Evaluating text quality: judging output texts without a clear source

Anthony Hartley and Donia Scott
Information Technology Research Institute,
University of Brighton
UK
{firstname.lastname}@itri.bton.ac.uk

Abstract

We consider how far two attributes of text quality commonly used in MT evaluation – intelligibility and fidelity – apply within NLG. While the former appears to transfer directly, the latter needs to be completely re-interpreted. We make a crucial distinction between the needs of *symbolic authors* and those of *end-readers*. We describe a form of textual feedback, based on a controlled language used for specifying software requirements that appears well suited to authors' needs, and an approach for incrementally improving the fidelity of this feedback text to the content model.

1 Introduction

Probably the most critical questions that need to be addressed when evaluating automatically generated texts are: does the text actually say what it's supposed to say and is it fluent, coherent, clear and grammatical? The answers to these questions say something important about how good the target texts are and — perhaps more to the point — how good the system that generated them is. There is no *a priori* reason why the target texts should be any better or worse when they result from natural language generation (NLG) or from machine translation (MT): indeed, they could result from the same language generator. Given this, it may be natural to assume that NLG could appropriately adopt evaluation methods developed for its more mature sister, MT. However, while this holds true for issues related to intelligibility (the second critical question), it does not apply as readily to issues of fidelity (the first question). We go beyond our recent experience of evaluating the AGILE system for producing multilingual versions of software user manuals (Hartley, Scott *et al*., 2000; Kruijff *et al*., 2000) and raise some open questions about how best to evaluate the faithfulness of an output text with respect to its input specification.

2 Evaluating intelligibility

The use of rating scales to assess the intelligibility of MT output has been widespread since the early days in the field. Typically, monolingual raters assign a score to each sentence in the output text. However, this does not amount to an agreed methodology, since the number of points on the scale and their definition have varied considerably. For example, Carroll (1966) used a nine-point scale where point 1 was defined as "hopelessly unintelligible" and point 9 as "perfectly clear and intelligible"; Nagao and colleagues (Nagao *et al.*, 1985), in contrast, used a five-point scale, while Arnold and his colleagues (Arnold *et al*., 1994) suggest a four-point discrimination. In evaluating the intelligibility of the AGILE output, we asked professional translators and authors who were native speakers of the languages concerned—Bulgarian, Czech and Russian—to score individual text fragments on a four-point scale. The evaluators were also asked to give a summative assessment of the output's suitability as the first draft of a manual.

In a single pass, AGILE is capable of generating several types of text, each constituting a section of a typical software user manual—i.e., overview, short instructions, full instructions, and functional descriptions—and appearing in one of two styles (personal/direct or impersonal/indirect). We evaluated all of these text types using the same method. The intelligibility evaluation was complemented by an assessment of the grammaticality of the output, conducted by independent native speakers trained in linguistics. Following an approach widely used in MT (e.g., Lehrberger and Bourbeau, 1987), the judges referred to a list of error categories for their annotations.

3 Evaluating fidelity

In MT, evaluating fidelity (or "accuracy") entails a judgment about the extent to which two texts "say the same thing". Usually, the two texts in question are the source (i.e., original) text and the (machine-)translated text and the judges are expert translators who are again invited to rate the relative information content of pairs of sentences on an anchored scale (e.g., Nagao *et al*., 1985). But others (e.g., Caroll, 1966) have also compared the informativeness of the machine translation and a human translation deemed to serve as a benchmark. Interestingly, both of these researchers found a high correlation between the intelligibility evaluations and the fidelity evaluations, which suggests that it may be possible to infer fidelity from the (less costly) evaluation of intelligibility. However, at the current state-of-the-art this approach does not guarantee to detect cases where the translation is perfectly fluent but also quite wrong.

For NLG, the story is rather different. Lacking a source text, we are denied the relatively straightforward approach of detecting discrepancies between artifacts of the same type: texts. The question is, instead, whether the generated text "says the same thing" as the message — i.e., the model of the intended semantic content together with the pragmatic force of the utterance.

The message is clearly only available through an external representation. In translation generally, this external representation is the source text and the task is commonly characterized as *identifying the message* — which originates in the writer's mental model — in order to re-express it in the target language. In an NLG system, the one external representation that is commonly available is the particular domain model that serves as input to the generation system. This model may have been provided directly by an artificial agent, such as an expert system. Alternatively, it may have been constructed by a human agent as the intended instantiation of their mental model. Yet, whatever its origins, directly comparing this intermediate representation to the output text is problematic.

