Internal Use Only（非公開）

TR－SLT－0082

A feasibility study of analogy-based translation using the same source and target language

Erwann THORAVAL & Yves LEPAGE

2004年9月6日

概要

A method based on analogy for machine translation is being developed at SLT-NLP. By now, it is already possible to get translations for some short utterances. So as to evaluate this new method, two experiments have been proposed. The first one deals with the evaluation of the analogical equation solver. The evaluation allowed us to assess the quality of the program: the accuracy is very high (98%) in number of characters), but the coverage remains to be improved. The second one concerns the recursivity of the method necessary to increase the coverage.

（株）国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒619－0288「けいはんな学研都市」光台二丁目2番地2 TEL：0774－95－1301

Advanced Telecommunications Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai "Keihanna Science City" 619-0288,Japan
Telephone:+81-774-95-1301
Fax :+81-774-95-1308

Abstract

With the arrival of new technologies few years ago (cellular phone, internet, cable and satellite television), the linguistic borders keep softening off. It has become easier and easier to access information in foreign languages. However, our individual linguistic capabilities do not allow us to understand everything. Thus, translation systems able to quickly translate information for us are needed. Machine translation is one of the applications aimed at by natural language processing like, for instance, automatic summary, document indexing and classification, automatic learning in artificial intelligence.

It already exists several machine translation systems based on different concepts, like rule-based systems, example-based systems or statistical methods. This document presents a feasibility study of a new approach in machine translation which is based on *analogy* between strings of characters. This document presents some experiments and their results so on to evaluate the method.

Contents

Glossary

API Application Program Interface

EBMT Example-Based Machine Translation

MT Machine Translation

SQL Structured Query Language

Key words

Machine translation, analogy, evaluation, recursivity

Introduction

By now, machine translation based on analogy is not very developped. However, the basic principle is quite simple, and this method can generate many new utterances as soon as a set of data is available. Here, we shall be interested in the evaluation of this method. We propose two experiments. The first one deals with the evaluation of an analogical equation solver. The second one concerns the recursivity of the method, necessary to increase the coverage of the method.

This work has been done in the framework of an internship, at ATR, Japan. After a presentation of ATR laboratories, we shall see in more details the background and the objectives. A chapter will then give basic knowledge on analogy. Finally, we shall describe the experiments we conducted and their results.

Chapter 1

Background and objectives

In the first part of this chapter, we shall discuss about the necessity of inflating corpora to improve the quality of machine translation.

1.1 Background

Let us start by giving the definition of a corpus. In linguistics, a *corpus* (plural *corpora*) is a large and structured set of attested texts. A corpus may contain a single text in a single language (monolingual corpus) or text data in several languages (multilingual corpus). Multilingual corpora that have been specially formatted for side-by-side comparison are called *aligned parallel corpora.*

In the design of MT[1] systems, corpora are often used as the basic source of knowledge.

Why is it so important to increase the size of corpora? There are several answers to this question. Firstly, to allow translation into many languages, data have to be available in all these languages. Secondly, the aim of machine translation systems is to provide the most correct translations, in terms of *syntax* but also of *meaning*, that is to say, to improve the quality of translations. To do so, systems need more and more data of different meanings. Thirdly, it is also important to add sentences with the *same meaning* in corpora. Such sentences are called *paraphrases*. Paraphrases are necessary in automatic evaluation methods, like BLEU[2].

The same reasons can be applied to human beings. When we try to learn a new language, the quality of our productions strongly depends on the knowledge we have of this new language, but also of our own language. The more we know grammar rules, vocabulary, synonyms, exceptions, the more correct we can produce sentences. The same applies to machine translation systems.

[1] Machine Translation

[2] for more informations about BLEU, see Appendix C on page 41.

1.2 Specific problems

There are several methods that can be used to inflate a corpus. One of these methods is based on *analogy* [3]. Basically, an analogy is a relation between four elements, and is noted as following:

$$A : B :: C : D$$

Here is an example of an analogy of meaning:

$$\textit{planets} : \textit{sun} :: \textit{electrons} : \textit{nucleus}$$

which can be understood as *the planets are to the sun as the electrons are to the nucleus.* Here is an analogy between strings of characters:

I like Japanese food. : *I prefer Japanese food* :: *I like Mexican food.* : *I prefer Mexican food.*

The use of analogy has many advantages, especially:

- simplicity: one single engine for the whole process;
- bidirectionality: the system can translate either Japanese → English or English → Japanese;
- universality: the system works for any language;
- easiness of adding new languages: just need to translate the sentences of the corpus in the new language (no training needed).

We shall consider in more detail the analogy-based method in chapter 2 on the next page. The aims of the internship, according to these specifics problems, will be presented in section 2.2 on page 13.

Chapter 2

The analogy-based method

We shall here discuss about the chosen method to inflate a corpus. Firstly, we shall focus on the analogy principle and its application to translation; then we shall propose experiments to evaluate the method; finally, we shall present the aims of the internship[1].

2.1 Overview

2.1.1 Definition of an analogy

An analogy is a relation between four objects. Between strings of characters, *successful* : *unsuccessful* :: *pleasant* : *unpleasant* or *I drive* : *he drives* :: *I speak* : *he speaks* are valid analogies. We can also write *bird* : *wings* :: *fish* : *fin*. This analogy has a meaning for human beings. However, in the case of this study, we can not work with such analogies. To do so, world knowledge or language knowledge are required. Here we shall focus on analogies between strings of characters.

Analogies can be seen as equalities:

$$\textit{fable} : \textit{fabulous} :: \textit{miracle} : \textit{miraculous}$$

but also as equations:

$$\textit{fable} : \textit{fabulous} :: \textit{miracle} : x \quad \Rightarrow \quad x = \textit{miraculous}$$

In the case of strings of characters, the following relations can be established:

$$A : B :: C : D \Longrightarrow \begin{cases} (1) & |A|_a + |D|_a = |B|_a + |C|_a, \ a \\ (2) & d(A,B) = d(C,D) \\ (3) & d(A,C) = d(B,D) \end{cases}$$

[1] details about the implementation will be discussed in the next chapter.

(1): the number of occurrences of the symbol in a the string A added to the number of occurrences of the symbol a in the string D is equal to the number of occurrences of the symbol a in the string B added to the number of occurrences of the symbol a in the string C. This relation is valid for any symbol (letters, numbers, punctuation symbols).

(2): the distance[2] between A and B is equal to the distance between C and D.

