Using Lexical Chains for Text Summarization

Regina Barzilay
Mathematics and Computer Science Dept.
Ben Gurion University in the Negev
Beer-Sheva, 84105 Israel
regina@cs.bgu.ac.il

Michael Elhadad
Mathematics and Computer Science Dept.
Ben Gurion University in the Negev
Beer-Sheva, 84105 Israel
http://www.cs.bgu.ac.il/ elhadad

Abstract

We investigate one technique to produce a summary of an original text without requiring its full semantic interpretation, but instead relying on a model of the topic progression in the text derived from lexical chains. We present a new algorithm to compute lexical chains in a text, merging several robust knowledge sources: the WordNet thesaurus, a part-of-speech tagger and shallow parser for the identification of nominal groups, and a segmentation algorithm derived from (Hearst, 1994). Summarization proceeds in three steps: the original text is first segmented, lexical chains are constructed, *strong* chains are identified and *significant* sentences are extracted from the text.

We present in this paper empirical results on the identification of strong chains and of significant sentences. Preliminary results indicate that quality indicative summaries are produced and are extensively documented in `http://www.cs.bgu.ac.il/summarization-test`. Pending problems are identified: the need for anaphora resolution, a model for reconstructing a coherent summary out of the selected sentences, a method to handle long sentences and a method to control the degree of condensation of the original text. Plans to address these short-comings are briefly presented.

1 Introduction

Summarization is the process of condensing a source text into a shorter version preserving its information content. It can serve several goals — from survey analysis of a scientific field to quick indicative notes on the general topic of a text. Producing a quality informative summary of an arbitrary text remains a challenge which requires full understanding of the text. Indicative summaries, which can be used to quickly decide whether a text is worth reading, are naturally easier to produce. In this paper we investigate a method for the production of such indicative summaries from arbitrary text.

(Jones, 1993) describes summarization as a two-step process: (1) Building from the source text a source representation; (2) Summary generation — forming summary representation from the source representation built in the first step and synthesizing the output summary text.

Within this framework, the relevant question is what information has to be included in the source representation in order to create a summary. There are three types of source text information: linguistic, domain and communicative. Each of these text aspects can be chosen as a basis for source representation.

Summaries can be built on a deep semantic analysis of the source text. For example, in (McKeown and Radev, 1995), McKeown and Radev investigate ways to produce a coherent summary of several texts describing the same event, when a detailed semantic representation of the source texts is available (in their case, they use MUC-style systems to interpret the source texts).

Alternatively, early summarization systems (Luhn, 1968) used only linguistic source information. The intuition was that the most frequent words represent the important concepts of the text. In this approach the source representation was the frequency table of text words. This representation abstracts the text into the union of its words without considering any connection among them.

In contrast to these two extreme positions (using as a source representation a full semantic representation of the text or reducing it to a simple frequency

table), we deal in this paper with the issue of producing a summary from an arbitrary text without requiring its full understanding, but using widely available knowledge sources. Our main goal is therefore to find a middle ground for source representation, rich enough to build quality indicative summaries, but easy enough to extract from the source text to work on arbitrary text.

Over-simplification can harm the quality of the source representation. As a trivial illustration, consider the following two sequences:

1. "*Dr.Kenny has invented an anesthetic machine . This device controls the rate at which an anaesthetic is pumped into the blood.*"

2. "*Dr.Kenny has invented an anesthetic machine. The Doctor spent two years on this research.*"

"Dr.Kenny" appears once in both sequences and so does *"machine"*. But sequence 1 is about the *machine*, and sequence 2 is about the *"doctor"*. This example indicates that if the source representation does not supply information about semantically related terms, one cannot capture the "aboutness" of the text, and therefore the summary will not capture the main point of the original text.

The notion of cohesion, introduced in (Halliday and Hasan, 1976) captures part of the intuition. Cohesion is a device for "sticking together" different parts of the text. Cohesion is achieved through the use of semantically related terms, reference, ellipsis and conjunctions.

Among these different means, the most easily identifiable and the most frequent type is lexical cohesion (as discussed in (Hoey, 1991)). Lexical cohesion is created by using semantically related words. Halliday and Hasan classified lexical cohesion into reiteration category and collocation category. Reiteration can be achieved by repetition, synonyms and hyponyms. Collocation relations specify the relation between words that tend to co-occur in the same lexical contexts (e.g., "*She works as a* **teacher** *in the* **School**").

