UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA
FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI
Corso di Laurea in Informatica

Construction of a Thesaurus to use with semantic proximity algorithm
Summary

DISCo – LET
Dipartimento Informatica, Sistemistica e COmunicazione –
Laboratorio Elaborazione Testi

Supervisori:
Prof. Huu LE VAN
Prof. Luca BERNARDINELLO

Relazione della prova finale di:
Roberto FERMI
Matricola n°048091

Anno Accademico 2004 – 2005

Introduction

The content of my stage regards the Information Retrieval, the discipline that takes care to study, to plan and to realize information technology systems finalized in document finding.
This thematic has attracted my interest because I think that in the modern society the Information Retrieval represents and will always represent more the faster and effective way to store and to search information on any argument. The scope of this stage is to realize a system for the finding of information that establishes a valid methodology for the comparison between the informative need of the user (query) and the logical representation of documents (replaces of documents).
The input parameters of an IRS (Information Retrieval System) are the relevant information taken from the original documents and the user request. The retrieval process can be schematized in the following steps:

- creation of a representation of an object, based on its description;
- creation of a representation of the informative need of the user, query;
- comparison between the two representations and choose of that better represent the query of the customer.

The optimal objective is to supply to the user only results totally pertinent to his request, excluding all the rest. In practical this goal is not reachable, so two parameters are used to evaluate the effectiveness of a retrieval process: recall and precision. The precision measures the system ability in recovering only documents important for the request of the user. The recall measures the system ability in recovering all the important documents.
There are many ways to do text researches, but the common idea is to reduce the original document to a document substitute (a significant summary) and, then, compare its sentences with the user query.
There are a lot of methodologies concerning this confront activity; those most widespread are:

- Boolean model
- Fuzzy model
- Probabilistic model
- Vector model

Before a document can be used as a base for the finding of information, it is necessary to execute some operations to obtain only relevant words.
The main steps of this process are:

- stop-list
- stemming
- weighting
- thesaurus

Through the union of the previous operations it is possible to realize a thesaurus that substantially contains couples of terms that have a likeness degree between them.
As far as the comparison between the sentences is concerned, the major characteristics of a good semantic proximity algorithm are the ability of understand the request of the user (precision) and the speed with which the result is given back to the user.
This stage has been developed in two parts: first of all it has been realized an algorithm for the thesaurus construction and, then, it has been possible to implement an algorithm for the research of semantic proximity between the sentences. The first part is about the thesaurus' construction algorithm, while the second part is about the implementation of the Paice-Ramirez's algorithm.

Part 1: Thesaurus

The use of thesaurus in semantic proximity's algorithms is very important, although, this implementation takes a lot of computing time and, it is often inapplicable in common information technology systems. It rises, therefore, the problem to optimize the execution time of the algorithm, maintaining at the same time an optimal level of search of the terms' similarity.

In a thesaurus are memorized the likeness value of the words' couples. There are many ways for the automatic construction of a thesaurus; the one utilized for this stage is based on the search in the document of terms that can be considered similar words. Among the various algorithms oriented to the calculation of semantic likeness between the words, it has been chosen the one relative to the Cosine coefficient.

The implementation of a good thesaurus is realized using the previous techniques and, in particular, the fundamental phases are:

- normalization of the input text;
- applying of the Cosine coefficient for the analysis of the semantic likeness between terms;
- improve of the output data.

Implementation

The normalization of the original document is carried out in various phases. First of all every punctuation character and all the multiple spaces are eliminated, except for the dot. Then, using and apposite dictionary, all not significant words in the document context are deleted. Finally the normalization of the document's words (that means the research of a common radix) is realized using the stemming algorithm proposed by Porter. This procedure is based on the sequential elimination of the word's suffixes, until the basic form of the term is reached. The elaborated word is confronted with a dictionary of the used language for verifying to have obtained an existing word and not have committed stemming errors; in case the term is not found, the original value is replaced.

Therefore, the Porter's algorithm appliance is limited to the English language for the presence of a less number of grammatical exceptions. Using this elaborations is possible to reduce remarkably the number of text's words, decreasing the time needed to execute the research of the semantic likeness between couple of terms.

For the construction of the thesaurus the Cosine's coefficient had been used because it succeeds in supplying better results in comparison with other coefficients.

