[*Proceedings of the Evaluators' Forum*, April 21st – 24th, 1991, Les Rasses, Vaud, Switzerland; ed. Kirsten Falkedal (Geneva: ISSCO).]

Developer-Oriented Evaluation of MT Systems

Andrew Way
Eurotra Project
Department of Language and Linguistics
University of Essex

Abstract

The issue of developer-oriented evaluation of MT systems has received little attention in the past. We assume that one of the main tools at the developer's disposal is a **test suite** of sentences. The best known appears in [2], and this particular suite of sentences is investigated here, together with discussions regarding the number of suites required and the usefulness of test suites in general.

1 Introduction

Currently one of the main areas of study of our research group at Essex is the evaluation of MT systems. We have principally been concerned with user-oriented evaluation of commercially available MT systems, in particular "Globalink", and have examined both declarative [4, 3] and operational [5, 6] means of evaluation.

Although there has been a good deal of discussion regarding the use of test suites, most of these papers focus on their usefulness for end users rather than for system developers [1, 7]. In this paper the problems particular to system developers are discussed, as are the options open to them for testing and evaluating their modules. We assume that one of the main tools at the developer's disposal is a test suite of sentences. There is a significant lack of such suites available, of which the best known appears in [2]. We discuss here this particular suite of sentences as well as the usefulness of test suites in general. Consideration is also given to the question of how many test suites are needed, given that most if not all which have been published to date have been monolingual.

2 Types of Evaluation

Three broad classes of MT evaluation strategy may be discerned:

Typological Evaluation seeks to specify which particular linguistic constructions the system handles satisfactorily and which it does not. The principal tool for such an investigation is a *test suite* – a set of sentences which individually represent specified constructions (e.g. Dative-shift passive) and hence constitute performance probes.

Declarative Evaluation seeks to specify how an MT system performs relative to various dimensions of translation quality.

Operational Evaluation seeks to establish how effective an MT system is likely to be (i.e. in terms of cost effects) as part of a given translation process.

Detailed descriptions of both declarative [3, 8] and operational evaluation [5, 11] appear elsewhere and, being more oriented towards the needs of the potential user are beyond the scope of this paper.

2.1 Typological Evaluation

Typological evaluation is primarily of interest to system developers: potential users may not be familiar with the linguistic descriptions used, nor is it likely to be apparent how frequently some missing or badly-handled construction might occur in their particular text type.

We assume that the principal tool at the developer's disposal for testing his system is a suite of sentences. These sentences exemplify particular types of linguistic constructions that the system is likely to encounter in its lifetime. If the system is intended to operate within a particular subject field then it is obvious that its design will reflect this sublanguage (i.e. if the MT system is intended to translate car manuals then it is much more important for it to be able to analyse and translate imperatives rather than intricate sentences containing relative clauses, modifiers and transconstructionals, and coordinate structures, for instance).

We ignore here traditional objections to approaches advocating the use of test suites, including the fact that as test suites concentrate on syntactic constructions rather than lexical coverage, some other means of assessing this coverage is required. If we include different verb and nounframes in our test suite, then it is a relatively trivial matter to add lexical entries to our MT dictionaries. The test suite approach has an advantage over corpus-based approaches in that the corpus contains a large amount of redundancy, i.e. most constructions will be encountered more than once, whereas in the test

suite each combination of concepts appears once and once only[1], although we still consider that the testing of the MT system against a corpus constitutes the final stage in its development.

Once a corpus has been established for the task in hand, statistical information regarding the type and frequency of lexical and grammatical phenomena contained therein should be obtained in order to be able to evaluate the capability of the system to successfully translate sentences contained in the corpus. If good observed frequency data were not available then this would probably lead to an over (or under) estimate of the system's potential. At present, however, this statistical information will almost certainly be gained by hand (a laborious process), as the necessary tool capable of parsing texts in this way is not available.

Assuming that we have established the relative frequency of the phenomena contained in the corpus, the test suite can now be constructed. One point to consider before beginning this task is how many suites one needs. For instance, the simplest set of sentences will be those which contain one linguistic concept (this set will be very small – simple imperatives, perhaps – and of little worth), or perhaps slightly more realistically two in combination. Such a suite would be of use to the developer in that he would be able to see if the rules he has just written can handle that concept in isolation. This is merely a first step, of course.