A recent survey of complete NLG systems (Cahill et al., 1999) found that half of the 18 systems examined accepted input directly from another system[1]. A typical example is the Caption Generation System (Mittal et al., 1998), which produces paragraph-sized captions to accompany the complex graphics generated by SAGE (Roth et al., 1994). The input to generation includes definitions of the graphical constituents that are used to by SAGE to convey information: "spaces (e.g., charts, maps, tables), graphemes (e.g., labels, marks, bars), their properties (e.g., color, shape) and encoders—the frames of reference that enable their properties to be interpreted/translated back to data values (e.g., axes, graphical keys)."[2] For obvious reasons, this does not readily lend itself to direct comparison with the generated text caption.

In the remaining half of the systems covered, the domain model is constructed by the user (usually a domain expert) through a technique that has come to be known as *symbolic authoring*: the 'author' uses a specially-built knowledge editor to construct the symbolic source of the target text. These editors are interfaces that allow authors to build the domain model using a representation that is more 'natural' to them than the artificial language of the knowledge base.[3] The purpose of these representations is to provide feedback intended to make the content of the domain model more available to casual inspection than the knowledge representation language of the

[1] By complete systems, we refer to systems that determine both "what to say" and "how to say it", taking as input a specification that is not a hand-crafted simulation of some intermediate representation.
[2] Mittal et al., 1998, pg. 438.
[3] See Scott, Power and Evans, 1998.

domain model. As such, they are obvious candidates as the standard against which to measure the content of the texts that are generated from them.

We first consider the case of feedback presented in graphical mode, and then the option of textual feedback, using the WYSIWYM technology (Power and Scott, 1998; Scott, Power and Evans, 1998). We go on to make recommendations concerning the desirable properties of the feedback text.

4 Graphical representations of content

Symbolic authoring systems typically make use of graphical representations of the content of the domain model—for example, conceptual graphs (Caldwell and Korelsky, 1994). Once trained in the language of the interface, the domain specialist uses standard text-editing devices such as menu selection and navigation with a cursor, together with standard text-editing actions (e.g., select, copy, paste, delete) to create and edit the content specification of the text to be generated in one or several selected languages.

The user of AGILE, conceived to be a specialist in the domain of the particular software for which the manual is required (i.e., CAD/CAM), models the procedures for how to use the software. AGILE's graphical user interface (Hartley, Power *et al.*, 2000) closely resembles the interface that was developed for an earlier system, DRAFTER, which generates software manuals in English and French (Paris *et al.*, 1995). The design of the interface represents the components of the procedures (e.g., goals, methods, preconditions, sub-steps, side-effects) as differently coloured boxes. The user builds a model of the procedures for using the software by constructing a series of nested boxes and assigning labels to them via menus that enable the selection of concepts from the underlying domain ontology.

4.1 The input specification for the user

As part of our evaluation of AGILE, we asked 18 IT professionals[4] to construct a number of predetermined content models of various degrees of complexity and to have the system generate text from them in specified styles in their native language. Since the evaluation was not conducted *in situ* with real CAD/CAM system designers creating real draft manuals, we needed to find a way to describe to the evaluators what domain models we wanted them to build. Among the possible options were to give them a copy of either:

- the desired model as it would appear to them in the interface (e.g., Figure 1);
- the target text that would be produced from the model (e.g., Figure 2);
- a 'pseudo-text' that described the model in a form of English that was closer to the language of the AGILE interface than to fluent English (e.g., Figure 3).

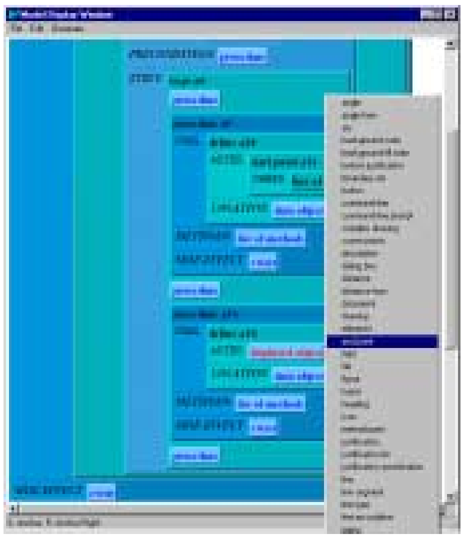

Figure 1: Graphical display of content model

Draw a line by specifying its start and end points.

Figure 2: Target text

To draw a line
    Specify the start point of the line.
    Specify the end point of the line.

