Machine Translation by Case Generalization

Hiroshi Nomiyama

IBM Research, Tokyo Research Laboratory
5-19 Sanbancho, Chiyoda-ku,Tokyo 102 Japan
E-Mail:nomiyama@trl.vnet.ibm.com

Abstract

Case-based machine translation is a promising approach to resolving problems in rule-based machine translation systems, such as difficulties in control of rules and low adaptability to specific domains. We propose a new mechanism for case-based machine translation, in which a large set of cases is generalized into a smaller set of cases by using a thesaurus.

1 Introduction

Case-Based/Example-Based Machine Translation (CBMT/EBMT) has been proposed as a way of overcoming the knowledge acquisition bottleneck in machine translation. This approach is based on the simple concept of translating sentences by analogy with similar cases stored in a set of cases(a case-base) [1, 2, 3, 4].

This approach has two advantages in terms of knowledge acquisition. CBMT/EBMT ensures that (1) if the same case as the input exists in the case-base, then the same result will be obtained, and (2) if a similar case exists in the case-base, then a similar result will be obtained. In the first instance, which cases are regarded as the same depends on the equality metrics of the system. In the second instance, which cases are regarded as similar depends on the similarity metrics. Rule developers or users can control the system on the basis of equality and similarity without understanding the global flow of controls.

In applying this idea to practical machine translation systems, there are still two serious problems. One is that CBMT/EBMT requires a great deal of computation because of its inherent need to retrieve a huge number of cases and calculate their similarities to the input. For practical systems, several hundreds of thousands of cases must be accessible.

CBMT/EBMT systems should not impose any restrictions on cases to be added to the case-base in an effort to keep the case-base small, since the similarity metrics depends on the frequencies of cases. If cases are restricted, sufficient information to control the rules is not acquired.

The other problem of CBMT/EBMT is the difficulty of defining a semantic distance. Though thesauri are used as bases for semantic distance calculation in CBMT/EBMT, it may be impossible to define a general semantic distance by using thesauri alone. Semantic distances between words are defined according to which specific words are related to their translations. For example, in translating the word "食べる"(eat,feed,. . .), "犬(dog)-が-食べる"is equivalent to "a dog eats," "牛 (cow)-が-食べる" is equivalent to "a cow feeds," and "馬 (horse)-が-食べる" is equivalent to "a horse feeds." In these cases, "牛"(cow) is closer to "馬"(horse) than "犬"(dog), because different words are selected for each translation of "食べる" with "牛"(cow) and "犬"(dog). But in translating "走る"(run,gallop,. . .), "犬 (dog)-が-走る" is equivalent to "a dog runs," "牛 (cow)-が-走る" is equivalent to "a cow runs," and "馬 (horse)-が-走る" is equivalent to "a horse gallops." In these cases, "牛"(cow) is closer to "犬"(dog) than "馬"(horse).

If such incomplete semantic distances calculated by thesauri alone are used for CBMT/EBMT, exceptional cases may be interpreted as general ones(over-generalization). Over-generalization is a major problem in translating idiomatic expressions. For example, "頭(head)-が-切れる" has two translations: "hurt one's head " or, idiomatically, "be smart." But "頭部(head)-が-切れる" has only one interpretation, "hurt one's head," though the word "頭部" has almost the same meaning as "頭".

It is obvious that "頭部-が-切れる" can be translated correctly by adding this translation pair into the case-base. The addition, however, cannot prevent the idiomatic expression "頭-が-切れる" from being interpreted generally. The idiomatic interpretation still may be adopted for "X-が-切れる" if X is more similar to the word "頭" than the words in the case-base whose pattern is "X-が-切れる."

Sato [3] and Sumita [4] weigh each slot depending on how much it affects the translation. However, since such weights are calculated only for each slot, the over-generalization that occurs inside of a slot is not resolved. To avoid over-generalization, we need some mechanism to encapsulate exceptions rather than to adjust the semantic distance.

2 Machine Translation by Case Generalization

A case-base, in contrast to a set of rules, has inherent redundancy, because cases are collected without preselection. In the simplest case, if the sentence "A" has only one translation equivalent "a," then the single case "A" → "a" is enough to translate "A."

