AUTOMATED TEXT SUMMARIZATION AND THE SUMMARIST SYSTEM

Eduard Hovy and Chin-Yew Lin

Information Sciences Institute
of the University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292-6695
email: {hovy,cyl}@isi.edu
tel: 310-822-1511

Abstract

This paper consists of three parts: a preliminary typology of summaries in general; a description of the current and planned modules and performance of the SUMMARIST automated multilingual text summarization system being built sat ISI, and a discussion of three methods to evaluate summaries.

1. THE NATURE OF SUMMARIES

Early experimentation in the late 1950's and early 60's suggested that text summarization by computer was feasible though not straightforward (Luhn, 59; Edmundson, 68). The methods developed then were fairly unsophisticated, relying primarily on surface level phenomena such as sentence position and word frequency counts, and focused on producing extracts (passages selected from the text, reproduced verbatim) rather than abstracts (interpreted portions of the text, newly generated).

After a hiatus of some decades, the growing presence of large amounts of online text—in corpora and especially on the Web—renewed the interest in automated text summarization. During these intervening decades, progress in Natural Language Processing (NLP), coupled with great increases of computer memory and speed, made possible more sophisticated techniques, with very encouraging results. In the late 1990's, some relatively small research investments in the US (not more than 10 projects, including commercial efforts at Microsoft, Lexis-Nexis, Oracle, SRA, and TextWise, and university efforts at CMU, NMSU, UPenn, and USC/ISI) over three or four years have produced several systems that exhibit potential marketability, as well as several innovations that promise continued improvement. In addition, several recent workshops, a book collection, and several tutorials testify that automated text summarization has become a hot area.

However, when one takes a moment to study the various systems and to consider what has really been achieved, one cannot help being struck by their underlying similarity, by the narrowness of their focus, and by the large numbers of unknown factors that surround the problem. For example, what precisely is a summary? No-one seems to know exactly. In our work, we use *summary* as the generic term and define it as follows:

> A summary is a text that is produced out of one or more (possibly multimedia) texts, that contains (some of) the same information of the original text(s), and that is no longer than half of the original text(s).

To clarify the picture a little, we follow and extend (Spärck Jones, 97) by identifying the following aspects of variation. Any summary can be characterized by (at least) three major classes of characteristics:

Input: characteristics of the source text(s)

Source size: single-document *vs.* **multi-document**: A single-document summary derives from a single input text (though the summarization process itself may employ information compiled earlier from other texts). A multi-document summary is one text that covers the content of more than one input text, and is usually used only when the input texts are thematically related.

Specificity: domain-specific *vs.* **general**: When the input texts all pertain to a single domain, it may be appropriate to apply domain-specific summarization techniques, focus on specific content, and output specific formats, compared to the general case. A domain-specific summary derives from input text(s) whose theme(s) pertain to a single restricted domain. As such, it can assume less term ambiguity, idiosyncratic word and grammar usage, specialized formatting, etc., and can reflect them in the summary.

A general-domain summary derives from input text(s) in any domain, and can make no such assumptions.

Genre and scale: Typical input genres include newspaper articles, newspaper editorials or opinion pieces, novels, short stories, non-fiction books, progress reports, business reports, and so on. The scale may vary from book-length to paragraph-length. Different summarization techniques may apply to some genres and scales and not others.

Output: characteristics of the summary as a text

Derivation: Extract *vs.* **abstract**: An extract is a collection of passages (ranging from single words to whole paragraphs) extracted from the input text(s) and produced verbatim as the summary. An abstract is a newly generated text, produced from some computer-internal representation that results after analysis of the input.

Coherence: fluent *vs.* **disfluent**: A fluent summary is written in full, grammatical sentences, and the sentences are related and follow one another according to the rules of coherent discourse structure. A disfluent summary is fragmented, consisting of individual words or text portions that are either not composed into grammatical sentences or not composed into coherent paragraphs.

Partiality: neutral *vs.* **evaluative**: This characteristic applies principally when the input material is subject to opinion or bias. A neutral summary reflects the content of the input text(s), partial or impartial as it may be. An evaluative summary includes some of the system's own bias, whether explicitly (using statements of opinion) or implicitly (through inclusion of material with one bias and omission of material with another).

Conventionality: fixed *vs.* **floating**: A fixed-situation summary is created for a specific use, reader (or class of reader), and situation. As such, it can conform to appropriate in-house conventions of highlighting, formatting, and so on. A floating-situation summary cannot assume fixed conventions, but is created and displayed in a variety of settings to a variety of readers for a variety of purposes.

Purpose: characteristics of the summary usage

Audience: Generic *vs.* **query-oriented**: A generic summary provides the author's point of view of the input text(s), giving equal import to all major themes in it. A query-oriented (or user-oriented) summary favors specific themes or aspect(s) of the text, in response to a user's desire to learn about just those themes in particular. It may do so explicitly, by highlighting pertinent themes, or implicitly, by omitting themes that do not match the user's interests.

Usage: Indicative *vs.* **informative**: An indicative summary provides merely an indication of the principal subject matter or domain of the input text(s) without including its contents. After reading an informative summary, one can explain what the input text was about, but not necessarily what was contained in it. An informative summary reflects (some of) the content, and allows one to describe (parts of) what was in the input text.

Expansiveness: Background *vs.* **just-the-news**: A background summary assumes the reader's prior knowledge of the general setting of the input text(s) content is poor, and hence includes explanatory material, such as circumstances of place, time, and actors. A just-the-news summary contains just the new or principal themes, assuming that the reader knows enough background to interpret them in context.

At this time, apart from early work by Spärck Jones and students, such as (Tait and Spärck Jones, 83), we know of few linguistic or computational studies of these and other aspects of summaries; the work by (Van Dijk and Kintsch, 83) and (Endres-Niggemeyer, 97) focus on the psycholinguistic aspects of humans when they create summaries. We believe that the typology of summaries is a fruitful area for further study, both by linguists performing text analysis and by computational linguists trying to create techniques to create summaries conforming to one or more of the characteristics listed above. A better understanding of the types of summary will facilitate the construction of techniques and systems that better serve the various purposes of summarization in general.

Our own work is computational. Over the past two years, under the TIPSTER program, we have been developing the text summarization system SUMMARIST (Hovy and Lin, 98; Lin, 98). Our goal is to investigate the nature of text summarization, using SUMMARIST both as a research tool and as an engine to produce summaries for people upon demand.

In this paper, we first describe the architecture of SUMMARIST and provide details on the evaluated results of several of its modules in Sections 3, 4, and 5. Finally, since the evaluation of summaries (and of summarization) is a little-understood business, we describe some preliminary experiments in this regard in Section 6.

2. SUMMARIST

The goal of SUMMARIST is to create summaries of arbitrary text in English and selected other languages (Hovy and Lin, 98). By eschewing language-specific methods for the relatively surface-level processing, it is possible to create a multi-lingual summarizer fairly easily. Eventually, however, SUMMARIST will include language-specific techniques of parsing and semantic analysis, and will combine robust NLP processing (using Information Retrieval and statistical techniques) with symbolic world knowledge (embodied in the concept thesaurus SENSUS (Knight and Luk, 94; Hovy, 98), derived from WordNet (Miller et al., 90) and augmented by dictionaries and similar resources) to overcome the problems endemic to either approach alone. These problems arise because existing robust NLP methods tend to operate at the word level, and hence miss concept-level generalizations (which are provided by symbolic world knowledge), while on the other hand symbolic knowledge is too difficult to acquire in large enough scale to provide adequate coverage and robustness. For high-quality yet robust summarization, both aspects are needed.