Collocation relations are more problematic for identification than reiteration, but both of these categories are identifiable on the surface of the text. Lexical cohesion occurs not only between two terms, but among sequences of related words — called *lexical chains* (Morris and Hirst, 1991). Lexical chains provide a representation of the lexical cohesive structure of the text. Lexical chains have also been used for information retrieval (Stairmand, 1996) and for correction of malapropisms (Hirst and St-Onge, 1997 (to appear)). In this paper, we investigate how lexical chains can be used as a source representation for summarization.

Another important dimension of the linguistic structure of a source text is captured under the related notion of *coherence*. Coherence defines the macro-level semantic structure of a connected discourse, while cohesion creates connectedness in a non-structural manner. Coherence is represented in terms of coherence relations between text segments, such as *elaboration*, *cause* and *explanation*. Some researchers, *e.g.*, (Ono, Kazuo, and Seiji, 1994), use discourse structure (encoded using RST (Mann and Thompson, 1987) as a source representation for summarization). Clearly, this representation is expressive enough; the question is whether it is computable. In contrast to lexical cohesion, coherence is difficult to identify without complete understanding of the text and complex inference. In addition, there is no precise criteria for classification of different relations. Consider the following example from Hobbs(1978): "*John can open the safe. He knows the combination.*"

(Morris and Hirst, 1991) show that the relation between these two sentences can be interpreted as *elaboration* or as *explanation*, depending on "context, knowledge and beliefs."

There is, however, a close connection between discourse structure and cohesion. Related words tend to co-occur within a discourse unit of the text. So cohesion is one of the surface signs of discourse structure and lexical chains can be used to identify it. Other signs can be used to identify discourse structure as well (connectives, paragraph markers, tense shifts).

In this paper, we investigate the use of lexical chains as a model of the source text for the purpose of producing a summary. Obviously, other aspects of the source text need to be integrated in the text representation to produce quality summaries; but we want to empirically investigate how far one can go exploiting mainly lexical chains. In the rest of the paper we first present our algorithm for lexical chain construction. We then present empirical results on the identification of strong chains among the possible candidates produced by our algorithm. Finally, we describe how lexical chains are used to identify significant sentences within the source text and eventually produce a summary.

2 Algorithm for Chain Computing

One of the chief advantages of lexical cohesion is that it is an easily recognizable relation, enabling lexical chains computation. The first computational model for lexical chains was presented in (Morris and

Hirst, 1991). They define lexical cohesion relations in terms of categories, index entries and pointers in Roget's Thesaurus. Morris and Hirst evaluated that their relatedness criterion covered over 90% of the intuitive lexical relations. Chains are created by taking a new text word and finding a related chain for it according to relatedness criteria. Morris and Hirst introduce the notion of "activated chain" and "chain returns", to take into account the distance between occurrences of related words. They also analyze factors contributing to the strength of a chain — repetition, density and length. Morris and Hirst did not implement their algorithm, because there was no machine-readable version of Roget's Thesaurus at the time.

One of the drawbacks of their approach was that they did not require the same word to appear with the same sense in its different occurrences for it to belong to a chain. For semantically ambiguous words, this can lead to confusions (e.g., mixing two senses of *table* as a piece of furniture or an array). Note that choosing the appropriate chain for a word is equivalent to disambiguating this word in context, which is a well-known difficult problem in text understanding.

More recently, two algorithms for the calculation of lexical chains have been presented in Hirst and St-Onge (1995) and Stairmand (1996). Both of these algorithms use the WordNet lexical database for determining relatedness of the words (Miller et al., 1990). Senses in the WordNet database are represented relationally by synonym sets ('synsets') — which are the sets of all the words sharing a common sense. For example two senses of "*computer*" are represented as: {calculator, reckoner, figurer, estimator, computer} (*i.e.*, a person who computes) and {computer, data processor, electronic computer, information processing system}. WordNet contains more than 118,000 different word forms. Words of the same category are linked through semantic relations like synonymy and hyponymy.

Polysemous words appear in more than one synsets (for example, *computer* occurs in two synsets). Approximately 17% of the words in WordNet are polysemous. But, as noted by Stairmand, this figure is very misleading: "a significant proportion of WordNet nouns are Latin labels for biological entities, which by their nature are monosemous and our experience with the news-report texts we have processed is that approximately half of the nouns encountered are polysemous." (Stairmand, 1996).

Generally, a procedure for constructing lexical chains follows three steps: (1) Select a set of candidate words; (2) For each candidate word, find an appropriate chain relying on a relatedness criterion among members of the chains; (3) If it is found, insert the word in the chain and update it accordingly.

