Sentence Reduction for Automatic Text Summarization

Hongyan Jing
Department of Computer Science
Columbia University
New York, NY 10027, USA
hjing@cs.columbia.edu

Abstract

We present a novel sentence reduction system for automatically removing extraneous phrases from sentences that are extracted from a document for summarization purposes. The system uses multiple sources of knowledge to decide which phrases in an extracted sentence can be removed, including syntactic knowledge, context information, and statistics computed from a corpus which consists of examples written by human professionals. Reduction can significantly improve the conciseness of automatic summaries.

1 Motivation

Current automatic summarizers usually rely on sentence extraction to produce summaries. Human professionals also often reuse the input documents to generate summaries; however, rather than simply extracting sentences and stringing them together, as most current summarizers do, humans often "edit" the extracted sentences in some way so that the resulting summary is concise and coherent. We analyzed a set of articles and identified six major operations that can be used for editing the extracted sentences, including removing extraneous phrases from an extracted sentence, combining a reduced sentence with other sentences, syntactic transformation, substituting phrases in an extracted sentence with their paraphrases, substituting phrases with more general or specific descriptions, and reordering the extracted sentences (Jing and McKeown, 1999; Jing and McKeown, 2000).

We call the operation of removing extraneous phrases from an extracted sentence *sentence reduction*. It is one of the most effective operations that can be used to edit the extracted sentences. Reduction can remove material at any granularity: a word, a prepositional phrase, a gerund, a to-infinitive or a clause. We use the term "phrase" here to refer to any of the above components that can be removed in reduction. The following example shows an original sentence and its reduced form written by a human professional:

Original sentence:
When it arrives sometime next year in new TV sets, the V-chip will give parents a *new and potentially revolutionary* device to block out programs they don't want their children to see.

Reduced sentence by humans:
The V-chip will give parents a device to block out programs they don't want their children to see.

We implemented an automatic sentence reduction system. Input to the reduction system includes extracted sentences, as well as the original document. Output of reduction are reduced forms of the extracted sentences, which can either be used to produce summaries directly, or be merged with other sentences. The reduction system uses multiple sources of knowledge to make reduction decisions, including syntactic knowledge, context, and statistics computed from a training corpus. We evaluated the system against the output of human professionals. The program achieved a success rate of 81.3%, meaning that 81.3% of reduction decisions made by the system agreed with those of humans.

Sentence reduction improves the conciseness of automatically generated summaries, making it concise and on target. It can also improve the coherence of generated summaries, since extraneous phrases that can potentially introduce incoherece are removed. We collected 500 sentences and their corresponding reduced forms written by humans, and found that humans reduced the length of these 500 sentences by 44.2% on average. This indicates that a good sentence reduction system can improve the conciseness of generated summaries significantly.

In the next section, we describe the sentence reduction algorithm in detail. In Section 3, we introduce the evaluation scheme used to assess the performance of the system and present evaluation results. In Section 4, we discuss other applications of sentence reduction, the interaction between reduction and other modules in a summarization system, and related work on sentence simplication. Finally, we conclude with future work.

2 Sentence reduction based on multiple sources of knowledge

The goal of sentence reduction is to "reduce without major loss"; that is, we want to remove as many extraneous phrases as possible from an extracted sentence so that it can be concise, but without detracting from the main idea the sentence conveys. Ideally, we want to remove a phrase from an extracted sentence *only if it is irrelevant to the main topic*. To achieve this, the system relies on multiple sources of knowledge to make reduction decisions. We first introduce the resources in the system and then describe the reduction algorithm.

2.1 The resources

(1) The corpus. One of the key features of the system is that it uses a corpus consisting of original sentences and their corresponding reduced forms written by humans for training and testing purposes. This corpus was created using an automatic program we have developed to automatically analyze human-written abstracts. The program, called the decomposition program, matches phrases in a human-written summary sentence to phrases in the original document (Jing and McKeown, 1999). The human-written abstracts were collected from the free daily news service "Communications-related headlines", provided by the Benton Foundation (http://www.benton.org). The articles in the corpus are news reports on telecommunication related issues, but they cover a wide range of topics, such as law, labor, and company mergers.