Figure 3: Pseudo-text input specification

We rejected the first option because it amounted to a task of replication which could be accomplished successfully even without users having any real understanding of the meaning of

[4] There were six for each of the three Eastern European languages; all had some (albeit limited) experience of CAD/CAM systems and were fluent speakers of English.

the model they were building. Therefore, it would shed no light on how users might be able to build a graphical model externalising their own mental model.

We discarded the second because a text may not necessarily make any explicit linguistic distinction between different components of the model—for example, between a precondition on a method and the first step in a method consisting of several steps[5]. Thus, in general, target texts may not reflect every distinction available in the underlying domain model (without this necessarily causing any confusion in the mind of the reader). As a result of such underspecification, they are ill-suited to serving as a staring point from which a symbolic author could build a formal model.

We opted, then, for providing our evaluators with a pseudo-text in which there was an explicit and regular relationship between the components of the procedures and their pseudo-textual expression. Figure 4 is one of the pseudo-texts used in the evaluation.

Draw **an arc**
First, start-tool the *ARC* command.
M1. Using the *Windows* operating system: choose the *3 Points* option from the *Arc* flyout on the *Draw* toolbar.
M2. Using the *DOS* or *UNIX* operating system:
choose the *Arc* option from the *Draw* menu.
choose *3 Points* option.
Specify **the start point** of the arc.

Figure 4: fragment of a typical pseudo-text

4.2 Evaluating the fidelity of the output

This particular set-up afforded us the possibility of judging the fidelity of the 'translation' between the following representations:

a) desired model and model produced
b) model produced and output text
c) pseudo-text and model produced
d) pseudo-text and the output text

[5] For example, between: "To cook a goose: Before starting, make sure the goose has been plucked. Put the goose in a medium oven for 1.5 hours." and "To cook a goose: First pluck the goose. Then put it in a medium oven for 1.5 hours."

We focused on (a), which was of course mediated by (c); that is, we focused on the issue of creating an accurate model. This is an easier issue than that of the fidelity of the output text to the model (b), while the representations in (d) are too remote from one another to permit useful comparison.

To measure the correspondence between the actual models and the desired/target models, we adopted the *Generation String Accuracy* (GSA) metric (Bangalore, Rambow and Whittaker, 2000; Bangalore and Rambow, 2000) used in evaluating the output of a NLG system. It extends the simple *Word Accuracy* metric suggested in the MT literature (Alshawi et al., 1998), based on the string edit distance between some reference text and the output of the system. As it stands, this metric fails to account for some of the special properties of the text generation task, which involves ordering word tokens. Thus, corrections may involve re-ordering tokens. In order not to penalise a misplaced constituent twice—as both a deletion and an insertion—the generation accuracy metric treats the deletion (D) of a token from one location and its insertion (I) at another location as a single movement (M). The remaining deletions, insertions, and substitutions (S) are counted separately. Generation accuracy is given by the following equation, where R is the number of (word) tokens in the reference text.

$$GenerationAccuracy = \left(1 - \frac{M + I + D + S}{R}\right)$$

For Bangalore and his colleagues, the reference text is the desired text; it is a gold standard given *a priori* by a corpus representing the target output of the system. The generation accuracy of a string from the actual output of the system is computed on the basis of the number of movements, substitutions, deletions and insertions required to edit the string into the desired form.

In our case, the correspondence was measured between models rather than texts, but we found the metric 'portable'. The tokens are no longer textual strings but semantic entities. Although this method provided a useful quantitative measure of the closeness of the fit of the actual generated text to what was intended, it is not without problems, some of which apply irrespective of whether the metric is applied to texts or to semantic models. For example, it does not capture qualitative differences between the generated object and the reference object, that is, it does not distinguish trivial from serious mistakes. Thus, representing an action as the first step in a procedure rather than as a precondition would have less impact on the end-reader's ability to follow the instructions than would representing a goal as a side-effect.[6]

5 Textual representations of content

Once the model they represent becomes moderately complex, graphical representations prove to be difficult to interpret and unwieldy to visualise and manipulate (Kim, 1990; Petre, 1995). WYSIWYM offers an alternative, *textual modality of feedback,* which is more intuitive and natural. As we will discuss below, there is a sense in which, in its current form, the feedback text may be *too* natural.

5.1 Current status of WYSIWYM feedback text

The main purpose of the text generated in *feedback* mode, as currently conceived, is to show the symbolic author the possibilities for further expanding the model under development.

