LG-Eval: A Toolkit for Creating Online Language Evaluation Experiments

Eric Kow, Anja Belz

School of Computing, Engineering and Mathematics
University of Brighton
Brighton BN2 4GJ, UK
a.s.belz@brighton.ac.uk,eric.kow@gmail.com

Abstract

In this paper we describe the LG-Eval toolkit for creating online language evaluation experiments. LG-Eval is the direct result of our work setting up and carrying out the human evaluation experiments in several of the Generation Challenges shared tasks. It provides tools for creating experiments with different kinds of rating tools, allocating items to evaluators, and collecting the evaluation scores.

Keywords: Natural Language Generation, Evaluation Methods, Evaluation Resources.

1. Comparative Evaluation of Automatically Generated Language

The past six years have seen big changes in how Natural Language Generation (NLG) systems and modules are evaluated. In 2005, a discussion began in the NLG community about sharing data and system components, and making it possible to directly compare independently developed systems and components. Informal discussions at the 2005 NLG workshops (ENLG'05 and UCNLG'05[1].) were followed by the INLG'06[2] Special Session on Sharing Data and Comparative Evaluation[3].

Both sides of the discussion came together at the NSF Workshop on Shared Tasks and Comparative Evaluation in NLG[4] where some of the break-out working groups started hammering out the details of some potential shared task evaluation challenges (STECs). All four specific task proposals that were first presented at the NSF workshop have since come to fruition (TUNA-REG,[5] GREC,[6] GIVE[7] and QG[8]). A further two shared tasks have been run (SR[9] and HOO[10]), and two more have been proposed and are likely to run in the near future (a Spanish surface realisation task and GRUVE[11]).

That makes a total of eight NLG STECs in five years, covering a range of NLG subfields from referring expression generation (REG) to generating instructions in virtual environments (GIVE, GRUVE). The variety of tasks has been mirrored by variety in evaluation methods. NLG Shared Tasks have seen diverse extrinsic and intrinsic, automatically and human-evaluated methods. The overall benefits from the shared tasks have been substantial: legacy task definitions, data and evaluation resources have been made available to the wider community and have led to numerous associated publications and follow-on research. Many researchers from outside traditional NLG have been drawn in as a result of the shared tasks (Basile and Bos, 2011; Boyd and Meurers, 2011), comparative evaluation is now standard, and the Generation Challenges (GenChal) sessions, where results from current STECs and ideas for new ones are presented, have been part and parcel of the alternating INLG and ENLG conferences since 2008.

Apart from the legacy data sets, task definitions and detailed documentation, evaluation software has been an important by-product of GenChal shared tasks.[12] Many of these are for automatic evaluation of automatically generated language. However, we also carried out human evaluations in every GenChal shared task we ran, and in this paper we describe the software tool we created and tested in the course of the numerous human evaluation experiments we carried out for the shared tasks as well as many follow-on experiments. We start with an overview of the three main dimensions of experimental design according to which language evaluation experiments differ (Section 2.), followed by a description of the LG-eval toolkit itself (Section 3.), and some concluding remarks (Section 4.).

2. Language Evaluation Methods and Tools

The benefit of setting up a language evaluation experiment with the LG-eval toolkit is that it takes care of implementational aspects and leaves the experimenter free to focus on

[1] 10th European Workshop on Natural Language Generation, http://www.ling.helsinki.fi/~gwilcock/ENLG-05/; and 1st Workshop on Using Corpora for Natural Language Generation, http://www.itri.brighton.ac.uk/ucnlg/ucnlg05/

[2] 4th International Natural Language Generation Conference, http://www.ict.csiro.au/inlg2006/.

[3] http://www.itri.bton.ac.uk/~Anja.Belz/inlg06-specsess.html.

[4] http://www.ling.ohio-state.edu/nlgeval07/

[5] On Referring Expression Generation, http://www.itri.brighton.ac.uk/research/genchal09/tuna/

[6] On Generating Referring Expressions in Context, http://www.itri.brighton.ac.uk/research/genchal10/grec/

[7] On Giving Instructions in Virtual Environments, http://www.give-challenge.org/research/