(3): the distance between A and C is equal to the distance between B and D.

Although this relation is an implication, in the present work we shall use it as if it were an equivalence.

plurality of solutions An analogical equation may have several solutions. For instance:

$$aba : aa = cbcbcb : x \Rightarrow x = ccbcbc \vee x = cbccbc \vee x = cbcbcc$$

In the above example, the letter b has been pulled out of the string *aba*. Since there are three b in the string *cbcbcb*, the three solutions above are correct.

For detailed information about analogy, refer to [3].

2.1.2 Description of the method

The analgoy-based method permits the inflation of a corpus because it is able to *generate* new objects using three other objects. Here is a summary of the method:

- in the corpus find triples which can produce an analogy;
- try to solve the analogical equation made up by these three sentences;
- if a solution is found, check if the produced sentence is valid (i.e. if it is syntactically correct and if it makes sense).

An example is shown on fig. 2.1 on the facing page. Using three sentences existing in the corpus, a fourth sentence is produced (*I prefer Mexican food.*) which was not already in the corpus.

2.1.3 Application to translation

Let us have a look to the analogy-based translation. Figure 2.2 on page 14 shows the method to translate a sentence from English to French. We would like to translate in French the sentence D = *I prefer Mexican food.* We have a set of sentences in English (source language) and in French (target language).

[2]for more information about the distance, see appendix B on page 39.

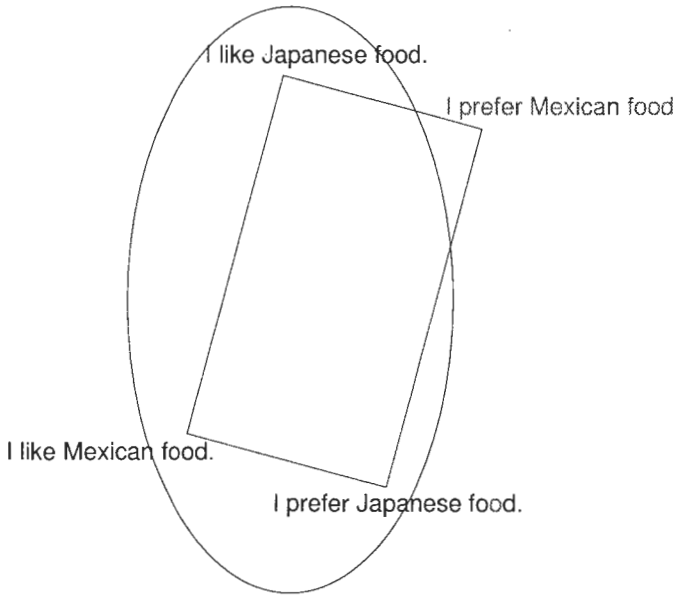


FIGURE 2.1: Generation of a new sentence

These set are actually an aligned parallel corpus which contains sentences in English and their translations in French.

Fortunately, there is an analogy between D and three others sentences which is:

A	:	B	::	C	:	D
I like Japanese food.	:	*I prefer Japanese food.*	::	*I like Mexican food.*	:	*I prefer Mexican food.*

Since we know the French translations of A, B and C, we can *transpose* the same analogy in the target language:

I like Japanese food.	:	*I prefer Japanese food.*	::	*I like Mexican food.*	:	*I prefer Mexican food.*
↓		↓		↓		↓
J'aime la cuisine japonaise.	:	*Je préfère la cuisine japonaise.*	::	*J'aime la cuisine mexicaine.*	:	$x \Rightarrow x = Je$ *préfère la cuisine mexicaine.*

2.2 Goals

The aim of this internship is to evaluate the analogy-based method. Here are the provided means:

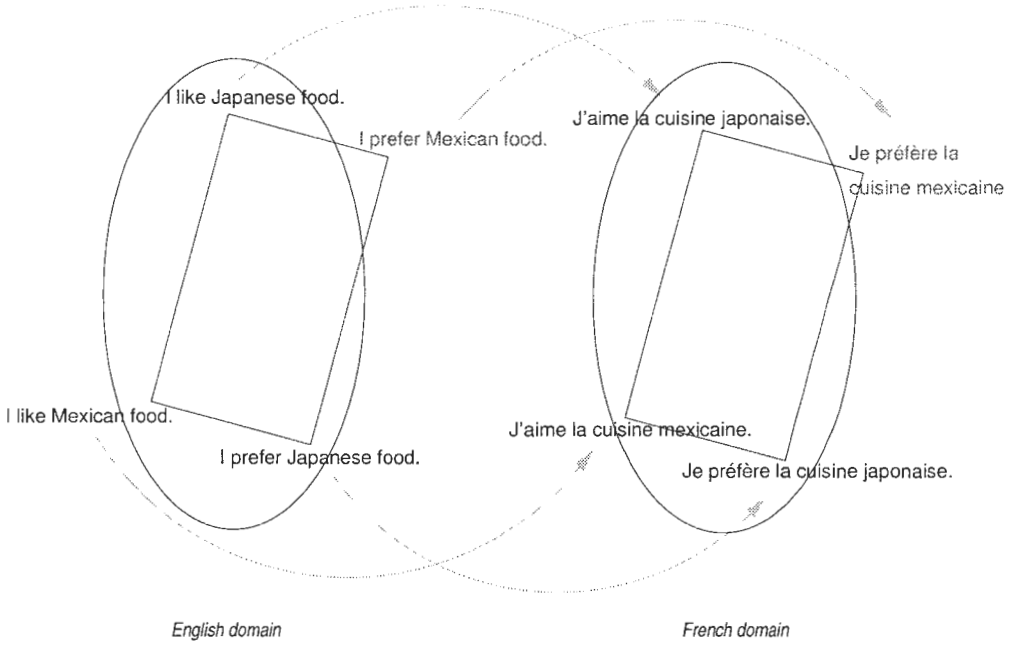


FIGURE 2.2: Analogy-based translation principle

- the BTEC-CSTAR corpus[3] available in four languages (Japanese, English, Chinese and Korean);
- a program which inspects a set of sentences to find analogic relations between these sentences[4];
- a program which solves analogical equations given by three sentences[5].

In order to do the evaluation of this method, we will make translation experiments using the same language, both in the source language and in the target language (English). We will then check the quality of the results refering to the source sentences.

The internship can be split in two parts. The first part deals with the evaluation of the program for the resolution of analogical equation. The second part proposes a recursive method to increase the number of results provided.

