A Machine Learning Approach To Sentence Ordering For Multi-Document Summarization and its Evaluation

Danushka Tarupathi Bollegala
Ishizuka Laboratory,
Department of Electronics and Information,
Faculty of Engineering,
The University of Tokyo.

Abstract

With the increasing popularity of the Internet,millions of texts available in electronic format. Availability of information is no longer a problem. Efficient searching techniques and search engines that can retrieve information for a user specified query exist. However, for a given user specified query, these searching methods yield lots of information. It is not possible for a human reader to simply go through all these texts to find the information. Automatic text summarization is vital in such situations. Apart from summarization there are lots of other applications which presents a computer generated text to the user. For example in Question and Answering (QA) applications, the user enters a question to the system and the system returns the answer to the user as a human readable text. There are also more advanced applications of text generation such as concpet-to-text generation application. In all these natural language generation applications, a text is produced by the computer as the final output. The generated text needs to be coherent for a human to read and easily comprehend. One important step towards generating a coherent text is to order the sentences in a logical manner. In this thesis, I investigate the problem of ordering sentences with special regard to multidocument summarization. Ordering information is a difficult but an important task for natural language generation applications. A wrong order of information does not only make it difficult to understand, but also conveys entirely different idea to the reader.

I propose an algorithm that will learn orderings from a set of human ordered texts. The model consists of a set of ordering experts. When a pair of sentences are presented, each expert gives its preference over one sentence to another. Each expert is associated with a weight. These weights are learnt using a set of human ordered summaries. Each expert's individual preferences are combined to produce a total preference function. Sentences are then ordered to satisfy this total preferences. Finding the optimal ordering that satisfies the total preference function is NP complete. Therefore, a greedy ordering algorithm that approximates the optimal ordering is used. I propose two new metrics, Weighted Kendall Coefficient and Average Continuity, for the evaluation of sentence orderings in addition to the existing evaluation metrics such as Kendall rank correlation coefficient, Spearman correlation coefficient and Continuity metric. My experimental results show that the proposed algorithm outperforms the existing methods in all these evaluation metrics. Finally, I conclude this thesis with a brief explanation of my intended future research in this field.

Contents

List of Figures

Chapter 1

Introduction

Multidocument summarization is the task of generating a summary from a given set of multiple documents. The documents might contain repeating information as well as contradictory information. Therefore, the task of selecting sentences to include in the summary should be done with utmost care. The task of sentence selection has been investigated and many efficient algorithms exist. However, the problem of organizing information for multidocument summarization, so that the generated summary is coherent, has received relatively little attention. While sentence ordering for single document summarization can be determined from the ordering of sentences in the input article, this is not the case for multidocument summarization. Information presented in a wrong order not only conveys a wrong message to the reader but also makes it difficult to comprehend. Barzilay [1] shows experimental evidence to this fact. In this research, I have focused my attention to this problem. I have studied the sentence ordering problem in multidocument summarization of news paper articles. However, ordering information arises in many other fields such as Summarization, Question and Answering systems and concept to text generating systems and the proposed algorithm of this research can be easy extended to these fields.

The task of ordering things has various difficulties attached to it. To begin with, searching for an order which optimizes a given condition, is NP-complete [3]. However, there exist efficient algorithms that approximate optimal orderings as we shall later discuss in this thesis. Things get even difficult when the task is ordering sentences written in natural languages. Each sentence conveys information to the reader and some sentences need a proper context to be comprehensible. There can be multiple possible orderings for a given set of sentences and a "good order" for one user might not necessarily be good for another. However, there are some orderings where a majority of readers shall find easy to follow. Analyzing summaries made by human professionals, Jing [8] shows that the rules behind summarization can be modelled using a hidden markov model.

Previous studies on sentence ordering can be categorized into two categories. One is based on chronological ordering, where sentences are ordered according to

the publication dates of the documents [17]. However, one of the major draw backs in this method is that it is unable to order sentences which have same publication dates. Chronological ordering does not consider the information conveyed by the sentences. For example, two sentences referring to the same topic, but appearing in distinct dates, shall be placed far apart in the summary. However, Barzilay [1] gives experimental evidence to the fact that chronological ordering works satisfactory for the task of sentence ordering in news paper summarization. She proposes an improved chronological ordering algorithm which segments documents into blocks of topics before ordering chronologically.

The other approach to sentence ordering uses a probabilistic language model [11]. Lapata proposes a probabilistic model which calculates conditional probabilities among sentences. Her system assumes that the position of a sentence in the text can be determined only from the sentences that appear before that sentence in the text. We shall discuss this model later in section 2.3. Her system uses a text corpus to calculate the conditional probabilities. This can be viewed as an attempt to model the vast background knowledge humans use when they order sentences. However, the calculations are badly affected by sparsity and the model iteself is too naive for this task. Taking this probabilistic approach one step ahead, Barzilay and Lee [2] proposes a Hidden Markov Model to catch the transition probabilities among sentences.

Apart from these major trends in this field of sentence ordering, Okazaki [19] proposes a method which uses precedence relations to predict sentence order. The ordering of sentences in the original documents convey valuable information and can be used to predict the order among sentences in the extract. I shall explain these precedence relations in detail in section 2.7. The above mentioned previous work in this field gives us several independent ordering techniques to work with. However, these methods tend to be domain specific and the best combination of these methods are unknown. In the proposed algorithm, I integrate all the existing ordering methods and introduce some new methods that can be used to order sentences. The algorithm is machine learnt from a set of human ordered texts as a linear combination of existing methods. We shall discuss this algorithm in next chapter. The trainable nature of the algorithm makes it flexible enought to be used in many other fields which require a sentence ordering step.

This research does not stop by proposing a sentence ordering algorithm. It extends to the evaluation metrics for sentence orderings. Traditionally, researches working with orderings used the famous Kendall's τ coefficients and Spearman's rank correlation coefficients to evaluate their results. Okazaki [19] shows that these pairwise discordant comparison metrics are insufficient to evaluate sentence order. He proposes Continuity metric as a partial solution to this problem. I extend this metric and proposes Average Continuity in section 3.6. Moreover, as an extension to the traditional Kendall's τ coefficient, I propose the Weighted Kendall coefficient as a new evaluation metric for sentence ordering. Chapter 3 contains more details on these evaluation metrics. Finally, I conclude this thesis with a brief explanation of future research I intend to carry out in this field.

Chapter 2

Method

The first step in multidocument summarization (MDS) is extracting a set of sentences from the given document set. This set of sentences could be relevant to a user specified query or it can be a general summary. The second step of MDS will be to order these sentences to make a coherent summary. This chapter describes the algorithm I propose to order a given set of extracted sentences.

The author of a particular document is likely to order the sentences logically to make the document coherent. Therefore, for sentences belonging to a particular document, we can safely retain this original order given by the author. In single document summarization this simple ordering strategy would be sufficient to order all the extracted sentences as they belong to a one single document. However, we cannot apply this simple method to MDS because the sentences belong to different documents. Such documents may have been written by various authors on various dates.