(2) The lexicon. The system also uses a large-scale, reusable lexicon we combined from multiple resources (Jing and McKeown, 1998). The resources that were combined include COMLEX syntactic dictionary (Macleod and Grishman, 1995), English Verb Classes and Alternations (Levin, 1993), the WordNet lexical database (Miller et al., 1990), the Brown Corpus tagged with WordNet senses (Miller et al., 1993). The lexicon includes subcategorizations for over 5,000 verbs. This information is used to identify the obligatory arguments of verb phrases.

(3) The WordNet lexical database. WordNet (Miller et al., 1990) is the largest lexical database to date. It provides lexical relations between words, including synonymy, antonymy, meronymy, entailment (e.g., *eat* $\rightarrow$ *chew*), or causation (e.g., *kill* $\rightarrow$ *die*). These lexical links are used to identify the focus in the local context.

(4) The syntactic parser. We use the English Slot Grammar(ESG) parser developed at IBM (McCord, 1990) to analyze the syntactic structure of an input sentence and produce a sentence parse tree. The ESG parser not only annotates the syntactic category of a phrase (e.g., "np" or "vp"), it also annotates the thematic role of a phrase (e.g., "subject" or "object").

2.2 The algorithm

There are five steps in the reduction program:

Step 1: Syntactic parsing.

We first parse the input sentence using the ESG parser and produce the sentence parse tree. The operations in all other steps are performed based on this parse tree. Each following step annotates each node in the parse tree with additional information, such as syntactic or context importance, which are used later to determine which phrases (they are represented as subtrees in a parse tree) can be considered extraneous and thus removed.

Step 2: Grammar checking.

In this step, we determine which components of a sentence *must not* be deleted to keep the sentence grammatical. To do this, we traverse the parse tree produced in the first step in top-down order and mark, for each node in the parse tree, which of its children are grammatically obligatory. We use two sources of knowledge for this purpose. One source includes simple, linguistic-based rules that use the thematic role structure produced by the ESG parser. For instance, for a sentence, the main verb, the subject, and the object(s) are essential if they exist, but a prepositional phrase is not; for a noun phrase, the head noun is essential, but an adjective modifier of the head noun is not. The other source we rely on is the large-scale lexicon we described earlier. The information in the lexicon is used to mark the obligatory arguments of verb phrases. For example, for the verb "convince", the lexicon has the following entry:

```
convince
   sense 1:
       NP-PP :PVAL (''of'')
       NP-TO-INF-OC
   sense 2:
       NP
```

This entry indicates that the verb "convince" can be followed by a noun phrase and a prepositional phrase starting with the preposition "of" (e.g., he convinced me of his innocence). It can also be followed by a noun phrase and a to-infinitive phrase (e.g., he convinced me to go to the party). This information prevents the system from deleting the "of" prepositional phrase or the to-infinitive that is part of the verb phrase.

At the end of this step, each node in the parse tree — including both leaf nodes and intermediate nodes — is annotated with a value indicating whether it is grammatically obligatory. Note that whether a node is obligatory is relative to its parent node only. For example, whether a determiner is obligatory is relative to the noun phrase it is in; whether a prepositional phrase is obligatory is relative to the sentence or the phrase it is in.

Step 3: Context information.

In this step, the system decides which components in the sentence are most related to the main topic being discussed. To measure the importance of a phrase in the local context, the system relies on lexical links between words. The hypothesis is that the more connected a word is with other words in the local context, the more likely it is to be the focus of the local context. We link the words in the extracted sentence with words in its local context, if they are repetitions, morphologically related, or linked in WordNet through one of the lexical relations. The system then computes an importance score for each word in the extracted sentence, based on the number of links it has with other words and the types of links. The formula for computing the context importance score for a word w is as follows:

$$ContextWeight(w) = \sum_{i=1}^{9}(L_i \times NUM_i(w))$$

Here, i represents the different types of lexical relations the system considered, including repetition, inflectional relation, derivational relation, and the lexical relations from WordNet. We assigned a weight to each type of lexical relation, represented by L_i in the formula. Relations such as repetition or inflectional relation are considered more important and are assigned higher weights, while relations such as hypernym are considered less important and assigned lower weights. $NUM_i(w)$ in the formula represents the number of a particular type of lexical links the word w has with words in the local context.

After an importance score is computed for each word, each phrase in the sentence gets a score by adding up the scores of its children nodes in the parse tree. This score indicates how important the phrase is in the local context.

Step 4: Corpus evidence.