2.2.1 First part: incomplete checking

The aim of this part is to evaluate the analogical equation resolution algorithm written by Yves Lepage [3, pages 207–243].

[3]for further details, see A on page 37.

[4]whose name is `extork`, written by Yves Lepage.

[5]whose name is `solveanalogy`, written by Yves Lepage.

The `extork` programs creates a file containing sentences which are in analogical relation between other sentences. The aim is to pull out one of the four sentences in an analogy and try to retrieve it resolving the analogical equation resulting from the three remaining sentences. In an ideal case, every sentence should be retrieved. However, in practice, this is not true (due to some problems in the resolution algorithm and in its implementation). Implementation details are discussed in section 3.1 on page 17. The results of this experiment are detailed in section 4.1 on page 29.

2.2.2 Second part: using recursivity to improve checking

Even when huge data are available, it is not always easy to obtain analogies. Though, it is possible to increase the number of analogies using recursivity. For instance, assume we have the following data :

I like rock music.	I own classical clothes.
I own funny clothes.	I like funny music.
I listen to rock music.	

We would like to obtain *I listen to classical music.* With the above data, it is impossible to make a direct analogical equation which solution would be *I listen to classical music.* However, we can make:

$$\textit{I own funny clothes.} : \textit{I like funny music.} :: \textit{I own classical clothes.} : x \quad \Rightarrow \quad x = \textit{I like classical music.}$$

With this new sentence, we are now able to make:

$$\textit{I like rock music.} : \textit{I listen to rock music.} :: \textit{I like classical music.} : x \quad \Rightarrow \quad x = \textit{I listen to classical music.}$$

The example above shows that it is possible to use several levels of recursivity to compute a sentence.

The goal is to create a program which uses recursivity to compute translations (still from English to English). Implementation details are discussed in section 3.2 on page 19. The results of this experiment are detailed in section 4.2 on page 32.

Chapter 3

Implementation

Details about the experiments are discussed in this chapter. The first part presents briefly the first experiment protocol. The second part is the most important since it required the most of the time. This part deals with the implementation of the recursive algorithm, the data structures and the heuristics required for the algorithm.

3.1 First experiment

The objective of this first experiment is to evaluate the quality of the analogical equation resolution algorithm. A schematic view of this experiment is presented on fig. 3.1.

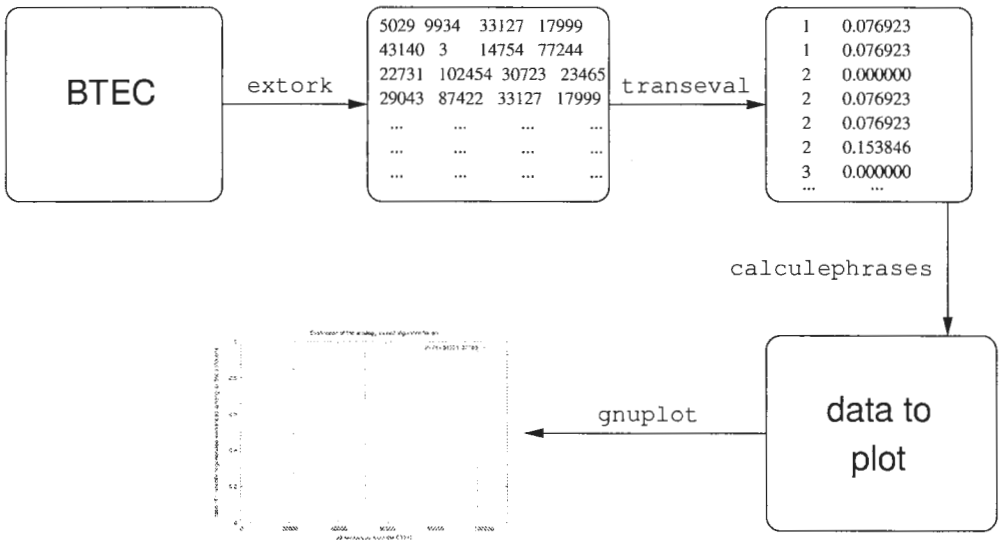


FIGURE 3.1: Schematic view of the first experiment

As mentioned in the previous chapter, the `extork` program tries to find the

analogic relations between sentences from the BTEC.

Here is a sample of the `extork` output:

24320	10664	119721	10665
77874	49615	45697	38577
36337	158114	128792	98720

The numbers are the line numbers in the BTEC. They correspond to the following sentences :

I'm here alone.	I am traveling alone.	I'm here with a group.	I am traveling with a group.
I have a good idea.	I don't have any idea.	I have a good appetite.	I don't have any appetite.
I'm hurt.	I've been hurt.	I'm running a fever.	I've been running a fever.

Then, for each analogical relation (i.e. for each line), we pull out one of the four sentences and we give the three remaining sentences to `solveanalogy`. Three cases can occur (please do not forget that an analogical equation can have several solutions[1]):

good result this is obviously the better case. The sentence is computed without any mistake;

bad result the algorithm succeeded in solving the analogical equation. However, the result is not exactly what was expected[2];

no result this is the worse case: the program did not succeeded in solving the analogical equation.

The program which performs this operation is called `transeval`. Its output contains, for each found solutions (good or bad) the edit distance[3] to the original sentence. These data have then to be organized in order to be plotted. This is the work of `calculephrases`. This program generates a file which can be directly used by `gnuplot`[4].

Basically, this experiment had to be done only with the English language. However, as data were available in Japanese, Chinese and Korean, we decided to make the same experiment on these three different languages. The results for these four languages are detailed in section 4.1 on page 29.

[1] it often occurs that there is one good solution, and several bad solutions per sentence.

[2] usually, some characters in the sentence are inverted, due to a problem of contiguity in the algorithm.

[3] for more informations about the edit distance, refer to appendix B on page 39.

[4] `gnuplot` is a interactive datafile and function plotting utility, which is available for Unix, MS Windows, Apple Macintosh and others platforms and can be freely distributed. `http://www.gnuplot.info`

3.2 Second experiment

As seen in the previous chapter (subsection 2.1.3 on page 12), it is possible to increase the number of sentences involved in analogies using recursivity. This is the purpose of this second experiment. The goal is to make a program able to translate a sentence from a source language to a target language, using recursivity. We will first have a look to the recursive principle. Then we will see in more details the implementation of this recursive method.

