Example-Based Machine Translation at Carnegie Mellon University

Ralf D. Brown
Carnegie Mellon University, Language Technologies Institute
5000 Forbes Avenue
Pittsburgh, PA 15213-3890
ralf+@cs.cmu.edu

January 22, 2000

Example-Based Machine Translation (EBMT) is fundamentally translation by analogy. Given a corpus of pretranslated examples, one translates previously-unseen text by finding the best match(es) in the corpus and using the associated translation(s). If this sounds a lot like a translation memory (TM) system, it is because EBMT is a superset of translation memory.

There are a variety of approaches to finding the "best match" for a new sentence. One can parse the corpus into syntactic parse trees and match the trees [5]. One can find the nearest single match for the complete input (as is typically done in translation memory systems) and then attempt to modify both the matched example and its associated translation to create an exact match for the input [6]. Or one can find the complete set of exact phrasal matches and piece together a full translation from the fragments, which is the approach taken at Carnegie Mellon for both the Pangloss and DIPLOMAT projects[2].

The advantage of using exact phrasal matching is that one does not need to build a parser (as would be the case for parse-tree matching) or implement language-specific modification rules. As a result, our EBMT system is perfectly suited for rapid development of new language pairs – given a large sentence-aligned corpus such as the Hansards[4], an initial version of a new bidirectional translator can be created in one or two *days*.

The disadvantage of exact matching is that it requires several million words of parallel text for reasonably broad coverage of unrestricted text, which may be difficult to obtain. Thus, we have embarked on a project (funded by the National Science Foundation) to add generalization of the pretranslated examples, thereby dramatically reducing the required size of the training corpus. These generalizations may be syntactic (noun, verb, adjective) or semantic (weekday, color, company, country, etc.), and are implemented by pattern replacement in both the corpus and the input to be translated.

The underlying idea behind generalization is that there are equivalence classes of words and phrases that may be used interchangeably in a particular context and still yield grammatical results. For instance, any pretranslated example which uses the word "Monday" could be modified to use "Tuesday" instead (or any other day of the week). At a more general level, if one can identify noun phrases in the source-language text, one can then (within limits) substitute any other noun phrase whereever a noun phrase occurs. One does not need expert linguists to create a full grammar of the language – a small subset grammar capturing the most frequent patterns and phenomena will suffice; it can always be extended later as needs and available resources dictate.

The basic algorithm in the Carnegie Mellon EBMT system is to search the example base for the largest phrases from the input which are contained in each pre-translated sentence. For each match,

;;;(TOKEN <NOUN-M>)	;;;(TOKEN <NP-M>)
book	the <NOUN-M>
livre	le <NOUN-M>
;;;(TOKEN <NP-M>)	;;;(TOKEN <NP-F>)
<POSS> <ADJ-M> <NOUN-M>	the <NOUN-F>
<POSS> <NOUN-M> <ADJ-M>	la <NOUN-F>

Figure 1: Sample English-French Production Rules

the corresponding translation is determined by performing a word-level alignment [2] of the two halves of the translation example. The overall translation is assembled from the partial translations using a statistical language modeler in our multi-engine machine-translation architecture [1].

This partial exact matching is extended by allowing the equivalence classes mentioned above. Equivalence classes are applied by replacing any matching words or phrases with the name of the equivalence class, appending a disambiguating number if that equivalence class has already been used in the sentence (referred to as tokenizing in the remainder of this article). The process is repeated until no more replacements are possible, at which time a partial exact match against the example base is performed, just as previously without equivalence classes. Since the example base has also been tokenized, this allows interchangeable use of the members of an equivalence class.

In the input to be translated, phrases belonging to an equivalence class are *always* replaced by the class name. In the example base, the class members are only replaced if an appropriate translation is present in the target-language half of the example. To permit proper matching against the example base, ambiguous "words" are permitted which match any one of several alternatives at that location. Whenever a single word is replaced by its class name, the original word is retained as an alternative for matching; unfortunately, this is not possible for phrases as the difference in length would cause erroneous matches when examining the index. This capability for ambiguous terms also allows words to be in multiple equivalence classes provided that the translations are mutually distinct. (If there were a common translation between different equivalence classes, the system would be unable to decide which to use).