The program uses a corpus consisting of sentences reduced by human professionals and their corresponding original sentences to compute how likely humans remove a certain phrase. The system first parsed the sentences in the corpus using the ESG parser. It then marked which subtrees in these parse trees (i.e., phrases in the sentences) were removed by humans. Using this corpus of marked parse trees, we can compute how likely a subtree is removed from its parent node. For example, we can compute the probability that the "when" temporal clause is removed when the main verb is "give", represented as $Prob$("when-clause is removed"|"v=give"), or the probability that the to-infinitive modifier of the head noun "device" is removed, represented as $Prob$("to-infinitive modifier is removed"|"n=device"). These probabilities are computed using Bayes's rule. For example, the probability that the "when" temporal clause is removed when the main verb is "give", $Prob$("when-clause is removed"|"v=give"), is computed as the product of $Prob$("v=give"|"when-clause is removed") (i.e., the probability that the main verb is "give" when the "when" clause is removed) and $Prob$("when-clause is removed") (i.e., the probability that the "when" clause is removed), divided by $Prob$("v=give") (i.e., the probability that the main verb is "give").

Besides computing the probability that a phrase is removed, we also compute two other types of probabilities: the probability that a phrase is reduced (i.e., the phrase is not removed as a whole, but some components in the phrase are removed), and the probability that a phrase is unchanged at all (i.e., neither removed nor reduced).

These corpus probabilities help us capture human practice. For example, for sentences like "The agency reported that ...", "The other source says that ...", "The new study suggests that ...", the that-clause following the say-verb (i.e., report, say, and suggest) in each sentence is very rarely changed at all by professionals. The system can capture this human practice, since the probability that *that-clause* of the verb *say* or *report* being unchanged at all will be relatively high, which will help the system to avoid removing components in the that-clause.

These corpus probabilities are computed beforehand using a training corpus. They are then stored in a table and loaded at running time.

Step 5: Final Decision.

The final reduction decisions are based on the results from all the earlier steps. To decide which phrases to remove, the system traverses the sentence parse tree, which now has been annotated with different types of information from earlier steps, in the top-down order and decides which subtrees should be removed, reduced or unchanged. A subtree (i.e., a phrase) is removed only if it is not grammatically obligatory, not the focus of the local context (indicated by a low importance score), and has a reasonable probability of being removed by humans.

Figure 1 shows sample output of the reduction program. The reduced sentences produced by humans are also provided for comparison.

3 Evaluation

3.1 The evaluation scheme

We define a measure called *success rate* to evaluate the performance of our sentence reduction program.

Example 1:
Original sentence : *When it arrives sometime next year in new TV sets,* **the V-chip will give parents a new and potentially revolutionary device to block out programs they don't want their children to see.**
Reduction program: The V-chip will give parents a new and potentially revolutionary device to block out programs they don't want their children to see.
Professionals : The V-chip will give parents a device to block out programs they don't want their children to see.

Example 2:
Original sentence : **Som and Hoffman's creation would allow broadcasters to insert multiple ratings into a show,** *enabling the V-chip to filter out racy or violent material but leave unexceptional portions of a show alone.*
Reduction Program: Som and Hoffman's creation would allow broadcasters to insert multiple ratings into a show.
Professionals : *(the same)*

Figure 1: Sample output of sentence reduction program

The success rate computes the percentage of the system's reduction decisions that agree with those of humans.

We compute the *success rate* in the following way. The reduction process can be considered as a series of decision-making processes along the edges of a sentence parse tree. At each node of the parse tree, both the human and the program make a decision whether to remove the node or to keep it. If a node is removed, the subtree with that node as the root is removed as a whole, thus no decisions are needed for the descendants of the removed node. If the node is kept, we consider that node as the root and repeat this process.

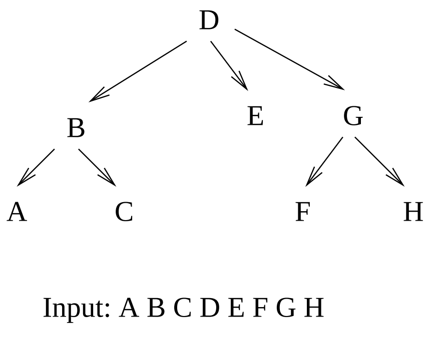

Figure 2: Sample sentence and parse tree

Suppose we have an input sentence *(ABCDEFGH)*, which has a parse tree shown in Figure 2. Suppose a human reduces the sentence to *(ABDGH)*, which can be translated to a series of decisions made along edges in the sentence parse tree as shown in Figure 3. The symbol "y" along an edge means the node it points to will be kept, and "n" means the node will be removed. Suppose the program reduces the sentence to *(BCD)*, which can be translated similarly to the annotated tree shown in Figure 4.