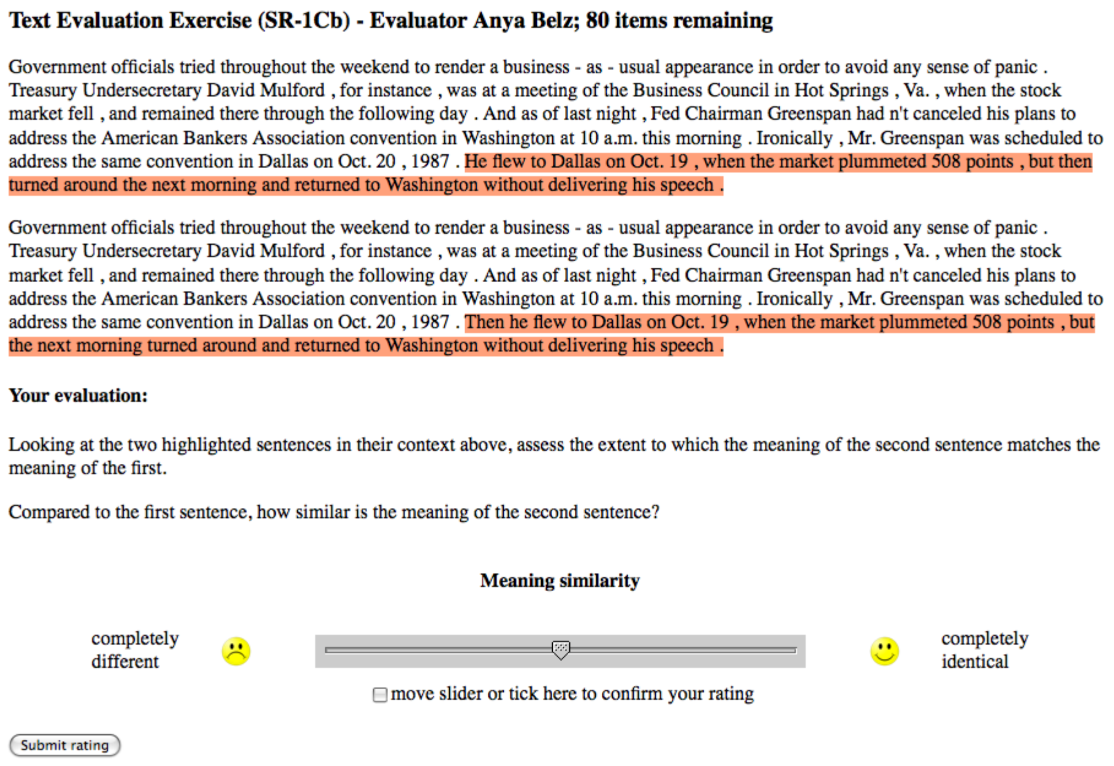
[8] On Question Generation, http://www.questiongeneration.org/

[9] On Surface Realisation, http://www.itri.brighton.ac.uk/research/sr-task/

[10] On text correction, http://clt.mq.edu.au/research/projects/hoo/

[11] Generating Routes Under Uncertainty in Virtual Environments.

[12] In this volume, we also report on the newly created online repository of data, tasks, documentation and evaluation software that have resulted from the GenChal shared tasks (Belz and Gatt, 2012). See http://sites.google.com/site/genchalrepository/.


Text Evaluation Exercise (SR-1Cb) - Evaluator Anya Belz; 80 items remaining

Government officials tried throughout the weekend to render a business - as - usual appearance in order to avoid any sense of panic . Treasury Undersecretary David Mulford , for instance , was at a meeting of the Business Council in Hot Springs , Va. , when the stock market fell , and remained there through the following day . And as of last night , Fed Chairman Greenspan had n't canceled his plans to address the American Bankers Association convention in Washington at 10 a.m. this morning . Ironically , Mr. Greenspan was scheduled to address the same convention in Dallas on Oct. 20 , 1987 . He flew to Dallas on Oct. 19 , when the market plummeted 508 points , but then turned around the next morning and returned to Washington without delivering his speech .

Government officials tried throughout the weekend to render a business - as - usual appearance in order to avoid any sense of panic . Treasury Undersecretary David Mulford , for instance , was at a meeting of the Business Council in Hot Springs , Va. , when the stock market fell , and remained there through the following day . And as of last night , Fed Chairman Greenspan had n't canceled his plans to address the American Bankers Association convention in Washington at 10 a.m. this morning . Ironically , Mr. Greenspan was scheduled to address the same convention in Dallas on Oct. 20 , 1987 . Then he flew to Dallas on Oct. 19 , when the market plummeted 508 points , but the next morning turned around and returned to Washington without delivering his speech .

Your evaluation:

Looking at the two highlighted sentences in their context above, assess the extent to which the meaning of the second sentence matches the meaning of the first.

Compared to the first sentence, how similar is the meaning of the second sentence?

Figure 1: Example of an evaluation experiment (from the SR'11 evaluations) with absolute quality judgements and visual analogue scales.

choosing an appropriate set of quality criteria, and a suitable evaluation tool and method, selecting the data to be evaluated, and writing instructions for evaluators.

