Approach to scalable statistical text summarization

Nguyen Minh Le and Susumu Horiguchi

June 27, 2002

IS-RR-2002-16

School of information science

Japan Advance Instiute of Science and Technology, Hokuriku

Ashahidai 1-1, Tatsunokuchi-machi

Nomi-gun, Ishikawa-ken, 923-12, JAPAN

nguyenml@jaist.ac.jp hori@jaist.ac.jp

ISSN 0918-7553

Approach to scalable statistical text summarization

Nguyen Minh Le and Susumu Horiguchi
Japan Advanced Institute and Technology
Asahidai 1-1, Tatsunokuchi-machi
Nomi-gun, Ishikawa-ken, 923-1292, JAPAN
{nguyenml; hori}@jaist.ac.jp

Abstract
This paper analysts some aspect of applying statistical machine translation method to summary text document. After considering this text summarization as statistical machine translation system we apply translation model to test on a corpus consist of long sentence and its reduced, which was produced from our decomposition program. We also revised several translation model and language model to discover a fix model for text summarization. After using training algorithm to cope with corpus, the most important in the remainder is complexity of decoder, for this reason, we will discuss the hierarchy of parallel algorithm for both training data and decoder process is essential.

1. Introduction

Much of the reported work has been concerned with producing is how to define the important part in a document. There are four approaches when working in text summarization problem: statistical, knowledge based, shallow understanding based, and hybrid approaches. One of these approaches, statistical is fast, robust, scalable, in this report we would proposed a new method based on statistical machine translation for many statistical models, most of which have already been successfully used in other language application, are applied to text summarization.

One of earliest work [4] is use Vector Space Model in Information Retrieval to measure the similarities between paragraphs and find important paragraph(s), another work is [3], which formulates summarization as a statistical classification problem – dividing sentences into two categories: important and unimportant – and deploys the Baysesian classification for summarization. Knight and Marcus [1] apply statistical machine translation techniques for compressing sentences. Similar techniques for summarizing web pages are reported in [2].

Recently, Jing and Kewon [9] use Hidden Markov Model to decompose human-written summary sentences. They proposed a cut and paste text summarization system, it is process for demonstrating a professional summarizer do. They also proposed two phases, reduce, and combining sentence. The limitation of this method that is needs a lot of knowledge database.

In this paper, we would propose a method can be used available corpus of document and its summary as training data to learn. Our method also applies statistical machine translation similar [1], but we use IBM model [7]. The training data we obtain was generated from our decomposition program. The main reason to use statistical machine translation for text summarization is the similar between

translation and text summarization. Both systems are rewriting system. Moreover, in text summarization system the source language and target language is the same, thus if a specification model formulated it the sentence in original text document then summary text can be applied this model. Therefore, we will find a fix model to reduce the complexity of both training process and decoding. The first we focus on finding a fixed model for sentence compressing. This work related to [1]. We will revise several language model and translation model in SMT for compressing sentence problem. With this propose we assume the translation model in SMT can be applied for text summarization in reduction problem, then revising them to discover the best appropriate model. After that, we would extent our fixed model for summarization a full text document.

The remainder of this paper as follows: The first is reviewing of statistical machine translation, continuing with applying these technical in text summarization and then is presented with the application of statistical machine translation techniques for text summarization. Next two parts is experiment and conclusion.

2. Statistical machine translation

Statistical machine translation can be described as a noisy channel model. See finger 1

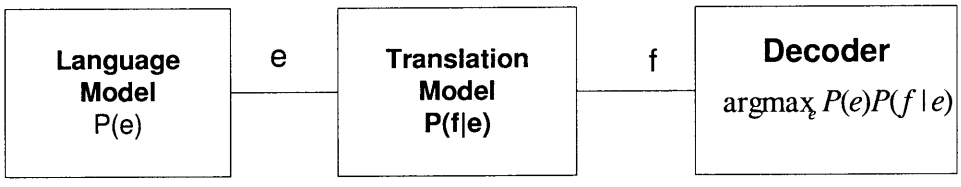

Figure 1: The noisy channel model in machine translation. The language model generates an English sentences e. The translation Model transmits e as French sentence f. The decoder finds English sentence e^ which is most likely to have given rise to f.

In this figure introduced the noisy channel in order to translate from English sentences into French sentences.

English sentences is called $e^i = e_1e_2...e_i$ was transformed into French sentence denoted $f^j = f_1f_2...f_j$. Then it sends the French f to decoder. The decoder then determiner the English sentence e^ that f is most likely to have arisen from.

There are three components in statistical machine translation that is: *language model*; *translation model*, and *decoder*.

2.1 Language model

This is model let us know the probability P (e) of the English sentences. We can build language model by many ways such as n-gram, probabilistic grammar, Hidden Markov model [5]; link grammar [9]. We review some aspect of these languages model bellow:

2.1.1 N-gram

This model allows defying the n^{th} word by n-1 words before. The task of predicting the next word can be stated as attempting to estimate the probability function P:

$P(w_n | w_{1,...}w_{n-1})$. This probability function let us define the word n based on n-1 word before. For n=2, 3, 4, and these alternatives are usually referred to as a *bigram*, *trigram*, and four-*gram* model, respectively. This model can be shown more detail in well – known book [5].

2.1.2 Probabilistic context free grammar

Context free grammar (CFG) did not cover natural language processing; sometime a sentence can be parsed into several syntax trees.

For example a sentence: *I saw a man in the park with a telescope.*
We don't know this sentence mean "I saw a man by the telescope and the man was in the park" or "I saw a man in the park and this man ware a telescope ". Probabilistic context free grammar gives a score for each syntax tree. This score was considered as probabilistic of syntax tree, and calculate it by using the probabilistic of each rule in context free grammar.