To decide the order among such sentences, I use six independent ordering strategies which I shall call "experts" in this thesis . When presented with a pair of sentences, each of these experts gives its preferences for a one sentence over another as a value in the range of $[0, 1]$. In my model each expert generates a pair-wise preference defined as follows,

$$\mathrm{PREF}_e(u, v, Q) \in [0, 1]\,. \tag{2.1}$$

Here,u, v are two sentences that we want to decide the order; Q is the set of sentences which we have already ordered. The expert e returns its preference of u to v. A value greater than 0.5 indicates that the expert prefers u to v, whereas a value smaller than 0.5 shows that the experts prefers v to u. When the expert is undecided about its preferences, it will return 0.5. For each expert in our model we define a preference function as in 2.1. Then, the linear weighed sum of these individual preference functions is taken as the total preference by the set of experts. Cohen [3] proposes an elegant learning model that works with preference functions and I adopt this learning model to the task. We use the on-line weight allocation algorithm proposed by Cohen [3] based on the Hedge algorithm to learn

the weights associated with each expert's preference function. Then we use the greedy algorithm proposed by Cohen [3] to get an ordering which approximately satisfies the total preference.The learning algorithm and the ordering algorithm is discussed in detail in sections 2.13 and 2.12.

2.1 The Set of Experts

We designed the following six ranking experts.

1. Chronological Expert.
2. Probabilistic Expert.
3. Heuristic Expert.
4. Topical Relevance Expert.
5. Precedent Expert.
6. Succedent Expert.

Next, I shall explain the concepts behind these experts and the definitions of their preference functions.

2.2 Chronological expert

Chronological expert emulates conventional chronological ordering [17, 12] which arranges sentences according to the dates on which the documents were published. It preserves the appearance order for sentences in the same document. We define a preference function for the expert as follows:

$$\mathrm{PREF}_{chro}(u, v, Q) = \begin{cases} 1 & T(u) < T(v) \\ 1 & [D(u) = D(v)] \wedge [N(u) < N(v)] \\ 0.5 & [T(u) = T(v)] \wedge [D(u) \neq D(v)] \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

Therein: $T(u)$ is the publication date of sentence u; $D(u)$ presents the unique identifier of the document to which sentence u belongs; $N(u)$ denotes the line number of sentence u in the original document. Chronological expert gives 1 (preference) to the newly published sentence over the old and to the prior over the posterior in the same article. Chronological expert returns 0.5 (undecided) when comparing two sentences which are not in the same article but have the same publication date. Experimental results by Barzilay [1] implies that chronological ordering works well in ordering news summaries. News articles are generally published in the order as the events occurred. Therefore, ordering sentences according to the publication date works satisfactorily in news summaries. However, in certain cases a

news article may refer to an even which occurred previously. In such situations a naive chronological ordering will not work. Besides, when the articles are published on the same date, chronological ordering is not possible.

2.3 Probabilistic expert

Lapata [11] proposes a probabilistic model to predict sentence order. Her model assumes that the position of a sentence in the summary depends only upon the sentences which precede it in the summary. For example let us consider a summary T which contains sentences $S_1, \ldots, S_n$ in that order. The probability $P(T)$ of observing this order is given by,

$$P(T) = \prod_{i=1}^{n} P(S_n|S_1, \ldots, S_{n-i}) \tag{2.3}$$

Using the bi-gram approximation she further reduces this probability to,

$$P(T) = \prod_{i=1}^{n} P(S_i|S_{i-1}) \tag{2.4}$$

Since, exact occurrences of two sentences in a corpus is rare, she breaks each sentence into features and takes the vector product of features.

$$P(S_i|S_{i-1}) = \prod_{(a_{<i,j>}, a_{<i-1,k>}) \in S_i \times S_{i-1}} P(a_{<i,j>}, a_{<i-1,k>}) \tag{2.5}$$

Feature conditional probabilities can be calculated using frequency counts of features as follows.

$$P(a_{<i,j>}|a_{<i-1,k>}) = \frac{f(a_{<i,j>}, a_{<i-1,k>})}{\sum_{a_{<i,j>}} f(a_{<i,j>}, a_{<i-1,k>})} \tag{2.6}$$

Lapata [11] uses Nouns,Verbs and dependency structures as features. Where as in our expert we implemented only Nouns and Verbs as features. Once these conditional probabilities are calculated, we can define the preference function for the probabilistic expert as follows,

$$\mathrm{PREF}_{prob}(u, v) = \frac{1 + P(v|u) - P(u|v)}{2}. \tag{2.7}$$

where u, v are two sentences in the extract. When u is preferred to v, i.e. $P(v|u) > P(v|u)$, according to definition 2.7 a preference value greater than 0.5 is returned. If v is preferred to u, i.e. $P(v|u) < P(u|v)$, we have a preference value smaller than 0.5. When $P(v|u) = P(u|v)$, the expert is undecided and it gives the value 0.5.

2.4 Back-off smoothing

We performed back-off smoothing [10] on the frequency counts in equation 2.6 as these values were sparse. In back-off smoothing, a portion of probabilities of frequently occurring terms are transferred to sparsely occurring terms. For simplicity, I shall write w_1^m to denote the n-gram of length m, $w_1, w_2, \ldots, w_m$. $C(w_1^m)$ is the count of w_1^m in the corpus. Then the smoothed conditional probability $P_s(w_i|w_{i-n+1}^{i-1})$ of seeing w_i after $w_{i-n+1}, \ldots, w_{i-1}$ is given recursively as follows, [14]

$$P_s(w_i|w_{i-n+1}^{i-1}) = \begin{cases} (1 - d_{w_{i-n+1}^{i-1}}) \frac{C(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})} & C(w_{i-n+1}^{i}) > k \\ \alpha_{w_{i-n+1}^{i-1}} P_s(w_i|w_{i-n+2} \ldots w_{i-1}) & \text{otherwise} \end{cases} . \quad (2.8)$$

In definition 2.8 the first condition applies to terms w_{i-n+1}^{i} which exceeds a certain value k of counts. When using this model to smooth probabilities in sparse data k is set to zero. Therefore, for terms appearing one or more times in the corpus the conditional probabilities are reduced by a factor of $0 < d_{w_{i-n+1}^{i-1}} < 1$. Setting this value to 0, does not reserve any probabilities to be assigned for sparse data. These reserved probabilities are then assigned to the unseen n-grams as shown in the second condition in definition 2.8. The factor $\alpha_{w_{i-n+1}^{i-1}}$ is selected as in equation 2.9 so that the total probability remains a constant.

$$\alpha_{w_{i-n+1}^{i-1}} = \frac{1 - \sum_{C(w_{i-n+1}^{i})>k} (1 - d_{w_{i-n+1}^{i-1}}) \frac{C(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})}}{1 - \sum_{C(w_{i-n+1}^{i})<k} P_s(w_i|w_{i-n+2})} \quad (2.9)$$