2.1. Quality Criteria

The LG-eval toolkit permits any quality criteria to be used and defined in the introduction text. The following are the intrinsic quality criteria we have used so far:

Fluency/Readability: This is a single quality criterion intended to capture language quality as distinct from its meaning, i.e. how well a piece of text reads.[13] In our evaluation experiments we have sometimes called it Fluency, and sometimes Readability.

Adequacy/Clarity: In the TUNA evaluation experiments, this quality criterion was called Adequacy, explained as "how clear the description is", and "how easy it would be to identify the image from the description". This criterion was called Clarity in other experiments, explained as "how easy it is to understand what is being described".

Coherence: To score highly on this criterion, a text should be well structured and well organised. The text should not just be a heap of related information, but should build from sentence to sentence to a coherent body of information about a topic (wording from DUC).

Meaning Similarity: This criterion requires evaluators to read two pieces of text, and then to decide how close in meaning one text is to the other.

[13]It was not our intention to capture 'reading ease' which is sometimes also referred to as readability.

2.2. Relative vs. Absolute Quality Judgements

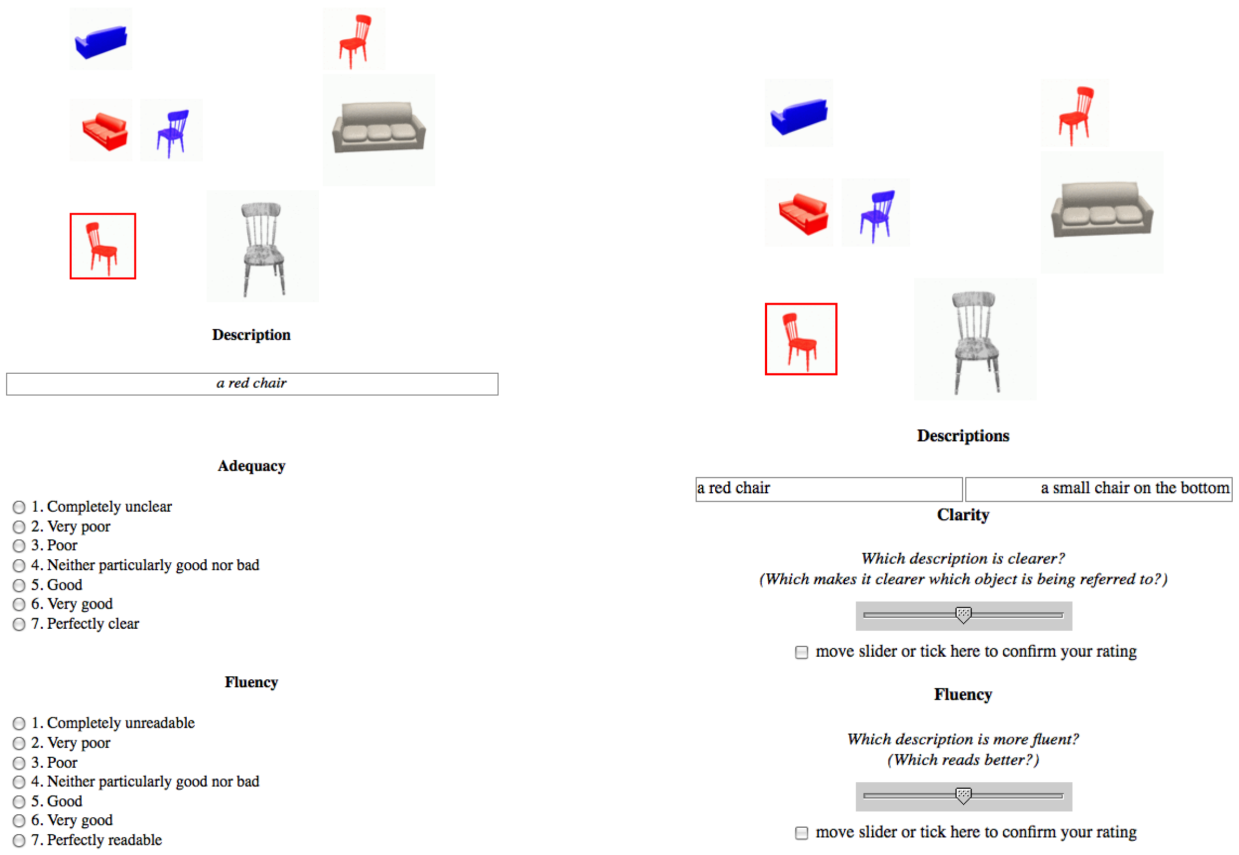
One of the dimensions along which the rating tools we have used in intrinsic human evaluations differ is that they either require evaluators to make judgements of individual pieces of text in isolation (**absolute quality judgements**), or they require evaluators to compare two pieces of text and decide which is better (**relative quality judgements**). Absolute quality judgements are by far the more commonly used in Natural Language Processing (NLP). Relative quality judgements had also been used in NLP system evaluation previously (Reiter et al., 2005), but to our knowledge ours was the first systematic investigation of relative quality judgements where evaluators express, in addition to their preference (which system do you prefer?), also the strength of their preference (how strongly do you prefer the system you prefer?).