A PCFG consist of:

+ A set of terminal $\{w_k\}$; k=1,2,..,V
+ A set of non-terminals, $\{N^i\}$,i=1,2...n
+ A designated start symbol, N^1
+ A set of rules, $\{N^i \rightarrow \zeta^j\}$, (where ζ^j is a sequence of terminals and non-terminals)

A corresponding set of probabilities on rule such that:
$\forall i \ \Sigma_j P(N^i \rightarrow \zeta^j)=1$

Example:

S-> NP VP (1.0)	NP-> NP PP 0.4
PP-> prep NP (1.0)	NP -> astronomers 0.1
VP-> verb NP (0.7)	NP -> ears 0.18
VP-> VP PP (0.3)	NP ->saw 0.04
prep -> with (1.0)	NP-> stars 0.18
verb -> saw (1.0)	NP -> telescope 0.1

(Note that NP: noun phrase, PP: prep phrase, VP: verb phrase; S: sentence)

The sentence: Astronomer *saw stars with ears*

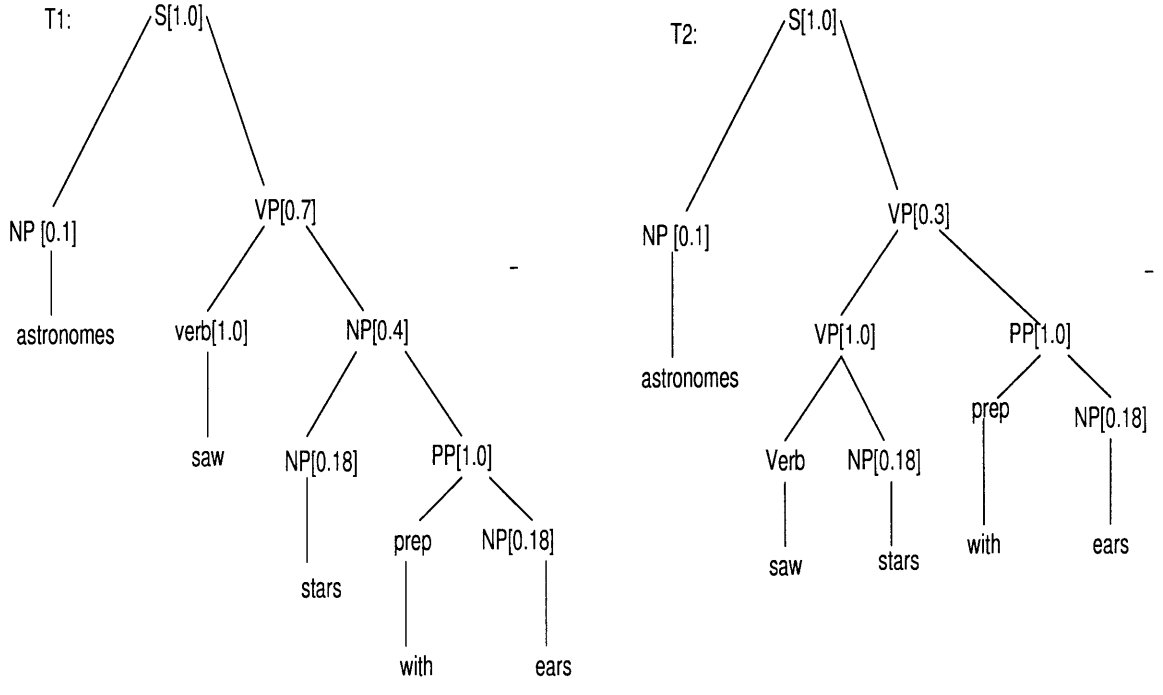

Figure 2: an probabilistic tree example

P (T1) =1.0 x 0.1 x0.7 x 1.0 x0.4 x0.18 x 1.0 x 1.0 x 0.18=0.0009072
P (T2) =1.0 x 0.1 x 0.3 x 0.7 x 1.0 x0.18 x 1.0 x 1.0 x 0.18=0.0006804

Therefore, the probabilistic to parse this sentence is:
P(S) =P (T1) +P (T2)
We also know the best syntax tree is T1.
PCFG was applied in several domains in NLP; especially in text summarization it is used as a preprocessing step to parse both sentence and its compression for generating a training data. This work was described in [1].

2.1.3 Link grammar

In this part we will introduce some major aspect of link grammar, it is described more detail in [11]. A Link grammar consists of a set of words (the terminal symbols of the grammar), each of which has a linking requirement. A sequence of words is a sentence of the language defined by the grammar if there exists a way O draw arcs among the words so as to satisfy the following conditions:

Planarity: The links do not cross (when drawn above the words)

Connectivity: The links suffice to connect all the words of the sequence together.

Satisfaction: The links satisfy the linking requirements of each word in the sequence. The linking requirement of each word is contained in a dictionary. Let see the example bellow:

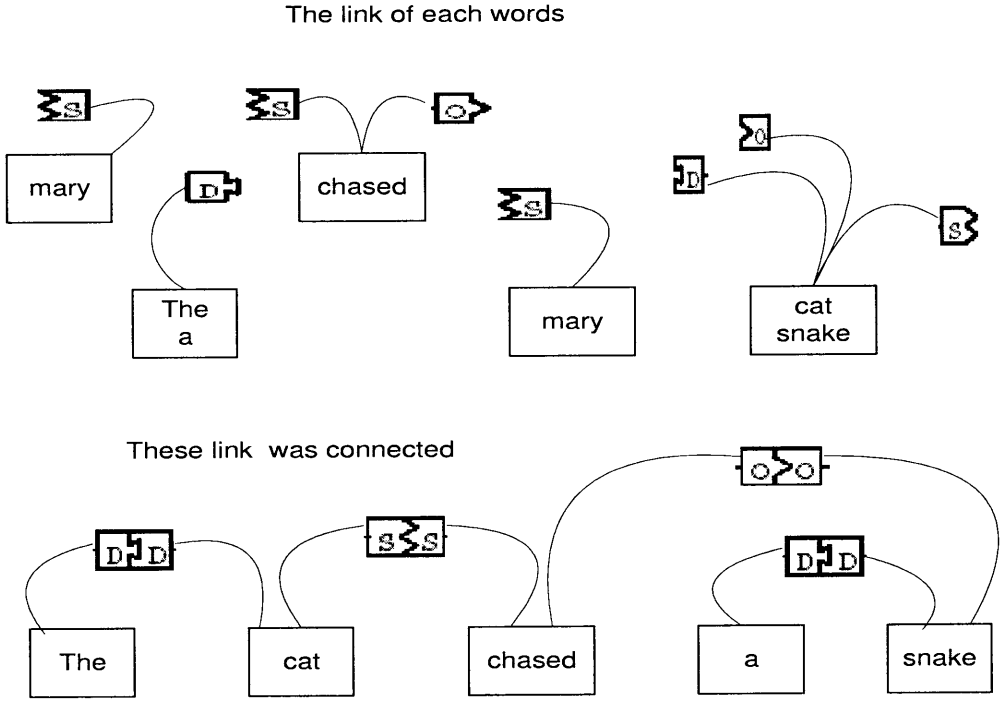

Figure 3: An example of parsing a sentence using Link Gramar

In the figure above, each of the intricately shaped labeled boxes in s connector. A connector is satisfied by "plugging it into "a compatible connector. If the mating end of a connector is drawn facing to the right, then its mate must be to its right facing to the left. Exactly one of the connector attached to a given black dot must be satisfied.

Therefore, "cat" is needed a D connector to its left, and either an O connector to its left or a S connector to its right.
Thus, we have a diagram that shows how the linking requirements are satisfied in the sentence "The cat chased a snake" (see figure 2)
A set of links that prove that a sequence of words is in the language of a link grammar is called a linkage. They used a succinct, computer-readable notation for expressing the dictionary of linking requirements. The following dictionary encodes the linking requirement of the previous example.

Word	Formula
a the	D+
snake cat	D- & (O- or S+)
Mary	O- or S+
ran	S-
chased	S- & O+

Table 1: example word dictionary

We can see that the linking requirement for each word is expressed as a formula involving the operator ***&,*** and ***or***, ***parentheses***, and connector names (D, O, S). The suffix on a connector name indicates the direction (relative to the word being defined) in which the matching connector (if any) must lie. The & of two formula is satisfied by satisfying both the formulas. The "or" of two formulas requires that exactly one of its formulas be satisfied. The order of the arguments of an "&" operator is significant. Therefore, with the dictionary like this, given a sentence we have to define the set of link that satisfied all of link requirement. The complexity of parsing sentence with link grammar is O (n^3); it is the same with other parsing like EARLY or CYK for context free grammar.
It can be seen more detail in [12]. Link grammar is one of new language model, it haven't used in text summarization problem. In this report, we would like to compare it with other method in parsing sentence.

2.2 Translation model

As mention above, translation model define the probabilistic to translate from source sentence e into target sentence f. Assume that $e=e_1e_2...e_m$; and $f=f_1f_2...f_n$, we need to define e_i map with f_j. This is called alignment from e_i to f_j and all of that alignment possible let us know the alignment from e to f.
Note that, e_i can be word as IBM-1 or phrases (set of words) as template model [22].
We now sketch the structure of the six models, and we will focus on introducing the simplest model; model 1 (IBM-1).

+ In IBM-1 all alignment has the same probability,

+ IBM-2 uses a zero-order alignment model $P(a_j | j, I,J)$ where different alignment positions are independent from each other.

+ The HMM user a fist-order model $p(a_j | a_{j-1})$ where the alignment position aj depends on the previous alignment position a_{j-1}.

+ In IBM-3 we have an (inverted) zero-order alignment model $p(j | a_j, I,J)$ with an additional fertility model $p(\Phi|e)$ which describes the number of words Φ aligned to and English word e.

+ In IBM-4 we have a (inverted) first order alignment

This is simplest translation model based on word alignment model p(j|j') and a fertility model $p(\Phi|e)$.

+ The models IBM-3 and IBM-4 are different as they waste probability mass on non-strings. IBM-5 is a reformulation of IBM-4 with a suitably refined alignment model in order to avoid deficiency.

The main differences of these models lie in the alignment model (which may zero-order or first), in the existence of an explicit fertility model and whether the model is deficient or not.

+ One other translation model is based on syntax tree [14], this model aim to solve in case two language is too different in syntactic.

As mention above, we will describer IBM-1 model in detail, so it is simplest model on all model listed above. First of all, we consider the notation bellow:

$$P(\text{fle}) = \frac{1}{z}\sum_{a_1=0}^{l}\cdots\sum_{a_m=0}^{l}\prod_{j=1}^{m}P(f_j \mid e_{a_j})$$

This is notation of Brown et al [7]: e is the English sentence; l is the length of e in words; f is the French sentence; m is the length off;

fj is word j in f; a_j is the position in e that f_j is aligned with; e a_j is the word in e that fj is aligned with; P ($w_f | w_e$) is the translation probability, the probability that we will see w_f in the French sentences given that we see we in the English sentence; and Z is normalization constant.

The basic idea of this formula is fairly straightforward. The m sums $\sum_{a_1=0}^{l}\cdots\sum_{a_m=0}^{l}$ sum over all possible alignments of French words to word. The meaning of a_i=0 for an a_j is that word j in the French sentence is aligned with the empty element, that is, it has no translation. One English word can be aligned with multiple French words, but one French word is aligned with at most one English word.

To make clear we consider an example as bellow:

English sentence: Tom Loves Mary

French sentence: Jean aime Marie

P(Jean aime Marie/ Tom loves Mary) for the alignment (Jean, Jonh), (aime, loves), and (Marie, Mary), then we multiply the three corresponding translation probabilities.

P(Jean | John) x P(aime | loves) x P(Marie | Mary) .