Problems arise when one attempts to integrate new rules into the grammar as a whole, for they may combine in a wrong or unexpected manner with existing rules. One might therefore decide in favour of a new suite to test this phase of the development process. We know that this new set of sentences has again to faithfully represent the statistical data already gathered, i.e. if the most common combination of concepts in a corpus proves to be **Subject-Verb Agreement** and **Simple Present Tense** then it would be disastrous if one's system failed to correctly analyse sentences containing such constructions. Obviously sentences which at first sight seem fairly simple involve the interaction of a number of linguistic concepts, viz:

(1) These men might not be paid for a week.

This sentence contains the following combination of linguistic concepts:

- Number agreement between determiner and noun
- Subject-Verb Agreement
- Tense
- Modality

[1] For other refutations of such criticisms, see [2] pp. 2-3

- Negation
- Passive
- Argument-Modifier distinction (cf "for a week" vs "for their work")

As can be seen, it is very easy to increase the complexity of the sentence by adding what on the surface level seem to be trivialities. In developing a test suite one has to limit the number of combinations of concepts, else the sentences become intolerably difficult. To the above lists we could add **Comparison, Coordination, Subclauses, Participial Constructions, Raising and Control Verbs, Reflexives, Support Verbs and Long Distance Dependencies,** all of which can be combined in various ways to produce complicated sentences which our MT systems have no hope of translating in a reasonable time[2]. How do we count these concepts, and assuming we can, what upper limit is it reasonable to impose on the combination of linguistic concepts occurring in the sentences of one's test suite?

As to the first question, the simplest position to take is to say that the presence of any of the above list (which is, of course, incomplete) adds to the complexity of the sentence by a factor of one, so if we add a relative clause to a main clause then the resultant sentence becomes more difficult to translate when the relative clause itself contains more of these linguistic concepts than another. For instance, if we examine the following:

(2) a. The terrorist who might have been responsible for the explosion was captured by the police.
b. The class that I teach is advanced.

one can see that both the main clause and relative clause in (a) contain more concepts than those in (b), and so we can safely deduce that it is a more complex sentence. We all know, however, that certain concepts pose more problems than others – coordination is a notorious example here – so in some instances we would need to allocate a weighting to each concept occurring in a sentence. It is simple enough to state that a sentence containing coordination is more complex than a similar sentence without it, but which of the following is easiest to translate:

(3) a. John may be coming.
b. John is not coming.

i.e. is it easier to implement modal verbs than negation? This is not easy to say.

[2] We all know the syndrome: there is always one person at a demo who has thought up such a sentence, and it is invariably the case that this person knows nothing about the practicalities of MT.

Regarding the second question, five or six might be a first approximation, depending on the degree of difficulty associated with each phenomenon. For instance, **Tense** occurs in (almost) every sentence, but it is obviously much easier to implement the Simple Present than the English Auxiliary System. Another example is that it is occasionally stated that Subject-Verb Agreement is little short of irrelevant for English as the number of the subject is already given (from the lexicon). Nevertheless in more complicated sentences such as:

(4) I know a lecturer with clever students who try/tries to obtain grants from the Government.

if one's rules did not explicitly state that the subject had to agree with the verb then the meanings of the above would be indistinguishable to one's system. If, however, it was the case that such sentences occurred only rarely (or not at all) in one's corpus then one might be fully justified in deciding to omit Subject-Verb Agreement from one's coverage[3].

This raises another point that affects all MT systems which are not reversible (i.e. which use different grammars for analysis and generation). Analysis and Synthesis are different tasks, so should one test them with different sets of sentences? Examine the following sentences:

(5) a. Even from 1965 accessible intercontinental telephone links were simply created by undersea cable.
b. The next good financial figures are due on Monday.

In the first example above, the analysis component must produce **all** permissible structures, i.e. with the sentential modifiers in different positions, whereas in synthesis one can choose to generate only one of these. Not all problems are, however, exacerbated in analysis. It is a reasonable position for writers of an Analysis component to take that they need not concern themselves with filtering out ungrammatical input, but this remains the principal task of the Synthesis writer. Thus the order of the string of adjectives in the second example above poses no problem for analysis, as one merely assumes that the sentence will be input to the system in the form above, but for synthesis one must explicitly rule out illegal combinations such as "The good financial next figures".

Given these facts, it seems that we are forced to posit different suites depending on the task in hand. It makes little sense for an analysis component to be tested against ungrammatical input (except that of course the component should not "parse" such input, i.e. assign it structure) when one can reasonably assume that the module will never be confronted with it. Con-

[3] This is the approach taken by the writers of the English Analysis module at UMIST.

versely we assume that a suite designed for testing a generation component **must** contain ill-formed strings so that the filtering out of such input by that module can be tested[4].