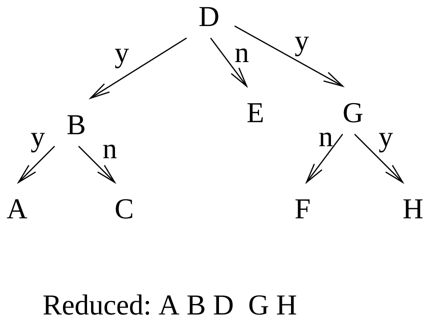

Figure 3: Reduced form by a human

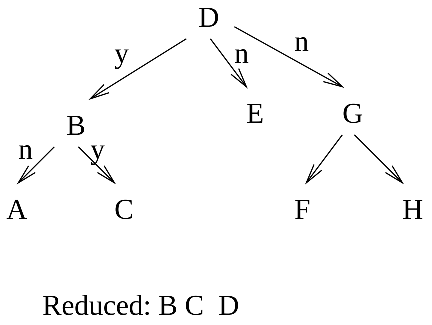

Figure 4: Reduced form by the program

We can see that along five edges (they are D→B, D→E, D→G, B→A, B→C), both the human and the program made decisions. Two out of the five decisions agree (they are D→B and D→E), so the success rate is 2/5 (40%). The *success rate* is defined as:

$$\text{success rate} = \frac{\begin{array}{l}\text{\# of edges along which the hu-}\\ \text{man and the program have made}\\ \text{the same decision}\end{array}}{\begin{array}{l}\text{the total \# of edges along which}\\ \text{both the human and the progam}\\ \text{have made decisions}\end{array}}$$

Note that the edges along which only the human or the program has made a decision (e.g., G→F and G→F in Figure 3 and Figure 4) are not considered in the computation of success rate, since there is no agreement issue in such cases.

3.2 Evaluation result

In the evaluation, we used 400 sentences in the corpus to compute the probabilities that a phrase is removed, reduced, or unchanged. We tested the program on the rest 100 sentences.

Using five-fold validation (i.e., choose different 100 sentences for testing each time and repeating the experiment five times), The program achieved an average success rate of 81.3%. If we consider the baseline as removing all the prepositional phrases, clauses, to-infinitives and gerunds, the baseline performance is 43.2%.

We also computed the success rate of the program's decisions on particular types of phrases. For the decisions on removing or keeping a clause, the system has a success rate of 78.1%; for the decisions on removing or keeping a to-infinitive, the system has a success rate of 85.2%. We found out that the system has a low success rate on removing adjectives of noun phrases or removing adverbs of a sentence or a verb phrase. One reason for this is that our probability model can hardly capture the dependencies between a particular adjective and the head noun since the training corpus is not large enough, while the other sources of information, including grammar or context information, provide little evidence on whether an adjective or an adverb should be removed. Given that whether or not an adjective or an adverb is removed does not affect the conciseness of the sentence significantly and the system lacks of reliability in making such decisions, we decided not to remove adjectives and adverbs.

On average, the system reduced the length of the 500 sentences by 32.7% (based on the number of words), while humans reduced them by 41.8%.

The probabilities we computed from the training corpus covered 58% of instances in the test corpus. When the corpus probability is absent for a case, the system makes decisions based on the other two sources of knowledge.

Some of the errors made by the system result from the errors by the syntactic parser. We randomly checked 50 sentences, and found that 8% of the errors made by the system are due to parsing errors. There are two main reasons responsible for this relatively low percentage of errors resulting from mistakes in parsing. One reason is that we have taken some special measures to avoid errors introduced by mistakes in parsing. For example, PP attachment is a difficult problem in parsing and it is not rare that a PP is wrongly attached. Therefore, we take this into account when marking the obligatory components using subcategorization knowledge from the lexicon (step 2) – we not only look at the PPs that are attached to a verb phrase, but also PPs that are next to the verb phrase but not attached, in case it is part of the verb phrase. We also wrote a preprocessor to deal with particular structures that the parser often has problems with, such as appositions. The other reason is that parsing errors do not always result in reduction errors. For example, given a sentence "The spokesperson of the University said that ...", although the that-clause in the sentence may have a complicated structure and the parser gets it wrong, the reduction system is not necessarily affected since it may decide in this case to keep the that-clause as it is, as humans often do, so the parsing errors will not matter in this example.