In the probabilistic expert we need to consider only bi-grams of words which appear in consecutive sentences. Therefore, the recursive formula in 2.8 considers only bi-grams and unigrams of words. The only remaining parameter in formula 2.8 is $d_{w_{i-n+1}^{i-1}}$. Katz [10] proposes a method based on Turing's estimate to decide the value of $d_{w_{i-n+1}^{i-1}}$). Before, explaining this method we shall redefine $d_{w_{i-n+1}^{i-1}}$ as D_r, where $r = C(w_{i-n+1}^{i-1})$. For higher r values we shall not discount the probabilities because higher frequencies are reliable.

$$D_r = 1 \text{ for } r > R \quad (2.10)$$

In my experiments I took frequencies over five to be reliable. Therefore, in my experiments I took $R = 5$. When, n_r is the number of words (n-grams of words) which occurred exactly r times in the corpus, Turing's estimate P_T for a probability of a word (n-grams of words) which occurred in the sample r times is,

$$P_T = \frac{r^*}{N}. \quad (2.11)$$

where,

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \tag{2.12}$$

We shall select D_r such that the contribution of probabilities yielded by this method is proportional to the contributions by the Turing [7] estimate. Taking the proportional coefficient to be μ, we can write this relation as,

$$(1 - D_r) = \mu(1 - \frac{r^*}{r}). \tag{2.13}$$

The unique solution to the equation 2.13 is,

$$D_r = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \text{ for } 1 \leq r \leq k \tag{2.14}$$

2.5 Heuristic Expert

There are language dependent heuristic rules that helps human readers to decide the order among sentences. Connection phrases such as although,however,therefore convey such information about sentence order. As a naive rule, we can consider an expert which shall always order sentences having connection phrases as the first word after sentences which do not have connection phrases as the first word. We can design a preference function as following to express this rule.

$$PREF_{sem}(u,v) = \begin{cases} 1 & \neg Con(u) \wedge Con(v) \\ 0 & Con(u) \wedge \neg Con(v) \\ 0.5 & else \end{cases} . \tag{2.15}$$

Here,the boolean function $Con(u)$ returns true only if the sentence u contains a connection phrase as the first word in it. In Japanese language the “wa” and “ga” particles match the introduction of a subject to its references. As another naive heuristic rule, if the noun phrases preceding these particles in two sentences match, we can always order the sentence containing the particle wa with the noun phrase before the sentence containing the particle ga. In fact, there are many such language dependent heuristic ordering rules that can be used to decide the sentence order. However, the reliability of such rules are not clear. In order to keep the proposed algorithm language independent, I only implemented the rule shown in 2.15.

2.6 Topical Relevance Expert

In a summary, sentences conveying information to a particular topic tend to appear in groups. Similarity of a sentence with the sentences that appear before it, can be taken as a measure of this topical relevance. Grouping sentences which are topically related, improves coherence of the summary. Motivated by this fact, we designed an expert which prefers sentences which are topically more relevant than

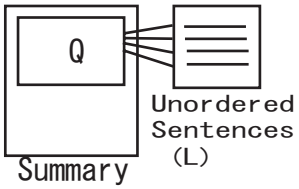

Figure 2.1: Topical relevance expert

the others to the ordered summary. Let us consider the situation illustrated in figure 2.1. Here, the the block Q represents the sentences that have already been ordered in the summary and block L represents the sentences that are not yet been ordered in the extract. For each sentence $l \in L$ we calculate its topical relevance $\mathrm{topic}(l)$ as follows.

$$\mathrm{topic}(l) = \max_{q \in Q} \mathrm{sim}(l, q) \tag{2.16}$$

$$\mathrm{sim}(l, q) = \frac{\vec{l}.\vec{q}}{\sqrt{|\vec{l}||\vec{q}|}} \tag{2.17}$$

Here, $\mathrm{sim}(l, q)$ is the cosine similarity of term vectors $\vec{l}$ and $\vec{q}$, and it is given by the equation 2.17. When Q is empty we are unable to evaluate equation 2.16 and the expert is undecided. It returns the value 0.5. The preference function for this expert can be written as follows,

$$\mathrm{PREF}_{topic}(u, v, Q) = \begin{cases} 0.5 & [Q = \Phi] \vee [\mathrm{topic}(u) = \mathrm{topic}(v)] \\ 1 & [Q \neq \Phi] \wedge [\mathrm{topic}(u) > \mathrm{topic}(v)] \\ 0 & \text{otherwise} \end{cases} . \tag{2.18}$$

2.7 Precedent expert

Okazaki [19] proposes precedence relations as a method to improve the chronological ordering of sentences. He considers the information stated in the documents where sentences were extracted from, to judge the order. If the sentences preceding the extracted sentence in the original document match well with the so far ordered summary, it is suitable to be ordered next in the summary. We designed an expert which takes into consideration these precedence relations in ordering sentences.
Consider the situation shown in figure 2.2, where we have already ordered a block Q of sentences in the summary. We have to select a sentence l out of the remaining extract L. Let us assume that the sentence l belongs to document D,in which l is preceded by the block P of sentences. We find the maximum similarity between all pairs of sentences taken from blocks P and Q. For each sentence $l \in L$ we calculate its precedence $\mathrm{pre}(l)$ as follows,

$$\mathrm{pre}(l) = \max_{p \in P, q \in Q} \mathrm{sim}(p, q) \tag{2.19}$$

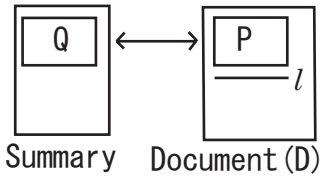

Figure 2.2: Precedent expert

We take the cosine similarity defined in (2.17) for sim(p,q) in equation 2.19. However, when Q is empty we are unable to evaluate equation 2.19 and the expert is undecided. It then returns the value 0.5. The preference function for this expert can be written as follows,

$$\mathrm{PREF}_{pre}(u, v, Q) = \begin{cases} 0.5 & [Q = \Phi] \vee [\mathrm{pre}(u) = \mathrm{pre}(v)] \\ 1 & [Q \neq \Phi] \wedge [\mathrm{pre}(u) > \mathrm{pre}(v)] \\ 0 & \text{otherwise} \end{cases} . \tag{2.20}$$

2.8 Succedent expert

In the sentence extraction stage of summarization, there can be multiple candidate sentences. Due to the limited length of the summary and to avoid repetitions, the sentence extraction algorithm might deliberately leave behind an important sentence. Most sentence extraction algorithms do not consider whether the selected sentences go in a good order. However, when ordering sentences, such left behind candidates might turn out to be more suitable than the extracted sentences. One option would be to replace the extracted sentences by these candidates at the ordering phase. However, the ordering algorithm does not have any information regarding the process of sentence extraction and does not know these left behind candidates. Therefore, we cannot replace any extracted sentences. However, we can design an expert which will match the portions that appear after the extracted sentences in their documents with the remaining extract sentences to identify such candidates. Having identified these candidates, we can order the extracted sentences according to the order between candidates.