In order to maintain functionality while we experiment with new aspects, and since not all kinds of summary require the same processing steps, we have adopted a very open, modular design. Since it is still under development, not all the modules of SUMMARIST are equally mature.

To create extracts, one needs procedures to identify the most important passages in the input text. To create abstracts, the core procedure is a process of interpretation. In this step, two or more topics are fused together to form a third, more succinctly stated, one. (We define *topic* as a particular subject that we write about or discuss.) This step must occur after the identification step. Finally, to produce the summary, a concluding step of sentence generation is needed. Thus SUMMARIST is structured according to the following 'equation':

summarization =
topic identification + *interpretation* + *generation*

For **identification**, the goal is to filter the input to retain only the most important, central, topics. Once they have been identified, they can simply be output, to form an extract. Typically, topic identification can be achieved using various complementary techniques. This stage of SUMMARIST is by far the most developed; making it, at present, an extract-only summarizer. See Section 3.

For **interpretation**, the goal is to perform compaction through re-interpreting and fusing the extracted topics into more succinct ones. This is necessary because abstracts are usually much shorter than their equivalent extracts. All the variations of fusion are yet to be discovered, but they include at least simple concept generalization (*he ate pears, apples, and bananas→ he ate fruit*) and script identification (*he sat down, read the menu, ordered, ate, and left→ he visited the restaurant*). We discuss interpretation in Section 4.

For **generation**, the goal is to produce the output summary. In the case of extracts, generation is a null step, but in the case of abstracts, the generator has to reformulate the extracted and fused material into a coherent, densely phrased, new text. The modules planned for SUMMARIST are described in Section 5.

Prior to topic identification, the system **preprocesses** the input text. This stage converts all inputs into a standard format we call SNF (Summarist Normal Form). Preprocessing includes tokenizing (to read non-English texts and output word-segmented tokens); part-of-speech tagging (the tagger is based on Brill's (1992) part-of-speech tagger); demorphing (to find root forms of each input token, using a modification of WordNet's (Miller et al., 90) demorphing program); phrasing (to find collocations and multi-word phrases, as recorded in WordNet); token frequency counting; *tf.idf* weight calculation (to calculate the *tf.idf* weight (Salton, 88) for each input token, and rank the tokens accordingly); and query relevance calculation (to record with each sentence the number of demorphed content words in the user's query that also appear in that sentence).

An example text in Indonesian, preprocessed into SNF, is shown in Figures 1(a) and 1(b). Figure 1(a) indicates that the text contained 1618 characters, and that it had been processed by the following modules: tokenization and part of speech tagging, title treatment, demorphing, WordNet categorization and common word identification, *tf.idf* computation, and OPP processing (see Section 3.1). It also records the top-scoring words in the text, together with their scores, as given by the modules computing term frequency (*tf_keywords*), *tf.idf*, and the OPP. The field *opp_rule* shows the most important sentence positions as 0 (the title); sentence 1; sentences 2 and 3 (tied), in that order. Figure 1(b) contains the processed text itself, one word per line, with the features added to each word by various modules. The features include paragraph and sentence number (*pno* and *sno*), part of speech (*pos*, empty for Indonesian), common word indicator (*cwd*), presence of word in title (*ttl*), morphology (*mph*), WordNet count (*wnc*), word frequency in text (*frq*), and *tf.idf* and OPP scores (see Sections 3.3 and 3.1 resp.).

3. Phase 1: Topic Identification

Summarization systems that perform topic identification only produce extract summaries; these include the current

operational version of SUMMARIST, as well as the systems of (Aone et al., 98; Strzalkowski et al., 98; Bagga and Baldwin, 98; and Mitra et al., 97).

```
<*topic=#???>
<*docno=HTML-DOC>
<*title="Senator Dari Demokrat Sarankan ClintonMemberi Kesaksian Di DepanKongres ">
<*token_title="sarankan|demokrat|clintonmemberi|kongres|kesaksian|senator">
<*pos_title="-">
<*char_count=2300>
<*module=PRE|TTL|MPH|CAT|TFIDF|OPP>
<*tf_keywords=kesaksian,6.000|kongres,6.000|lewinsky,5.000|demokrat,3.000|
 hadapan,3.000|video,3.000|'saya,2.000|agung,2.000|digambarkan,2.000|jawaban,2.000>
<*tfidf_keywords=lewinsky,35.088|kesaksian,32.448|kongres,27.718|video,16.894|
 demokrat,13.859|hadapan,12.219|digambarkan,11.838|senator,11.262|pembaruan,10.451>
<*opp_rule=p:0,1|1,2|2,4|3,4 s:-,->
<*opp_keywords=kongres,26.917|kesaksian,25.667|demokrat,16.333|lewinsky,14.167|
   senator,12.667|sarankan,10.667|video,9.333|screen,9.000|the,9.000|hadapan,8.917>
```

Figure 1(a). Indonesian text: preamble, after preprocessing.

```
Dari     <pno=2 sno=3 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Demokrat <pno=2 sno=3 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=3 tfidf=13.859 opp=16.333,3.667>
Sarankan <pno=2 sno=3 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=1 tfidf=5.631 opp=10.667,3.667>
Clinton <pno=2 sno=3 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Memberi <pno=3 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Kesaksian <pno=3 sno=1 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=6 tfidf=32.448 opp=25.667,3.250>
Di       <pno=3 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Depan    <pno=3 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Kongres <pno=3 sno=1 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=6 tfidf=27.718 opp=26.917,3.250>
Washington <pno=4 sno=1 pos=NA cwd=0 ttl=0 mph=- wnc=- frq=1 tfidf=3.434 opp=2.833,2.833>
,        <pno=4 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
Pembaruan <pno=4 sno=1 pos=NA cwd=0 ttl=0 mph=- wnc=- frq=2 tfidf=10.451 opp=6.500,2.833>
Seorang <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
senator <pno=5 sno=1 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=2 tfidf=11.262 opp=12.667,2.833>
dari     <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Partai   <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Demokrat <pno=5 sno=1 pos=NA cwd=0 ttl=1 mph=- wnc=- frq=3 tfidf=13.859 opp=16.333,2.833>
,        <pno=5 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
hari     <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Minggu   <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
(        <pno=5 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
20       <pno=5 sno=1 pos=CD cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
/        <pno=5 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
9        <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
)        <pno=5 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
,        <pno=5 sno=1 pos=PUN cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0 opp=-,->
menyarankan <pno=5 sno=1 pos=NA cwd=0 ttl=0 mph=- wnc=- frq=1 tfidf=3.759 opp=2.833,2.833>
agar     <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Presiden <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
Clinton <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
secara   <pno=5 sno=1 pos=NA cwd=1 ttl=0 mph=- wnc=- frq=0 tfidf=0.000 opp=-,->
…continued…
```

Figure 1(b). Indonesian text: words plus their attributes, after preprocessing.

We assume that a text can have many (sub)-topics, and that the topic extraction process can be parameterized in at least two ways: first, to include more or fewer topics to produce longer or shorter summaries, and second, to include only topics relating to the user's expressed interests.

Typically, topic identification can be achieved using various complementary techniques, including those based on stereotypical text structure, cue words, high-frequency indicator phrases, and discourse structure. Modules for all of these have been completed or are under construction for SUMMARIST. In processing, each module assigns a numeric score to each sentence. When all modules are done, the Topic Id Integration Module combines their scores to produce the overall ranking. The final result is the top-ranked $n\%$ of sentences as its final result, where n is specified by the user.