3.2.1 Overview

Here, we have a set of 510 sentences that we would like to translate. This set is a subset of the BTEC and is often used in ATR-SLT for experiment purposes. We also use the same source and target language (English) for this experiment. Figure 3.2 describes the sequence of operations. As in the previous experiment, the output sentences are not all perfect. We will evaluate these outputs three times (two times using BLEU[5], and once using edit distances).

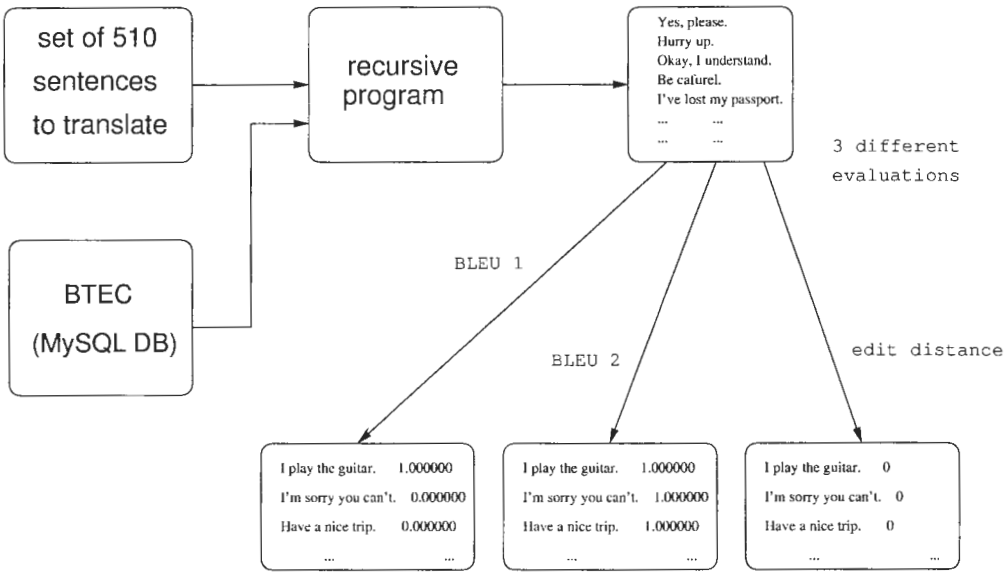


FIGURE 3.2: Schematic view of the second experiment

Now, let us have a look at the recursive algorithm.

[5]BLEUr13n4 and BLEUr14n4. More details in section 4.2 on page 32.

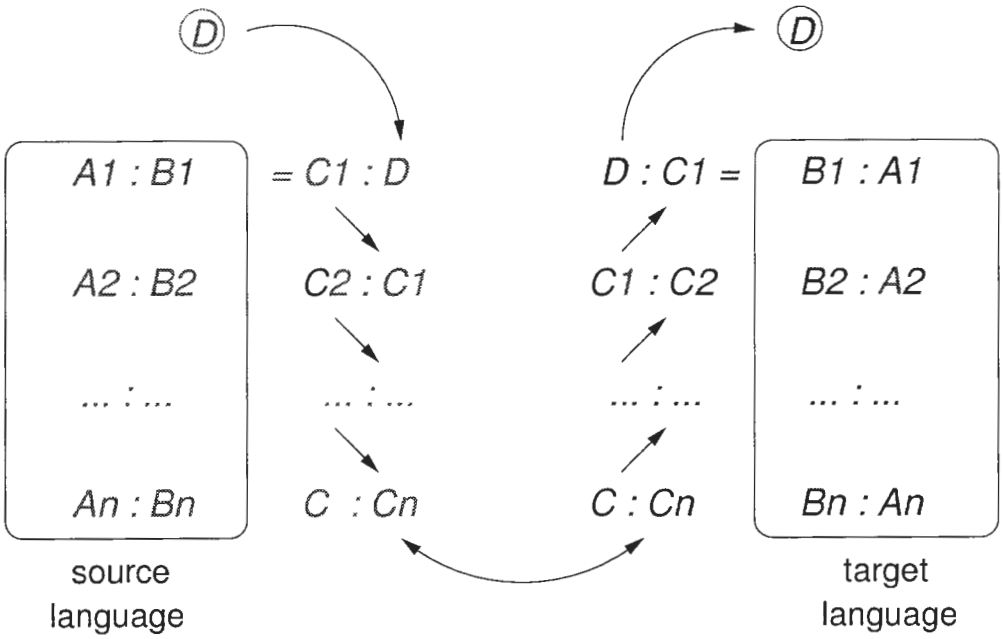


FIGURE 3.3: Principle of recursive translation

notations:

$\mathcal{A}$	the source language
$\mathcal{A}$	the target language (same as source language in this setting)
A, B, C or D	sentences from the source language
A, B, C or D	sentences from the target language

Assume that D is the sentence to translate. Here is the procedure to follow:

step 1 find two sentences (A and B)[6] which are in analogic relation with D ;

step 2 solve the analogical equation $A : B :: x : D$, $x = C$;

step 3 **if** $C \in \mathcal{A}$ **then** get A, B, C from $\mathcal{A}$ and solve the analogical equation
$A : B :: C : x$, $x = D$
else return to step 1 to translate C.

Figure 3.3 is the schematic view of the above procedure.

3.2.2 Algorithms

This section presents the algorithm corresponding to the procedure described before. The following table explains the notations used in the algorithm. Then, the four main functions of the algorithm are detailed.

[6] using heuristics, see subsection 3.2.4 on page 26.

notations

i	the source language
j	the target language
D_i	the sentence to translate (the source sentence)
D_j	the translation of D_i in the target language (the target sentence)
A_i, B_i, C_i	sentences involved in an analogy

functions part

Translate() The main function. It first looks for an existing translation of D_i. If no translation is found, then it calls **Compute()**.

Compute() First, it gets a set of sentences which may form an analogy with D_i (which are A_i and B_i, already in the corpus). Then, it gets the solutions of the analogical equation in the source language. Finally, it calls **Transpose()**.

Transpose() It transposes the analogical equation from the source language to the target language. It first gets the translations of A_i and B_i (respectively A_j and B_j). Then it tries to get translations of $\{C_i\}$. The recursive call is made here[7].

functions description

Translate(D_i, i, j)
begin
if D_i is already being computed
  then exit
  else **if** translations of D_i already exist
    then return D_j
    else **return Compute**(D_i, i, j)
  endif
endif
end

[7] you can refer to fig. 3.3 on the preceding page.