As with AGILE's box representation, clicking on a coloured 'anchor' brings up a menu of legitimate fillers for that particular slot in the content representation. Instantiating green anchors is optional, but all red anchors must be instantiated for a model to be potentially complete (Figure 5). Once this is the case, authors tend to switch to *output* mode, which produces a natural text reflecting the specified model and nothing else.

1. Do <red>this action</red> by using <green>this method</green>.
2. Schedule <red>this event</red> by using <green>this method</green>.
3. Schedule the appointment by using <green>this method</green>.

Figure 5: fragment of a typical feedback text

In WYSIWYM systems the same generator is used to produce both the feedback and output texts; this means that the feedback text can be as fluent as the output text. In its current instantiations, this is precisely what is produced, even when the generator is capable of producing texts of rather different styles for the different purposes.[7]

5.2 Feedback in a controlled language

The motivation for generating a new type of feedback text comes from two sources.

The first is the pseudo-texts that we constructed by hand for the AGILE evaluation. As far as the form of the models actually constructed is concerned, they proved consistently reliable guides for the symbolic authors. Where they proved inadequate was in their identification of multiple references to the same domain model entity; several authors tended to create multiple instances of an entity rather than multiple pointers to a single instance. Let us now turn from the testing scenario, where authors have a defined target to hit, and consider instead a production setting where the author is seeking to record a mental model. It is a simple matter to have the system generate a second feedback text, complementing the present one, this time in the style of the pseudo-texts[8] for the purpose of describing unambiguously, if rebarbatively, the state of a potentially complete model.

The second is Attempto Controlled English (ACE: Fuchs and Schwitter, 1996; Fuchs, Schwertel and Schwitter, 1999), which allows domain specialists to interactively formulate software requirements specifications. The specialists are required to learn a number of compositional rules which they must then apply when writing their specifications. These are parsed by the system.

For all sentences that it accepts, the system creates a paraphrase (Figure 6) that indicates its interpretations by means of brackets. These interpretations concern phenomena like anaphoric reference, conjunction and disjunction, attachment of prepositional phrases, relative clauses and quantifier scope. The user

[6] See Hartley et al (2000) for further discussion of this issue and the results of the AGILE evaluation.

[7] As, for example, in the ICONOCLAST system (see http://www.itri.bton.ac.uk/projects/iconoclast).

[8] *Modulo* the reference problems, for which a solution is indicated below

either accepts the interpretation or rephrases the input to change it.

Input:
The customer enters a card and a numeric personal code. If it is not valid then SM rejects the card.

Paraphrase:
The customer enters a card and [the customer enters] a numeric personal code. If [the personal code] is not valid then [SimpleMat] rejects the card.

Figure 6: ACE paraphrases

The principle of making interpretations explicit appears to be good one in the NLG context too, especially for the person constructing the domain model. Moreover, in the context where the output text is required to be in a controlled language, the use of WYSIWYM relieves the symbolic author of the burden of learning the specialized writing rules of the given control language.

Optimising the formulation of the controlled language feedback is matter of iteratively revising it via the testing scenario, using GSA as the metric, until authors consistently achieve total fidelity of the models they construct with the reference models.

6 Conclusions

So how can go about judging whether the products of NLG systems express the intended message? A first step towards this goal is to enable symbolic authors to satisfy themselves that they have built the domain model they had in mind. Graphical feedback is too difficult to interpret, while natural language output that is optimised for the end-reader may not show the unequivocal fidelity to the domain model that the symbolic author requires.

We have suggested that textual feedback in a form close to a controlled language used for specifying software requirements is a good candidate for this task. We have further outlined a method for incrementally refining this controlled language by monitoring symbolic authors' ability to construct reference domain models on the basis of controlled language feedback. The trade-off between transparency and naturalness in the output text intended for the end-reader will involve design decisions based on, among other things, reader profiling. Assessing the fidelity of the end-reader text to the model is also a necessary step, but not one that can be conflated with or precede that of validating the accuracy of the model with respect to the author's intentions.

Acknowledgements

The work described in the paper has been supported by EC INCO-COPERNICUS project PL961104 AGILE "Automatic generation of Instructions in Languages of Eastern Europe". The authors express their gratitude to all the partners of, and participants in, the AGILE project, upon whose work this paper reports.

References

Alshawi, H., Bangalore, S. and Douglas, S. (1998). Automatic acquisition of hierarchical transduction models for machine translation. Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL'98), Montreal, Canada, pp. 41 – 47

Arnold, D., Balkan, L., Lee Humphreys, R., Meijer, S. and Sadler, L. (1994). *Machine translation: an introductory guide.* Blackwell.