Once every word of the document is normalized, it is executed the calculus of the Cosine's coefficient over every couple of unique terms, using the following formula:

$$somiglianza\_lessicale = \frac{\sum_{i=1}^{N_f} V_{px}[i] \cdot V_{py}[i]}{\sqrt{\sum_{i=1}^{N_f} V_{px}[i]^2 \cdot \sum_{i=1}^{N_f} V_{py}[i]^2}}$$

where,

N_f	represents the total number of sentences
N_s	represents the number of significant sentences
$P_1, P_2, ..., P_n$	represents the significant words
$V_{pi}[N_f]$	vector that contains in the position $V_{pi}[j]$ the number of occurrences of the word p_i in the $j-th$ sentence

The normalized words are organized in the rows of a table; on the columns of the same table, instead, are memorized the occurrences of every sentence's terms. The Cosine's coefficient uses this matrix to calculate the semantic likeness. In particular, to reduce the space occupied by the database and calculus' time, it is chosen to find direct dependences only; in other words, if is found the semantic likeness for the couple of terms "A B", the value for "B A" is not calculated. Also in the case is found "A A" the analysis is avoided, because the value should be 1.
The improving of the output data consists in finding a semantic likeness value threshold, under which the couple of word should not be memorized in the thesaurus. In this way is possible to define the precision of the constructed thesaurus. It is chosen to set the threshold value equals to 0.3 because after detailed analysis was revealed that under that threshold the words are rarely linked together. To obtain good search's results it is chosen to ignore words under that limit. It has been placed an under limit for the length of the considered terms: words with a number of characters lower than 2 are ignored.
In the case the same couple of words comes analyzed more than once in different documents, the memorized semantic likeness value is the average between all the values found since that time. This sagacity is needed to obtain a realistic value of the likeness between two terms. If there is a large number of documents, for sure the same couple of words will be analyzed more than once time and it will have different values from document to document. Using the average, it is possible to maintain a trace of the value of each document, increasing the precision of the thesaurus.
To calculate the average value correctly it is necessary that in the database is memorized also the number of times that the couple has been analyzed. The database structure should be like this:

Termine1	Termine2	Somiglianza	Occorrenze

The code for the thesaurus' construction has been developed in a multithread environment to make possible the parallel execution of various instances of each class. This permits to take advantage of potentiality of the modern personal computer, reducing remarkably the time of execution.
The goal of this stage is to realize a thesaurus usable from the semantic proximity's algorithms in different subjects. So, the number of terms in the database should be sufficiently big to permit to obtain good results for any request.
Since the thesaurus construction is based only on the terms' frequency, it is necessary that every document considered has a well defined subject. In this way it is possible to avoid ambiguity between the values taken.
For this stage 42 documents have been analyzed with an average of 10.000 words, obtaining a thesaurus of 11.405 significant relations and 408 important relations.

Part 2: Paice-Ramirez algorithm

After careful researches it has been found that ones of the simpler, but reliable algorithms for the semantic proximity research, based on the use of a thesaurus, is the one developed by Paice and Ramirez.
The operation is based on the quantification of the vicinity between two sentences in relation of the existing likeness among the words of the first and the second phrase. For this reason it is essential to have an available thesaurus containing couples of words that present a likeness.

We supply some definitions that will be used to opportunely define the working of the algorithm:

- word: any finite sequence not null of alphabetical characters having length greater than or equal to a minimum defined level
- term: any word that does not appear in any stop-list
- phrase: a finite sequence not null of terms that ends with a dot.

Now it is possible to consider

$S=(S_1, S_2, ..., S_m)$

and

$T=(T_1, T_2, ..., T_n)$

two phrases composed, respectively, by m and n terms. The function to calculate the semantic proximity R between the two sentences S and T is like the following:

$$R(S,T) = f(S_x, T_y)$$

Implementation

In order to correctly evaluate the weight of the words it is necessary to define a mapping, namely an association between the terms of the first sentence and the second one, in a way that the two associated terms presents a not null likeness. It is possible to assume that: every S_x term in S can be associate to a different term T_y in T, or not to be mapped. One term of the phrase S can not be associated to more than one of the sentence T.

We can define $J(x)$ as a function that associate the term $T_{J(x)}$ in T to a term S_x in the sentence S. For every S_x in S that it does not present a mapping with the terms in T, it is worth that $J(x) = 0$.