Compute(D_i, i, j)
begin
$\{C_i\} \leftarrow \emptyset$, $\{D_j\} \leftarrow \emptyset$
$\{A_i, B_i\} \leftarrow$ computing heuristics
for each couple (A_i, B_i)
do
    $\{C_i\} \leftarrow$ **Solve_Analogy**(B_i, A_i, D_i)
    $\{D_j\} \leftarrow$ **Transpose**(A_i, B_i, $\{C_i\}$, i, j)
done
return $\{D_j\}$
end

Transpose(A_i, B_i, $\{C_i\}$, i, j)
begin
$\{A_j\} \leftarrow \emptyset$, $\{B_j\} \leftarrow \emptyset$, $\{C_j\} \leftarrow \emptyset$
$\{A_j\} \leftarrow$ **Give_Existing_Translations**(A_i, i, j)
$\{B_j\} \leftarrow$ **Give_Existing_Translations**(B_i, i, j)
for each C_i
do
    co recursive call **endco**
    $\{C_j\} \leftarrow \{C_j\} \cup$ **Translate**(C_i, i, j)
done
return Solve_Group_Analogy($\{A_j\}$, $\{B_j\}$, $\{C_j\}$, i, j)
end

Solve_Group_Analogy($\{A_j\}$, $\{B_j\}$, $\{C_j\}$, i, j)
begin
$Solutions \leftarrow \emptyset$
for each A_j
do
    for each B_j
    do
        for each C_j
        do
            $Solutions \leftarrow Solutions \cup$ **Solve_Analogy**(A_j, B_j, C_j)
        done
    done
done
return $\{Solutions\}$
end

Note For the experiment purpose, both source and target languages are English. Nevertheless, we did not want to limit the domain of application of the program to this only experiment. This program can therefore be used for real translation if aligned corpora in the source and target languages are available.

This algorithm has been written in the C language. This language has been used to program `solveanalogy`. Using the same language for all these programs allowed an easy communication between them. Moreover, a MySQL interface is easily for C.

3.2.3 Database : MySQL

As seen before, the data used for this experiment come from the BTEC. Physically, BTEC is a set of text files. Text files are not very adapted to our purpose. Managing text files in a C-language program is rather difficult. Thus, we preferred to use a database which is more flexible and reliable. In addition, the database can be stored on a different machine than the one which executes the program.

The database chosen is MySQL[8]. MySQL is an open-source software and its license is free for non-commercial usage. An API[9] is available for the C-language, that is to say, it is easy to access to the MySQL database from C programs: a set of functions has been developped by the MySQL team and allows our program to connect to the database, to make requests, to insert data and so on.

[8] `http://www.mysql.com`
[9] Application Program Interface

Several SQL tables have been made to store the BTEC into the database. They can be seen in fig. 3.1 to 3.6 on the next page. Here is a description of these tables:

id	nb_car	cntgh1 ... cntgh8	freq
	unsigned int		
SQL identifier	number of characters in the sentence	contiguity hash code	number of occurences of the sentence

TABLE 3.1: data_{en, ja, ko, zh} — contains informations about the sentences

id	string
unsigned int	*blob*
SQL identifier	the sentence

TABLE 3.2: data_all — contains the sentences and their respective id.

id_sql	id_btec
unsigned int	*varchar(50)*
SQL identifier	BTEC identifier

TABLE 3.3: btec — link between the BTEC id. and the SQL id.

data_{en, ja, ko, zh} contains statistics on the sentences stored. Each sentence has a SQL identifier (an integer). It is easier to identify sentences with a number rather than with a string of characters. The contiguity hash code will not be used by now, but it has been introduced for further applications.

data_all makes the relation between the sentences and their SQL identifier. The data type chosen to store the sentences si a *blob*. This type is a binary object which can contain anything, up to 2^{16} bytes[10]. A *text* type is available, but it is not case sensitive. Moreover, except for English, a character requires two bytes to be stored.

btec makes the relation between the SQL identifier and the BTEC identifier. The BTEC identifier is a complicated mix of letters and numbers. It is not easy to handle for an identifier.

[10] 65,536 bytes

lang	bytes
varchar(20)	*unsigned int*
{en, ja, ko, zh}	# of bytes for one character

TABLE 3.4: desc_domaines — contains the number of bytes by character

src	cib	chaine
varchar(20)		*blob*
source language	target language	sentence being computed

TABLE 3.5: trad_en_cours — contains the sentences which are being computed at the m moment

lang1	lang2	id1	id2	freq
varchar(20)		*unsigned int*		
lang. of sentence id1	lang. of sentence id2	SQL id1	SQL id2	# of occurence of this translation

TABLE 3.6: memoire_trad — contains all translations

desc_domaines stores the number of bytes for one character. It is used by the program in functions which manipulate characters. Japanese, Chinese and Korean are encoded on two bytes.

trad_en_cours is a kind of latch to prevent the program from computing several times the same sentence. When the program tries to translate a sentence, it puts this sentence into the table. Thus, no loops are made.

memoire_trad may be the most important table since it contains all translation. It establishes the relation between the source and the target language. The `freq` field is important. It contains the number of occurences of every translation. Thus, we know which translation is more often produced.

3.2.4 Heuristics

Heuristics are used to find, into the corpus, two sentences capable of making an analogical equation with the sentence to translate. The rapidity to find translations depends strongly on the quality of the heuristics. Therefore, we have to find the best heuristics in terms of rapidity of resolution.

Stopping the recursivity

First, let us focus on the recursivity. We have to take care of stopping the recursion process, otherwise, our program will never stop. The stop condition depends on the heuristics. As seen in chapter 2, sentence lengths are kept in an analogy. Thus, if we have $A : B :: C : D$ and $|A| \geq |B|$[11] then we will have $|C| \geq |D|$. Here, C is the sentence which has to be translated on the next recursion level. Thus, if the length of C increases at each recursion level, our program will never stop. We therefore have to set a constraint on the respective lengths of A and B which is:

$$|A| < |B|$$

Naïve method

Now, we can see in more details the way to find a pair (A, B) which is relevant to make analogical equations with the sentence to translate. The naïve method does not take care of the sentence to translate. Thus, all arrangements of A and B will be used. As there are about 100,000 different sentences in the BTEC, the number of combinations would be:

$$100,000 \times 100,000 = 10^{10}$$

[11] $|A|$ stands for the *length* (i.e. the number of characters) in the string A.