To summary, we computer P(f | e) by summing the probabilities of all alignments. For each alignment, we make two simplifying assumptions: Each French word is generated by exactly one English word; and the generation of each French word is independent of the generation of all other French words in the sentence.

2.3 Decoder

Based on the observation that we have P(f) is fixed value:

With a specification sentence e we would find the sentence f that's most satisfied with it. We have:

$$\arg\max_e P(\text{fle}) = \arg\max_e \frac{P(e)P(f \mid e)}{P(f)} = \arg\max_e P(e)P(f \mid e)$$

The problem is that the search space is infinite, so we need a heuristic search algorithm. One possibility is to use stack search.[19] The basic idea is that we build an English sentence incrementally. We keep a stack of partial translation hypothesis. At each point, we extend these hypotheses with a small of words and alignments and then prune the stack to its previous size by discarding the least likely extended hypothesis. This algorithm is not guaranteed to find the best translation, but can be implemented efficiently. If the source and target language are constrained to have the same word order, then the linear Viterbi algorithm [10] can be applied. If re-ordering is limited to rotations around nodes in a binary tree, then optimal decoding can be

carried out by a high–polynomial algorithm [21]. For arbitrary word reordering, the decoding problem is NP-complete [20]. Fortunately, in text summarization the target language and source language is the same, the order word in source language and target language is the same too if we remarks some phrases as component that its order changing. Almost remainder parts are constrained with the order of original text. (They used to cut and paste from document to produce the summary). Almost the decoder was described above will cost a long time when handing with a long sentence. This suggest us to implement on parallel computer

3. Text summarization based on statistical machine translation

As mention above, in machine translation says pair sentences are English and French, the length of English is sentence is affect with the time of decoder because of too much alignment possible. On these other hand, the data training is almost including a long text document and its summary, as several available newspapers on the Internet. To handle this problem, firstly, we use decompose [9] program to find the alignment between summary sentence with the part of original text document; we denote it as *link-part*. Secondly, we use the set of pair of summary sentence and its link-part as training data, then we apply model of statistical machine translation to find the best summarization alignment. The remainder of this part can be organizing as follows: the first we describe the decomposition program that applied for defining the summary sentence with its *link-part*, continuing with the fixed model for summarization.

3.1 Decomposition

Using Hidden Markov Model for decomposition text document and its summary was described in [9]. This work reports that 81% of summary sentence produced by human are based on cutting and pasting. This work also uses decomposing process as decoder from original document and its summary. The decoder aim to answer three questions:

i) Do cutting and pasting text from the original document construct this summary sentence?
ii) If so, what components in the sentence come from the original document?
iii) And where in the document do the components come from?

In my work, we use decomposing process aim to define the summary sentence with original *link-part*. One of problem for decomposition task that describes in [9] is the formulation. In this work, they use the position of sentence and position of word within that sentence to formulate. This is difficult to define the boundary of sentences in the document; we restrict the problem by considering the term of the position sentence as position of paragraph. Moreover, this work didn't define the position of the summary word that is not in original document. In this report I will present a method to define the position of sequence word in which it may or may not in original document.

3.1.1 HMM Solution

An input summary sentence can be represented as a word sequence: $(I_1,.. I_N)$, where I_1 is the first word of the sentence and I_N is the last word.

Each word within summary sentence maybe occurs in the original document, or exits some word is its similar.

We find in the original document a set of word that has distance semantic to the word Ij greater than const. We denote the position of each word within original document by the sentence position and the word position within the sentence. Multiple occurrence of a word in the document can be represented by a set of word positions:

$\{(s_1,w_1), (s_2, w_2),\ldots,(s_n, w_n)\}$, thus, multiple similarity occurrence of a word in the document was represented as:
$\{(s_1,w_1,d_1), (s_2,w_2,d_2),\ldots(s_n,w_n,d_n)\}$ with di is the ***distance semantic*** [15,16,17,18] between this word and a word in the document. The value of di is always greater than **const.** Using this notation, the decomposition problem can be formulated as follows:
Given a word sequence $\{I_1,..,I_n\}$ *and the sequence feature* $\{(s_1, w_1,d_1),(s_2,w_2,d_2),\ldots(s_n,w_n,d_n)\}$ *for each word in the sequence, determine the most likely document feature for each word in the sequence.*
One problem emerges in here that is how to define the distance between two words. To define distance di between two words (Ii, Ij) we used an online WordNet the same method was described [9]. We assume that the distance semantic between two words is 1 if they are the same. We can see figure 1, there are total 1,936 features possible. We have to define the most likely feature should be. To define the most likely document feature we extend the HMM from [9] based on some heuristic rules and apply Viterbi algorithm [9] for that HMM.
The probabilistic transition function between two sates in our HMM as bellow:
$PROP((s_i,w_i,d_i))|(s_j,w_j,d_j))=PROP((s_i,w_i)|(s_j,w_j))$x di x dj
The position transition function $PROP((s_j,w_j)|(s_j,w_i))$ was defined in [1], this function based on some heuristic rule as follow:

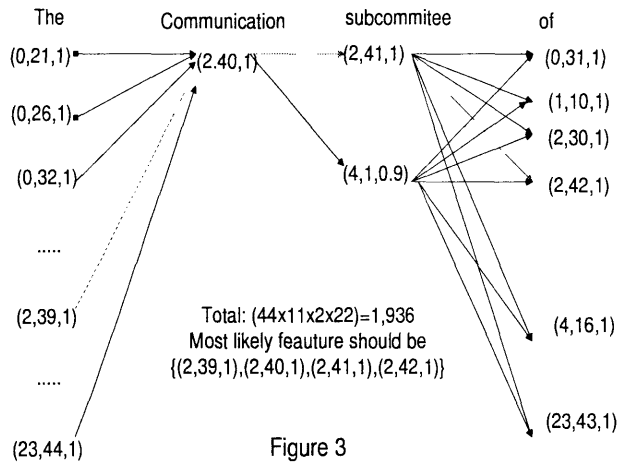

Figure 3

Assume that: the position of Ii (S1, W1) and Ii+1 (S2, W2) are two adjacent word in a summary sentence and Ii is before Ii+1.