A factor that is important to deal with is the order of the terms in the two analyzed phrases, It is necessary to introduce a function that represents the tidiness (accuracy) of the terms orders.

$$\Psi(J) = \psi_{\min} + \Phi(J) \cdot (1 - \psi_{\min})$$

where,

$$\Phi(J) = \frac{1}{m-1} \sum_{x=1}^{m-1} K_x$$

and

$$K_x = \begin{cases} 1 & se \quad J(x+1) = J(x)+1 \\ 0 & se \quad J(x+1) \neq J(x)+1 \end{cases}$$

Even in the worst sort, this function should not bring the result to zero. The function for the calculus of the semantic proximity, considering also the tidiness's tie became:

$$R(S,T) = \frac{1}{F} \sum_{x=1}^{m} W_x \cdot V \cdot (S_x \cdot T_{J(x)}) \cdot \Psi(J)$$

where:

- for every couple of associate terms S_x e $T_{J(x)}$ we want to determinate the value that defines the affinity. W_x is weight associated to the word S_x
- F is a normalization factor

- $V \cdot (S_x \cdot T_{J(x)})$ denote the semantic or lexical proximity between the two terms S_x e $T_{J(x)}$. V is calculated using the values contained in the thesaurus.
- $J(x)$ denote the mapping function
- $\Psi(J)$ denote the tidiness function

The mapping is calculated using a matrix $m \times n$ where, in the rows are indicated the phrase's terms of the document and in the columns the words of the user query. In each intersection it is inserted the semantic likeness value between the two terms considered, taking it from the thesaurus.
The direct mapping of the terms is a very simple and quick technique to determine an association between words, but it is not efficient because it is "short-sighted". To avoid this problem the mapping has been realized using an algorithm that hold memory of the weights distribution of the matrix, creating a second matrix W. The values contained in W are weights that penalize an eventual choice of the current term.

$$w_{ij} = \sum_{k=1}^{n,k\neq i} m_{ik} + \sum_{k=1}^{m,k\neq j} m_{ik} + (1 - m_{ij})$$

The selected value is the lower one, that is the one corresponding to the minimal loss.
Once the value is chosen, the algorithm delete the row and the column corresponding to the selected element. In this way, it is possible to avoid to chose it again in the next phases.
The choice of the ordering for the correspondence between the terms is memorized in a vector, that will be used to calculate the tidiness's function, that is to analyze if the terms of the query are placed in the same order of the words of the current phrase.
A particular attention must be given to the final value of semantic proximity returned by the algorithm. This is made by doing a weighted average between the semantic proximity's coefficients of each document's phrase. The weight is calculated by an apposite formula and depends by the number of existing phrases in the text and by the value of each single sentence:

$$peso = e^{(prossimità * coefficiente)}$$

$$Coefficiente = \ln(numero\_parole) \cdot \frac{100}{80}$$

The complexity of the used formulas in this algorithm, make difficulty a total evaluation of the code's quality. To avoid this problem, it has been chosen to execute separated tests to estimate each single parts of the algorithm.

Synonymous evaluation

The test carried out consists in executing different searches over the same text, inserting as a query synonymous. It has been verified that the algorithm uses correctly the thesaurus instrument because it gives back results comparable with the effective semantic likeness among the analyzed words.

Ordering evaluation

In order to estimate the ordering of the words of the query regarding the analyzed phrase of the document, an appropriate phrase containing three key words has been introduced in the text. The tests have been conducted changing the terms' order and analyzing the found semantic proximity's variation. From the tests we can see that the algorithm can analyze the term's sorting of the user query in relation with the document's phrase, to optimize the semantic proximity's value.

Length evaluation of the phrases

Another aspect to analyze is the influence that the phrase's length has on the semantic proximity's value returned by the algorithm. To carry out this evaluation it is necessary to modify the documents' phrases' length and to analyze the variation of the results.

From the outcomes of the tests it can be deduced that the semantic proximity's value is correctly influenced by the phrase's length considered.

Precision and recall evaluation

In order to verifying the algorithm efficiency it is necessary to analyze the precision and recall parameters that characterize the information retrieval systems. From the results obtained it is deduced that the algorithm has a good ability in retrieving only the relevant document for the user, but its capability of retrieving all the documents effectively important for the user's search is lower.

Conclusions

The main difficulties met during the realization of this stage have been to reduce to a minimum the calculus time to yield the results in an acceptable time for the user and to give a correct interpretation of the general Paice-Ramirez's formula. From the collected outcomes it is possible to say that the general goal of this stage has been reached in a more than satisfactory way: the calculus time are limited, the algorithms adapt themselves to the more updated personal computers taking advantage of their potentialities, and the output data are coherent.

The experience of this stage has allowed me to deepen the crucial aspects of the IR and to study new elements of the programming, of the optimization of the code and the hardware configuration of the computers