Of course, this huge number of combinations is not suitable. Every combination would be done, like *Good night* : *I like fried chicken.* :: *You should have a drink.* : x which will obviously not deliver any solution.

Distance and similarity-based method

We have to set more precise constraints on A and B in order to form analogical equations which may be solved. We propose the following, in two steps:

1. first, we try to find a set of Bs which are as close as possible to D;
2. then, we use this set of Bs to find a set of As.

The first point is done using the edit distance[12]. We keep the sentences with the smallest distance to D. The second point is based on similarity. We want to find sentences (A) which have the longer substring included into B.

Experiments showed that good results are obtained (i.e. that the sets of As and Bs can make analogical equation with D) using a set of ten sentences for Bs and a set of one hundred sentences for As.

[12]you can refer to appendix B on page 39 for details about the edit distance.

Chapter 4

Results

4.1 First experiment : without recursivity

Table 4.1 on the next page shows a sample of the results of analogical equations. Since an analogical equation may have several solutions and since a sentence may be involved in several analogies, there obviously are many solutions for one sentence. As we said before, the algorithm and the program which solve the equations are not perfect; bad results are sometimes output, and sometimes, no result is output. Here we are interested in the proportion of perfect outputs among all the outputs. In this sample, we can see that the bad sentences occur rarely. On the other hand, perfect sentences occur often. Tables 4.2 on the following page and 4.3 on page 31 show quantitative results.

4.1.1 Table 4.2

The number of sentences involved in analogies is about half of the number of sentences, except in Korean. This may be due to the morphological richness of Korean. For instance, there are many conjugations in this language. Thus, it is more difficult to obtain analogical relations between sentences in Korean.

The number of analogies by sentence involved is high — between thirty and fourty, except in Korean. Nevertheless, the standard deviation is high. This means that some sentences are very often involved in analogies whereas others are only involved in one or two analogies.

4.1.2 Table 4.3

The first column shows the average distance to the reference. As we saw before, we tried to retrieve each sentence which was involved in an analogy. The edit distance is shown in percentage. For Japanese, it means that on a one hundred characters sentence, the distance to the reference is less than 1.

# of occ.	sentence
25	**Really?**
2	lReady?
232	**No, thanks.**
23	,No thanks.
2	No th,anks.
1	N,o thanks.
1	No t,hanks.
1	Nos, thank.
1	ne, thasNo.
1	s, thankNo.
1571	**Excuse me.**
4	Excmee us.
3	Excus mee.
1	.Excuse me
1	Excue mes.
1	Excuse m.e
679	**I'm sorry.**
6	I'm srryo.
2	I'm .sorry
985	**Thank you.**
4	Thank yuo.
3	Thank ouy.
2	edyuThank.
670	**Where is the boarding gate?**
4	iWhere's the boarding gate?
2	Where is the boardig gatne?
765	**I'd like to reconfirm my reservation.**
3	I'd like re toconfirm my reservation.
3	I'd like to reconfirm my resrvatione.
2	I'd t likeo reconfirm my reservation.
1	I reconfirm my reservation'd like to.
1	I'd like to confirm my reerservation.
121	**I'm very hot.**
8	I' verym hot.
4	I very'm hot.
2	v eyI'm hot.
1	I'me vry hot.
1	aevI'mry hot.
1	i eyI'm hot.
1	nevI'mry hot.
1	o oeyI'm hot.
1	t veyI'm hot

TABLE 4.1: Sample of the solutions to analogical equations

The ratio of no result ranges from five to ten per cent. This is due to a bug in `solveanalogy`[1]; even when a solution exists, the program may not find it.

The ratio of bad results is quite low. For Chinese, it means that for hundred sentences solved, only six are bad.

The last column is the complementary of the third one. For hundred sentences solved, around ninety five per cent are correctly retrieved.

	number of different sentences	number of sentences involved	number of analogies	analogies by sentence involved
Japanese	103,274	53,572	1,910,062	35.65 ± 58.5
English	97,769	53,250	2,384,202	44.77 ± 98.6
Korean	92,628	25,088	266,504	10,62 ± 21.1
Chinese	96,234	49,675	1,639,068	33 ± 51.7

TABLE 4.2: results for the first experiment – 1/2

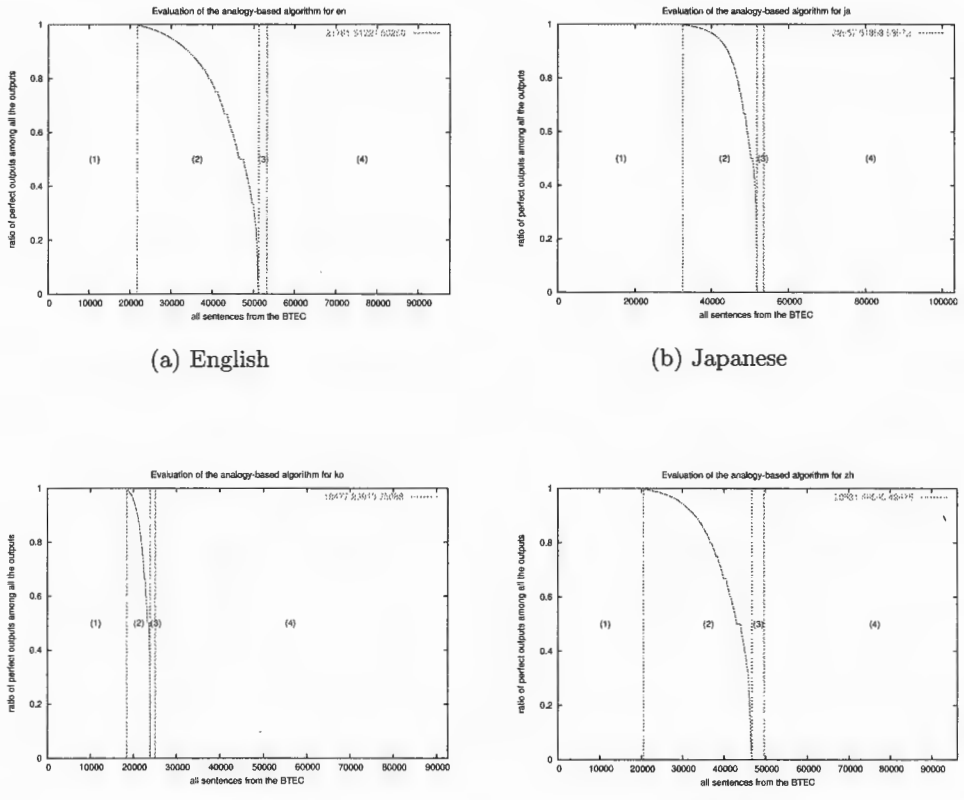
[1]the program which solves the analogical equations.