4 Discussion and related work

The reduction algorithm we present assumes generic summarization; that is, we want to generate a summary that includes the most important information in an article. We can tailor the reduction system to query-based summarization. In that case, the task of the reduction is not to remove phrases that are extraneous in terms of the main topic of an article, but phrases that are not very relevant to users' queries. We extended our sentence reduction program to query-based summarization by adding another step in the algorithm to measure the relevance of users' queries to phrases in the sentence. In the last step of reduction when the system makes the final decision, the relevance of a phrase to the query is taken into account, together with syntactic, context, and corpus information.

Ideally, the sentence reduction module should interact with other modules in a summarization system. It should be able to send feedback to the extraction module if it finds that a sentence selected by the extraction module may be inappropriate (for example, having a very low context importance score). It should also be able to interact with the modules that run after it, such as the sentence combination module, so that it can revise reduction decisions according to the feedback from these modules.

Some researchers suggested removing phrases or clauses from sentences for certain applications. (Grefenstette, 1998) proposed to remove phrases in sentences to produce a telegraphic text that can be used to provide audio scanning service for the blind. (Corston-Oliver and Dolan, 1999) proposed to remove clauses in sentences before indexing documents for Information Retrieval. Both studies removed phrases based only on their syntactic categories, while the focus of our system is on deciding *when it is appropriate to remove a phrase*.

Other researchers worked on the text simplification problem, which usually involves simplifying text but not removing any phrases. For example, (Carroll et al., 1998) discussed simplifying newspaper text by replacing uncommon words with common words, or replacing complicated syntactic structures with simpler structures to assist people with reading disabilities. (Chandrasekar et al., 1996) discussed text simplification in general. The difference between these studies on text simplification and our system is that a text simplification system usually does not *remove* anything from an original sentence, although it may change its structure or words, but our system removes extraneous phrases from the extracted sentences.

5 Conclusions and future work

We present a novel sentence reduction system which removes extraneous phrases from sentences that are extracted from an article in text summarization. The deleted phrases can be prepositional phrases, clauses, to-infinitives, or gerunds, and multiple phrases can be removed form a single sentence. The focus of this work is on determining, for a sentence in a particular context, which phrases in the sentence are less important and can be removed. Our system makes intelligent reduction decisions based on multiple sources of knowledge, including syntactic knowledge, context, and probabilities computed from corpus analysis. We also created a corpus consisting of 500 sentences and their reduced forms produced by human professionals, and used this corpus for training and testing the system. The evaluation shows that 81.3% of reduction decisions made by the system agreed with those of humans.

In the future, we would like to integrate our sentence reduction system with extraction-based summarization systems other than the one we have developed, improve the performance of the system further by introducing other sources of knowledge necessary for reduction, and explore other interesting applications of the reduction system.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. IRI 96-19124 and IRI 96-18797. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

John Carroll, Guido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical simplification of English newspaper text to assist aphasic readers. In *Proceedings of AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*, Madison, Wisconsin, July.

R. Chandrasekar, C. Doran, and B. Srinivas. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)*, Copenhagen, Denmark, August.

Simon H. Corston-Oliver and William B. Dolan. 1999. Less is more: Eliminating index terms from subordinate clauses. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics(ACL'99)*, pages 349–356, University of Maryland, Maryland, June.

Gregory Grefenstette. 1998. Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In *Working Notes of AAAI 1998 Spring Symposium on Intelligent Text Summarization*, Stanford University, Standford, California, March.

Hongyan Jing and Kathleen R. McKeown. 1998. Combining multiple, large-scale resources in a reusable lexicon for natural language generation. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, volume *1*, pages 607–613, Université de Montréal, Quebec, Canada, August.

Hongyan Jing and Kathleen R. McKeown. 1999. The decomposition of human-written summary sentences. In *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 129–136, University of Berkeley, CA, August.

Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of NAACL 2000.*

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation.* University of Chicago Press, Chicago, Illinois.

Catherine Macleod and Ralph Grishman, 1995. *COMLEX Syntax Reference Manual.* Proteus Project, New York University.

Michael McCord, 1990. *English Slot Grammar.* IBM.

George A. Miller, 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.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. Cognitive Science Laboratory, Princeton University.