[*Translating and the Computer 19*. Papers from the Aslib conference… 13 & 14 November 1997]

EVALUATION OF LANGUAGE TOOLS

Bente Maegaard
Center for Sprogteknologi, Njalsgade 80, DK-2300 Copenhagen S, Denmark.
E-mail: bente@cst.ku.dk

Abstract: In this paper we describe the evaluation methodology developed by EAGLES/TEMAA and give indications about its possible application on translation tools.

INTRODUCTION

One of the effects of language technology becoming more and more widespread and more important for more people is the growing interest in evaluation techniques for language technology products. Evaluation of computational linguistic programs have existed in the research world for as long as computational linguistics has existed: one parser is faster than others, one parser gives richer information, one linguistic theory gives better explanations about language etc. Evaluation of MT has existed at least since the ALPAC report 1966 and of course has developed in the course of the years, so that now a clear understanding has been developed that objective, reliable, comparable tests are not only needed, but to a certain extent obtainable.

The first step towards a thorough methodology consists in description, structuring and standardisation of the elements of an evaluation. The next step is to develop as far as possible computer tools to support the testing and evaluation. One of the major sources of input for this paper is the EAGLES and TEMAA projects, EU projects with the goal of providing a framework for the evaluation of language engineering products and projects. EAGLES started and TEMAA further developed the framework and implemented an instance of the methodology for spelling checkers. These projects were a collective effort and I am particularly indebted to the other contributors to the final TEMAA report: Louis des Tombe, Shona Douglas, Maghi King, Steven Krauwer, Sandra Manzi, Patrizia Paggio, Gurli Rohde, Merle Tenney, Nancy Underwood.

WORKING TOWARDS STANDARDS

Commonly agreed evaluation methodologies necessarily build upon standards. Where do we find standards for evaluation? The fact is that ISO in their 9000 series on software production provide guidelines for the evaluation of software, cf. e.g. ISO

1991. This proves a very good starting point for elaborating an evaluation methodology. You can distinguish two types of evaluation: 1) *adequacy evaluation*, where the performance of a product is seen from the point of view of the user/purchaser, and 2) *progress evaluation*, where development progress is measured against plans and previous versions. The ISO documents are mostly concerned with progress evaluation, in that they are aimed at software production environments, whereas in this paper we will be focussing on adequacy evaluation.

The elements of an evaluation methodology are:

- Description of
    - User
    - System
    - Measure
    - Method
- Test materials

and, if possible

- Computer tools for automatic testing and report generation

The description of user, system, method and measure should be formalised as far as possible. Methods for measurement should be both formalised and automated. There are two reasons for attempting formalisation. First of all, formalised descriptions are more easily standardised and it is easier to check conformity. Secondly, formal measures and automated methods are more reliable. In particular automation allows for much larger text samples to be tested and therefore provides much better and more reliable results. Of course, not all characteristics of a system and its behaviour, nor all characteristics of a user role, can be formalised, but it is possible to get quite far.

ISO 9126 *Quality characteristics and guidelines for their use* gives a list of quality characteristics for software: functionality, reliability, usability, efficiency, maintainability and portability. We have added an extra quality characteristic for language tools: customisability. The quality characteristics are formalised as attributes and values; an attribute can be further specified in subattributes etc. This way a system may be described in terms of a structure of attributes.

Exactly the same way, users are described. A user has a certain background and most importantly, a user has a *task* to perform. So, the description of a user hopefully ends up using the same attributes as the system description, and gives desired values or thresholds for the relevant attributes and weightings for each attribute. This description is called a *user profile.* It is an important feature of the EAGLES/TEMAA framework to place the user and the task in such a central role.

The Evaluator's Workbench

A computer program, the *Evaluator's Workbench*, was made which can take a system and the associated list of measurable attributes, a list of appropriate tests and carry out the tests using existing data sets. If the system being evaluated is a spelling checker - which was the case for the TEMAA project – the user tasks or roles can be described as the writer, the end user, the reader, the customer. A writer in turn may have characteristics as technical writer, native speaker, foreign language speaker. The reason that we need to distinguish these different classes of users is their different language competence which leads to different importance of e.g. the spelling checker's suggestion adequacy ('how often is the first suggestion for a wrongly spelled word correct? how often is the correct word among those suggested at all?).