3.1 Position Module

The Position Module is based on the well-known fact that certain genres exhibit such structural and expositional regularity that one can reliably locate important sentences in certain fixed positions in the text. In early studies, Luhn (1959) and Edmundson (1968) identified several privileged positions, such as first and last sentences.

We generalized their results (Lin and Hovy, 97), developing a method for automatically identifying the sentence positions most likely to yield good summary sentences. The training phase of this method calculates the yield of each sentence position by comparing the similarity between human-created abstracts and the contents of the sentence in each ordinal position in the corresponding texts. By summing over a large collection of text-abstract pairs from the same corpus and appropriately normalizing, we create the *Optimal Position Policy* (OPP), a ranked list that indicates in what ordinal positions in the text the high-topic-bearing sentences tend to occur. We tested this method on two corpora: the Ziff-Davis texts (13,000 newspaper articles announcing computer products) and a set of several thousand *Wall Street Journal* newspaper articles. For the Ziff-Davis corpus we found the OPP to be

[T1, P2S1, P3S1, P4S1, P1S1, P2S2,
{P3S2, P4S2, P5S1, P1S2}, P6S1, ...]

i.e., the title (T1) is the most likely to bear topics, followed by the first sentence of paragraph 2, the first sentence of paragraph 3, etc. (Paragraph 1 is invariably a teaser sentence in this corpus.) In contrast, for the *Wall Street Journal,* we found the OPP to be

[T1, P1S1, P1S2, ...]

We evaluated the OPP method in various ways. One measured *coverage*, the fraction of the (human-supplied) keywords that are included verbatim in the sentences selected under the policy. (A random selection policy would extract sentences with a random distribution of topics; a good position policy would extract rich topic-bearing sentences.) We measured the effectiveness of an OPP by taking cumulatively more of its sentences: first just the title, then the title plus P2S1, and so on. Summing together the multi-word contributions in the top ten sentence positions, 10-sentence extracts (approx. 15% of a typical Ziff-Davis text) intersected with 95% of the corresponding human abstracts.

In addition to the OPP itself, we created OPP keywords, by counting the number of times each open-class word appeared in an OPP-selected sentence, and sorting them by frequency. Any other sentence with a high number of these keywords can also be rewarded with an appropriate score.

In operation, the Position Module simply selects an appropriate OPP for the input text, assigns a score to each sentence in order of the OPP, and then computes the OPP keyword list for the text. It then assigns additional scores to sentences according to how many OPP keywords they contain. These scores can be seen in Figure 1(b), in the item *opp=x,y* on each line. The first number provides the global OPP score (the score of this word, summed over the whole text) and the second score the local OPP (the score of this sentence in the OPP).

3.2 Cue Phrase Module

In pioneering work, Baxendale (1958) identified two sets of phrases—*bonus* phrases and *stigma* phrases—that tend to signal when a sentence is a likely candidate for inclusion in a summary and when it is definitely not a candidate, respectively. Bonus phrases such as "in summary", "in conclusion", and superlatives such as "the best", "the most important" can be good indicators of important content. During processing, the Cue Phrase Module simply rewards each sentence containing a cue phrase with an appropriate score (constant per cue phrase) and penalizes those containing stigma phrases.

Unfortunately, cue phrases are genre dependent. For example, "Abstract" and "in conclusion" are more likely to occur in scientific literature than in newspaper articles. Given this genre-dependence, the major problem with cue phrases is identifying them. A natural method is to identify high-yield sentences in texts (compared to their human-made abstracts) and then to identify common phrases in those sentences. A careful study on the automated collection of cue phrases is reported in (Teufel and Moens, 98).

In the context of SUMMARIST, we have tried several methods of acquiring cue phrases. In one experiment, we manually compiled a list of cue phrases from a training corpus of paragraphs that themselves were summaries of texts. In this corpus, sentences containing phrases such as "this paper", "this article", "this document", and "we conclude" fairly reliably reflected the major content of the paragraphs. This indicated to us the possibility of summarizing a summary.

In another experiment, we examined methods to automatically generate cue phrases (Liu and Hovy, in prep.). We examined various counting methods, all of them comparing the ratios of occurrence densities of words in summaries and in the corresponding texts in various ways, and then extracted the words showing the highest increase in occurrence density between text and associated abstract. Finally, we searched for frequent concatenations of such privileged words into phrases. While we found no useful phrases in a corpus of 1,000 newspaper articles, we found the following in 87 articles on Computational Linguistics:

Method 1		**Method 2**	
S1	*phrase*	*S2*	*phrase*
11.50	multiling. natural lang.	3.304	in this paper
8.500	paper presents the	2.907	this paper we
7.500	paper gives	2.723	base on the
6.000	paper presents	2.221	a set of
6.000	now present	2.192	the result of
5.199	this paper presents	2.000	the number of
4.555	paper describes	1.896	in order to

In method 1, $S1 = wc_s$, the total number of words occurring in the summary that co-occur with the word w in any sentence, normalized by the total number of words. In method 2, $S2 = wc_s * df / D$, where D is the total number of training documents and df is the number of documents in which the word being counted appears.

The Cue Phrase Module was not applied in the example in Figure 1(b), since we have not trained cue phrases for Indonesian.

3.3 Topic Signature Module

In a straightforward application of word counting, one might surmise that words that occur most frequently in the text may possibly indicate important material. Naturally, one has to rule out closed-class words such as "the" and "in". A common method is to create a list of words using *tf.idf*, a measure that rewards words for being *relatively* frequent—much more frequent in one text than on average, across the corpus. This method, pioneered a decade ago (Salton, 88), is used in IR systems to achieve query term expansion.

The same idea can be used in topic identification. On the assumption that semantically related words tend to co-occur, one can construct *word families* and then count the frequency of word families instead of individual words. A frequent word family will indicate the importance of its common semantic notion(s) in the text. To implement this idea, we define a *Topic Signature* as a topic word (the *head*) together with a list of associated (*keyword weight*) pairs. Each topic signature represents a semantic concept using word co-occurrence patterns. We describe in Section 4.2 how we automatically build Topic Signatures and plan to use them for topic interpretation.

For use in topic identification, we created a Topic Signature for each of five groups of 200 documents, drawn from five domains. When performing topic identification for a document, the Topic Id Signature Module scanned each sentence, assigning to each word that occurred in a signature the weight of that keyword in the signature. Each sentence then received a signature score equal to the total of all signature word scores it contained, normalized by sentence length. This score indicated the relevance of the sentence to the signature topic.

Since we have no signatures for Indonesian word families, no signature score appears in Figure 1(b). However, the *tf.idf* score of each word (comparing the frequencies of each term in the text and across a collection of Indonesian texts) appears in the item *tfidf=x*.

3.4 Discourse Structure Module

A new module that uses discourse structure is under construction for SUMMARIST. This module, being built by Daniel Marcu, is an extension of his Ph.D. work (Marcu, 97). It is based on the fact that texts are not simply flat lists of sentences; they have a hierarchical structure, one in which certain clauses are more important than others. After parsing the hierarchical structure of an input text and then identifying the important clauses in this structure, Marcu discards unimportant clauses and retains only the most important ones, still bound together within the discourse structure, and hence still forming a coherent text.