1. If ((S1=S2) and (W1=W2-1), then P(Ii+1|Ii) is assigned the maximal value P_1.
2. if ((S1=S2)) and (W1<W2-1), then P(Ii+1|Ii) assigned the 2nd highest value P_2.
3. if ((S1=S2) and (W1>W2), then P(Ii+1|Ii) is assigned as the 3rd highest value P_3.
4. if ((S2-CONST <S1<S2), then P(Ii+1|Ii) is assigned the 4th highest value P_4.
5. If (S2<S1<S2+CONST), then P(Ii+1|Ii) is assigned the 5th highest value P_5.
6. if (|S2-S1|>=CONST), then P(Ii+1|Ii) is assigned a small value P_6.

With using Bigram model and probabilistic transition function was defined like that, we have:

$$\Pr ob(I_1, I_2,..I_n) = \prod_{i=1}^{n-1} \Pr ob(I_i \mid I_{i+1})$$

Therefore, we may apply Viterbi algorithm, one dynamic programming method to determine the most likely document feature for each word in the sequence.

3.1.2 Checking Position to find maximize likely sequence feature

In previous section we were formulated the decomposition as HMM solution to determine the most likely document feature for each word in the sequence through Viterbi algorithm. In this section, we modify the Viterbi algorithm by adding the checking position function for each two repetition words. The modifier Viterbi algorithm can be shown as bellow:

Viterbi algorithm:

Input: Given word sequence (W_1,...W_n), and a list of possible document features for each word Wi
Li: (S_1,W_1,D_1),...(Sim_i, Wm_j,Di) and bigram probabilities, find the most likely sequence of features.

Initialization Step:

– For I =1 to m1 do
•SORE(i,1)=1;
•BACK(i,1)=0;
•Iteration step:
– For j =2 to N
• *If L_j is not empty*
For I =1 to Mj SCORE(i,j)= Max (SCORE(k,j-1) x PROP(Wj=(Sji, Wji,Dji)|Wj-1=(Sj-1k,Wj-1k,Dj-1k)
with (k=1,Mj-1)
BACKP(i,j)=index of k that gave the max
Else For i=1 to Mj-1 SCORE(i,j)=SEQSCORE(i,j-1)
BACK(i,j)=j-1
•Sequence Identification step:
•P(N)=i that maximizes SCORE(i,n)
•For i=n-1 to 1 P(i)=BACK(P(i+1),i+1)
•For i=1 to n if Lj is empty P(i)=empty

Checking position function will be added in finding the maximal of formulate:

Max (SCORE (k, j-1) x PROP (Wj= (Sji, Wji,Dji) |
W j-1= (Sj-1k, Wj-1k, Dj-1k) with (k = 1, Mj-1)

To make more easy, we define T[k,j] is sequence of feature that has probabilistic is SCORE[k,j]. The propose of this function aim to prevent the feature (Sji,Wji,Dji) appear in T[k,j-1]. With adding the checking position function Viterbi algorithm could have avoided a wrong case: When the feature (Sji, Wji, Dji) appear in T[k,j-1] and SCORE(k,j-1) x PROP(Wj=(Sji, Wji, Dji)|Wj-1=(Sj-1k, Wj-1k, Dj-1k) is maximal..

3.2 Summarization based on Statistical machine translation

3.2.1 Language model

To select an efficient for text summarization we have revised some language model that includes: n-gram, probabilistic grammar, and probabilistic link - grammar.

The language model we used in our system is both probabilistic grammar and link grammar. After using decomposition process to align summary sentences with some sentences in the original document, we collect all of it to build a corpus for training.

Probabilistic grammar was published in [1], they used this model to handle the compression problem. The reason for revising an efficient model to handle text summarization is the language model can be used for both original language and its summary.

3.2.2 *Summarization model*

To find out the novel model for text summarization we first apply some model available in statistical machine translation such as IBM-1 – IBM-5 [3]. In this paper, we assume that these models in machine translation are fixed for summarization. The related work that apply noisy channel model (in here summarization model) can be referred in [1]. In this work, they parse both input sentence and its compression sentence with probabilistic context free grammar (PCFG), and then the training data are set of *<syntax tree, syntax tree>*. To make clear their method, we can obverse an example in the figure bellow:

In this figure, the syntax tree (t) is compressed to syntax tree (s_1). They observe the different between two tree to find out a relevant probabilistic.

For example, in a figure bellow: The relevant probabilistic is: P (A-> CBD| A->CD)

To produce the new node for its compression, we generate 2^{n-1} sub-tree as:

P(A->CBD|A->CB); P(A->CBD|A->CD) ; P(A->CBD|A->BD); and find the most fix its sub-tree (that is A->CD) based on the relevant probabilistic from corpus.

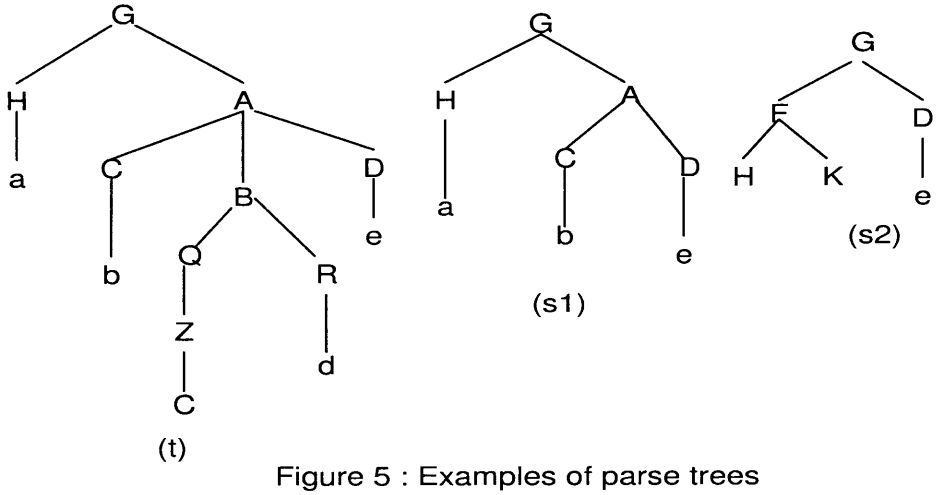

Figure 5 : Examples of parse trees

In work before there is no comparison of using several model translations applying text summarization problem. In my work we will revised several model in SMT for text summarization. We assume that:

The translation model (IBM) model in statistical machine translation [7] will be fixed with text summarization problem; here is compression from a long sentence into short sentence.