But if we view the case-base as a collection of sentences, the same sentences rarely seem to occur [1]. Sentences can, however, be divided into smaller fragments which are meaningful units for translation according to the some linguistic models, which we call translation patterns.

These fragments are combined for use in translating sentences. Fragments divided on the basis of translation patterns are obviously more effective than sentences, because smaller fragments are more likely to match than full sentences.

We generalize such fragments extracted according to each translation pattern, using a thesaurus, by replacing the words that occur in cases by more general concepts in the thesaurus. The words to be replaced are determined by their frequencies in the case-base. Frequent occurring fragments should be assigned more weight than less frequent by occurring fragments. The frequencies of fragments are used to weigh generalized cases in generalization.

Semantic distances are calculated for each translation pattern as the importances of generalized cases. Only meaningful categories for the translation pattern are stored as generalized cases, except that the most meaningful category is taken as a default. For example, the word "犬"(dog) may be generalized into the concept <dog> [2] for translation of "鳴く"("a dog barks"), whereas it may be replaced by the more general concept <animal>, for other translation patterns in which the concept <dog> is not meaningful.

While generalizing cases, we can identify exceptional cases as those which cannot be generalized. Once we identify exceptions, then we can prevent such exceptions from being interpreted generally.

In this way, cases are generalized according to the translation pattern into generalized cases with concepts as the values of their variables.

In addition to generalized cases, rules can be formulated according to translation patterns. Generalized cases and manually written rules are assumed to be the same as objects in CBMT. It is valuable to have rules available as well as cases, especially when the case-base contains insufficient cases. If rules are not available, there must be sufficient cases from the time the system is first used. Incremental development of any domain is possible only if general rules are available.

In accordance with these basic ideas, we propose a method of machine translation in which cases are generalized. In our approach, we define linguistic patterns in translation. According to these patterns, the cases in the case-base are divided into smaller fragments and are generalized. Both rules and generalized cases are used to translate sentences.

CBMT is divided into two sub-processes: (1) best matching, to search for the most similar cases in the case-base, and (2) application control, to control the combination of similar cases for translation. Application control is a general problem in machine translation, whereas best matching is a problem unique to CBMT. If the best matching process returns certainty factors, the system is controlled using these factors on the basis of the some other model such as Watanabe's [5].

In this paper, we concentrate on best matching using a thesaurus.

[1] The case-base should contain natural sentences rather than examples which are only the smallest fragments effective for translation. We distinguish CBMT from EBMT in accordance with this viewpoint.

[2] Concepts are enclosed between arrowheads (< and >) in this paper.

3 Generalizing Cases

3.1 Division and Linearlization of Cases

At first, we define a translation pattern (TP_i) as follows.

$TP_i = [P_s, V_s, P_t, V_t]$

P_s : Structural Pattern in Source Language (SL)
V_s : List of Lexical Variables in SL
P_t : Transformation into Target Language (TL)
V_t : List of Variables in TL

We call the number of variables in V_s the term number (M_i) of TP_i.

Next, we extract translation pattern cases (TPC_i) from the case-base by applying the pattern matches described in TP_i to all cases in the case-base.

$TPC_i = [L_s, C_s, L_t]$

L_s : List of Values of Lexical Variables in SL
C_s : List of Constraints in SL
L_t : List of Values of Variables in TL

If some patterns other than those specified in P_s are related in translation, those patterns are described in constraints (C_s).

These TPC_is are linearlized into linearlized translation pattern cases ($LTPC_i$).

$LTPCi : L_s \rightarrow (C_s, L_t)$

We call the right-hand part of $LTPC_i$ the value (V). The examples in Fig. 1 are extracted $LTPC_i$s in Japanese-to-English translations of "NOUN ni VERB," where we assume a translation pattern in which an English preposition is determined by a binary relation of a Japanese noun and a Japanese verb.

In the following section, we show how to generalize $LTPC_i$s into generalized linear translation pattern cases ($GLTPC_i$) by replacing words with more general concepts in the thesaurus, and calculate degrees of importance for them.