To produce the text's discourse structure, Marcu adapted Rhetorical Structure Theory (Mann and Thompson, 88), which postulates approximately 25 relations that bind clauses (or groups of clauses) together if they exhibit certain semantic and pragmatic properties. These relations are signaled by so-called cue phrases such as "*but*" and "*however*" (for the relation Contrast), "*in order to*" and "*because*" (for the relation Cause), "*then*"

and "*next*" (for Sequence), and so on. Most relations are binary, having a principal component (the Nucleus) and a subsidiary one (the Satellite). Relations can be nested recursively; a text is only coherent if all its clauses can be linked together, first in local subtrees and then in progressively larger ones, under a single overarching relation. Marcu uses a constraint satisfaction algorithm to assemble all the trees that legally organize the input text, and then employs several heuristics to prefer one tree over the others.

To produce an extract summary of the input text, Marcu simply discards the least salient material, in order, by traversing the discourse structure top-down, following Satellite links only.

Marcu's subsequent this work combines the discourse structure paradigm with several other surface-based methods, including cue phrases, the discourse tree shape, title words, position, and so on (Marcu, 98a). Using an automated coefficient learning method, he finds that the best linear combination of values for all these methods. Evaluation shows that the resulting extracts approximate human performance on both newspaper articles and *Scientific American* texts:

Extract	**F-score**	
10% (clauses)	**news**	**Sci. Am.**
Human	79.41%	71.27%
System	68.86%	70.42%

3.5 Topic Identification Integration Module

After SUMMARIST applies some or all the above modules, each sentence has been assigned several different scores. Some method is required to combine these scores into a single score, so that the most important topic-bearing sentence can be ranked first. However, it is not immediately clear how the various scores should be combined for the best result. Various approaches have been described in the literature. Most of them employ some sort of combination function, in which coefficients assign various weights to the individual scores, which are then summed. (Kupiec et al., 95) and (Aone et al., 97) employ the Expectation Maximization algorithm to derive coefficients for their systems.

Initially, we implemented for SUMMARIST a straightforward linear combination function, in which we specified the coefficients manually, by experimentation. This hand tuning method is good for getting a feeling of how various modules can affect the SUMMARIST output, but it does not guarantee consistent performance over a large collection. As we found in the formal TIPSTER-SUMMAC evaluation of various summarization systems (Firmin Hand and Sundheim, 98), the results of this function were decidedly non-optimal, and did not show the potential and power of the system! Since consistent performance and graceful degradation are very important for SUMMARIST, alternative combination functions were needed.

In subsequent work, we tested two automated methods of creating better combination functions. These methods assumed that a set of texts with their ideal summaries are available, and that information contained in the ideal summaries can be reliably recovered from their corresponding original texts. The sentences in the original texts that were also included in the summaries we called *target sentences.*

Unfortunately, we know of no large corpus of texts with truly ideal summaries. Therefore, as training data, we used a portion of the results of the TIPSTER-SUMMAC summarization evaluation dry run, annotated to indicate the relevance and popularity of each sentence, as aggregated over the ratings of several systems (Baldwin, 98). This collection contains 403 summaries containing 4,830 training instances/sentences, which are judged as relevant to TREC topics 110, 132, 138, 141, and 151. (Note that contributions from topics are not uniform. Specific (topic/contribution) numbers are 110/226, 132/122, 138/50, 141/42, and 151/63.) Since the sentence scores are based on the consensus votes of the summaries resulting from six different experimental summarization systems participating in the dry run, no single sentence relevance judgement is available to construct a truly ideal summary set. Thus the consensus selected target sentences should be called 'pseudo-ideal sentences'. Please refer to (Firmin Hand and Sundheim, 98) for a more detail description of the TIPSTER-SUMMAC dry run evaluation procedure and setup.

We then employed two methods to automatically learn the combination function(s) that identified in each training text the most target sentences.

The first method is a decision tree learning algorithm based on C4.5 (Quinlan, 86). Each module's outcome is used as a feature in the learning space. The normalized score (from 0 to 1 inclusive) of each module for each sentence is used as its feature value. A feature vector is a six-tuple: (*TTL*: v1, *TF*: v2, *TFIDF*: v3, *SIG*: v4, *OPP*: v5, *QRY*: v6). *TTL* indicates the score from the title module; *TF*, the term frequency module; *TFIDF*, the *tf.idf* module; *SIG*, the topic signature module; *OPP*, the position module; and *QRY*, the query signature module. All sentences included in the ideal summary of a text are positive examples, all others are negative examples.

To fully utilize limited training data, we followed the standard decision tree training and validation procedure

and conducted a 5-way cross-validation test. The algorithm generated a tree of 1,611 nodes, of which the top (most informative) questions pertain to the query signature, term frequency, overlap with title, and OPP. Compared with the manually built function, the decision tree is considerably better. With the linear combination function, SUMMARIST used to score 33.02% (Recall and Precision) on an unseen test set of 82 dry-run texts. On the same data, SUMMARIST now scores 58.07% (Recall and Precision) in the 5-way cross-validation test. This represents an improvement of 25%. It is important to understand that this 58.07% score should not be interpreted as how frequently SUMMARIST produces and recovers relevant summaries. This figure is obtained by evaluating against every sentence contained in the pseudo-ideal summaries. Thus it is simply a measure of how well SUMMARIST correctly recovers the pseudo-ideal sentences. The final judgement has to be made by human analysts who judge the sentences extracted by SUMMARIST as a whole, a judgement that unfortunately we cannot carry out on a large scale with our limited staff. However, the figure is still a valuable performance indicator. No summaries can be called good summaries if they do not contain any summary-worthy sentences.

For the second method, we followed the same setup as the decision tree training method mentioned above but implemented a 6-node perceptron as the learning engine. Training it on the same data produced results within 1% of the decision tree.

When measuring overall summarization performance, we conjecture that the performance of SUMMARIST with the decision tree, tested in a relevance judgement setting such as the TIPSTER-SUMMAC evaluation, should lie somewhere in the 70% range. As explained above, using the decision tree, SUMMARIST's performance increased by 25% (= 57% of its initial score). Adding this improvement to its score of 49% in the Ad Hoc normalized best summary category (Firmin Hand and Sundheim, 98) places it with that range. We look forward to new evaluations like SUMMAC.

We have recently trained a new decision tree, using different data. The new training data derives from the Question and Answer summary evaluation data provided by TIPSTER-SUMMAC. The principal difference between the (Baldwin, 98) data and the new Q&A data is that the latter contains essential text fragments (phrases, clauses, and sentences) which must be included in summaries to answer some TREC topics. These fragments are judged by two human judges and are thus much more accurate training data. SUMMARIST trained on the Q&A data should therefore perform better than the version trained on the older data.

An example extract summary of the Indonesian text in Figure 1, using the latest decision tree as combination function, appears in Figure 2. A detailed description of the aforementioned training experiments and the improved combination function appears in (Lin, in prep.).

```
<DOC>
 <SUMMARIZER>ISI</SUMMARIZER>
 <TASKTYPE>qanda</TASKTYPE>
 <SUMMARYTYPE>15%</SUMMARYTYPE>
 <QNUM> ???</QNUM>
 <DOCNO>HTML-DOC</DOCNO>
 <TITLE> Senator Dari Demokrat Sarankan Clinton Memberi Kesaksian Di DepanKongres
 </TITLE>
 <TEXT>
Seorang senator dari Partai Demokrat , hari Minggu ( 20 / 9 ) , menyarankan agar
Presiden  Clinton secara sukarela memberikan kesaksian di hadapan Kongres guna
menghentikan " siksaan politik " yang memanas Senin pagi ketika rakyat AS
menyaksikan testimoni Clinton di hadapan juri agung menyangkut Monica Lewinsky .
Senator John Kerry , anggota Demokrat dari Massachusetts mengusulkan Clinton
bersaksi menjawab pertanyaan Komite Hukum Kongres . Dengan menyiarkan video
kesaksian Clinton , Kongres sebetulnya menghadapi pukulan balik ..
 </TEXT>
 </DOC>
```

Figure 2. Summary of Indonesian text in Figure 1.