An example of such a procedure was represented by Hirst and St-Onge (H&S). In the preprocessor step, all words that appear as a noun entry in WordNet are chosen. Relatedness of words is determined in terms of the distance between their occurrences and the shape of the path connecting them in the WordNet thesaurus. Three kinds of relation are defined: extra-strong (between a word and its repetition), strong (between two words connected by a Wordnet relation) and medium-strong when the link between the synsets of the words is longer than one (only paths satisfying certain restrictions are accepted as valid connections).

The maximum distance between related words depends on the kind of relation: for extra-strong relations, there is not limit in distance, for strong relations, it is limited to a window of seven sentences; and for medium-strong relations, it is within three sentences back.

To find a chain in which to insert a given candidate word, extra-strong relations are preferred to strong-relations and both of them are preferred to medium-strong relations. If a chain is found, then the candidate word is inserted with the appropriate sense, and the senses of the other words in the receiving chain are updated, so that every word connected to the new word in the chain relates to its selected senses only. If no chain is found, then a new chain is created and the candidate word is inserted with all its possible senses in WordNet.

The greedy disambiguation strategy implemented in this algorithm has some limitations illustrated by the following example:

> **Mr.** *Kenny is the* **person** *that invented an anaesthetic* **machine** *which uses* **micro-computers** *to control the rate at which an anaesthetic is pumped into the blood. Such* **machines** *are nothing new. But his* **device** *uses two* **micro-computers** *to achieve much closer monitoring of the* **pump** *feeding the anaesthetic into the patient.*

According to H&S's algorithm, the chain for the word "*Mr.*" is first created `[lex "Mr.", sense {mister, Mr.}]`. "*Mr.*" belongs only to one synset, so it is disambiguated from the beginning. The word "*person*" is related to this chain in the sense "*a human being*" by a medium-strong relation, so the chain now contains two entries:
`[lex "Mr.", sense {mister, Mr.}]`

`[lex "person", sense {person, individual, someone, man, mortal, human, soul}]`.
When the algorithm processes the word "*machine*", it relates it to this chain, because "*machine*" in the first WordNet sense ("*an efficient person*") is a holonym of "*person*" in the chosen sense. In other words, "*machine*" and "*person*" are related by a strong relation. In this case, "*machine*" is disambiguated in the wrong way, even though after this first occurrence of "*machine*", there is strong evidence supporting the selection of its more common sense: "*micro-computer*", "*device*" and "*pump*" all point to its correct sense in this context — "*any mechanical or electrical device that performs or assists in the performance*".

This example indicates that disambiguation cannot be a greedy decision. In order to choose the right sense of the word the 'whole picture' of chain distribution in the text must be considered. We propose to develop a chaining model according to all possible alternatives of word senses and then choose the best one among them.

Let us illustrate this method on the above example. First, a node for the word "*Mr.*" is created `[lex "Mr.", sense {mister, Mr.}]`. The next candidate word is "*person*". It has two senses: "*human being*" (*person* − 1) and "*grammatical category of pronouns and verb forms*" (*person* − 2). The choice of sense for "person" splits the chain world to two different interpretations as shown in Figure 1.

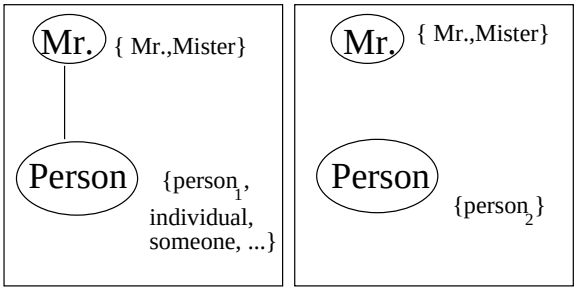

Figure 1: Step 1: Interpretations 1 and 2

We define a *component* as a list of interpretations that are exclusive of each other. Component words influence each other in the selection of their respective senses.

The next candidate word "*anaesthetic*" is not related to any word in the first component, so we create a new component for it with a single interpretation.

The word "*machine*" has 5 senses $machine_1$ to $machine_5$. In its first sense, "*an efficient person*", it is related to the senses "*person*" and "*Mr.*". It therefore influences the selection of their senses, thus