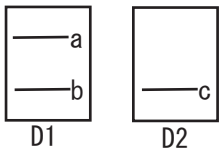

Figure 2.3: Reasoning behind succedence

For example consider the situation depicted in figure 2.3, where we want to decide the order between sentences b and c which belong to the documents $D1$ and

$D2$ respectively. Let us assume that these two documents were published on the same day. In this case,the chronological expert will be unable to decide the order between b and c. However, if there exists a sentence a in document $D1$ which is similar to the lastly ordered sentence r in the partially ordered summary Q, then we can order b before c. In fact, the sentence extraction algorithm might have deliberately left out sentence a because it selected sentence r. We design an expert which takes these succedence relations into consideration when ordering. Let us consider the situation depicted in figure 2.4. r is the last sentence we have ordered in the summary so far and it belongs to document D. In document D there are some sentences K, appearing after R. The succedence of sentence $l \in L$ is defined as follows,

$$\mathrm{succ}(l) = \max_{k \in K} \mathrm{sim}(l, k) \tag{2.21}$$

We take cosine similarity of term vectors for sim(l,k) as defined in (2.17). However, when Q is empty we are unable to evaluate equation 2.21 and the expert is undecided. It then returns the value 0.5. The preference function for this expert can be written as follows,

$$\mathrm{PREF}_{succ}(u, v, Q) = \begin{cases} 0.5 & [Q = \Phi] \vee [u_{succ} = v_{succ}] \\ 1 & [Q \neq \Phi] \wedge [u_{succ} > v_{succ}] \\ 0 & \text{otherwise} \end{cases} . \tag{2.22}$$

2.9 Total Preference Function

Total preference function is calculated as the weighted sum of individual expert's preference function. Taking E to be the set of experts, I define the total preference function as follows,

$$\mathrm{PREF}_{total}(u, v, Q) = \sum_{e \in E} w_e \mathrm{PREF}_e(u, v, Q). \tag{2.23}$$

Weights w_e are normalized such that their sum is 1.

$$\sum_{e \in E} w_e = 1 \tag{2.24}$$

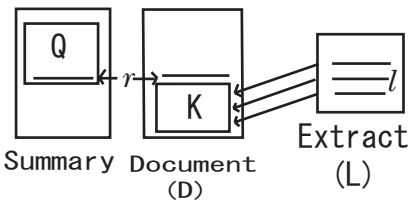

Figure 2.4: Succedent expert

2.10 Measuring the Quality of an Ordering

Once the total preference function is computed as in 2.23, the next step is to find an ordering which will satisfy this total preference function. However, we must first quantify the notion of agreement between a preference function PREF and an ordering ρ. As a one possible notion we could define the degree of agreement $\mathrm{AGREE}(\mathrm{PREF}_{total}, \rho)$ between PREF and ρ as follows.

$$\mathrm{AGREE}(\mathrm{PREF}_{total}, \rho) = \sum_{u,v;\rho(u)>\rho(v)} \mathrm{PREF}_{total}(u, v, Q) \tag{2.25}$$

Therein: u and v are two extract sentences; the ordering function ρ orders u before v. As it is obvious from definition 2.25 if the ordering function ρ perfectly agrees with PREF_{total}, then for each $\mathrm{PREF}_{total}(u, v, Q)$ inside the summation we shall get a preference value of 1. Hence, maximize $\mathrm{AGREE}(\mathrm{PREF}_{total}, \rho)$.

2.11 Complexity of Ordering

Having a notion of agreement defined as in 2.25, we are in a position to analyze the complexity of the problem of finding an optimal ordering function which maximizes $\mathrm{AGREE}(\mathrm{PREF}_{total}, \rho)$ for a given total preference function PREF_{total}. However, Cohen [3] proves this problem to be NP-complete when the number of sentences to be ordered exceeds three. He proves this by reducing from CYCLIC-ORDERING [5]. The original theorem which he states in his paper can be stated in the following form,

Theorem 1 *The following decision problem is NP-complete for any set S with* $|S| \geq 3$*:*
Input: A rational number κ*; a set* X*; a collection of* N *ordering functions* $f_i : X \to S$*; and a preference function* PREF *defined as,*

$$\mathrm{PREF}(u, v) = \sum_{i=1}^{N} w_i R_{f_i}(u, v) \tag{2.26}$$

where $\mathbf{w} = (\mathbf{w_1}, \ldots, \mathbf{w_N})$ *is a rational weight vector in* $[0, 1]^N$ *with* $\sum_{i=1}^{N} w_i = 1$*. Then the problem of finding a total order* ρ *such that* $\mathrm{AGREE}(\rho, \mathrm{PREF}) \geq \kappa$ *is NP-complete.*

The ranking functions R_{f_i} in Cohen's theorem do not take into consideration the so far ordered set of elements Q. Therefore, thorem 1 cannot be directly applied to our ordering problem. However, the introduction of Q in ranking functions (in our case each experts preference functions) makes the issue of ordering even harder because the preference functions cannot be calculated in advance. Preference functions themselves are affected by the ordering algorithm and this makes any theoretical analysis of complexity difficult. However, the problem at hand is clearly NP since a

nondeterministic algorithm can guess a total order and check the weighted number of arguments in polynomial time.

2.12 Ordering Algorithm

As shown in previous section, finding the optimal order for a given total preference is difficult. For a set of N sentences there are $N!$ possible orderings. Searching all these possible orderings to find the optimal ordering becomes computationally impossible when N gets larger. However, Cohen [3] proposes a greedy algorithm that approximates the optimal ordering. He gives a theoretical proof to the fact that this greedy algorithm approximates the optimal ordering to a factor of half. Once the unordered extract X and total preference (equation 2.23) are given, this greedy algorithm can be used to generate an approximately optimal ordering function $\hat{\rho}$.

However, there are some fundamental differences between the algorithm proposed by Cohen [3] and the one used by us. Our preference function has Q, the so far ordered summary, as a parameter in it. Therefore, the value of preference function changes while ordering.

Greedy Ordering Algorithm

let $V = X$
for each $v \in V$ **do**
    $\pi(v) = \sum_{u \in V} \mathrm{PREF}(v, u, Q) - \sum_{u \in V} \mathrm{PREF}(u, v, Q)$
while V is non-empty **do**
    let $t = \arg\max_{u \in V} \pi(u)$
    let $\hat{\rho}(t) = |V|$
    $V = V - \{t\}$
    for each $v \in V$ **do**
        $\pi(v) = \pi(v) + \mathrm{PREF}(t, u) - \mathrm{PREF}(v, t)$
endwhile

2.13 Learning Algorithm

Upto now, we have designed a set of experts which are able to give their preference among sentences and an ordering algorithm that approximates these preferences. However, the preference value for a particular pair of sentences may vary from expert to expert and we cannot know in advance which experts are correct. Besides an expert which gave a wrong decision for a particular sentence pair may give correct disicion for another. Therefore, we need a method to evaluate the reliability of each expert's decision. These reliabilities can be learnt from a given set of human orderings as weights allocated to each expert. Cohen [3] proposes an on-line weight allocation algorithm to learn the weights associated with each expert in equation 2.23 based on the Hedge algorithm [4]. We shall explain this algorithm in regard

to our model of six experts.