	average distance to the reference	ratio of no result	bad results	good results
Japanese	0.7%	5.1%	3.2%	96.8%
English	2.2%	10.7%	3.8%	96.2%
Korean	0.8%	4.9%	4.7%	95.3%
Chinese	1.8%	11.2%	6.0%	94.0%

TABLE 4.3: results for the first experiment – 2/2

4.1.3 Diagrams

For each of the four languages, a diagram has been drawn. The x axis represents all different sentences of the BTEC. The y axis shows the ratio of perfect



(a) English

(b) Japanese

(c) Korean

(d) Chinese

FIGURE 4.1: Graphical view of the results above

outputs among all the outputs. Each diagram can be split into four parts, numbered from 1 to 4.

area (1) all these sentences have *perfect outputs*;

area (2) *at least one* ouput is perfect (this ratio decreases);

area (3) none of the outputs are perfect;

area (4) sentences not reachable by immediate analogy.

As it is shown in the previous tables, only few sentences were not retrieved by the program (around five per cent). This corresponds to the thin area on the diagrams.

4.1.4 Discussion

These results are encouraging. Even if the analogical equation solver is not perfect, the accuracy is very high (more than ninety-five per cent). Now, we would like to increase the coverage. Only half of the BTEC sentences are directly covered by analogies. Introducing recursivity may increse the coverage. The next section presents the result obtained using recursivity.

4.2 Second experiment : with recursivity

This second experiment has been made using the same corpus than the previous one. Results have been obtained on a set of 510 sentences used in all experiments to evaluate the quality of the translation systems implemented in ATR-SLT. In the set of 510 sentences, 199 are in direct analogical relation with other sentences from the BTEC. In the ideal case, none of these 510 sentences should be in direct analogical relation with other sentences. Indeed, we use recursivity to reach sentences which were not reachable before.

We used three evaluation methods which are:

BLEUr13n4 [2] for each candidate, we have 13 references at our disposal, not necessary different one from each other. These references are paraphrases of the original sentence, given by humans. It is important to note that the original sentence (i.e. the sentence to translate) *is not present* in the 13 references. For instance, the references for the sentences *Have a nice trip.* are:

[2] BLEUrXnY stands for the BLEU method applied with X references and Y for the maximum order of n-grams.

Have a good trip.	Enjoy your trip.
Bon voyage.	Enjoy your trip.
Bon voyage!	Have a good time on your trip.
Bon voyage.	Have a good trip!
Happy trails!	Bon voyage.
Enjoy your trip.	Have a good trip!
Be sure to have a great time on your trip.	

BLEUr14n4 it is the same method as BLEUr13n4, but we added the original sentence to the 13 references, so there are now 14 references.

edit distance this evaluation gives the number of basic edition operations needed to transform the translated sentence to the original one. More detailed informations about the edit distance are available in appendix B on page 39.

# of retrieved sentences	# of bad sentences	average score		
		BLEU 1	BLEU 2	edit distance
329 (64.5%)	9 (2.7%)	**0.22** ± 0.36	**0.98** ± 0.12	**0.15** ± 1.15

TABLE 4.4: Results of the second experiment – English

4.2.1 Discussion

Table 4.4 presents the results of the second experiment performed on English. The coverage is not so bad, but remember that among these 329 sentences, 199 are in direct analogical relation with other sentences from the BTEC. Nevertheless, it means that 130 sentences have been rebuilt thanks to the recursivity.

Among the 329 sentences, only 9 are bad. For instance, the program gave *I don't have timet righ now.* instead of *I don't have time right now.* or *Is this all right table?* instead of *Is this table all right?*. All other sentences (320) were perfectly rebuilt.

Now, let us have a look at the BLEU scores. The difference between the two scores is huge. However, the two methods used were very similar. The only difference is that we added the original sentence to the set of references for BLEUr14n4. It means that, even with a set of references of same meaning than the candidate, BLEU may give a bad score to a very good candidate. For instance, the sentence *Have a nice trip.* has been perfectly retrieved. BLEUr13n4 gave a score of 0 whereas BLEUr14n4 gave a score of 1, as would be expected.

Finally, the last score is the edit distance. The average distance between the candidates and their respective references is very low: 0.15 character. A score of 1 character would mean that, in average, a character would have been added or deleted on each sentence. In other words, there is only one character added or deleted every 7 sentences.

4.3 Conclusion and future work

These experiments give a good overview of the capabilities of the method. Some improvements to the analogical solver (both to the algorithm and to the program) may give better results. We can notice that the accuracy is very high. Thus, we should focus on coverage. The second experiment shows that the addition of recursivity can improve the coverage. But solving analogical equation is not sufficient to obtain many good translations. If there is no analogical relations in the sentences of the corpus, the method will not give results. Therefore, the most important point is to add *relevant* sentences into the corpus, in order to increase the number of *different* analogies. The problem is to find what a *relevant* language model is. Examining the sentences which were not involved in analogies may give some elements of answer. Other experiments could then be proposed by adding different sentences than those which already are in the corpus and evaluating the coverage increasing.

During the recursive experiment, we saw that adding a dictionary to the corpus could really improve the coverage. A dictionary is the same as an aligned corpus, but data are words instead of plain sentences. This allows more analogical combinations between sentences. For instance, if we have the sentence *I am eating chicken.* and a food term dictionary, we can easily produce *I am eating beef*, *I am eating rabbit.*, *I am eating cookies.* and so on. During the experiments, we saw this phenomenon twice. We added

$$FAX \leftrightarrow FAX$$

into the data. This gave us a sentence which was not given before: *Where can I send a FAX?*. We also added

$$\textit{This man is an idiot.} \leftrightarrow \textit{This man is an idiot.}$$

which gave *That man is an asshole.*[3] Before adding *This man is an idiot.*, we had

$$\textit{That man is a ashnsole.}^{4}$$

Indeed, in the data, there were no sentence like ... *is* **an** {*idiot, angel, animal, orange, oven, ...*}. That is why the program had difficulties to built *That man is a ashnsole.* and this explains the misplaced letters.

[3] sic! This sentence is one of the 510 test sentences.