Bangalore, S. and Rambow, O. (2000). Exploiting a Hierarchical Model for Generation. Proceedings of the 18th International Conference on Computational Linguistics (COLING'2000), Saarbruecken, Germany, pp. 42 – 48.

Bangalore, S., Rambow, O. and Whittaker, S. (2000). Evaluation Metrics for Generation. Proceedings of the 1st International Conference on Natural Language Generation, Mitzpe Ramon, Israel, pp. 1 – 8.

Cahill, L., Doran, C., Evans, R., Mellish, C., Paiva, D., Reape, M., Scott, D. and Tipper, N. (1999). In search of a reference architecture for NLG systems. Proceedings of the 7th European Workshop on Natural Language Generation (EWNLG'99), Toulouse, France, pp 77 – 85.

Caldwell, T. and Korelsky, T. (1994). Bilingual generation of job descriptions from quasi-conceptual forms. Proceedings of the Fourth Conference on Applied Natural Language Processing (ANLP'94), pp. 1 – 6.

Carroll, J.B. (1966). An experiment in evaluating the quality of translations. In J. Pierce. *Language and machines: computers in translation and linguistics.* Report by the Automatic Language Processing Advisory Committee (ALPAC). Publication 1416. National Academy of Sciences National Research Council, pp. 67 – 75.

Fuchs, N.E. and Schwitter, R. (1996). Attempto Controlled English (ACE). Proceedings of the 1st International Workshop on Controlled Language Applications (CLAW'96), Leuven, Belgium.

Fuchs, N.E., Schwertel, U. and Schwitter, R. (1999). Attempto Controlled English (ACE) Language Manual Version 3.0, Technical Report 99.03, Department of Computer Science, University of Zurich, August 1999.

Hartley, A., Power, R., Scott, D. and Varbanov, S. (2000). Design specification of the user interface for the AGILE final prototype. Deliverable INTF2 of INCO-COPERNICUS project PL961104 AGILE: "Automatic. Generation of Instructions in Languages of Eastern Europe". Available at http://www.itri.bton.ac.uk.

Hartley, A., Scott, D., Kruijff-Korbayova, I., Sharoff, S. *et al.* (2000). Evaluation of the final prototype. Deliverable EVAL2 of INCO-COPERNICUS project PL961104 AGILE: "Automatic. Generation of Instructions in Languages of Eastern Europe". Available at http://www.itri.bton.ac.uk.

Kim, Y. (1990). Effects of conceptual data modelling formalisms on user validation and analyst modelling of information requirements. PhD thesis, University of Minnesota.

Kruijff, G-J., Teich, E., Bateman, J., Kruijff-Korbayova, I. *et al.* (2000). Multilinguality in a text generation system for three Slavic languages. Proceedings of the 18th International Conference on Computational Linguistics (COLING'2000), Saarbruecken, Germany, pp. 474 – 480.

Lehrberger, J. & Bourbeau, L. (1987) *Machine translation: linguistic characterisitics of MT systems and general methodology of evaluation.* John Benjamins.

Mittal, V.O, Moore, J., Carenini, G. and Roth, S. (1998). Describing Complex Charts in Natural Language: A Caption Generation System. *Computational Linguistics*, 24(3), pp. 431 – 468.

Nagao, M. Tsujii, J. and Nakamura, J. (1985). The Japanese government project for machine translation. *Computational Linguistics,* 11(2-3), pp. 91 – 109.

Paris, C., Vander Linden, K., Fischer, M., Hartley, T., Pemberton, L., Power, R. and Scott, D. (1995). A Support Tool for Writing Multilingual Instructions. Proceedings of the Fourteenth International Joint Conference in Artificial Intelligence (IJCAI'95), pp. 1395 – 1404.

Petre, M. (1995). Why looking isn't always seeing: readership skills and graphical programming, *Communications of the ACM*, 38(6), pp. 33 – 42.

Power, R. and Scott, D. (1998) Multilingual Authoring Using Feedback Texts. Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics, Montreal, Canada, pp. 1053 – 1059.

Roth, S.F., Kolojejchick, J., Mattis, J. and Goldstein, J. (1994) Interactive graphics design using automatic presentation knowledge. Proceedings of CHI'94: Human Factors in Computing Systems, Boston, M.A.

Scott, D.R., Power, R., and Evans, R. (1998) Generation as a Solution to Its Own Problem. Proceedings of the 9th International Workshop on Natural Language Generation (INLG'98), Niagara-on-the-Lake, Canada, pp. 256 –265.