One further point to consider is whether the various test suites contain purely monolingual sentences. This is obviously the case with the analysis suite, but what about synthesis? Some MT systems can easily be adapted to parsers (e.g. in a multilevel system a particular level can be loaded and then tested by adapting the input in a manner appropriate for that level), and so such systems can also be fed monolingual input. There are, however, many systems where this is not possible, so some other strategy is required.

If we decide to create a testbank to test transfer and target language, there are obvious advantages and disadvantages in choosing a set of monolingual sentences. The advantage is that the test is realistic, for the input to transfer is the output of the analysis phase, and as we have already constructed an analysis test suite this serves to test transfer as well[5]. The problem is that the suite will also need to contain "artificial" sentences, i.e. sentences in the **source** language containing constructions which the developer knows **in advance** will be problematic for the **target** language (cases of "head-switching", for instance [9, 10]). Even then, we need to be sure that a sentence in the source language which is intended to test a particular phenomenon in the target language actually attempts to produce that target sentence, rather than some unforeseen paraphrase.

3 Conclusions

What types of test suites arise from this discussion? For a non-reversible transfer-based MT system it seems as if we need at least the following:

- An initial development suite
- An analysis suite
- A synthesis suite
- A transfer suite

[4] It is assumed here that a system which can inherently distinguish between correct and ill-formed input is preferable to one which cannot. As Flickinger et al state (p.4, [2]), "Although it is clearly desirable to be as tolerant of user mistakes as is practical, it is more important for the system to correctly interpret well-formed input" and "... a system that fails to detect ungrammaticality will introduce ambiguity and therefore interpret incorrectly".

[5] Another point to note is that where systems have different grammars for analysis and synthesis, one can use the output of analysis as direct input to the synthesis component of the same language if both components use the same set of features.

The development suite is used in the initial stages to test rules to see if they cope with the particular linguistic construction in mind in isolation. The analysis suite is monolingual, and contains purely grammatical strings. The synthesis suite is preferably monolingual, and contains both grammatical and ill-formed strings[6]. The transfer suite is constructed possibly from the analysis suite in addition to some artificially created source language sentences purely to test the target language.

If we are fortunate it may well prove possible to have just one basic suite per language; given that we have to test the analysis component against the main testbank, we can collect the objects at the level preceding transfer and use them together with some ill-formed data as the synthesis test suite. The same output of analysis is extended with a set of objects designed to test transfer and target language. This was the strategy selected in [12].

The final stage of testing is to select input freely from the chosen corpus, not just testing each and only each sentence from the corpus (such a corpus could be translated by constructing a sentence dictionary, for instance). This stage should indicate to us the stability of the system with regard to the chosen sublanguage and its suitability for use in small applications.

References

[1] Kirsten Falkedal & Maghi King. *Using test suites in evaluation of machine translation systems.* "13th International Conference on Computational Linguistics", pp. 211-216, 1990.

[2] Dan Flickinger, John Nerbonne, Ivan Sag & Tom Wasow. *Toward evaluation of NLP systems.* "25th Annual Meeting of the Association for Computational Linguistics", 1987.

[3] Essex MT Evaluation Group. *Assessing a PC-based commercial MT system.* Technical report, The University of Essex, 1991.

[4] Essex MT Evaluation Group. *The Globalink Translation System.* "Personal Computer World", pp. 162-166, 1991.

[5] Essex MT Evaluation Group. *Operational Evaluation of MT Systems.* Technical Report, The University of Essex, 1991, forthcoming.

[6] R. Lee Humphreys. *User-Oriented Evaluation of MT Systems.* Technical Report, The University of Essex, 1991.

[6] These may not be sentences, of course, depending on the input to synthesis for a particular MT system.

[7] John Lehrberger & Laurent Bourbeau. *Machine Translation: Linguistic Characteristics of MT Systems and General Methodology of Evaluation.* John Benjamins, Amsterdam, 1987.

[8] John R. Pierce, John B. Carroll, et al. *Language and Machines - Computers in Translation and Linguistics.* (ALPAC Report), 1966.

[9] Louisa Sadler, Ian Crookston, Doug Arnold & Andy Way. *LFG and Translation.* "Third International Conference on Theoretical and Methodological Issues in Machine Translation" 11-13 June 1990, 1990.

[10] Louisa Sadler, Ian Crookston & Andy Way. *Co-description, projection and 'difficul'' translation.* Technical report, Department of Language and Linguistics, University of Essex, December, 1989.

[11] G. van Slype. *Conception d'une méthodologie générale d'évaluation de la traduction automatique.* "Multilingua", (l-4):221-237, 1982.

[12] Andrew Way. *A practical developer-oriented evaluation of two MT systems.* Technical report, The University of Essex, 1991.