["Sangatu"(March) ,"Kowasu"(destroy)	] → ([],["in"])
["Sigatu"(April) ,"Gironsuru"(discuss)	] → ([],["in"])
["Gogatu"(May) ,"Saiketusuru"(vote)	] → ([],["in"])
["Rokugatu"(June) ,"Hieru"(cool)	] → ([],["in"])
["Getuyou"(Monday) ,"Arau"(wash)	] → ([],["on"])
["Kayou"(Tuesday) ,"Kimaru"(decide)	] → ([],["on"])
["Syuumatu"(weekend),"Agaru"(raise)	] → ([],["on"])
["Higasi"(east) ,"Uturu"(move)	] → ([],["to"])
["Toukyou"(Tokyo) ,"Idousuru"(move)	] → ([],["to"])
["Sitigatu"(July) ,"Idousuru"(move)	] → ([],["in"])

Figure 1: Translations of "NOUN ni VERB"

3.2 Case Generalization by Means of a Thesaurus

3.2.1 Creation of N-Term Partial Thesauri

We create working thesauri, $PTH_i(j)$ $(1 \leq j \leq M_i)$, for each term. They include every word in the j-th term, and set pairs of values and their frequencies in each word node.

Here we define the importances used to weigh generalized cases.

Importance of a Link (IL) The importance of a link (IL) is the probability of occurrence of cases that occurred in the subtree of $PTH_i(j)$. IL is defined as follows.

$$IL = \frac{S}{C_i}$$

where S is the total number of cases in the subtree connected with the link, and C_i is the total number of $LPTC_i$s extracted from the case-base according to TP_i.

Importance of a Node (IN) The importance of a node (IN) shows the degree of variance of values in a subtree. IN is defined as follows.

$$IN = \sqrt{\sum p_k^2}$$

where p_k is the probability of each value in the subtree [3].

Importance of a Value (IV) The importance of a value L (IV) in the node k is defined as follows.

If node k is a word node, then
IV_{kl} = frequency of value L in node k

[3] We adopt the same expression as that used by Stanfill [6] and Sumita [4].

else

$$IV_{kl} = IN_{kl}\sum(IL_m \times IV_{ml})$$

where m is a node linked to node k, and IV_{ml} is the importance of value L in node m.

Importance of a Generalized Case (IC) The importance of a $GLTPC_i$ (IC) is defined as follows.

$$IC = \sum_{j=1}^{M_i} IV_{jl}$$

where IV_{jl} is the importance of value L, which is the same as the value of the $GLTPC_i$.

3.2.2 Subdivision of Conceptual Leaf Nodes

According to the definitions given in the previous section, at first ILs and INs are set in all the links and nodes in $PTH_i(j)$, and IVs are calculated in conceptual leaf nodes in $PTH_i(j)$.

If IV is not the maximum value in a conceptual leaf node and is greater than the pre-defined threshold value and its frequency is greater than 2, the node is subdivided into more specific concepts.

Subdivision occurs because a specific category which doesn't exist in the thesaurus is effective for a specific translation pattern. Only the difference from the thesaurus is kept as the translation pattern thesaurus i ($TPTH_i$).

3.2.3 Propagation of Importance of Values

Next, we calculate IV in all nodes other than conceptual leaf nodes by propagating IV. The propagation is done by multiplying the importances of values by the importances of links, and the sum of all the propagated values is multiplied by the importance of the node. At first, the propagation is done upward, starting from the conceptual leaf nodes. During upward propagation, downward propagation is done if a child node is a conceptual node and a propagated value is greater than the maximum importance of values in the child node. Downward propagation prevents overgeneralization.

We show examples of results of importance calculation in Fig. 2 and Fig. 3, for the first and second terms respectively. In Fig. 2, the subdivision occurred in the node <Time> and the new node <*X*> was created. A downward propagation occurred in the node <Concrete> in Fig. 2. The word "in" was made more important than the word "to" in the node <Concrete>.

