D = 7 and for slightly higher B and D = 5. No setting produced a better score than Position's 0.269 (S) and 0.299 (W).

So for both extract sizes, the settings I selected for the DUC 2002 submission weren't optimal, but it was only for the 200-word extract that I could have outscored the positional baseline through optimizing settings.

**Feature selection**

Another thing you can vary is the feature cocktail's set of ingredients. I examined two variations:

- I ran the system with only a single feature in each case to measure the effectiveness of features by themselves.
- I left single features out of the feature cocktail and measured each feature's contribution as a final ingredient by subtracting the cocktail's performance without the feature from the cocktail's performance with the feature. A negative contribution means that removing the feature actually improved performance.

Tables 3 and 4 show the results from these experiments for 200-word extracts and 400-word extracts, respectively.

In both cases, sentence position was indispensable and the distance to previous occurrence was very useful. Certain features appeared to be a hindrance rather than an asset, although none of the variants performed significantly worse than the system I actually used in the contest. However, I measured only single removals. That a feature had no valuable contribution as a final ingredient doesn't mean it had no contribution at all. But it seems that you could remove sentence length (which, by itself, performed hardly above chance level—about 0.030/0.032) when creating 200-word extracts. On the other hand, sentence length's reduced value might be caused by technical aspects of WPDV-XTR, such as the summing over all tokens in each sentence; it actually might be useful in other systems.

Again, the role of the data type used here (newswire and newspaper text) greatly favors positional features. For another type of data or task, you should reappraise each feature's relative worth, as positional features might be less dominant.

**Training data**

The biggest experimental change was the

replacement of the training data I used in the competition (400-word single-document extracts) with the DUC 2002 extracts (200- and 400-word multidocument extracts). For each collection, I retrained the system on all other collections and created a new submission for the collection in question. This experiment used reasonably optimal conditions. The training data was the exact same type as the test data, as it was selected for the same purpose and given extracts by the same group of extractors. However, I carefully separated the training and test data, so the experiment was still fair.

I tried the system with this new data and without changing any of the parameters. However, because of computation time, I had to train the WPDV models for this experiment only on those features and combinations that occurred at least twice in the training data. As this generally leads to quality loss, the results should be slightly worse than when using a full model. But even these sub-optimal models performed strikingly well (see Table 5).

For both extract sizes, retraining raised performance above the positional baseline and the parameter setting and feature selection variants (although not significantly). My DUC 2002 submission's disappointing performance seems primarily caused by the discrepancy between the kind of training data and the intended task.

### Combined changes

Because retraining, changing feature selection, and optimizing parameters all led to improved results, I reasoned that combining the three could lead to even better improvement. As an example, I trained a system on the 200-word DUC 2002 extracts described earlier but without the frequency threshold and without features for sentence length and relative token frequency.

This system did well for both extract sizes, although at completely different parameter settings. For the 200-word extracts, the best settings were where I expected them by now—at high D and lower weights. But for the 400-word extracts, the lower D suddenly did well, and even high weights started producing results.

The new best score for 400-word extracts was 0.287 (S) and 0.317 (W), which wasn't a significant improvement and a mere 8 percent over the positional baseline, but the model was trained to produce 200-word extracts. For 200-word extracts, the new best score—0.245 (S)

Table 5. Results with new training data.

	F-values for 200-word extracts (S/W)	F-values for 400-word extracts (S/W)
WPDV-XTR	0.188/0.211	0.258/0.290
Position	0.191/0.215	0.265/0.294
Retrained on 200-word extracts	0.217/0.237	0.276/0.308
Retrained on 400-word extracts	0.207/0.230	0.277/0.309

and 0.265 (W)—was an impressive and statistically significant improvement, a full 28 percent (S) and 23 percent (W) better than the positional baseline.