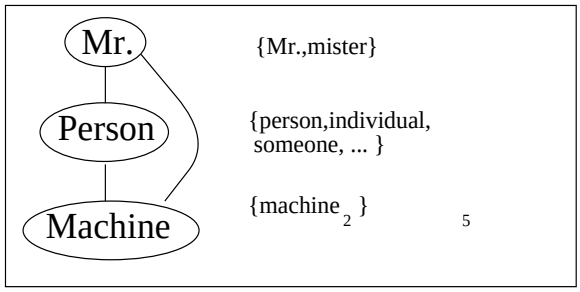

Figure 2: Step 2: Interpretation 1

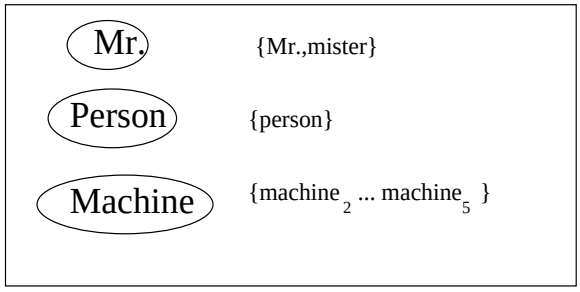

Figure 3: Step 2: Interpretation 2

"*machine*" has to be inserted in the first component. After its insertion the picture of the first component becomes the one shown in Figures 2 to 5.

But if we continue the process and insert the words "*micro-computer*", "*device*" and "*pump*", the number of alternative greatly increases. The strongest interpretations are given in Figures 6 and 7.

Under the assumption that the text is *cohesive*, we define the best interpretation as the interpretation with the most connections (edges in the graph). In this case, the second interpretation at the end of Step 3 is selected, which predicts the right sense for "*machine*". We define the score of an interpretation as the sum of its chain scores. Chain score is determined by the number and weight of the relations between chain members. Experimentally, we fixed the weight of reiteration and synonym to 10, of antonym to 7, and of hyperonym and holonym to 4. Our al-

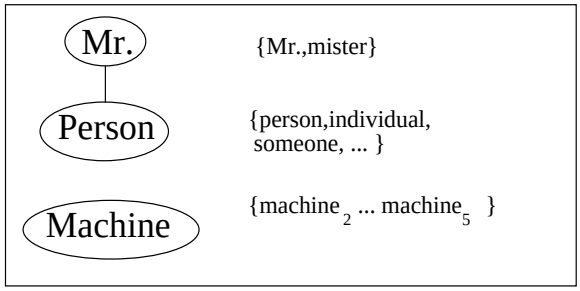

Figure 4: Step 2: Interpretation 3

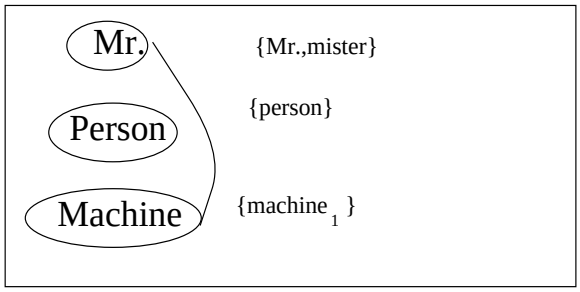

Figure 5: Step 2: Interpretation 4

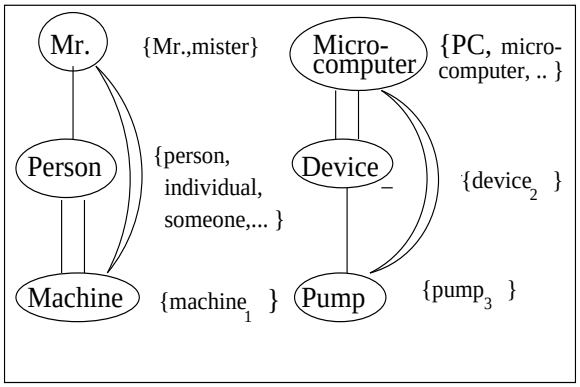

Figure 6: Step 3: Interpretation 1

gorithm develops all possible interpretations, maintaining each one without self contradiction. When the number of possible interpretations is larger than a certain threshold, we prune the weak interpretations according to this criteria. In the end, we select from each component the strongest interpretation.

In summary, our algorithm differs from H&S's algorithm in that it introduces, in addition to the relatedness criterion for membership to a chain, a non-greedy disambiguation heuristic to select the appropriate senses of chain members.

The two algorithms differ in two other major aspects: the criterion for the selection of candidate words and the operative definition of a text unit.

We choose as candidate words simple nouns and noun compounds. As mentioned above, nouns are the main contributors to the "aboutness" of a text, and noun synsets dominate in WordNet. Both (Stairmand, 1996) and H&S rely only on nouns as candidate words. In our algorithm, we rely on the results of Brill's part-of-speech tagging algorithm to identify nouns, while H&S do not go through this step and only select tokens that happen to occur as nouns in WordNet.