[<>,<Destruction>] → ([],["in"])
[<>,<Speech>] → ([],["in"])
[<>,"Saiketusuru"(decide)] → ([],["in"])
[<>,<>] → ([],["in"])
[<*X*>,<Action>] → ([],["on"])
[<*X*>,"Kimaru"(decide)] → ([],["on"])
[<*X*>,<Up-Down>] → ([],["on"])
[<Location>,<Abstract>] → ([],["to"])
[<Direction>,<Abstract>] → ([],["to"])
[<>,"Idousuru"(move)] → ([],["to"])

Figure 4: Result of the Intra-Term Generalization

3.2.4 Intra-Term Generalization of $LTPC_i$

According to importances calculated according to the method described in the previous section, $LTPC_i$s are generalized in the j-th term. If the value with the highest IV in the child node is the same as the value with the highest IV in the parent node, then the word in the term is generalized by the concept in the parent node. This process of generalization is repeated until no further generalization is possible, and only the most generalized cases are kept. If identical cases are obtained as a result, only one case is kept.

We show an example of intra-term generalization of ["Kayou"(Tuesday),"Kimaru"(decide)] → ([],["on"]). Initially, the firts term "Kayou"(Tuesday) is generalized. The value of this case ([],["on"]) is the same as the value with the highest IV in the parent node <*X*> (see Fig. 2), so "Kayou"(Tuesday) is replaced by <*X*>. The value ([],["on"]) is not the value with the highest IV in the parent node of <*X*>, and therefore generalization stops at the first term. Next, the second term "Kimaru"(decide) is generalized. In the parent node <Decision> of "Kimaru"(decide), the value that is the same as the value of the case is one of the values with the highest IV. Consequently, parent nodes are checked to determine which value is more important. In the root node, ([],["on"]) is less important than ([],["in"]), so no generalization occurs for the second term. Finally, [<*X*>,"Kimaru"(decide)] → ([],["on"]) is obtained as the result of intra-term generalization.

The result of intra-term generalization for all the $LTPC_i$s in Fig. 1 is shown in Fig. 4.

3.2.5 Inter-Term Generalization of $LTPC_i$

Next we generalize cases over terms. Inter-term generalization takes ICs into consideration. If $M_i = 1$,