4. Phase 2: Topic Interpretation

The second phase of summarization—going from extract to abstract—is considerably more complex than the first, for it requires that the system have recourse to world knowledge. Without knowledge of the world, no system can fuse together the topics extracted to produce a smaller number of topics to form an abstract. Yet one cannot simply ignore abstraction: extracts are not adequate for many tasks. In one study, (Marcu, 98b) counted how many clauses had to be extracted from a text in order to fully contain all the material included in a human abstract of that text. Working with a newspaper corpus of 10 texts and 14 judges, he found a compression factor of 2.76—in that genre, extracts are almost three times as long (counting words) as their corresponding abstracts! Results of this kind indicate the need for summarization systems to further process extracted material: to remove redundancies, rephrase sentences to pack material more densely, and, importantly, to merge or fuse related topics into more 'general' ones using world knowledge.

The major problem encountered in the abstraction process is the acquisition of such world knowledge. In this section we describe two experiments performed in the context of SUMMARIST that investigate topic interpretation.

4.1. Concept Counting and the Wavefront

One of the most straightforward examples of topic fusion is concept generalization:

John bought some apples, pears, and oranges.
→ *John bought some fruit.*

Using a concept generalization taxonomy called WordNet (Miller et al., 90), we have developed a method to recognize that *apple, pear*, etc., can be summarized as *fruit*. The idea is simple. We first identify the WordNet equivalent of each topic word in the text, and then locate an appropriate generalization concept.

To identify an appropriate generalization concept, we need to count frequencies of occurrence of concepts (after all, *apple* and *pear* can equally well be generalized as *plant-product* or *physical-object*). Our algorithm (Lin 95) first counts the number of occurrences of each content word in the text, and assigns that number to the word's associated concept in WordNet. It then propagates all these weights upward, assigning to each node the sum of its weight plus all its childrens' weights. Next, it proceeds back down, deciding at each node whether to stop or to continue downward. The algorithm stops when the node is an appropriate generalization of its children; that is, when its weight derives so equally from two or more of its children that no child is the clear majority contributor to its weight. To find the wavefront, we define a concept's *weight* to be the sum of the frequency of occurrence of the concept C plus the weights of all its subconcepts. We then define the *concept frequency ratio* between a concept and its subconcepts:

$$R = \frac{MAX(\text{sum of all children of } C)}{SUM(\text{sum of all children of } C)}$$

This criterion selects the most specific generalization of a set of concepts as their fuser.

This algorithm can be extended to identify successive layers of fuser concepts. As described in (Lin 95), the first set of fuser concepts the algorithm locates need not be the last; one can continue downward, in order to locate more specific generalizations. Each stopping frontier we call an *interesting wavefront*. By repeating the wavefront location process until it reaches the leaf concepts of the hierarchy, the algorithm derives a set of interesting wavefronts. From all the interesting wavefronts, one can choose the most general one below a certain depth D to ensure a good balance of generality and specificity. For WordNet, we found $D=6$, by experimentation.

To evaluate the results of this type of fusion, we selected 26 articles about new computer products from *BusinessWeek* (1993–94) of average 750 words each. For each text we extracted the eight sentences containing the most interesting concepts using the wavefront technique, and compared them to the contents of a professional's abstracts of these 26 texts from an online service. We developed several weighting and scoring variations and tried various ratio and depth parameter settings for the algorithm. We also implemented a random sentence selection algorithm as a baseline comparison.

The results were promising, though not startling. Average recall (R) and precision (P) values over the three scoring variations were $R=0.32$ and $P=0.35$, when the system produces extracts of 8 sentences. In comparison, the random selection method scored $R=0.18$ and $P=0.22$ in the same experimental setting. These values show that semantic knowledge can help enable improvements over traditional IR word-based techniques. However, the limitations of WordNet are serious drawbacks: it contains almost no domain-specific knowledge.

4.2 Interpretation using Topic Signatures

Before addressing the problem of world knowledge acquisition head-on, we decided to investigate what type of knowledge would be useful for topic interpretation. After all, one can spend a lifetime acquiring knowledge in

just a small domain. How little knowledge does one need to enable effective concept fusion?

Our idea, again, was simple. We would collect a set of words that were typically associated with a target word, and then, during interpretation, replace the occurrence of the related words by the target word. For example, we would replace joint instances of *table, menu, waiter, order, eat, pay, tip*, and so on, by the single phrase *restaurant-visit*, in producing an indicative summary. We thus defined a *topic signature* as a family of related words, as follows:

$$TS = \{head, (w_1\ s_1), (w_2\ s_2), (w_3\ s_3), \ldots\}$$

where *head* is the target word and each w_i is an related word with association strength s_i.

As described in (Lin 97), we constructed signatures automatically from a set of 30,000 texts from the 1987 *Wall Street Journal* (WSJ) corpus. The paper's editors have classified each text into one of 32 classes. Within the texts of each class, we counted the occurrences of each content word (demorphed to remove plurals, etc.), relative to the number of times they occur in the whole corpus, using the standard *tf.idf* method. We then selected the top-scoring 300 terms for each category and created a signature with the category name as its head. The top terms of four example signatures are shown in Figure 3. It is quite easy to determine the identity of the signature head just by inspecting the top few signature words.

In order to evaluate the quality of the signatures formed by the algorithm, we evaluated the effectiveness of each signature by seeing how well it served as a selection criterion on texts. While this is not our intended use of signatures, document categorization is a well-known task with enough results in the literature to give us a sense of the performance of our methods. As data we used a set of 2,204 previously unseen WSJ news articles from 1988. For each test text, we created a single-text 'document signature', again using *tf.idf*, and then matched this document signature against the category signatures. The closest match provided the class into which the text was categorized. We tested several matching functions, including a simple *binary* match (count 1 if a term match occurs; 0 otherwise); *curve-fit* match (minimize the difference in occurrence frequency of each term between document and concept signatures), and *cosine* match (minimize the cosine angle in the hyperspace formed when each signature is viewed as a vector and each word frequency specifies the distance along the dimension for that word). These matching functions all provided approximately the same results. The values for Recall and Precision (R=0.7566 and P=0.6931) are encouraging and compare well with recent IR results (TREC, 95). Current experiments are investigating the use of contexts smaller than a full text to create more accurate signatures.

RANK	Aerospace	Banking	Environment	Telecomm.
1	contract	bank	epa	at&t
2	air_force	thrift	waste	network
3	aircraft	banking	environmental	fcc
4	navy	loan	water	cbs
5	army	mr.	ozone	cable
6	space	deposit	state	bell
7	missile	board	incinerator	long-distance
8	equipment	fslic	agency	telephone
9	mcdonnell	fed	clean	telecomm.
10	northrop	institution	landfill	mci
11	nasa	federal	hazardous	mr.
12	pentagon	fdic	acid_rain	doctrine
13	defense	volcker	standard	service
14	receive	henkel	federal	news
15	boeing	banker	lake	turner

Figure 3. Portions of the topic signatures of several concepts.

These results are encouraging enough to allow us to continue with topic signatures as the vehicle for a first approximation to world knowledge, as useful for topic interpretation. Considerable subsequent experimentation (Hovy and Junk, in prep.) with a variety of methods, including Latent Semantic Analysis, pairwise signatures, etc., indicates that the most promising method of creating signatures is χ^2. We are now busy creating a large number of signatures in an attempt to overcome the world knowledge acquisition problem.