In addition, we extend the set of candidate words to include *noun compound*. We first empirically evaluated the importance of noun compounds by taking

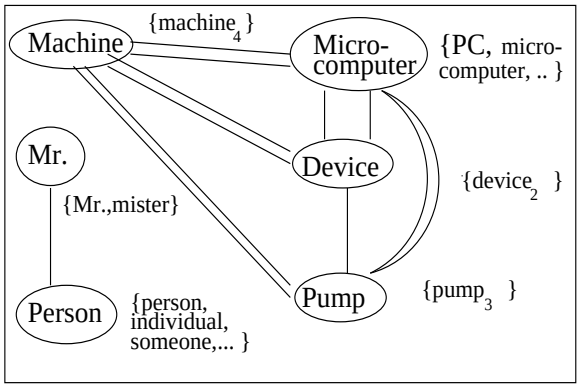

Figure 7: Step 3: Interpretation 2

into account the noun compounds explicitly present in WordNet (some 50,000 entries in WordNet are noun compounds such as "sea level" or collocations such as "digital computer"). However, English includes a productive system of noun compounds, and in each domain, new noun-compounds and collocations not present in WordNet play a major role.

We addressed the issue, by using a shallow parser (developed by Ido Dagan's team at Bar Ilan University) to identify noun-compounds using a simple characterization of noun sequences. This has two major benefits: (1) it identifies important concepts in the domain (for example, in a text on "*quantum computing*", the main token was the noun compound "*quantum computing*" which was not present in WordNet); (2) it eliminates words that occur as modifiers as possible candidates for chain membership. For example, when "*quantum computing*" is selected as a single unit, the word "*quantum*" is not selected. This is beneficial because in this example, the text was not about "*quantum*", but more about computers. When a noun compound is selected, the relatedness criterion in WordNet is used by considering its head noun only. Thus, "*quantum computer*" is related to "*machine*" as a "*computer*".

The second difference in our algorithm lies in the operative definition we give to the notion of text unit. We use as text units the segments obtained from Hearst's algorithm of text segmentation (Hearst, 1994). We build chains in every segment according to relatedness criteria, and in a second stage, we merge chains from the different segments using much stronger criteria for connectedness only: two chains are merged across a segment boundary only if they contain a common word with the same sense. Our intra-segment relatedness criterion is less strict: members of the same synsets are related, a node and its offspring in the hyperonym graph are related, siblings in the hyperonym graph are related

only if the length of the path is less than a threshold.

The relation between text segmentation and lexical chain is delicate, since they are both derived from partially common source of knowledge: lexical distribution and repetitions. In fact, lexical chains could serve as a basis for an algorithm for segmentation. We have found empirically, however, that Hearst's algorithm behaves well on the type of texts we checked and that it provides effectively a solid basis for lexical chains construction.

3 Building Summaries Using Lexical Chains

We now investigate how lexical chains can serve as a source representation of the original text to build a summary. The next question is how to build summary representation from this source representation.

The most prevalent discourse topic will play an important role in the summary. We first present the intuition why lexical chains are a good indicator of the central topic of a text. Given an appropriate measure of strength, we show that picking the concepts represented by strong lexical chains gives a better indication of the central topic of a text than simply picking the most frequent words in the text (which forms the zero-hypothesis).

For example, we show in Appendix A a sample text about Bayesian Network technology. There, the concept of network was represented by the words "*network*" with 6 occurrences, "*net*" with 2, and "*system*" with 4. But the summary representation has to reflect that all these words represent the **same** concept. Otherwise, the summary generation stage would extract information separately for each term. The chain representation approach avoids completely this problem, because all these terms occur in the same chain, which reflects that they represent the same concept.

An additional argument for the chain representation as opposed to a simple word frequency model is the case when a single concept is represented by a number of words, each with relatively low frequency. In the same Bayesian Network sample text, the concept of "*information*" was represented by the words "*information*" (3), "*datum*" (2), "*knowledge*" (3), "*concept*" (1) and "*model*" 1. In this text, "*information*" is a more important concept than "*computer*" which occurs 4 times. Because the "*information*" chain combines the number of occurrences of all its members, it can overcome the weight of the single word "*computer*".

3.1 Scoring Chains

In order to use lexical chains as outlined above, one must first identify the strongest chains among all those that are produced by our algorithm. As is frequent in summarization, there is no formal way to evaluate chain strength (as there is no formal method to evaluate a summary quality). We therefore rely on an empirical methodology. We have developed an environment to compute and graphically visualize lexical chains to evaluate experimentally how they capture the main topics of the texts. Figure 8 shows how lexical chains are visualized to help human testers evaluate their importance.