Whenever a term is replaced by its class name, the corresponding translation is remembered. Once a translation of the tokenized text has been found, each token is expanded by substituting the translation which was remembered when the text was initially tokenized. This back-substitution step yields the final translation which is output, and is what makes equivalence classes work.

As an example, consider the sentence

```
John Miller flew to Frankfurt on December 3rd.
```

This becomes

```
<firstname-m> <lastname> flew to <city> on <month> <ordinal>.
```

after an initial tokenization pass, and then

```
<person-m> flew to <city> on <date>.
```

after a second pass. The tokenized form will now match

```
Dr.  Howard Johnson flew to Ithaca on 7 April 1997.
```

among many other possibilities.

A further generalization of equivalence classes involves repeated (recursive) matching against the example base. For this extension, translation pairs in the example base are tagged with a token which preferably contains linguistic information such as gender and number. Tagged entries are not limited to literal strings – they may themselves contain tokens, allowing the use of paired

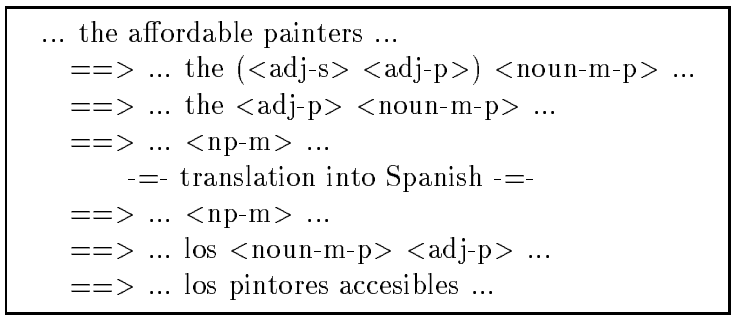

Figure 2: Disambiguation through Linguistic Constraints

production rules to create a grammar, as shown in Figure 1.

To perform a translation, the system first searches for phrases that completely match one or more tagged entries, and then substitutes the associated tags into the input text. This process is repeated until there are no more complete matches of tagged entries, at which point an extended form of the normal partial-exact match against all examples – including tagged entries – in the knowledge base is performed. As is the case when using equivalence classes, at each step the appropriate back-substitution is remembered so that it can be applied to the tokenized translation in order to produce the final output.

The process of matching against the corpus is more complex when grammar rules are involved, because not all alternative terms which will be matched represent the same number of words in the input. Each of the individual substitutions produced at any stage of the repeated tokenization described above may be matched against the corpus. Recursive matching permits a word to be in multiple equivalence classes even when the translations are not distinct, and can be applied more generally than simple tokenization because replacements are only made in the proper context.

When the tags contain linguistic information, this information can be used to enforce constraints and thus select the appropriate translation of a word. For the example shown in Figure 2, the English word "affordable" can be translated as either a singular or plural adjective; this is indicated by showing all alternatives for a given word as a list in parentheses. After a first recursive matching pass, both "affordable" and "painters" are tokenized. Searching the corpus for a further tagged match of this initial result yields only the masculine noun phrase in which both adjective and noun are plural, which disambiguates "affordable" as the plural form. Once no more matches are possible, the translation of the fully-tokenized input is determined, and the tokenization is reversed by back-substituting the appropriate translation for each tokenized term, as remembered during tokenization. The final result is a translation in which the correct alternative has been selected.

The effort of adding the grammar rules and linguistic information was quite modest, totalling an estimated 70-80 hours for the French system and 50-60 hours for the Spanish system. While the availability of morphological information for both French and Spanish considerably reduced the level of effort, for many language pairs much of the work can be performed automatically even without such data, given a bilingual dictionary which is required anyway. By matching suffixes or other lexical features, as was done for the Spanish system (and to a lesser extent for the French system), many of the most frequent morphological variations can be captured. Work is also underway on a method for automatically learning equivalence classes from the training corpus, which will significantly reduce the manual effort involved in adding linguistic information to generalize the corpus.