Figure 2 shows two evaluation experiments (both put together using the LG-eval kit) where one (on the left) has absolute quality judgements, whereas the other (on the right) has relative quality judgements.

2.3. Different Rating Tools

The LG-eval kit provides two types of rating tools that can be plugged into experiments (others can also be used but have to be created first).

With **Verbal Descriptor Scales** (VDSs), participants give responses on ordered lists of verbally described and/or numerically labelled response categories, typically varying in number from 2 to 11 (Svensson, 2000). Two examples of a VDS can be seen on the left in Figure 2, one for the criterion of Adequacy, one for Fluency. VDSs are used very



Description

a red chair

Adequacy

- ◯ 1. Completely unclear
- ◯ 2. Very poor
- ◯ 3. Poor
- ◯ 4. Neither particularly good nor bad
- ◯ 5. Good
- ◯ 6. Very good
- ◯ 7. Perfectly clear

Fluency

- ◯ 1. Completely unreadable
- ◯ 2. Very poor
- ◯ 3. Poor
- ◯ 4. Neither particularly good nor bad
- ◯ 5. Good
- ◯ 6. Very good
- ◯ 7. Perfectly readable

Descriptions

a red chair	a small chair on the bottom

Clarity

Which description is clearer?
(Which makes it clearer which object is being referred to?)

☐ move slider or tick here to confirm your rating

Fluency

Which description is more fluent?
(Which reads better?)

☐ move slider or tick here to confirm your rating

Figure 2: Example of two evaluation experiments (from TUNA evaluations) that are identical except that in one (on the left) absolute quality judgements and verbal descriptor scales are used, whereas in the other (on the right) relative quality judgements and visual analogue scales are used.

widely in contexts where computationally generated language is evaluated, including in dialogue, summarisation, MT and data-to-text generation.

Visual analogue scales (VASs) are far less common outside psychology and related areas than VDSs. Responses are given by selecting a point on a typically horizontal line (although vertical lines have also been used (Scott and Huskisson, 2003)), on which the two end points represent the extreme values of the variable to be measured. Such lines can be mono-polar or bi-polar, and the end points are labelled with an image (smiling/frowning face), and/or a brief verbal descriptor, to indicate which end of the line corresponds to which extreme of the variable. The labels are commonly chosen to represent a point beyond any response actually likely to be chosen by raters. The experiment shown in Figure 1 uses a VAS.

We have also used VASs for relative quality judgements, as can be seen on the right of Figure 2.

3. The LG-Eval Toolkit

3.1. Preliminaries

The following are explanations of some of the terms that we use in subsequent sections.

1. **Latin Square Experimental Design (LSED):** A Latin square is an $n \times n$ array the cells of which contain n different symbols, each occurring exactly once in each row and exactly once in each column. We use (sets of) Latin square(s) (collectively known as an LSED) to allocate trials to evaluators (corresponding to the rows), such that each evaluator sees each scenario (corresponding to the columns) and each of the systems or pairs of systems in the case of relative quality judgements (corresponding to the n symbols) the same number of times. For each experiment, a new (set of) Latin square(s) is randomly generated. The LSED is converted to an array of trials by assigning to cell $(i, j) = k$ the output of the kth system for the jth scenario. In the subsequent evaluation experiment, the ith evaluator will be presented with all and only the trials contained in the ith row of the array.

The following is a simple example of a single 4×4 Latin square for an experiment involving 4 scenarios and 4 evaluators:

		Scenarios			
		1	2	3	4
Evaluators	1	*S3*	*S1*	*S2*	*S4*
	2	*S4*	*S2*	*S1*	*S3*
	3	*S1*	*S4*	*S3*	*S2*
	4	*S2*	*S3*	*S4*	*S1*

For example, the first cell refers to the output produced by System *S3* for Scenario 1 which will be evaluated by Evaluator 1. The second row contains all the outputs that will be evaluated by Evaluator 2, etc.