**T**hese experiments show that WPDV-XTR, including features based on writing style recognition, can produce extracts that are on a par with, or better than, other state-of-the-art systems. The system can also perform significantly better than a positional baseline, but only if it is tuned to the task at hand. Factors in the tuning process are, in order of importance,

- Training data's appropriateness to the task
- Feature selection
- Parameter settings

The training data's appropriateness is easily the most important factor. This implies that researchers could further improve the machine learning approach to multidocument sentence extraction with even more specialized training data. The need for training data is an obvious weakness in the machine learning approach. However, understanding of summarization is not yet good enough to do without human summaries as examples. Also, enough document-abstract pairs seem to exist for our needs in principle—if only they were machine readable and available. What we lack mostly is human-made document-extract pairs, but you could approximate the extracts with an automatic mapping of an abstract to the sentences in the document containing the corresponding information.

The lack of overlap among extracts for the same competition collection by different human summarizers shows that each summarizer makes his or her own particular choices, possibly on the basis of his or her own interpretation of the competition's instructions (which didn't specify a goal or focus for the extracts). In other words, each

summarizer brings his or her own bias to the extraction task. The machine learning system's ability to overlap with the summarizers more than they overlap with each other might indicate that the system imitates the human summarizers' specific choices and biases. If true, when creating future training data, it's probably best to

- Give more exact instructions about the desired goal and focus of the extracts to prevent or at least reduce biases
- Limit the number of human summarizers, possibly even to one, to prevent biases or extract style differences other than goal- or focus-directed ones

Such training data should make mimicking the human extracts much easier for systems to learn than the current mix of biases. However, this approach only opts for a single bias; it would be better if we had greater understanding of biases' nature and could use them in creating different kinds of extracts.

Once researchers have selected their training material and task, they can optimize feature selection and parameter settings for their extraction system. My experiments show that you can't take much for granted here; optimal choices vary with task changes (for example, 200-word versus 400-word extracts) and features that you generally assume are useful might not be. Sentence length, for example, can prove less useful under some circumstances.

All in all, WPDV-XTR provides a sufficient basis for future research. However, such research will have another, much wider focus. I aimed the experiments in this article at investigating specific features and a specific type of measurement—precision and recall with regard to manual extracts. For actual applications, other criteria exist that must be optimized for. When using extracts as a basis for abstracts, recall is probably more important than precision, as subsequent transformations can further reduce abstract

The Author



**Hans van Halteren** is a researcher and lecturer in the Department of Language and Speech at the University of Nijmegen. His research interests are machine learning techniques and their integration with more linguistically oriented methods, language technology, computational linguistics, and corpus linguistics. He received his MS in mathematics and computer science and his PhD in corpus linguistics from the University of Nijmegen. Contact him at the Dept. of Language and Speech, Univ. of Nijmegen, PO Box 9103, NL-6500 HD Nijmegen, Netherlands; hvh@let.kun.nl.

length. When using extracts in a question-answering system, questions are available to give direction to the extracting process. Whatever the chosen task, researchers should complement a purely quantitative evaluation with more qualitative evaluations so that the system's most troublesome shortcomings can be tracked down and repaired. Furthermore, for such repairs, and for the larger task in which the extractor is embedded, researchers will probably have to look beyond the pure machine learning approach and include more linguistically motivated techniques. ■

**Acknowledgments**

I thank John Conroy and Mary Ellen Okurowski for providing the initial training data, and Wessel Kraaij for letting me use his positional baseline system.