[4] you can note that the n and the h are misplaced.

Conclusion

We proposed experiments to evaluate a translation method based on analogy. The results of these experiments are sufficiently encouraging. By now, it is already possible to get translations for some short utterances. Our work showed a high accuracy. Coverage can be improved using recursivity in the method.

Although it can seem paradoxical, analogy is a powerful method because it is simple. Unlike other machine translation systems, adding a new language can be done quickly and easily. No training or learning step is required. The only operation to do is to have a translation of the sentences in the language to add. So adding a new language basically consists in translating existing data.

Appendix A

BTEC

The BTEC[1] is a *multilingual parallel-corpus* which has been collected in ATR–SLT, and extended by research partners to their respective languages, within the framework of C-STAR[2]. The goal of the C-STAR consortium is to facilitate global cooperation in speech-to-speech translation research.

For now, four languages are available for the entire data :

- Japanese : ATR (Advanced Telecommunications Research Institute International)
- English : ISL (Interactive Systems Labs – Carnegie Mellon University)
- Chinese : CAS (Chinese Academy of Science)
- Korean : ETRI (Electronics and Telecommunications Research Institute)

Others languages are not yet fully available :

- French : CLIPS (Communication Langagière et Interaction Personne-Système – Institut d'Informatique et Mathématiques Appliquées de Grenoble)
- Italian : ITC (Istituto Trentino di Cultura)
- Spanish

The BTEC is the main data support used during the experiments. A sample of the contents is shown in fig. A.1 on the following page.

[1] Basic Traveler's Expression Corpus

[2] Consortium for Speech Translation Advanced Research – `http://www.c-star.org`

I'll stay at my friend's house.	友達の家に泊まります。
A round-trip ticket, please.	往復切符をお願いします。
Would you mind if I kissed you?	キスしていいかい。
Scotch with soda, please.	スコッチのソーダ割りをください。
Can you take me to the Flower Hotel on George Street?	ジョージ通りのフラワーホテル まで行ってくれますか。
Please bring your wife along.	奥さんも連れて来てください。
I'm Japanese.	私は日本人です。
How is your wife?	奥さんはいかがですか。

TABLE A.1: Example of sentences from BTEC (English, Japanese)

language	number of sentences	number of different sentences	size of sentences in characters mean ± std. dev.
Chinese	162,318	96,234	11.00 ± 5.77
English	162,318	97,769	35.14 ± 18.81
Japanese	162,318	103,274	16.21 ± 7.84

TABLE A.2: Some statistics on the BTEC multilingual corpus

Appendix B

Edit distance

The *edit distance* between two strings is the minimum number of operations needed to transform one string into the other. Here, the operations are insertion and deletion. Substitution can also be considered as an operation. Here its weight is twice the one of insertion or deletion (indeed, a substitution can be considered as a deletion followed by an insertion).

Definition

Let $\mathcal{E}$ be a set, δ be a function from $\mathcal{E} \times \mathcal{E}$ to $\mathbb{R}^+$. δ is a distance on $\mathcal{E}$ if and only if:

- $\forall (A, B) \in \mathcal{E}^2,\ \delta(A, B) = 0 \Leftrightarrow A = B$
- $\forall (A, B) \in \mathcal{E}^2,\ \delta(A, B) = \delta(B, A)$
- $\forall (A, B, C) \in \mathcal{E}^3,\ \delta(A, C) \leq \delta(A, B) + \delta(B, C)$

This mathematical definition also applies to edit distance between strings of characters.

Exemples

The distance between

- *fable* and *table* is 2: *fable* $\rightarrow$ *able* $\rightarrow$ *table*
- *kitten* and *sitting* is 5: *kitten* $\rightarrow$ *itten* $\rightarrow$ *sitten* $\rightarrow$ *sittn* $\rightarrow$ *sittin* $\rightarrow$ *sitting*
- *I'm very hot.* and *aevI'mry hot.* is 6: *I'm very hot.* $\rightarrow$ *aI'm very hot.* $\rightarrow$ *aI'm vry hot.* $\rightarrow$ *aeI'm vry hot.* $\rightarrow$ *aeI'm ry hot.* $\rightarrow$ *aevI'm ry hot.* $\rightarrow$ *aevI'mry hot.*

The edit distance is a good way to evaluate revision of translations since it really agree with the number of operations to be made (by hand, or with a computer) to go from a string to an other.

Appendix C

BLEU

BLEU[1][5] is a method for automatic evaluation of machine translation. The method has been developped at IBM Research Center in 2001.

BLEU performs an evaluation of a set of candidates against a set of references. There may be only one candidate and several references, or several candidates for one reference.

BLEU gives a measure of the inclusion of n-grams contained in the candidates in the set of references (the n-grams concentration of the references in the candidates).

Basically, BLEU is the product of two factors:

- a penality which depends on the length of the candidate. The shorter a candidate is, the more it will be penalized. The length of the candidates has to be higher than the length of the smallest reference;
- the geometric mean of the ratio of word-based n-grams present in the references, up to the maximum number in the references, over all values of n. In practice, n varies from 1 to 4.

BLEU is supposed to have a good correlation with human judgements about the fluency of a translation[1].

[1] BiLingual Evaluation Understudy

Bibliography

[1] Bogdan BABYCH, Debbie ELLIOTT, and Anthony HARTLEY. Extending MT evaluation tools with translation complexity measure. In *Proceedings of COLING 2004*, volume I, pages 106–112, Genève, August 2004.

[2] Kenji IMAMURA. *Automatic Construction of Translation Knowledge for Corpus-based Machine Translation.* PhD thesis, Nara Institute of Science and Technology, `http://www.slt.atr.jp/~kimamura/paper/dthesis2004.pdf`, 2004.

[3] Yves LEPAGE. *De l'Analogie rendant compte de la commutation en linguistique.* université Joseph Fourier – Grenoble I, `http://www.slt.atr.jp/~lepage/pdf/dhdryl.pdf`, 2003.

[4] Makoto NAGAO. A framework of mechanical translation between Japanese and English by analogy principle. In *Artificial and Human Intelligence*, pages 173–180, 1984. Amsterdam: North-Holland.

[5] Kishore PAPINENI, Salim ROUKOS, Todd WARD, and WEI-JING Zhu. BLEU : a Method for Automatic Evaluation of Machine Translation. Technical report, IBM J. T. Watson Research Center, `http://www.research.ibm.com/people/k/kishore/RC22176.pdf`, 2001.