With this assumption, using EM algorithm will generate the translation model.

EM-algorithm to train data from corpus

In this part we apply EM-algorithm [12] for training data from corpus. One of the main aspects of EM algorithm is it run too slow, it only find the local maximum and depend on starting point. After modeling the translation model as IBM-1, IBM-2 . [7] The training process for text summarization can be carried out as mention in statistical machine translation.

Decoder

The decoder process in text summarization problem is the same text translation.

Almost decoder applied for text translation is based on some improve searching algorithm like: *A*, stack – based* and *dynamic programming*. In text translation they have proven that this is slow process. Even In some worse case it is equal to a travel selling man problem. A decoder process is kernel of text summarization after training (training outstand this process), therefore the text summarization is scalable if it is parallelism.

4. Experiment

We collect training data from DUC2001, it consists of 60 sets of documents, and each set of document includes 10 documents and its summary. The average length of original document is 90 sentences and 1100 words. Moreover, we had a set of 1067 sentence pairs extracted from Ziff-Davis corpus, a collection of newspaper articles announcing computer product. We have done with four kind of test that consists of: *Spilt sentence*; this task split original document into set of sentences; *revised language model; Decomposition; training.*

4.1 Split sentence

Our splitter aim to split original document and its summary give the accuracy correct 95%. To evaluate the accuracy of this splitter is very simple; we only select randomly 10 documents in set of training data, and then split it manually. We compare each sentence in splintering by our program with each sentence that was colleted by hand.

4.2 Revised Language model

In this part we not only revised the accuracy of some language model with training data but also compare the running time of these parser while building language model.

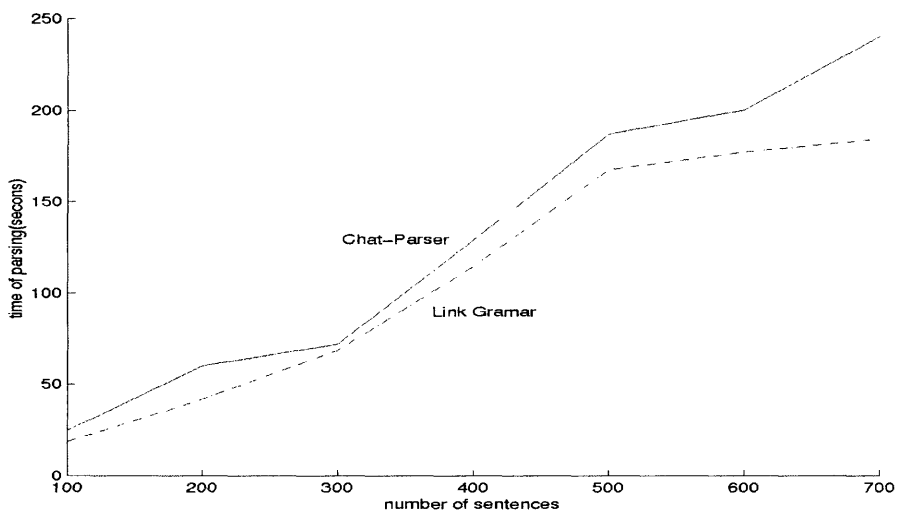

Figure 6: The running time of parsing process

In this experiment we compare two parser; chart parser (we use EARLY parser) and link-grammar. We have tested with input is a document that has number of sentence is 100 sentences to 700 sentence. The result obtaining from our experiment shows that: The time in parsing by Link-grammar is smaller than parsing by chat-parser.

Concerning about the accuracy of parsing we didn't report here because two parser can be improved. We will report the accuracy in comparing two parsers in later.

The complexity of two parser is O (n^3) with input is a sentence has n words.

If we apply parallel algorithm in parsing sentence then the complexity can be reduced O ($\log^2 n$) [13] time using n^6 processor. Moreover, every unambiguous context-free grammar can be parsed on a P-RAM in O (log n) time using a polynomial number of processor. The time in decomposing is depending on not only the total words in document but also the times of a word in summary document in original document. This is reason to explain for the result in the table above.

The complexity of decomposition process is complexity of Viterbi algorithm.

As we mentioned above, the complexity of Viterbi algorithm is K x N x M^2 (M is the average of one word in summary appear in document, N is total words, K is constant)

Can we parallel this algorithm?

4.3 Decomposition task

After splitting the document into set of sentence, we use distance semantic to formulate the decomposition task as mention above. We select a test set and evaluate the same way we do with splitter; that is we observer by hand to compare with decomposition problem. Average length of summary sentence is 100 words and average of total word in summary sentence is not in original document is 17.8 words, when applying distance semantic measure we have this value decrease to 7.2 words because some word in this set was corresponding semantically to a word in original document throughout semantic measure. The repetition position appears in the output of decomposition task for each summary sentence is 4.2. There is no repetition position (feature) in the output of decomposition task when using checking position function. Therefore, the output of our method is outperforming. To compare the result of two methods, we manually produce the correct output for each decomposition task. We evaluate the accuracy for each output in the input by calculating how many wrong positions (the output position is different from human output). With this evaluation we have the result of baseline method is 80.4%; the method with checking position and distance semantic is 88.2%; the method which use both method. In the figure bellow is the output of decomposition task. We use visualization to make a human easy in checking the result.