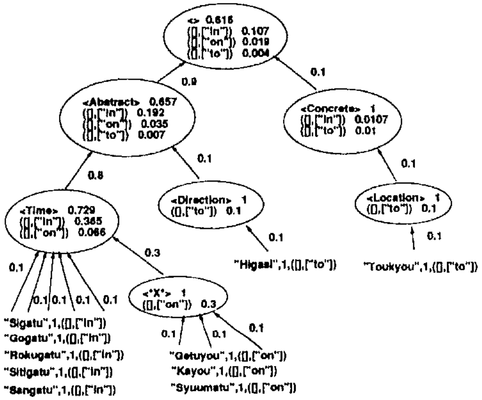

Figure 2: First-Term Partial Thesaurus

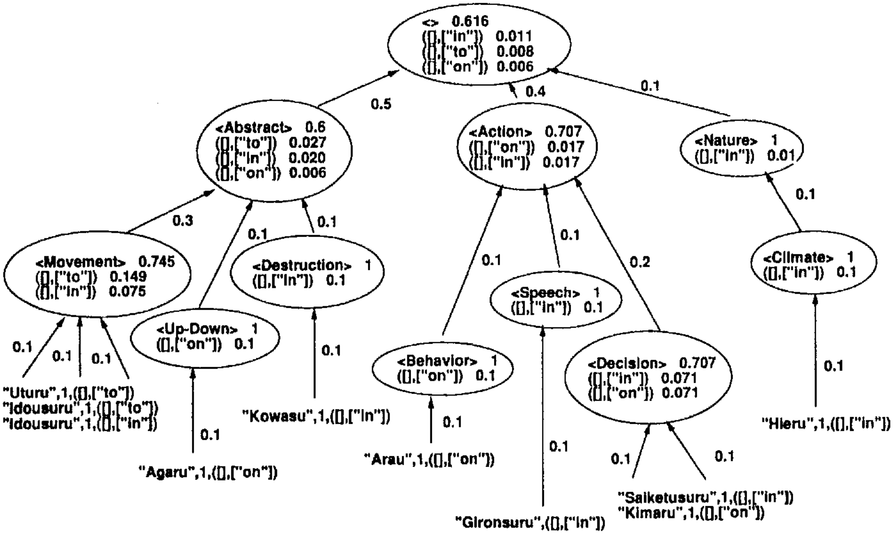

Figure 3: Second-Term Partial Thesaurus

then the result of intra-term generalization with ICs is the generalized linear translation pattern case i ($GLTPC_i$). If $M_i > 1$, j-th term maximum generalization ($1 \leq j \leq M_i$) is done for each term. In j-th term maximum generalization, terms other than the j-th term are fixed first and the j-th term is generalized as much as possible. Then, the maximum possible generalization is done for remaining of terms in turn. If $M_i > 1$, then $M_i \times (M_i - 1)$ $GLPTC_i$s are obtained. If identical cases are obtained as a result, only one case is kept.

We show an example of inter-term generalization of [<Direction>,<Abstract>] → ([],["to"]). Initially, first-term maximum generalization is done. IVs in the node <Abstract> are shown below (see $PTH_i(2)$ in Fig. 3).

([],["to"]) : 0.027
([],["in"]) : 0.020
([],["on"]) : 0.006

IVs in the node <Abstract>, which is the parent node of <Direction>, are shown below (see $PTH_i(1)$ in Fig. 2).

([],["in"]) : 0.192
([],["on"]) : 0.035
([],["to"]) : 0.007

Their totals are as follows.

([],["to"]) : 0.027 + 0.007 = 0.034
([],["in"]) : 0.020 + 0.192 = 0.212
([],["on"]) : 0.006 + 0.035 = 0.041

Since ([],["to"]) doesn't have the highest importance, the case is not generalized any further in the first term.

Next, the second term is generalized. The IVs in the node <Direction> are shown below (see Fig. 2).

([],["to"]) : 0.1

IVs in the node <>, which is the parent node of <Abstract>, are shown below (see Fig. 3).

([],["in"]) : 0.011
([],["to"]) : 0.008
([],["on"]) : 0.006

Their totals are as follows.

[<>,<>] → ([],["in"]) 0.118
[<*X*>,<>] → ([],["on"]) 0.306
[<Concrete>,<Abstract>] → ([],["to"]) 0.037
[<Location>,<>] → ([],["to"]) 0.108
[<Direction>,<>] → ([],["to"]) 0.108

Figure 5: Result of the Inter-Term Generalization

([],["to"]) : 0.1 + 0.008 = 0.108
([],["in"]) : 0 + 0.011 = 0.011
([],["on"]) : 0 + 0.006 = 0.006

Since ([],["to"]) has the highest importance, the second term is generalized into the root node <>, and the generalization stops because there are no more parent nodes. Therefore [<Direction>,<>] → ([],["to"]) 0.108 is the result of first-term maximum generalization of [<Direction>,<Abstract>] → ([],["to"]).

The result of inter-term generalization for all the $LTCP_i$s in Fig. 1 is shown in Fig. 5.

3.2.6 Addition of Translation Rules

Finally, translation rules (TR_is) are added to the set of $GLTPC_i$s. TR_is are descriptions in which concepts are specified as the values of variables of L_s of $LTPC_i$s. If the same case already exists in the set of $GLTPC_i$, then it is not added. If only the value of the case is different from TR_i, then it is replaced by TR_i. Otherwise, TR_i is added with its IC. The ICs for TR_is are calculated in the same way as for $GLTPC_i$s.

4 Best-Matching Algorithm

The TP_is, the set of $GLTPC_i$s, the TPH_is, and the thesaurus are used in best matching. The values of variables in V_s are extracted from the input sentence by applying pattern matching according to the description of TP_i. The best-matching process retrieves the most similar case from the set of $GLTPC_i$.

If $M_i = 1$, words which are equivalent to the word that is a value of the variable in V_s are first searched for in the value of the corresponding variable in L_s of $GLTPC_i$s. If none are found, upper concepts retrieved in either $TPTH_i$ or the thesaurus are searched in turn. The $GLTPC_i$ which is found first is the shortest-distance $GLTPC_i$ ($SDGLTPC_i$). If C_s in $GLTPC_i$ is not null, then it is also evaluated, whether it is true or false.