5. Phase 3: Summary Generation

We have devoted no effort to adapting an existing language generator or developing a new one for SUMMARIST. It has scarcely been necessary, since SUMMARIST is principally an extract-only system at present. However, we envisage the language generation needs of summarization systems in general, and SUMMARIST in particular, to require two major steps.

The microplanner: The task of a microplanner, in general, is to convert a discourse-level specification of a sequence of topics into a list of specifications at the level of one or a few clauses at a time. This involves making choices for several semi-independent aspects, including sentence length, internal sentence organization (order of preposition phrases, active or passive mood, etc.), identification of the principal theme and focus units, selection of the main verb and other important words, and so on. In the context of summarization, the microplanner's task is to ensure that the information selected by the topic identification module and (possibly) fused by the topic interpretation module is phrased compactly and as briefly as possible though still in a grammatical sentence.

The microplanner can be built to perform its work at two levels: the *textual* level, in which its input is a list of sentences or sentence fragments, and its output is a compacted list of sentences, and the *representational* level, in which its input is couched in an abstract notation (whether more or less explicitly syntactic depends on the implementation), and its output is a fairly syntactic abstract specification of each sentence. In the former case, the output is more or less directly readable by a human, while in the latter, the output has to be converted into grammatical sentences by the sentence generator.

Microplanning is an area still largely unexplored by computational linguists. The work that has been done, including some of the major systems (Nirenburg et al., 89; Rambow and Korelsky, 92; Hovy and Wanner, 96), is not really appropriate to the specific compaction-related needs of summarization. The most relevant study on microplanning for summarization is (McKeown and Radev, 95).

The sentence generator: The task of a sentence generator (often called realizer) is to convert a fairly detailed specification of one or a few clause-sized units into a grammatical sentence. A number of relatively easy to use sentence generators is available to the research community, including Penman (Penman, 88), FUF/SURGE (Elhadad, 92), RealPro (Lavoie and Rambow, 97), and NITROGEN (Langkilde and Knight, 98). We plan to employ one or more of these in SUMMARIST.

6. Summary Evaluation

6.1 Two Basic Measures

How can you evaluate the quality of a summary? We have found no literature on this fascinating question. Indeed, many anecdotes and experiences lead one to believe that the uses of summaries are so task-specific and user-oriented that no objective measurement is possible. When the inter-judge scoring variability is higher than half the average score, as small tests relating to summaries occasionally have suggested, then perhaps there is no hope.

However, it is possible to develop some general guidelines and approaches, and from them to develop some approximations to summarization evaluation. We give a very rough sketch here of some work performed in the context of SUMMARIST in early 1997; more details are in (Hovy, in prep.).

It is obvious that to be a summary, the summary must obey two requirements:

- it must be shorter than the original input text;
- it must contain (some of) the same information as the original, and not other, new, information.

One can then define two measures to capture the extent to which a summary *S* conforms to these requirements with regard to a text *T*:

Compression Ratio:

CR = (length S) / (length T)

Retention Ratio:

RR = (info in S) / (info in T)

However we choose to measure the length and the information content, we can say that a good summary is one in which *CR* is small (tending to zero) while *RR* is large (tending to unity). We can characterize summarization systems and/or text types by plotting the ratios of the summaries produced under varying conditions. For example, Figure 4(a) shows a fairly normal growth curve: as the summary gets longer (grows along the *x* axis, which measures *CR*), it contains more information (grows also along the *y* axis, which measures *RR*), until it is just as long at the original text and contains the same information. In contrast, Figure 4(b) shows a curve with a very desirable bend: at some special point, the addition of just a little more material to the summary adds a disproportionately large amount more of information. Figure 4(c) shows another desirable behavior: initially, all the important material is included in the summary; as it grows, the new material is less interesting.

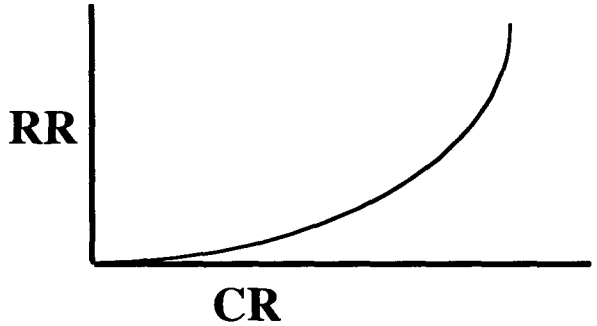

Figure 4(a). Compression Ratio (*CR*) vs. Retention Ratio (*RR*) for normal summary growth.

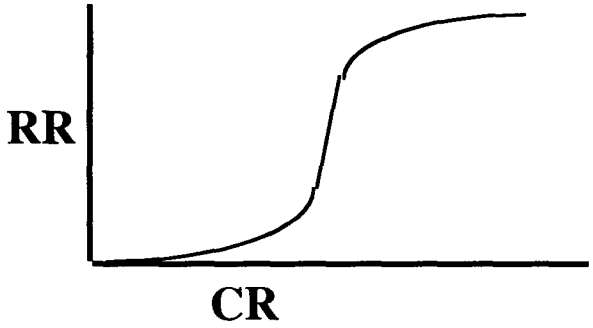

Figure 4(b). Compression Ratio (*CR*) vs. Retention Ratio (*RR*) for desirable summary growth.

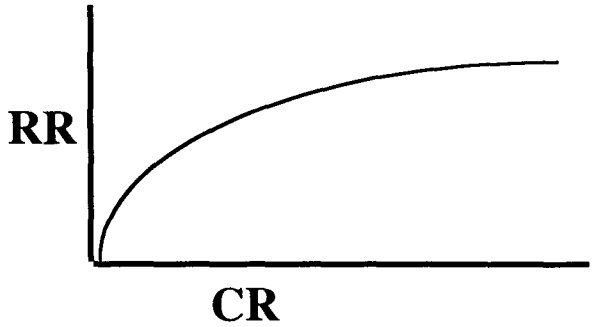

Figure 4(c). Compression Ratio (*CR*) vs. Retention Ratio (*RR*) for desirable summary growth.

Measuring length: Measuring length is relatively straightforward; one can choose as metric the number of words, of letters, of sentences, and so on. For a given genre and register level, there is a fairly fixed correlation between these metrics, in most cases.

Measuring information content: Ideally, one wants to measure not information content, but *interesting* information content only. The problem is that it is very hard to measure information content and almost impossible to define even what *interesting* information content may be. However, it is possible to approximate measures of information content in several ways. We describe three here.

The Shannon Game: This measure derives from a variant of a parlor game invented by Claude Shannon, when he was developing Information Theory (Shannon, 51). In this theory, the amount of information contained in a message is measured by $-p \log p$, where p is, roughly speaking, the probability of the reader guessing the message (or each piece thereof, individually). To test his theory, Shannon asked his wife and several helpers to receive and write down a message, by guessing one letter at a time, and being answered simply yes or no. Naturally, their knowledge of the typical letter frequencies in English (*e* being most frequent, then *t, a, o, i, n*, and so forth), coupled with their knowledge of English words and word endings (after *satisfac-* the chances for *–tion* are very high) helped them make informed guesses. And as soon as they started to make out the meaning of the transmitted message, their guesses got even better. Shannon's argument went as follows: the more the receiver could guess the message without any help, the less novel (and hence the less informative) it was to the receiver; and contrarily, the more guesses the receiver needed, the more informative the message was.