At CST we produced test methods and test data for the spelling checkers for Danish. To give some examples of test materials produced: 1) A spelling checker is supposed to flag only errors, so at least all common words must be known to the system, in order for it not to wrongly flag correct words. So, a list of commonly used words was constructed. 2) A spelling checker should find all spelling mistakes in a text. In order to automate the testing of this feature, lists of correct words were corrupted following rules of misspelling and the spelling checker was run on the list of misspelled words, automatically comparing the result with the original correct word.

As can be seen, testing of spelling checkers can be largely automatised this way. The Evaluator's Workbench then compares the result of the testing with the user profile and computes a report which gives the result of the evaluation of a specific spelling checker for a given user profile, i.e. when different spelling checkers have been tested, this report gives the input for the user to choose the most adequate system for a certain task/class of tasks.

EVALUATOR'S WORKBENCH FOR TRANSLATION TOOLS?

Above, we have seen the methodology used for spelling checkers. Spelling checkers are probably the simplest possible language tool. So, the interesting question is: Can this methodology be used also for more complicated systems and user roles, e.g. for grammar checkers, for information retrieval, message understanding, and for what is close to the heart of the Aslib audience: translation tools?

Translation is different from spelling in that whereas only one correct spelling of a word exists (or in some cases there are two alternative approved spellings), there will normally be more than one correct translation of a sentence. Also, the borderline between correct and incorrect translation is not so clearcut, which means that it will be more difficult to set up an automatic testing as

described above. However, even if it is difficult, one should not necessarily give up, and much can be achieved by formalising the features of systems and users.

The EAGLES subgroup on translation tools studied translation memories in particular. Feature check list examples were drawn up as e.g. the following for translation memory update and maintenance: Alignment, Importing an aligned SL- and TL-segment into the translation memory database, Adding an SL-segment and its translation to a TM while translating in TM mode, Modifying existing contents of TMs (apart from adding/importing). Each of these questions is broken down into a number of more detailed questions.

User profiles are described first by the *dimensions of translation,* i.e. quantity of translated text, text type, text domain, languages involved, translation quality, language policy of the organisation etc, then by *typical user profiles* such as freelance translator, small translation company, large translation company, translation department in small or large organisation, bilingual organisation etc. Typical user profiles in combination with dimensions of translation then provides input for the user requirements for the translation tool to be evaluated. Some of the evaluation will be factual in this case, i.e. the questions are answered by yes/no, no testing involving large data sets is involved. Others involve the building of test data sets and consequently may turn out to be a rather demanding job.

For machine translation systems, we are in a worse position wrt the production of test data and the evaluation itself, since only a minor part of the questions are factual (languages treated etc.). Here, a battery of test material, probably test suites as well as real text needs to be established, preferably together with the 'correct' translations. This opens the discussion about degrees of correctness and the possibility of measuring in a formal and automatic way. At the same time it opens the question about domain and test type orientedness of the system -- personally I would not like the PaTrans system which is designed for patent texts to be tested on a political or economic text!

In the absence of other reliable evaluation schemes, it has been suggested that the best way to evaluate an MT system is to measure the performance in terms of productivity gains. I.e. the number of errors is not computed (cf. the problems of defining a way of counting errors), but only the time it takes to obtain a correct translation. This is done e.g. by Lingtech while at the same time, the nature of the errors is analysed in order to provide feedback to the developers.

In conclusion, there is some way to go, but the development has started, and within some years, benchmarking for MT will exist, and it will build upon features as the ones outlined above.

List of references

EAGLES Evaluation of Natural Language Processing Systems, Draft, Center for Sprogteknologi, Copenhagen, July 1994. (an updated version is underway).

Hutchins, J.: Evaluation of Machine Translation and Translation Tools. In: Varile, N., A. Zampolli: *Survey of the State-of-the-Art in Human Language Technology*, Cambridge University Press, 1997.

Lehrberger, J. and L. Bourbeau: *Machine Translation: linguistic characteristics of MT systems and general methodology of evaluation.* John Benjamins, Amsterdam, Philadelphia, 1988.

ISO 9126 *Quality characteristics and guidelines for their use,* ISO, Geneva, 1991.

Maegaard, B. and V. Hansen: Lingtech/CST Collaboration: Patently Successful. In: *ELSNews, Vol 4.5,* Edinburgh, 1995.

Nomura, H. and H. Isahara: JEIDA's criteria on machine translation evaluation. In: *Proceedings of the International Symposium on Natural Language Understanding and AI,* Kyushu Institute of Technology, 1992.

TEMAA Final report, Center for Sprogteknologi, Copenhagen, 1996.