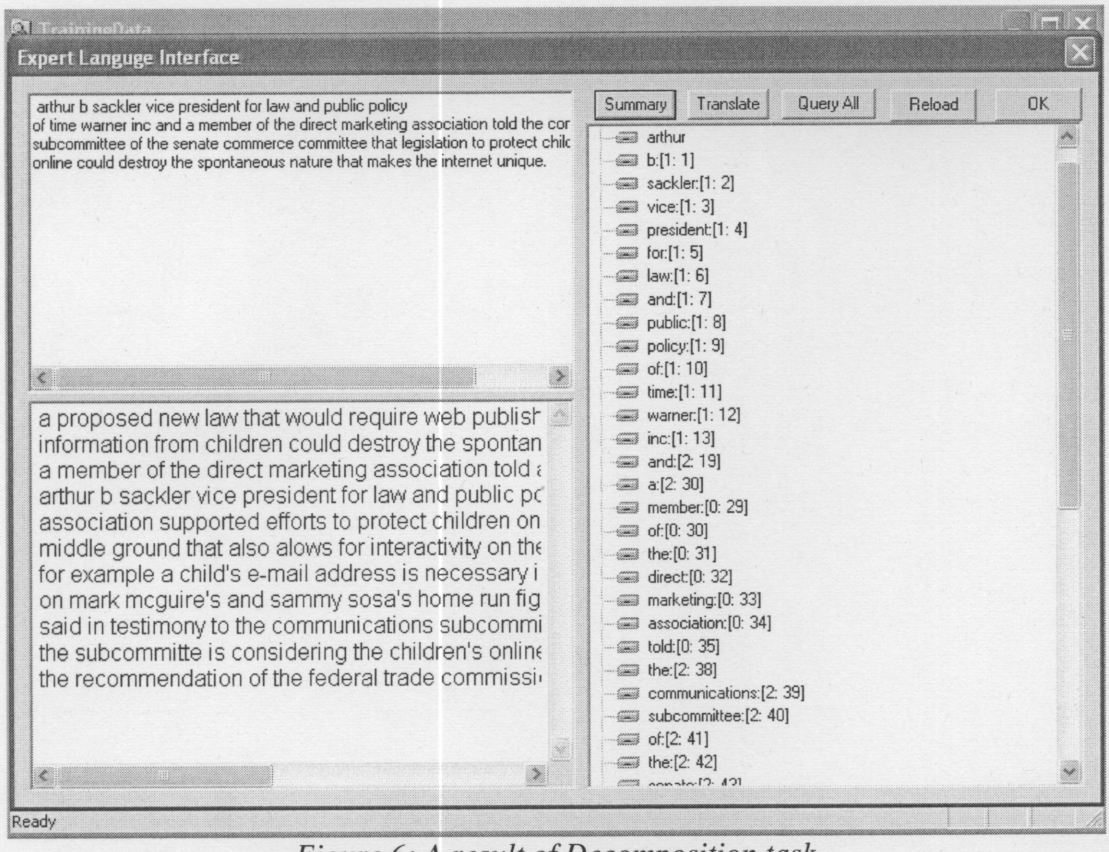

Figure 6: A result of Decomposition task

4.4 Training

This work will continue to compare our model with other model and accuracy is main goal we have to revise this part. Concerning about the time of this part, it is the complexity of EM algorithm, but this algorithm is too slow, it need to use parallel technical. In our experiment, we carry out EM training with several models from

IBM- 1 to IMB-3 and HMM model. The IBM-1; IBM-2; IMB-3 and HMM model was mention in section 2.2. It can be shown more detail in a well-know paper [7].
In future we will do with model based syntax tree but replacing PCFG with Probabilistic link grammar.

4.4.1 Preparing data for training

With data corpus is two files of sentence and its shorter (compares), we dived data corpus into two components; one is training data, other is testing data.
We applied language model is trigram (3-gram) and translation model is IBM-1, IBM-2, IBM-3, HMM model respectively.
There are total 1087 sentences with 25854 words; these words were sorting in a dictionary follow its frequency in corpus. This is an example for that dictionary.

5	and	765
6	to	552
7	of	527
8	a	512
9	is	400
10	for	363
11	The	313
12	in	280
13	that	210
14	's	181
15	are	176
16	with	171
17	on	161
18	as	141
19	will	130
20	be	126
21	from	122
22	can	116

Table 2: a dictionary
(Include word, word's id and word' frequency)

With a word was define as a number value, we have a pair of sentence and its compress will be mapped to a pair of a numeric sequence.

4.4.2 Accuracy of training process

To evaluate the accuracy of training process, we follow the method which was described in [8]; we let the training process generate the most probable alignment of the training corpus (*called the Viterbi alignment*). The alignment shows how the learned model induces the internal structure of the training data.
We allowed the human who performed the alignment to specify two different kinds of alignments: an S (sure) alignment which is used for alignments and P (possible) alignment which is used for alignments which might or might not exit.
The quality of an alignment $A=\{(j , a_j)\}$ is then measured using following error rate:

$$1-\frac{|A\cap S|+|A\cap P|}{|A|+|S|}$$ With |X| is the total number of element in X.

We collect in training corpus 50 sentences and generate alignment manually.
We use the evaluation before and executing a revised the accuracy alignment model with IBM-1, IBM-2, IBM-3, and HMM model. These models were defined in the section 2.2. It is referred more detail in [7].