One can use this technique to measure the information content of a summary S relative to that of its corresponding text T as follows. Assemble three sets of testers. One set first reads T, slowly, and must then try to re-create it, letter by letter, without any longer seeing T. When they guess a wrong letter, they must guess again; when they guess a correct one, they simply proceed to the next letter. The number of wrong guesses g_{wrong} and the total number of guesses g_{total} they make are recorded, and their score is computed as the ratio $R_T = g_{wrong} \; / \; g_{total}$. The second set of testers first reads S, slowly, and then, without any further recourse to S, proceeds to try to create T, letter by letter. They know that S is only a summary of their goal. As with the first set, the number of guesses, wrong and total, are recorded, and the ratio R_S is computed. The third set of testers reads nothing first, but starts immediately to create T, without knowing what it might be about. Also for them, the ratio R_N is computed.

The quality of S can be computed by comparing the three ratios, where R_N and R_T define the end points of a scale; the former quantifies how much a tester could guess from default world knowledge (and should hence not be attributed to the summary), and the latter quantifies how much a tester still has to guess, even with 'perfect' prior knowledge. The closer R_S is to R_T, the better the summary.

Section 6.2 describes a small experiment.

The Question Game: This measure approximates the information content of S by determining how many questions drawn up about T can be answered. Before starting, one or more people create a set of questions based on what they consider the principal content of T. These questions can be neutral (author's point of view) or query-oriented, as required. Then the testers are brought in, and asked to answer these questions three times in succession. In the first round, the testers are simply asked the questions without having read either S or T, and their answers are scored. In the second round, they are given S to read, and are asked to answer the same questions again. In the third round, they are given T, and asked to answer the questions a third time. After each round, the number of questions answered correctly is tallied.

The quality of S can be computed by comparing the three tallies, where the first and last tallies provide baselines—respectively, how many questions the testers would be able to answer from default world knowledge, and how many they would not be able to answer even when given the whole text T. The closer the testers' answer tally for the summary to the tally the full text, the better the summary.

The formal SUMMAC summarization evaluation (Firmin Hand and Sundheim, 98) contained a pilot test of the Question Game. A small experiment is described in Section 6.2.

The Classification Game: This measure approximates information content by testing how well people can perform a classification task on a collection of summaries S_i and on full texts T_i. A set of texts is drawn from a few different topics, several texts per topic. For each text, a summary is created. Some testers are then asked to classify either a text or its corresponding summary (but not both) into one of the topic classes; other testers are given the converse sets (a summary or its corresponding text) to classify. After classification, the correspondence between the classifications of full texts and their corresponding summaries is measured; the greater the agreement, the better the summary is at capturing that which causes the full text to be classified as it is.

Many variants of this game are possible. One may ask the testers to classify all summaries first and then all the full texts, as long as one takes care to test for prior classification distortion. Or, instead of asking testers to classify texts, one may ask them to rate the summaries and texts for relevance to the topic classes. In this case, the game reduces to classification into two piles: Relevant and Not Relevant.

The formal SUMMAC summarization evaluation (Firmin Hand and Sundheim, 98) contained tests for two variants of the Classification Game.

6.2 Some Evaluation Experiments

Experiment 1: In a small pilot test at ISI, we performed a test of the Shannon Game. We manually created two paragraph-length summaries of two 300-word newspaper articles for the *LA Times*, and had 3 groups of graduate students (native or near-native speakers of English) recreate the full text, as outlined above. The results were astounding: the students who had seen the full text required on average 11 additional letter guesses (i.e., they made on average only 11 errors); those who had seen the summaries required approximately 150 additional guesses each; and those who had seen no text at all required over 1,100 guesses each and in some cases did not complete the test, after 3 hours! It is seldom that one finds a phenomenon yielding an order of magnitude difference between contrast sets; we were quite pleased at the result.

Experiment 2: Taking advantage of the presence of a larger number of enthusiastic and able participants at the 1997 *AAAI Fall Symposium on Text Summarization*, we did a second evaluation study, focusing on the following questions:

1. How do different evaluation methods compare for each type of summary?
2. How do different summary types fare under different evaluation methods?
3. How much does the evaluator affect scores?
4. Is there a preferred evaluation method?

Preparation: We selected two newspaper articles, one about an art theft and one about a housing development, and created the following types of summary for each domain:

- Human abstract, full background
- Human abstract, just-the-news
- SUMMARIST extract
- SUMMARIST keywords-only
- Random sentence extract

We then created the materials for three evaluations: a Shannon Game, a Question Game, and a Classification Game. The Shannon and Question Games followed quite closely the setup outlined above. The Classification Game asked for an initial classification into some very distinct possible categories and then a subsequent one into some much closer categories.

Execution: At the Symposium, we had the participants double up into pairs. We then conducted each evaluation with at least four pairs. Each pair of people performed two different evaluations, seeing different texts each time. The whole exercise required one session of approx-imately three hours.

Results: Since the number of subjects was too small to support statistically meaningful scores, we provide in Figure 5 the relative order of magnitude results instead of the actual scores. The percentage difference between score categories is listed in the columns below the scores (for example, scores marked "1" in the Shannon Game column were 50% better than scores marked "2" in that column).

Most surprising was the lack of difference in the Classification Game: despite the difference in distinctiveness of the classification categories, the two categories were different enough to pose

no challenges. Surprising in the Shannon Game was the result that the Full Text, Abstract, and SUMMARIST Extract were all equally good in providing information content. Finally, the relative ranking of results in the Question Game deserves some discussion. As expected, the Full Text provides the most information and No Text the least. However, the SUMMARIST Extract provided approx. 30% higher scores on average than either type of (human-made) Abstract, and the Random sentence selection was just as good as the Abstracts! A larger study is required to determine whether this result is simply a result of the texts selected or whether something more unsuspected is going on.

While not large enough to warrant any conclusive statements, these evaluation trials make clear that different evaluation methods rate summaries differently (compare the Classification Game to the others, even though some correlations can be seen between scores in the Shannon and Question Games). It is very clear that different summary types give different evaluation results, especially within the class of Extracts. More studies of this type will help us learn which kinds of evaluations are good for which kinds of text and summary types, domains, and uses.

		Shannon	Questions	Classification
Full Text		1	1	1
Abstract	**Background**	1	3	1
	Just-the-News	*	3	1
	Regular	1	2	1
Extract	**Keywords**	2	4	1
	Random	*	3	1
No Text		3	5	

1–2: 50% diff.
2–3: 50% diff.
*: not performed

1–2: 30% diff.
2–3: 20% diff.
3–4: 20% diff.
4–5: 100% diff.

Figure 5. Results of evaluation experiments at AAAI Spring Symposium.

7. Conclusion

The study of automated text summarization still has a long way to go before we can really claim to understand the nature of summaries. That does not mean, however, that the results obtained over the past few years are not useful. Even fairly surface-oriented systems such as SUMMARIST and the other ones built under the TIPSTER program already exhibit some commercial potential. The kinds of techniques outlined in Sections 4 and 5, to move from extracts to abstracts and all the other types of summary, will require some careful and large-scale work over the next years. The potential payoff is enormous, however, and growing, as the amount of text available in corpora and on the web keeps growing.

Several bottlenecks impede current development of summarization systems. The primary bottleneck is the shortage of good training data in a variety of domains and genres, and for a variety of summary types. What is required is triples—full text, human abstract, and corresponding extract—from which it will be possible to empirically determine answers to open questions in all three phases of summarization. A second bottleneck is the shortage of world knowledge, as required to perform topic interpretation. A third bottleneck

is the lack of resources and funding to focus on microplanning in summary generation.