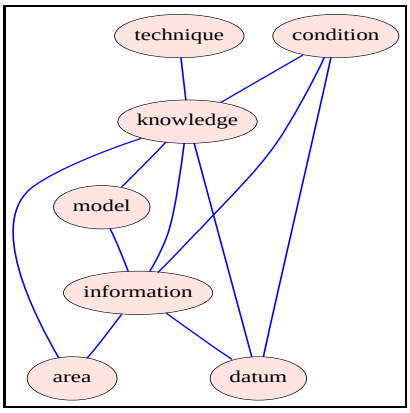

Figure 8: Visual representation of lexical chains

We have collected data for a set of 30 texts extracted from popular magazines (from "The Economist" and "Scientific American"), all of them are popular science genre. For each text, we manually ranked chains in terms of relevance to the main topics. We then computed different formal measures on the chains, including: chain length, distribution in the text, text span covered by the chain, density, graph topology (diameter of the graph of the words) and number of repetitions. The results on our data set indicate that only the following parameters are good predictors of the strength of a chain:

Length: The number of occurrences of members of the chain.

Homogeneity index: 1 - the number of distinct occurrences divided by the length.

We designed a score function for chains as:

$$Score(Chain) = Length * Homogeneity$$

When ranking chains according to their score, we evaluated that strong chains are those which satisfy our "Strength Criterion":

$$Score(Chain) > Average(Scores) + 2 * StandardDeviation(Scores)$$

These are preliminary results but they are confirmed by our experience on 30 texts analyzed extensively. We have experimented with different normalization methods for the score function, but they do not seem to improve the results. We plan on extending the empirical analysis in the future and to use formal learning methods to determine a good scoring function.

The average number of strong chains selected by this selection method was 5 for texts of 1055 words on average (474 words minimum, 3198 words maximum), when 32 chains were originally generated on average. The strongest chain of the sample text are represented in Appendix B.

3.2 Extracting Significant Sentences

Once strong chains have been selected, the next step of the summarization algorithm is to extract full sentences from the original text based on chain distribution.

We investigated three alternatives for this step:

Heuristic 1: For each chain in the summary representation choose the sentence that contains the first appearance of a chain member in the text.

This heuristic produced the following summary for the text shown in Appendix:

When Microsoft Senior Vice President Steve Ballmer first heard his company was planning to make a huge investment in an Internet service offering movie reviews and local entertainment information in major cities across the nation, he went to Chairman Bill Gates with his concerns. Microsoft's competitive advantage, he responded, was its expertise in Bayesian networks.

Bayesian networks are complex diagrams that organize the body of knowledge in any given area by mapping out cause and effect relationships among key variables and encoding them with numbers that represent the extent to which one variable is likely to affect another.

Programmed into computers, these systems can automatically generate optimal predictions or decisions even when key pieces of information are missing.

When Microsoft in 1993 hired Eric Horvitz, David Heckerman and Jack Breese, pioneers in the development of Bayesian systems, colleagues in the field were surprised.

The problem with this approach is that all words in a chain reflect the same concept, but to a different extent. For example, in the AI chain, (Appendix B, Chain 3) the token "*science*" is related to the concept "*AI*", but the words "*AI*" and "*field*" are more suitable to represent the main topic "*AI*" in the context of the text. That is, not all chain members are good representatives of the topic (even though they all contribute to its meaning).

We therefore defined a criterion to evaluate the appropriateness of a chain member to represent its chain based on its frequency of occurrence in the chain. We found experimentally that such words, call them *representative* words, have a frequency in the chain no less than the average word frequency in the chain. For example, in the third chain the representative words are "*field*" and "*AI*".

Heuristic 2: We therefore defined a second heuristic based on the notion of representative words:

For each chain in the summary representation, choose the sentence that contains the first appearance of a representative chain member in the text.

In this special case this heuristic gives the same result as the first one.

Heuristic 3: Often, the same topic is discussed in a number of places in the text, so its chain is distributed across the whole text. Still, in some text unit, this global topic is the central topic (focus) of the segment. We try to identify this unit and extract sentences related to the topic from this segment (or successive segments) only.

We characterize this text unit as a cluster of successive segments with high density of chain members. Our third heuristic is based on this approach.

For each chain, find the text unit where the chain is highly concentrated. Extract the sentence with the first chain appearance in this central unit. Concentration is computed as the number of chain members occurrences in a segment divided by the number of nouns in the segment. A chain has high concentration if its concentration is the maximum of all chains. Cluster is group of successive segments such that every segment contains chain members.

Note that in all these three techniques only one sentence is extracted for each chain (regardless of its strength).