Weight Learning Algorithm
Rate of learning $\beta \in [0,1]$, initial weight vector $\vec{w}^1 \in [0,1]^6$, s.t. $\sum_{e \in E} \vec{w}_e^1 = 1$.

Do for $t = 1, 2, \ldots, T$ where T is the number of training examples.

1. Get X^t; the set of sentences to be ordered.
2. Compute a total order $\hat{\rho}^t$ which approximates,

$$\mathrm{PREF}_{total}^t(u, v, Q) = \sum_{e \in E} \mathrm{PREF}_e^t(u, v, Q).$$

   We used the greedy ordering algorithm described in section 2.12 to get $\hat{\rho}^t$.
3. Order X^t using $\hat{\rho}^t$.
4. Get the human ordered set F^t of X^t. Calculate the loss for each expert.

$$\mathrm{Loss}(\mathrm{PREF}_e^t, F^t) = 1 - \frac{1}{|F|} \sum_{(u,v) \in F} \mathrm{PREF}_e^t(u, v, Q) \tag{2.27}$$

5. Set the new weight vector,

$$w_e^{t+1} = \frac{w_e^t \beta^{\mathrm{Loss}(\mathrm{PREF}_e^t, F^t)}}{Z_t} \tag{2.28}$$

   where, Z_t is a normalization constant, chosen so that, $\sum_{e \in E} w_e^{t+1} = 1$

In my experiments I set $\beta = 0.5$ and $w_i^1 = 1/5$. To explain equation 2.27 let us assume that sentence u comes before sentence v in the human ordered summary. Then the expert must return the value 1 for PREF(u,v,Q). However, if the expert returns any value less than 1, then the difference is taken as the loss. We do this for all such sentence pairs in F as follows,

$$\mathrm{Loss}(\mathrm{PREF}_e^t, F^t) = \frac{1}{|F|} \sum_{(u,v) \in F} [1 - \mathrm{PREF}_e^t(u, v, Q)]. \tag{2.29}$$

Equation 2.29 quickly reduces to equation 2.27. For a summary of length N we have $N(N-1)/2$ such pairs. Since this loss is taken to the power of β, a value smaller than 1, the new weight of the expert gets changed according to the loss as in equation 2.28.

Schapire [4] gives theoretical bounds of Hedge learning algorithm that can be directly used to infer bounds for the above mentioned learning algorithm. Their results imply that the cumulative loss of the total preference function PREF_{total} will not be much worse than that of the best ranking expert [3].

Chapter 3

Evaluation

Each ordering generated by the proposed algorithm in chapter 2 is compared against a reference order made by a human. Therefore, the task of evaluation is a task of comparing two sets of rankings. In statistics, Kendall's τ coefficient and Spearman's rank correlation coefficient are used to compare two sets of rankings. These are non-parametric statistical methods and can be performed on the ranks without considering the actual content (in this research sentences).

3.1 Kendall Correlation Coefficient

In order to explain the rank correlation coefficients, let us consider two permutations π,σ. Such permutations can be generated for a set of sentences by using integers to denote the ranks of the sentences. For example, permutations π and σ can be,

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 3 & 5 & 1 \end{pmatrix} \tag{3.1}$$

and

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 5 & 2 & 1 & 3 \end{pmatrix}. \tag{3.2}$$

The permutations have one to one correspondence with a given sentence ordering. For example for two orderings A and B of a summary, permutation π denotes that the extract sentence S_1 has a rank 2, S_2 has a rank 4 and so on. Where as in ordering B, expressed by permutation σ the extract sentence S_1 was given a rank 4, S_2 a rank 5 and so on.

Using permutations π and σ Kendall correlation coefficient τ_k can be defined as,

$$\tau_k = \frac{1}{n(n-1)/2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{sgn}(\pi(j) - \pi(i)).\mathrm{sgn}(\sigma(j) - \sigma(i)). \tag{3.3}$$

Therein: n represents the number of sentences; and $\mathrm{sgn}(x)$ is the sign function defined as follows,

$$\mathrm{sgn}(x) = \begin{cases} 1 & x > 0 \\ -1 & \text{otherwise} \end{cases}. \tag{3.4}$$

As it is clear from the definition 3.3, Kendall coefficient depends only upon the number of discordant pairs. When the number of concordant pairs exceeds the number of discordant pairs, τ_k is positive. If the number of discordant pairs exceeds the number of concordant pairs then it becomes negative. When discordants and concordants are equal in number, Kendall coefficient is zero. Since the total number of pairs that can be created from a set of n elements with replacement is $n(n-1)/2$, the coefficient can be normalized so that it becomes a metric in range $[-1, 1]$ irrespective of the number of elements n. Kendall coefficient has been used by Lapata [11] and Barzilay et al. [2] to evaluate sentence orderings. If a human had to correct an ordering by hand she will have to interchange all the discordant pairs. Therefore, Kendall coefficient can be seen as the effort she will have to put into when correcting an ordering. Larger the Kendall coefficient, lesser the effort to rectify the ordering.

3.2 Spearman Rank Correlation

For a given two orderings, we can compute the Spearman rank correlation coefficient τ_s using the corresponding permutations.

$$\tau_s = 1 - \frac{6}{n(n+1)(n-1)} \sum_{i=1}^{n} (\pi(i) - \sigma(i))^2 \tag{3.5}$$

Unlike Kendall coefficient which depends only upon the number of discordant pairs, Spearman coefficient considers the relative distance between all pairs.

3.3 Continuity Metric

A summary is usually read from top to bottom in one dimension. The reader brings together continuous sentences in a text and interpret their meaning. Therefore, if the summary has a lot of continuous blocks of texts, it helps the reader to easily comprehend the summary. Okazaki [19] proposes a metric to grasp the continuity of a summary. A summary which can be read continuously is better than a one with lots of discontinuities. Using the permutations π,σ of the two orderings, he defines the continuity metric τ_c as,

$$\tau_c = \frac{1}{n} \sum_{i=1}^{n} \mathrm{equals}(\pi\sigma^{-1}(i), \pi\sigma^{-1}(i-1)+1). \tag{3.6}$$

Therein: $\pi(0) = \sigma(0) = 0$;

$$\mathrm{equals}(x, y) = \begin{cases} 1 & x = y \\ 0 & \text{otherwise} \end{cases}. \tag{3.7}$$