In some sense, one might say that text summarization is one of the most difficult tasks of natural language processing. True abstracting requires true text understanding and true generation; it may quite possibly prove more difficult to find effective shortcuts than in machine translation or information extraction. But given how directly system improvements translate into useful results, text summarization is an exciting and very rewarding area in which to work.

Acknowledgements

We thank Louke van Wensveen for very useful initial discussions on evaluation, Daniel Marcu for the discourse work, Hao Liu and Mike Junk for their tireless experiments on topic signatures, Thérèse Firmin Hand, Sara Shelton, and Beth Sundheim for discussions about evaluation, and especially Sara Shelton for continued encouragement.

References

Aone, C., M.E. Okurowski, J. Gorlinsky, B. Larsen. 1998. A Scalable Summarization System using Robust NLP. In I. Mani and M. Maybury (eds), *Advances in Automated Text Summarization*. MIT Press.

Bagga, A. and B. Baldwin. 1998. Entity-Based Cross-Document Cross-Referencing using the Vector Space Model. *In Proceedings of COLING/ACL*, 79–86. Montreal, Canada.

Baldwin, B. 1998. Reworking of TIPSTER/SUMMAC Dry Run Evaluation Results. University of Pennsylvania Report.

Baxendale, P.B. 1958. Machine-Made Index for Technical Literature—An Experiment. *IBM Journal* (October) 354–361.

Brill, E. 1992. *A Corpus-Based Approach to Language Learning*. Ph.D. dissertation, University of Pennsylvania.

Edmundson, H.P. 1968. New Methods in Automatic Extraction. *Journal of the ACM* 16(2), 264–285.

Elhadad, M. 1992. *Using Argumentation to Control Lexical Choice: A Functional Unification-Based Approach*. Ph.D. dissertation, Columbia University.

Endres-Niggemeyer, B. 1997. SimSum: Simulation of Summatizing. In *Proceedings of the Workshop on Intelligent Scalable Summarization* at the ACL/EACL Conference, 89–96. Madrid, Spain.

Firmin Hand, T. and B. Sundheim. 1998. TIPSTER-SUMMAC Summarization Evaluation. *Proceedings of the TIPSTER Text Phase III Workshop*. Washington.

Hovy, E.H. and L. Wanner. 1996. Managing Sentence Planning Requirements. In *Proceedings of the Workshop on Gaps and Bridges in NL Planning and Generation*, 53–58. ECAI Conference. Budapest, Hungary.

Hovy, E.H. 1998. Combining and Standardizing Large-Scale, Practical Ontologies for Machine Translation and Other Uses. In *Proceedings of the First International Conference on Language Resources and Evaluation (LREC)*, 535–542. Granada, Spain.

Hovy, E.H. and M. Junk. In prep.

Hovy, E.H. and C-Y. Lin. 1998. Automating Text Summarization in SUMMARIST. In I. Mani and M. Maybury (eds), *Advances in Automated Text Summarization*. MIT Press.

Hovy, E.H. The Evaluation of Summaries. In prep.

Knight, K. and S.K. Luk. 1994. Building a Large-Scale Knowledge Base for Machine Translation. *Proceedings of the Conference of the American Association of Artificial Intelligence* (AAAI-94), 773–778. Seattle, WA.

Kupiec, J., J. Pedersen, and F. Chen. 1995. A Trainable Document Summarizer. In *Proceedings of the Eighteenth Annual International ACM Conference on Research and Development in Information Retrieval* (SIGIR), 68–73. Seattle, WA.

Langkilde, I. and K. Knight. 1998. Generation that Exploits Corpus Knowledge. In *Proceedings of the COLING/ACL Conference*, 704–709. Montreal, Canada.

Lavoie, B. and O. Rambow. 1997. A Fast and Portable Realizer for Text Generation

Systems. In *Proceedings of the Applied Natural Language Processing Conference (ANLP-97)*, 265–270. Washington, DC.

Lin, C-Y. 1995. Topic Identification by Concept Generalization. In *Proceedings of the Thirty-third Conference of the Association of Computational Linguistics (ACL-95)*, 308–310. Boston, MA.

Lin, C-Y. 1997. *Robust Automated Topic Identification.* Ph.D. dissertation, University of Southern California.

Lin, C-Y. and E.H. Hovy. 1997. Identifying Topics by Position. In *Proceedings of the Applied Natural Language Processing Conference (ANLP-97)*, 283–290. Washington, DC.

Lin, C-Y. and E.H. Hovy. 1998. Automatic Text Categorization: A Concept-Based Approach. In prep.

Lin, C-Y. Training a Selection Function for Extraction in SUMMARIST. In prep.

Liu, H. and E.H. Hovy. Automated Learning of Cue Phrases for Text Summarization. In prep.

Luhn, H.P. 1959. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, 159–165.

Mann, W.C. and S.A. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text* 8(3) (243–281).

Marcu, D. 1997. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts.* Ph.D. dissertation, University of Toronto.

Marcu, D. 1998. *The Automatic Construction of Large-scale Corpora for Summarization Research.* Forthcoming.

Marcu, D. 1998a. Improving Summarization through Rhetorical Parsing Tuning. *Proceedings of the COLING-ACL Workshop on Very Large Corpora.* Montreal, Canada.

Marcu, D. 1998b. Automated Creation of Extracts from Texts and Abstracts. In prep.

McKeown, K.R. and D.R. Radev. 1995. Generating Summaries of Multiple News Articles. In *Proceedings of the Eighteenth Annual International ACM Conference on Research and Development in Information Retrieval* (SIGIR), 74–82. Seattle, WA.

Miller, G., R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. 1990. Five papers on WordNet. CSL Report 43, Cognitive Science Laboratory, Princeton University.

Mitra, M., A. Singhal, and C. Buckley. 1997. Automatic Text Summarization by Paragraph Extraction. In *Proceedings of the Workshop on Intelligent Scalable Summarization* at the ACL/EACL Conference, 39–46. Madrid, Spain.

Nirenburg, S., V. Lesser, and E. Nyberg. 1989. Controlling a Language Generation Planner. In *Proceedings of IJCAI*, 1524–1530. Detroit, MI.

The Penman Primer, User Guide, and Reference Manual. 1988. Unpublished documentation, Information Sciences Institute, University of Southern California.

Quinlan, J.R. 1986. Induction of Decision Trees. *Machine Learning* 81–106.

Rambow, O. and T. Korelsky. 1992. Applied Text Generation. In *Proceedings of the Applied Natural Language Processing Conference (ANLP).* Trento, Italy

Salton, G. 1988. *Automatic Text Processing.* Reading, MA: Addison-Wesley.

Shannon, C. 1951. Prediction and Entropy of Printed English. *Bell System Technical Journal*, January 1951.

Spärck Jones, K. 1998. Introduction to Text Summarisation. In I. Mani and M. Maybury (eds), *Advances in Automated Text Summarization.* MIT Press.

Strzalkowski, T. et al., 1998. ? In I. Mani and M. Maybury (eds), *Advances in Automated Text Summarization.* MIT Press.

Tait, J.I. and K. Spärck Jones. 1982. *Automatic Summarising of English Texts.* Ph.D. dissertation, Cambridge University.

Teufel, S. and M. Moens. 1998. Sentence Extraction as a Classification Task. In I. Mani and M. Maybury (eds), *Advances in Automated Text Summarization.* MIT Press.

TREC. Harman, D. (ed). 1995. *Proceedings of the TREC Conference.*

Van Dijk, T.A. and W. Kintsch. 1983. *Strategies of Discourse Comprehension*. New York: Academic Press.