For most texts we tested, the first and second techniques produce the same results, but when they are different, the output of the second technique is better. Generally, the second technique produces the best summary. We checked these methods on our 30 texts data set. Surprisingly, the third heuristic, which intuition predicts as the most sophisticated, gives the least indicative results. This may be due to several factors: our criteria for 'centrality' or 'clustering' may be insufficient or, more likely, the problem seems to be related to the interaction with text structure. The third heuristics tends to extract sentences from the middle of the text and to extract several sentences from distant places in the text for a single chain. The complete results of our experiments are available on-line at `http://www.cs.bgu.ac.il/summarization-test`.

4 Limitations and Future Work

We have identified the following main problems with our method:

- Sentence granularity: all our methods extract whole sentences as single units. This has several drawbacks: long sentences have significantly higher likelihood to be selected, they also include many constituents which would not have been selected on their own merit. The alternative is extremely costly: it involves some parsing of the sentences, the extraction of only the central constituents from the source text and the regeneration of a summary text using text generation techniques.

- Extracted sentences contain anaphora links to the rest of the text. This has been investigated and observed by (Black, 1994). Several heuristics have been proposed in the literature to address this problem (Paice, 1990), (Paice and Husk, 1991) and (Black, 1994). The strongest seems to be to include together with the extracted sentence the one immediately preceding it. Unfortunately, when we select the first sentence in a segment, the preceding sentence does not belong to the paragraph and its insertion has a detrimental effect on the overall coherence of the summary. A preferable solution would be to replace anaphora with their referent, but again this is an extremely costly solution.

- Our method does not provide any way to control the length and level of detail of the summary. In all of the methods, we extract one sentence for each chain. The number of strong chains remains small (around 5 or 6 for the texts we have tested, regardless of their length), and the remaining chains would introduce too much noise to be of interest in adding details. The best solution seems to be to extract more material for the strongest chains.

The method presented in this paper is obviously partial in that it only considers lexical chains as a source representation, and ignores any other clues that could be gathered from the text. Still, our first informal evaluation indicates our results are of a quality superior to that of summarizers usually employed in commercial systems such as search systems on the World Wide Web on the texts we investigated. A large-scale evaluation of the method and how sensitive it is to the quality of the thesaurus and to its parameters is under way.

A Bayesian Networks Text

When Microsoft Senior Vice President Steve Ballmer first heard his company was planning to make a huge investment in an Internet service offering movie reviews and local entertainment information in major cities across the nation, he went to Chairman Bill Gates with his concerns.

After all, Ballmer has billions of dollars of his own money in Microsoft stock, and entertainment isn't exactly the company's strong point.

But Gates dismissed such reservations. Microsoft's competitive advantage, he responded, was its expertise in Bayesian networks.

Asked recently when computers would finally begin to understand human speech, Gates began discussing the critical role of "Bayesian" systems.

Ask any other software executive about anything Bayesian and you're liable to get a blank stare.

Is Gates onto something? Is this alien-sounding technology Microsoft's new secret weapon?

Bayesian networks are complex diagrams that organize the body of knowledge in any given area by mapping out cause-and-effect relationships among key variables and encoding them with numbers that represent the extent to which one variable is likely to affect another.

Programmed into computers, these systems can automatically generate optimal predictions or decisions even when key pieces of information are missing.

When Microsoft in 1993 hired Eric Horvitz, David Heckerman and Jack Breese, pioneers in the development of Bayesian systems, colleagues in the field were surprised. The field was still an obscure, largely academic enterprise.

Today the field is still obscure. But scratch the surface of a range of new Microsoft products and you're likely to find Bayesian networks embedded in the software. And Bayesian nets are being built into models that are used to predict oil and stock prices, control the space shuttle and diagnose disease.

Artificial intelligence (AI) experts, who saw their field discredited in the early 1980s after promising a wave of "thinking" computers that they ultimately couldn't produce, believe widening acceptance of the Bayesian approach could herald a renaissance in the field.

Bayesian nets provide "an overarching graphical framework" that brings together diverse elements of AI and increases the range of its likely application to the real world, says Michael Jordon, professor of brain and cognitive science at the Massachusetts Institute of Technology.

Microsoft is unquestionably the most aggressive in exploiting the new approach. The company offers a free Web service that helps customers diagnose printing problems with their computers and recommends the quickest way to resolve them. Another Web service helps parents diagnose their children's health problems.

The latest version of Microsoft Office software uses the technology to offer a user help based on past experience, how the mouse is being moved and what task is being done. "If his actions show he is distracted, he is likely to need help," Horvitz says. "If he's been working on a chart, chances are he needs help formatting the chart".