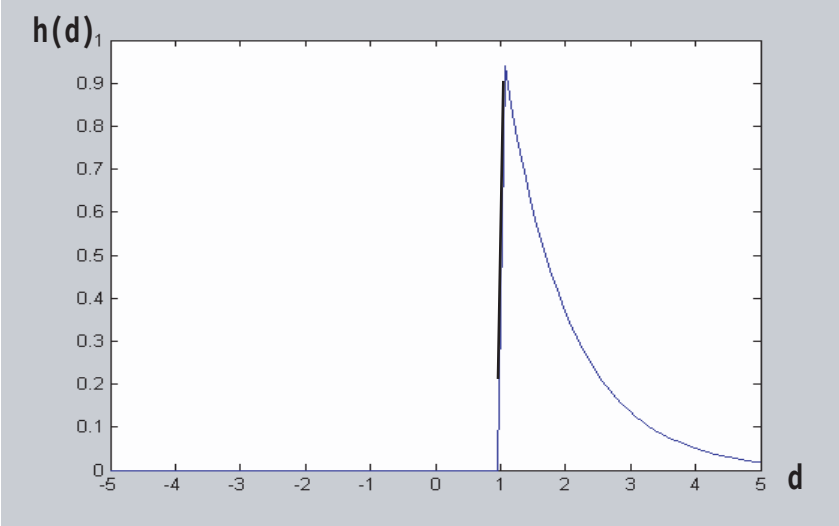

Figure 3.1: Weighting function

3.4 Proposed Metrics

Kendall, Spearman correlation coefficients and continuity metric are insufficient to evaluate a sentence ordering properly. Kendall coefficient neglects the relative distance between discordant pairs where as Spearman coefficient simply takes the squared difference between discordant pairs as well as concordant pairs. Continuity metric uses a simple bi-gram model when calculating sentence continuity and does not directly take into consideration the higher order sentence n-grams. To overcome these inheritant flaws of these metrics I propose two new metrics; Weighted Kendall coefficient and Average Continuity. Experimental results in chapter 4 show that these new metrics express the subtle differences between different sentence orderings better than the existing metrics.

3.5 Weighted Kendall Coefficient

Kendall correlation coefficient, τ_k was defined in 3.3 using permutations. It can also be stated more simply using the number of discordant pairs D as,

$$\tau_k = 1 - \frac{2D}{{}^nC_2}. \tag{3.8}$$

However, one major drawback of this metric when evaluating sentence orderings is that, it does not take into consideration the relative distance between the discordant pairs. On the other hand, Spearman correlation coefficient, τ_s, considers the relative distance between all pairs and does not distinguish between discordant and concordant pairs. None of the above mentioned metrics consider the position of

the sentences in the summary. However, when reading a text a human reader is likely to be more sensitive to a closer discordant pair than a pair far apart. Therefore, a closer discordant pair is more likely to harm the readability of the summary compared to a far apart discordant pair. In order to reflect these difference in our metric, we use an exponentially decreasing weight function as follows.

$$h(d) = \begin{cases} \exp(1-d) & d \geq 1 \\ 0 & \text{else} \end{cases} \tag{3.9}$$

This weighting function can be expressed graphically as in figure 3.5. Going by the traditional Kendall's τ coefficient we defined our weighted Kendall coefficient as following, so that it becomes a metric in $[1, -1]$ range.

$$\tau_w = 1 - \frac{2\sum_d h(d)}{\sum_{i=1}^{n} h(i)} \tag{3.10}$$

3.6 Average Continuity Metric

Both Kendall's τ coefficient and the weighted kendall coefficient measure discordants between ranks. However, in the case of summaries, we need a metric which expresses the continuity of the sentences. A summary which can be read continuously is better compared to a one that cannot. If the ordered extract contains most of the sentence blocks of the reference summary then we can safely assume that it is more readable and coherent than a one that does not. Sentence n-gram counts of continuous sentences give a rough idea of this kind of continuity.

For a summary of length N there are $(N - n + 1)$ possible sentence n-grams. Therefore, we can define a precision P_n of continuity length n as,

$$P_n = \frac{\text{number of matched n-grams}}{N - n + 1}. \tag{3.11}$$

As we shall explain later in figure 4.1, P_n decreases in an exponential-like curve with n. Therefore, we take the logarithmic average of P_n and define it as Average Continuity (AC).

$$\text{Average Continuity} = \exp(\sum_{n=2}^{4} \log(P_n)) \tag{3.12}$$

We add a small quantity α to both numerator and denominator of P_n in equation 3.11 so that the logarithm will not diverge when n-grams count is zero. We used $\alpha = 0.01$ in our evaluations. Experimental results showed that taking n-grams up to four gave better results because the n-grams tend to be sparse for larger n values. BLEU(BiLingual Evaluation Understudy) proposed by Papineni [20] for the task of evaluating machine translations has an analogical form to our average continuity. In BLEU, a machine translation is compared against multiple reference translations and precision values are calculated using word n-grams. BLEU is then defined as the logarithmic average of these precision values.

Chapter 4

Results and Discussion

Preparing 30 sets of summaries using the TSC-3 extract data set; I ordered each extract by five methods: Random Ordering (RO); Probabilistic Ordering (PO); Chronological Ordering (CO); Learned Ordering (LO); and Human-made Ordering (HO). CO is the ordering we get when we order sentences only using the chronological expert. We get PO when we order only using our probabilistic expert. Out of the 30 human-made orderings we used 20 to train our algorithm and order the rest of the 10 sets. LO represents these orderings. We measure closeness of respective orderings to the human-made one and evaluate each method using the metrics described in section 3. Experimental results are shown in table 4.1.

Figure 4.1 shows precision P_n, defined by equation 3.11, against the length of sentence continuity n for each of the ordering methods. Table 4.3 contains the actual precision values upon which figure 4.1 is drawn. The weights learned by our algorithm are shown in table 4.2.

From table 4.1 it is clear that the learned algorithm outperforms the chronological order. Although, when compared using the traditional evaluation metrics like Kendall and Spearman coefficients, the margin that the learned algorithm outperforms chronological ordering is quite narrow, the proposed metrics show a clear difference. Continuity metric considers only sentence bi-grams whereas Average Continuity considers sentence n-grams up to four. Therefore, we can think Average Continuity to be an extended version of Okazaki's [19] Continuity metric. I performed an ANOVA (Analysis of Variance) test on the experimental results and under the reliability factor $\alpha = 0.05$ the experimental results are statistically significant.

Figure 4.1 shows that for all sentence n-grams considered in calculating Average Continuity the trained algorithm has higher precisions. Figure 4.1 shows that precision falls in an exponential-like curve with the length of sentence n-gram. When the length of continuity increases the number of matching sentence n-grams decrease. On the other hand for two positive integers $2 < p < q$, p-grams contain the number of q-grams too by definition. This justifies the equation of Average Continuity 3.12 which takes the logarithmic average of precision instead of the

Table 4.1: Comparison with Human Ordering

Metric	Spearman	Kendall	Continuity	Weighted Kendall	Average Continuity
Random Order	-0.267	-0.160	-0.118	-0.003	0.024
Probabilistic Order	0.058	-0.019	-0.093	0.003	0.019
Chronological Order	0.774	0.735	0.629	0.688	0.511
Learned Order	0.783	0.746	0.706	0.717	0.546
Human Order	1.000	1.000	1.000	1.000	1.000

Table 4.2: Weights learned

Expert	Weights
Chronological	0.327947
Probabilistic	0.000039
Heuristic	0.015062
Topical relevance	0.016287
Precedent	0.196562
Succedent	0.444102