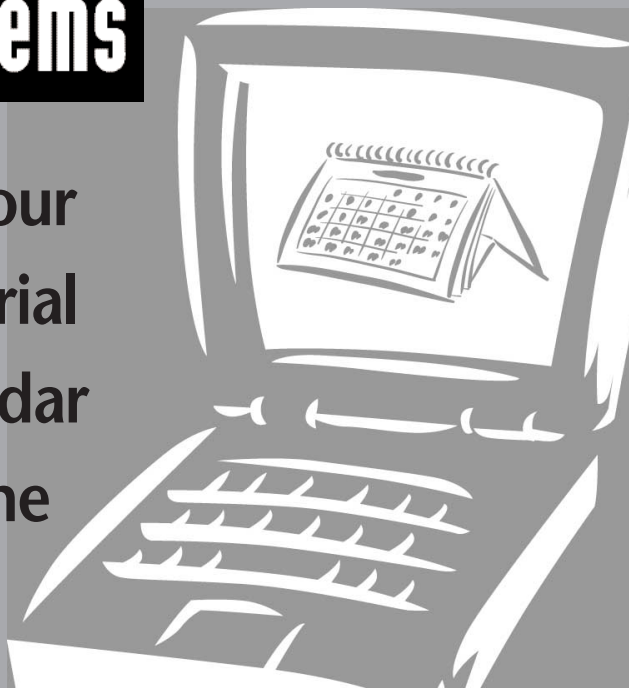
**References**

1. I. Mani and M.T. Maybury, eds., *Advances in Automatic Text Summarization*, MIT Press, 1999.
2. I. Mani, *Automatic Summarization*, John Benjamins Publishing, 2001.

3. D.R. Radev, E. Hovy, and K. McKeown, "Introduction to the Special Issue on Summarization," *Computational Linguistics*, vol. 28, no. 4, Dec. 2002, pp. 399–408.
4. D. Holmes, "Authorship Attribution," *Literary and Linguistic Computing*, vol. 13, no. 3, Sept. 1998, pp. 111–117.
5. J. Rudman, "The State of Authorship Attribution Studies: Some Problems and Solutions," *Computers and the Humanities*, vol. 31, no. 4, 1997–1998, pp. 351–365.
6. R.H. Baayen, H. van Halteren, and F. Tweedie, "Outside the Cave of Shadows: Using Syntactic Annotation to Enhance Authorship Attribution," *Literary and Linguistic Computing*, vol. 11, no. 3, Sept. 1996, pp. 121–132.
7. H. van Halteren, "The Detection of Inconsistency in Manually Tagged Text," *Proc. Workshop on Linguistically Interpreted Corpora 2000* (LINC 2000), 2000.
8. H. van Halteren, "A Default First Order Weight Determination Procedure for WPDV Models," *Proc. 5th Computational Natural Language Learning Workshop* (CoNLL 2001), 2001, pp. 119–122.
9. J.M. Conroy et al., "Using HMM and Logistic Regression to Generate Extract Summaries for DUC," *Proc. Document Understanding Conf. 2001* (DUC 01), Nat'l Inst. of Standards and Technology, 2001; [www-nlpir.nist.gov/projects/duc/pubs.html#2001](http://www-nlpir.nist.gov/projects/duc/pubs.html#2001).
10. H.P. Edmundson, "New Methods in Automatic Abstracting," *Comm. ACM*, vol. 16, no. 2, Apr. 1969, pp. 264–285.
11. W. Kraaij, M. Spitters, and M. van der Heijden, "Combining a Mixture Language Model and Naive Bayes for Multi-Document Summarization," *Proc. Document Understanding Conf. 2001* (DUC 01), Nat'l Inst. of Standards and Technology, 2001; [www-nlpir.nist.gov/projects/duc/pubs.html#2001](http://www-nlpir.nist.gov/projects/duc/pubs.html#2001).
12. S. Sekine and C. Nobata, "Sentence Extraction with Information Extraction Technique," *Proc. Document Understanding Conf. 2001* (DUC 01), Nat'l Inst. of Standards and Technology, 2001; [www-nlpir.nist.gov/projects/duc/pubs.html#2001](http://www-nlpir.nist.gov/projects/duc/pubs.html#2001).
13. C.J. van Rijsbergen, *Information Retrieval*, Butterworth, 1975.

IEEE  
**Intelligent  
Systems**

Visit our  
Editorial  
Calendar  
online



<http://computer.org/intelligent/edcal.htm>

For more information on this or any other computing topic, please visit our Digital Library at <http://computer.org/publications/dlib>.