Adding equivalence classes produces a small but noticeable improvement, and the greater infusion of generalization due to recursive matches produces a greater improvement. A series of

Input:
La motion de M. Lewis est adoptée par 147 voix contre 77.
(*Mr. Lewis' motion is adopted by 147 votes to 77.*)

707,000-word corpus, no generalization:

"la motion de m . lewis" (1)	"motion of Mr . Lewis"
"adoptée par" (0)	"adopted by"
"147 voix" (0)	"147 votes"
"77 ." (0)	"77 ."

307,000-word corpus, with full generalization:

"la motion de m ." (2.21)	"the motion for Mister"
"motion de m . lewis" (0.4)	"Mister Lewis's motion"
"est adoptée par 147" (1.825)	"is adopted by 147"
"par 147 voix contre 77 ." (1)	"by 147 voice against 77 ."

Figure 3: Comparison With and Without Generalization

experiments published last summer [3] showed as much as an order of magnitude reduction in the amount of training text required to be able to translate a given percentage of arbitrary input. Our French system reaches 80% coverage of the test text with less than 300,000 words of training material (nearly all of which consists of grammar rules and morphological entries) when using recursive matching, but requires one million words without grammar rules and 1.2 million words when relying solely on the translation examples. Since the performance curve flattens out, the difference is even greater for higher coverage values – achieving 90% coverage required less than half a million words with recursive matching versus 7 million words without. Similarly, the Spanish system reaches 80% coverage with only 350,000 words versus 2.5 million words and 90% with only 3 million words versus more than 11 million without generalization.

Translation quality is marginally lower when using the grammar rules, since it is easy to overgeneralize. A further cause of reduced quality is that generalizations only produce a single, preferred translation, rather than a number of closely related translations that may vary according to context.

Figure 3 illustrates the effect of generalization on the system's output. For this example, the system was told to use very terse output: only the very best-scoring translation for any match is shown, and no matches which are entirely contained within another, larger match are shown. The left-hand column indicates which phrase was matched and the penalty score for the generated translation (zero is considered perfect), while the right-hand column shows the translation. With less than half as much parallel text (including morphological and grammar entries), the generalized version covers more of the input, with generally longer matches, but also with lower quality. Thus, "voix" is translated as "voice" rather than "votes" because the generalization rules do not take the nature of the corpus (parliamentary proceedings) into account. The much smaller match without generalization, on the other hand, correctly uses "votes" because that is the usage in the corpus.

While increasing the effectiveness of the available translation examples is an interesting result for major languages such as French and Spanish, it is *vital* for languages which have little or no available parallel text; for such languages, being able to generalize the examples which can be found or manually generated may be what makes a translation system feasible at all.

References

[1] Ralf Brown and Robert Frederking. Applying Statistical English Language Modeling to Symbolic Machine Translation. In *Proceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-95)*, pages 221–239, Leuven, Belgium, July 1995. `http://www.cs.cmu.edu/~ralf/papers.html`.

[2] Ralf D. Brown. Example-Based Machine Translation in the PANGLOSS System. In *Proceedings of the Sixteenth International Conference on Computational Linguistics*, pages 169–174, Copenhagen, Denmark, 1996. `http://www.cs.cmu.edu/~ralf/papers.html`.

[3] Ralf D. Brown. Adding Linguistic Knowledge to a Lexical Example-Based Translation System. In *Proceedings of the Eighth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-99)*, pages 22–32, Chester, England, August 1999. `http://-www.cs.cmu.edu/~ralf/papers.html`.

[4] Linguistic Data Consortium. *Hansard Corpus of Parallel English and French.* Linguistic Data Consortium, December 1997. `http://www.ldc.upenn.edu/`.

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

[6] Tony Veale and Andy Way. Gaijin: A Template-Driven Bootstrapping Approach to Example-Based Machine Translation. In *Proceedings of the NeMNLP'97, New Methods in Natural Langauge Processessing*, Sofia, Bulgaria, September 1997. `http://www.compapp.dcu.ie/-~tonyv/papers/gaijin.html`.