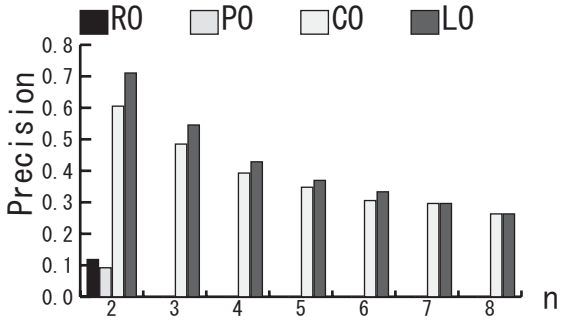

Figure 4.1: n-gram precision

Table 4.3: n-gram precision

n	Random Ordering	Probabilistic Ordering	Chronological Ordering	Learned Ordering
2	0.11	0.09	0.60	0.71
3	0	0	0.48	0.54
4	0	0	0.39	0.43
5	0	0	0.34	0.36

arithmetic mean. A similar curve can be observed for the BLEU metric proposed for evaluating machine translations by Papineni [20]. Mathematically, Calculating the arithmetic mean of logarithms for a quantity is equivalent to calculating the geometric mean of that quantity. Although, the Average Continuity 3.12 is defined so that the logarithms will not diverge when the n-grams were zero, considering too longer continuous lengths will diminish the subtle differences in non-sparse n-grams. Therefore, in my experiments I considered n-grams up to four.

The learned weights are shown in table 4.2. These weights can be considered as a rough estimate of the influence made by each of these experts when deciding the total order. The comparatively higher values of weights for chronological expert and succedent expert indicate that they are considered highly when deciding the order. In fact, the results in table 4.2 tell us that probabilistic expert had almost no influence in total preference. I consider this due to the too simple model by Lapata [11]. As I described in section 2.3, this model assumes that the sentence order can be predicted just by using the direct succedents. However, in texts written in natural languages the context of a sentence may span more than one sentence, and considering just the direct precedent sentence is not sufficient to perform an accurate calculation of sentence conditional probabilities.

We can also conclude from table 4.2 that the heuristic expert had almost no effect in the final learned ordering algorithm. This is mainly due to two reasons. Firstly, in most of the cases the heuristic expert is undecided because neither of the sentences have connection phrases and returns the preference value 0.5 as defined in section 2.5. Therefore, the loss for the heuristic expert accumulates and its weight gets reduced. Secondly, the judgments of the expert are not very reliable as it does not consider the objects connected by the connection phrases. An improvement would be to match the objects connected and give the decision. These experimental results show that when adopting language specific rules, one should have a whole set of rules rather than a single rule which is rarely triggred. However, these weights can vary upon the summary corpus and the sentence extraction algorithm. Once these weights are learnt for a specific sentence extracting algorithm, they can be used to order sentences from this algorithm.

Finally, I shall show one example of ordering by the learned algorithm. The summary is about the earthquake in Papua New Guinea in 1998. Figure 4.2 is the randomly ordered extract for the summary. Learned algorithm takes the set of sentences in figure 4.2 as input and orders them as in figure 4.3.

The major advantage of this machine learning method is that it is independent of the sentence extraction algorithm and once trained, can be used easily for different summary copora. The trainable nature of our algorithm lets us to use it even in other text structuring tasks, like QA and concept to text generation. We can improve the current set of experts and introduce new experts which are more suitable to these fields. The model is expandable and easily adaptable to these myriad tasks.

1. ただ、こうした後付けの推定はできても、事前に予測することは難しかった。
2. このため、東大地震研究所の都司嘉宣助教授は「日本周辺で同じタイプの地震が起きたら、大きな被害が予想される。基準の見直しが必要ではないか」と指摘する。
3. 津波の高さは７～１０メートルに達していたとみられる。
4. 今回の地震はパプアニューギニア北西部の沖合約１００キロを震源とし、地震の規模を示すマグニチュード（Ｍ）は７･0と推測される。
5. パプアニューギニア北方沖で１７日に起きた地震に伴う津波の被害は１９日になって拡大し、救援活動を指揮するため現地入りしたスケート首相は同日「約６００人が死亡したと聞いている。死者数はさらに増えるだろう」と語った。
6. 津波発生の仕組みはまだ分からないことが多いためだ。
7. 現地からの報道によると、大きな被害を受けたのはパプアニューギニア北西部のウェストセピク州アイタペの西方に位置する７村で、アロップ村（人口２５００人）など三つの村が完全に波にのみ込まれた。
8. 東大地震研究所の菊地正幸教授はアラスカやハワイなど世界１１地点で観測された地震波データを解析し、長さ４０キロにわたる断層が垂直に２メートルずれたと推測した。
9. 今回の地震は同州沖約百キロの海底下十五キロで発生した。
10. このため、断層が水平にずれる従来の地震に比べ、海水面への影響が大きく、津波も異常に大きくなったとみられる。

Figure 4.2: Randomly Ordered

1. パプアニューギニア北方沖で１７日に起きた地震に伴う津波の被害は１９日になって拡大し、救援活動を指揮するため現地入りしたスケート首相は同日「約６００人が死亡したと聞いている。死者数はさらに増えるだろう」と語った。
2. 現地からの報道によると、大きな被害を受けたのはパプアニューギニア北西部のウェストセピク州アイタペの西方に位置する７村で、アロップ村（人口２５００人）など三つの村が完全に波にのみ込まれた。
3. 津波の高さは７～１０メートルに達していたとみられる。
4. 今回の地震はパプアニューギニア北西部の沖合約１００キロを震源とし、地震の規模を示すマグニチュード（Ｍ）は７・０と推測される。
5. 東大地震研究所の菊地正幸教授はアラスカやハワイなど世界１１地点で観測された地震波データを解析し、長さ４０キロにわたる断層が垂直に２メートルずれたと推測した。
6. このため、断層が水平にずれる従来の地震に比べ、海水面への影響が大きく、津波も異常に大きくなったとみられる。
7. 今回の地震は同州沖約百キロの海底下十五キロで発生した。
8. ただ、こうした後付けの推定はできても、事前に予測することは難しかった。
9. 津波発生の仕組みはまだ分からないことが多いためだ。
10. このため、東大地震研究所の都司嘉宣助教授は「日本周辺で同じタイプの地震が起きたら、大きな被害が予想される。基準の見直しが必要ではないか」と指摘する。

Figure 4.3: Ordered by the Learned Algorithm

Chapter 5

Future Work

5.1 Hierarchical Clustering based approach to sentence ordering

There are fundamental limits in this pair-wise approach to ordering. A sentence once ordered by the greedy algorithm cannot be changed even if we come to know later that there is a more suitable position for it. However, when humans order sentences, they change their minds later on orderings they have done and reorder. To order a pair of sentences we have to know the context of sentences. One approach to this problem would be to use a bottom-up ordering model. Such a model will first order the sentences in clusters such that each cluster is locally coherent. These clusters are then merged and ordered to produce a globally coherent summary. Marcu [15] argues that global coherence can be achieved if the constraints for local coherence are satisfied.

For example we could first order the clusters of sentences that we are completely certain of their ordering and leave behind the ones which we are not sure of. Then we can combine these clusters in various ways and check whether we can find a place to insert the yet unordered sentences. We could carry on this process in a hierarchical manner until no more sentences are left. We need to define the criteria that we use to order sentences within the clusters as well as the criteria to order the clusters themselves. For example consider the situation illustrated in figure 5.1.