"Gates likes to talk about how computers are now deaf, dumb, blind and clueless. The Bayesian stuff helps deal with the clueless part," says Daniel T. Ling, director of Microsoft's research division and a former IBM scientist.

Horvitz and his two Microsoft colleagues, who were then classmates at Stanford University, began building Bayesian networks to help diagnose the condition of patients without turning to surgery.
The approach was efficient, says Horvitz, because you could combine historical data, which had been meticulously gathered, with the less precise but more intuitive knowledge of experts on how things work to get the optimal answer given the information available at a given time.
Horvitz, who with two colleagues founded Knowledge Industries to develop tools for developing Bayesian systems, says he and the others left the company to join Microsoft in part because they wanted to see their theoretical work more broadly applied.
Although the company did important work for the National Aeronautics and Space Administration and on medical diagnostics, Horvitz says, "It's not like your grandmother will use it".
Microsoft's activities in the field are now helping to build a groundswell of support for Bayesian ideas.
People look up to Microsoft," says Pearl, who wrote one of the key early texts on Bayesian networks in 1988 and has become an unofficial spokesman for the field. "They've given a boost to the whole area".
Microsoft is working on techniques that will enable the Bayesian networks to "learn" or update themselves automatically based on new knowledge, a task that is currently cumbersome.
The company is also working on using Bayesian techniques to improve upon popular AI approaches such as "data mining" and "collaborative filtering" that help draw out relevant pieces of information from massive databases. The latter will be used by Microsoft in its new online entertainment service to help people identify the kind of restaurants or entertainment they are most likely to enjoy.

B Bayesian Network Text: the Strongest Chain

The Criterion is 3.58, here are the five strong chains:

CHAIN 1: Score = 14.0
microsoft: 10 concern: 1 company: 6
entertainment-service: 1 enterprise: 1
massachusetts-institute: 1
CHAIN 2: Score = 9.0
bayesian-system: 2 system: 2 bayesian-net: 2
network: 1 bayesian-network: 5 weapon: 1
CHAIN 3: Score = 7.0
ai: 2 artificial-intelligence: 1
field: 7 technology: 1 science: 1
CHAIN 4: Score = 6.0
technique: 1 bayesian-technique: 1 condition: 1
datum: 2 model: 1 information: 3 area: 1
knowledge: 3
CHAIN 5: Score = 3.0
computer: 4

Acknowledgements

This work has been supported by the Israeli Ministry of Science. We are grateful to Graeme Hirst, Dragomir Radev, Claude Brisson for their feedback on a previous version.

References

Black, William J. 1994. Parsing, linguistic resources and semantic analysis, for abstracting and categorization.

Halliday, Michael and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Hearst, Marti A. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*.

Hirst, Graeme and David St-Onge. 1997 (to appear). Lexical chains as representation of context for the detection and correction of malapropisms. In Christiane Fellbaum, editor, *WordNet: An electronic lexical database and some of its applications*. Cambridge, MA: The MIT Press.

Hoey, M. 1991. *Patterns of Lexis in Text*. Oxford University Press, Oxford.

Jones, Karen Sparck. 1993. What might be in summary? *Information Retrieval*.

Luhn, H. P. 1968. The automatic creation of literature abstracts. In Schultz, editor, *H. P. Luhn: Pioneer of Information Science*. Spartan.

Mann, W. C. and S. Thompson. 1987. Rhetorical structure theory: description and constructions of text structures. In Gerard Kempen, editor, *Natural Language Generation: New Results in Artificial Intellligence, Psychology and Linguistics*. Martinus Ninjhoff Publishers, pages 85–96.

McKeown, Kathleen and Dragomir Radev. 1995. Generating summaries of multiple news articles. In *SIGIR 95 Proceedings*.

Miller, George A., Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography (special issue)*, 3(4):235–312.

Morris, J. and G. Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of the text. *Computational Linguistics*, 17(1):pp.21–45.

Ono, Kenji, Sumita Kazuo, and Miike Seiji. 1994. Abstract generation based on rhetorical structure extraction. In *Proceedings of the International Conference on Computational Linguistics (Coling 94)*, pages 344–348, Japan.

Paice, C. D and G. D. Husk. 1991. Towards the automatic recogmition of anaphoric features in english text: The impersonal pronoun "it". *Computer Speech and Language*, (2):pp.109–132.

Paice, Chris D. 1990. Constructing literature abstracts by computer : techniques and prospects. *Information Processing and Management*, 26(1):171–186.

Stairmand, Mark A. 1996. *A Computational Analysis of Lexical Cohesion with Applications in Information Retrieval*. Ph.D. thesis, Center for Computational Linguistics, UMIST, Manchester.