If $M_i > 1$, the j-th term shortest-distance $GLTPC_i$s ($SDGLTPC_j$) of each term are searched for. If $M_i = 2$, $SDGLTPC_1$ holds the shortest-distance word or concept in the first term, and $SDGLTPC_2$ holds the shortest-distance word or concept in the second term. If $M_i > 1$, $(M_i - 1)$ $SDLTPC_j$s are obtained for each j-th term. A total of $M_i \times (M_i - 1)$ $SDLTPC_j$s are obtained. The $SDGLTPC_j$ with the highest importance is selected as the $SDGLTPC$.

We will show an example in retrieving the most similar example for "Getuyou(Monday)ni-Huru(rain)." Suppose the parent node of "Huru" is <Climate>. At first, $SDGLTPC_1$ will be searched for in $GLTPC_i$s (see Fig. 5). "Getuyou" does not exist in any first terms in the set of $GLTPC_i$s. Therfore <*X*> which is the parent node of "Getuyou" is searched for and [<*X*>,<>] → ([],["on"]) 0.306 is found. The second term of this $GLTPC_i$ is a upper concept of "Huru," so this is $SDGLTPC_1$. Next, $SDGLTPC_2$ is searched for and is found to be the same as $SDGTPC_1$. Consequently, the most similar $GLTPC_i$ is [<*X*>,<>] → ([],["on"]) 0.306, and the word "on" is set as a preposition.

5 Discussion

In the CBMT approach, the linguistic model, which is a set of translation patterns, is important both for the compaction ratio of a case-base and for similarity metrics. If the model is not appropriate, most cases remains ungeneralized, and unnatural cases are retrieved as similar cases to the input. The problems of constructing linguistic models are the same as in rule-based systems.

However, our approach assumes that the linguistic model does not include controls of rules and generalized cases. Whether or not this assumption is correct, it is very difficult to define controls in such a way that any exceptional cases are encapsulated properly. Our approach provides an engineering solution to these difficulties.

In our approach, the quality of translations depends on the quantity of cases rather than the quality of the thesaurus. Therefore, it is important to explore (semi-)automatic case acquisition from bilingual corpora.

To construct a huge case-base is easier than to construct a well-defined thesaurus, because cases are constructed locally without taking account of side-effects. To define an effective thesaurus for translation, every effective category for translation must be included, and every intermediate category that is effective for translation must be included in order to calculate semantic distances properly.

If, on the other hand, thesauri can be developed independently from the case-base, developers or users can select the most appropriate thesaurus for the domain.

6 Concluding Remarks

This paper has described a framework for a machine translation using a mixture of rules and cases generalized by means of a thesaurus, which is much smaller than the case-base itself. Since the importances of rules and generalized cases are calculated in advance by generalization, it is not necessary to calculate them during the best-matching, which is done by exact matching of words or upper concepts in the thesaurus.

Acknowledgements

I would like to thank Masayuki Morohashi and the members of Japanese Processing Group of Tokyo Research Laboratory, IBM Japan, for their valuable suggestions and encouragement, and Michael McDonald for his helpful advice on the wording of this paper.

References

[1] Nagao, M., "A Framework of a Mechanical Translation between Japanese and English by Analogy Principle," Elithorn, A. and Banerji, R. (eds.): *Artificial and Human Intelligence*, NATO, 1984.

[2] Sadler, V., "Working with Analogical Semantics: Disambiguation Techniques in DLT," FORIS Publications, 1989.

[3] Sato, S., and Nagao, M., "Toward Memory-based Translation," Proc. of Coling '90.

[4] Sumita, E., Iida, H., and Kohyama, H., "Translating with Examples: A New Approach to Machine Translation," Proc. of the 3rd Int. Conf. on Theoretical and Methodological Issues in Machine Translation of Natural Languages, 1990.

[5] Watanabe, H., "A Similarity-Driven Transfer System," Proc. of Coling '92.

[6] Stanfill, C. and Waltz, D., "Toward Memory-Based Reasoning," Comm. of ACM, Vol.29, No.12, pp. 1213-1228, 1986.