In figure 5.1 sentences A and B are ordered first. Here, the notation $A > B$ means that the sentence A precedes sentence B. Then sentences C and D are ordered. Then we are left with three clusters of sentences; two clusters each of two sentences $[A > B]$, $[C > D]$ and a single cluster with one sentence $[E]$. Out of these three clusters let us assume that our criteria of cluster ordering prefers blocks $[A, B]$ and $[C, D]$ to be ordered and then order the block $[E]$ with the resulting block.

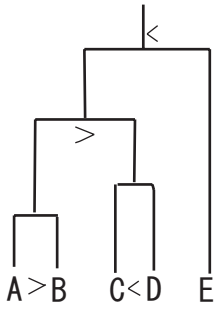

Figure 5.1: Hierarchical Clustering based Sentence Ordering

5.2 Sentence ordering as a problem of searching

For a given set of N sentences, there are $N!$ possible total orderings. As we saw in section 2.11 finding the optimal ordering among these $N!$ orderings is NP-complete. We can think of the task of ordering a set of sentences coherently, as a problem of finding the best ordering among these $N!$ orderings. In this model of sentence ordering as a problem of searching, there are two main factors we must consider; the criterion of optimization and the searching algorithm. We can explicitly define what is a coherent summary. We could give out a set of conditions a coherent summary should satisfy [13, 16]. Alternatively, we could prepare or collect a set of human ordered summaries and use a learning model to extract the rules behind a coherent summary. In this thesis I followed the latter approach and took total preference as the optimization criterion. However, much research is needed to identify the properties of a coherent summary. Once we have a criterion to optimize, we need a searching algorithm. In this thesis as well as Lapata [11] use the simple greedy algorithm proposed by Cohen [3]. Other alternatives too must be considered. One such method would be to use Genetic Algorithms(GA)[6]. GAs are known to find good sub-optimal solutions when the domain contains multiple solutions. In sentence orderings there exists many possible orderings. A GA could be used to find some of these possibilities. GAs have been successfully used in text planning tasks([18]). In short summaries we could even try a brute force or a best first search, as computations in factorial order is not so daunting for shorter summaries. Once the weights are learnt, the algorithm proposed in this thesis generates an ordering in less than one second. Therefore, trying out all the possibilities for shorter summaries is not computationally impossible.

5.3 Analysis of the effect of Q on the learning model.

Although we used the learning model proposed by Cohen [3], there are some fundamental differences between that model and our model. Firstly, the preference

function used by Cohen has just two variables and does not take into consideration Q, the order created so far. The introduction of Q in equation 2.1 disturbs some of the elegant properties of the model to an extent. For example, the greedy algorithm is no longer unaffected by previous decisions. Some expert's decisions affect others and they are not independent. This is specially true for topical relevance, precedent and succedent experts. Theses experts need others to guide them when Q is empty. Without properly analyzing the effect that Q makes upon the greedy ordering, we cannot be certain whether the properties of the original algorithm holds. Theoretical implications of this version of the greedy algorithm must be investigated.

5.4 Linear segmentation and Sentence Ordering

Figure 5.2: Order X

Figure 5.3: Order Y

Clustering sentences which belong to the same topic, increases the readability of a summary. For example consider the case illustrated in figure 5.2. Therein: the five sentences belong to two topics, A and B. In ordering X, shown in figure 5.2, the topic changes a total of four times, where as in ordering Y, shown in figure 5.3, topic changes just once. A user will likely to find the summary in figure 5.3 to be more readable than the summary in figure 5.2. Kan et al, [9] proposes a method to linearly segment a given text according to the topics covered by that text. Using this technique we can segment the summary in its topics. For two orderings of a given text, the ordering which yields the lower number of topic segments is more likely to be a comprehensive summary.

Acknowledgments

There are many people to thank for their support and encouragement, without whom this thesis would not have been possible. Firstly, I would like to thank Professor Mitsuru Ishizuka for his kind and wise advice given to me to fulfill this research. His guidance and experience was invaluable to make this research a success. Secondly, I would like to thank Mr. Naoaki Okazaki for providing me with the background knowledge in the summarization field. He was kind enough not only to share his vast knowledge in text summarization and sentence ordering, but also to read and comment on my papers. Without his progressive comments, this research would not be a reality. Lastly, but not least I would like to thank all the other members in Ishizuka lab for their kind co-operation and patience.

Bibliography

[1] Regina Barzilay, Noemie Elhadad, and Kathleen McKeown. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55, 2002.

[2] Regina Barzilay and Lillian Lee. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Proceedings of the Main Conference*, pages 113–120, 2004.

[3] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[4] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[5] Z. Galil and N. Megido. Cyclic ordering is np-complete. *Theoretical Computer Science*, 5:179–182, 1977.

[6] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company.Inc, The University of Alabama, 1989.

[7] I.J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, pages 40:237–264, 1953.

[8] Hongyan Jing. Using hidden markov modelling to decompose human-written summaries. *Computational Linguistics*, 28(4):527–543, 2002.

[9] M. Kan, J. Klavans, and K. McKeown. Linear segmentation and segment significance. *Proceedings of 6th Workshop on Very Large Corpora (WVLC-98)*, pages 197–205, 1998.

[10] Salva M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics Speech and Signal Processing*, 33(3):400–401, 1987.

[11] Mirella Lapata. Probabilistic text structuring: Experiments with sentence ordering. *Proceedings of the annual meeting of ACL, 2003.*, pages 545–552, 2003.

[12] C.Y. Lin and E. Hovy. Neats:a multidocument summarizer. *Proceedings of the Document Understanding Workshop(DUC)*, 2001.

[13] William C. Mann and Sandra A. Thompson. Rhetorical structure theory: Towards a functional theory of text organization. *Text*, 8(3):243–281, 1988.

[14] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts London, England, 2 edition, 2002.

[15] Daniel Marcu. From local to global coherence: A bottom-up approach to text planning. In *Proceedings of the 14th National Conference on Artificial Intelligence*, pages 629–635, Providence, Rhode Island, 1997.

[16] Daniel Marcu. The rhetorical parsing of unrestricted texts: A surface-based approach. *Association for Computational Linguistics*, pages 395–448, 2000.

[17] Kathleen McKeown, Judith Klavans, Vasileios Hatzivassiloglou, Regina Barzilay, and Eleazar Eskin. Towards multidocument summarization by reformulation: Progress and prospects. *AAAI/IAAI*, pages 453–460, 1999.

[18] Chris Mellish, Knott Alistair, Oberlander Jon, and O'Donnell Mick. Experiments using stochastic search for text planning. In *Proceedings of the 9th International Workshop on Natural Language Generation*, pages 98–107, Ontario, Canada, 1998.

[19] Naoaki Okazaki, Yutaka Matsuo, and Mitsuru Ishizuka. An integrated summarization system with sentence ordering using precedence relation. *ACM-TALIP, to appear in 2005.*, 2005.

[20] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu:a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, 2002.