This is a result of accuracy of training process. (Table 3)

Model	Errors [%]
IBM -1	31%
IBM -2	28%
IBM –3	26.2%
HMM	24.6%

(Table 3: Error rate of alignment model)

The other result of the training process that product the summarization table:
This table will report the probability of translation source words and target word. In summarization problem, two language is the same, therefore two word the same will be achieved the probability is 1. Observing our summarization table (equal to translation table) we have:

- ✓ Two words the same have probability is 1
- ✓ Two very different word have probability is very small
- ✓ Two word similar have probability is higher

Therefore, we can use it for measuring distance between two words, if we test on a large corpus.
An example of summarization table bellow: (table 4); each line in table consist of word in source; word in target; probabilistic of translation from source to target

```
Undersecretary Undersecretary 0.999677
user user 1
jacks jacks 0.5
subsystems subsystems 1
jacks electrically 0.5
box box 0.8
tested tested 0.975878
primary primary 0.474682
EtherneXt EtherneXt 0.960938
in advocates 0.00347343
extracts extracts 0.5
Hitachi Hitachi 0.666667
purchase CDC 0.303476
25-year-old 25-year-old 1
to 350M 0.00180684
good photo 0.0574109
modes modes 0.50216
```

(Table 4: An example summarization table)

5. Approach to parallel statistical text summarization

As mention above, the parallel of text summarization is very essential. We now sketch here a draft of parallel statistical text summarization system. The statistical text summarization based on SMT consists of four components; that is: Language model, Training, Decoder. We would to convert the sequential version into parallel version.

5.1 Language model

We would define language modeling for a sentence within text document, we can use several parsing as EARLY, CYK, Link Grammar. To enhance the time of this phase we would proposed a method which parsing in parallel. To know more detail about parallel version of parsing in context free grammar can be seen in [13].

5.2 Training and Decoder

The training process can be independent with a system of text summarization, so it didn't affect to the time in running of system. However, the algorithm for training is EM algorithm; it cost a lot of time. The most effective is parallel this algorithm.
Concerning about Decoder, this is a search algorithm to find the best alignment in space of translation model. Moreover, almost the decoder was described above will cost a long time when handing with a long sentence. This suggests us to implement on parallel computer.

6. Conclusion

We will finish the decoder process to evaluate the accuracy of system. In several parts we emphasize the necessary of parallel to build a scalable text summarization system. Moreover, our sentence splitter has the accuracy enough (95%) to carry out parsing, decomposing process. The link grammar can be replaced PCFG in generating a novel text summarization model, the time of parsing with Link grammar is smaller than chat parser for whole document. We also achieved the efficient result with decomposition task with our extending HMM model based on distance semantic and checking position. Concerning about revising translation model in SMT for text summarization, we achieved the error rate of HMM is smallest. However, the training data for testing was collected is small corpus, while IBM testing on the data size with 1 million words. That reason explains for us the result is a not enough. In future we will extend the full text summarization model and testing on the large corpus with fixed translation model. We also will parallel both training and decoding process.

Acknowledgement

This research was supported in part by the international research project grant, JAIST

Reference

[1] Knight, K and D. Marcu.2000. Statistic-based summarization-step one: sentence compression. In Proceeding of the 17th National conference of the American Association for Artificial Intelligent, Austin, Texas.
[2] Berger, A. and V.Mital. 2000. OCELOT: A system for summarizing web pages. In Proceedings of the 23rd Annual International ACM SIGIR. Conference on Research and Development in Information Retrieval, Athens, Greece.
[3]. Kupiec, J, J.Pedersen, and F.Chen. 1995. A trainable document summarizer. In Proceeding of the 18th international conference on Research and Development in Information Retrieval, pages 68-83, Seattle, Washington.
[4]. Saltom, G. A.Singhal, M.Mitta, and C.Buckley.1997. Automatic text structuring and summarization. Information Processing and Management, 33(2): 193-2008
[5]. Christopher D. Manning and Hinrich Schutze, 1999. Foundations of Statistical natural language Processing, The Mit Press

[6]. A. Dempster, N.Laird, and D.Rubin 1977 Maximum Likelihood from incomplete data via the em algorithm. Royal Statistical Society Series B, 39.
[7]. Brown, Della Pietra, Mercer, The mathematics of Machine translation: Parameter Estimation, Computational Linguistic 19(2), June 1993.
[8]. Kenji Yamada and Kevin Knight, A Syntax-Based Statistical Translation Model
[9]. Hongyan Jing, 2001, Cut-and-Paste Text summarization. Phd Thesis, Columbia University.
[10]. A.J.Viterbi. 1967. Errors bounds for convolution codes and an asymptotically optimal decoding algorithm. IEEE Transactions on Information theory, 13:260-269
[11]. F.Jelinek , John Lafferty R.L.Mercer, Basic methods of probabilistic context free grammar. In Speech Recognition and Understanding: Computer and System Sciences, vol.75, 1992
[12]. A. Dempster, N.Laird, and D.Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 39B,1977
[13]. Alan Gibbons & Wojciech Rytter. Efficient Parallel Algorithms, 1988
[14]. Adam-Berger, Statistical machine learning for information retrieval, Phd -Thesis 2001, CMU.
[15] Budanitsky A. and Hirst G. 2001. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics. Pittsburgh, PA.
[16] Jiang J. and Conrath D. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of International Conference on Research in Computational Linguistics, Taiwan
[17] Hirst G. and St-Onge D. 1998. Lexical Chains as representations of context for the detection and correction of malapropisms. In Fellbaum 1998, pp. 305-332.
[18] Leacock C. and Chodorow M. 1998. Combining local context and WordNet similarity for word sense identification. In Fellbaum 1998, pp. 265-283.
[19]. Ye-Yi Wang and Alex Waibel. 1997. Decoding algorithm in statistical translation. In Proc. 35th Annual Conf of the Association for Computational Linguistic, pages 366-372, Madrid, Spain, July.
[20]. K.Knight. 1999. Decoding complexity in word-replacement translation model. Computational Linguistic, 25(4)
[21]. D.Wu.1996. A polynomial-time algorithm for statistical machine translation. In Proc. ACL
[22]. F.J.Och and H.Ney.2000a. A comparison of alignment models for statistical machine translation. In COLING 2000: The 18th Int. Conf. On Computational Linguistic, pp 1086-1090, Germany, August.