2. **System:** A computer program or human author generating outputs for a given task. An LSED for an evaluation experiment involving n systems will be made up of at least one $n \times n$ Latin square.
3. **Scenario:** A single data item for which systems produce outputs. E.g. in the case of weather data to weather forecast generation, there may be a single set of weather data for each date and systems generate a single weather forecast for each date. In this case, each date can be said to provide one scenario. Scenarios correspond to columns in a given LSED.
4. **Trial:** A single system output (attribute set, word, phrase, sentence, etc.) to be evaluated (or pair of outputs, in the case of relative quality judgements), and the context within which it is to be evaluated.
5. **Trial set:** All the trials to be evaluated by the same evaluator, corresponding to a row in a given LSED.

3.2. Overview

The LG-eval toolkit (http://www.nltg.brighton.ac.uk/research/lg-eval) emerged from developing several similar but not identical online evaluations of automatically generated language. The ultimate goal of the toolkit is to reduce the work involved in, and (hopefully thus) human reluctance to creating, running and maintaining evaluation experiments. To this end, we aim to cut down on the amount of new code and human intervention needed to create a new evaluation experiment. The toolkit consists of two components:

- A set of Perl modules to provide experiment infrastructure, running as a CGI script, and
- a Haskell script to generate the LSED and assign trials to evaluators (participants) on its basis.

The toolkit comes with documentation written in the style of a quick start or walkthrough rather than a user manual (http://www.nltg.brighton.ac.uk/research/lg-eval). Our experience has been that revisiting experiments is easier if we have examples that we can copy and paste from rather than a list of options to wade through.

3.3. Experiment infrastructure

An online experiment consists of a list of *trial sets*. Each trial set is evaluated by a single *participant (or evaluator)*. A trial set consists of a set of *trials*. Each participant is allocated a trial set, this is assigned a random order and will normally have a small number of practice examples added at the beginning.

Running through such an ordered single trial set plus practice examples (which has to be done for each participant) can be described with a simple state machine (shown in Figure 3). At its heart, the LG-eval toolkit implements the state machine, taking care of details such as user authentication, handling incomplete forms, and saving the results. This leaves the experimenter with the task of designing the presentation of trials and rendering the design as an HTML form. Much of this task can be accomplished by copying and pasting from the example experiments provided with the toolkit.

In addition to minimising work for the experimenter, the toolkit aims to retain flexibility. It does this by maintaining agnosticism with respect to the precise design of trials and by requiring the use of callback functions for some parts where toolkit functionality must be interleaved with code specific to the experiment:

3.3.1. Trial design

The toolkit is deliberately agnostic to the contents of the trial file. It treats each trial as arbitrary data, which the experimenter is responsible for translating into an HTML form. LG-eval simply provides the system outputs and plugs them into the given trial file at run time.

The toolkit does not place any constraints on what type or size of item can be evaluated. E.g. texts to be evaluated can be of any length: in the experiment shown in Figure 1 two texts of paragraph length were compared, whereas in the experiments shown in Figure 2 short noun phrases of roughly 3-6 words on average were evaluated.

3.3.2. Callback functions

As an example of the use of callback functions, the toolkit has a function that saves the results of the evaluation of a single trial to disk. This function is not exposed to the experimenter as it is internal to the implementation of the toolkit's state machine; however, to do its job, the function needs to know what the results of evaluating the trial are. To keep this opaque to the experiment, the toolkit simply requires that the experimenter provide a callback function that renders the result of a given trial as a string, which the toolkit will in turn append to the appropriate results file.

3.4. Repeated Latin squares

The LG-eval toolkit uses a Latin Squares Experimental Design (LSED, see previous section) which ensures that each subject sees the same number of outputs from each system and for each scenario. There are modules for generating the squares themselves, given certain constraints, as well as for mapping squares to arrays of trials (see above). As certain constraints (such as having to have a given number of total trials, e.g. for comparability of experimental cost) make the use of LSEDs not entirely straightforward, we have aimed in the toolkit to simplify the process through both automation and documentation.

3.4.1. Automation

Automation for repeated Latin square generation comes in the form of a Haskell script that (i) reads a directory containing the experimental data along with parameters such as the square size, (ii) divides the data up to fit a given LSED, and outputs a set of files that can be used by the CGI script described above. Done by hand, the process is both complex and uninteresting, and thus error-prone. Automating this process allows the experiment to get started faster and, more importantly, to avoid mistakes. It also improves repeatability, allowing experiments to be revisited without requiring the experimenter to dredge up old processes from memory or relying on them to have documented them in the past.

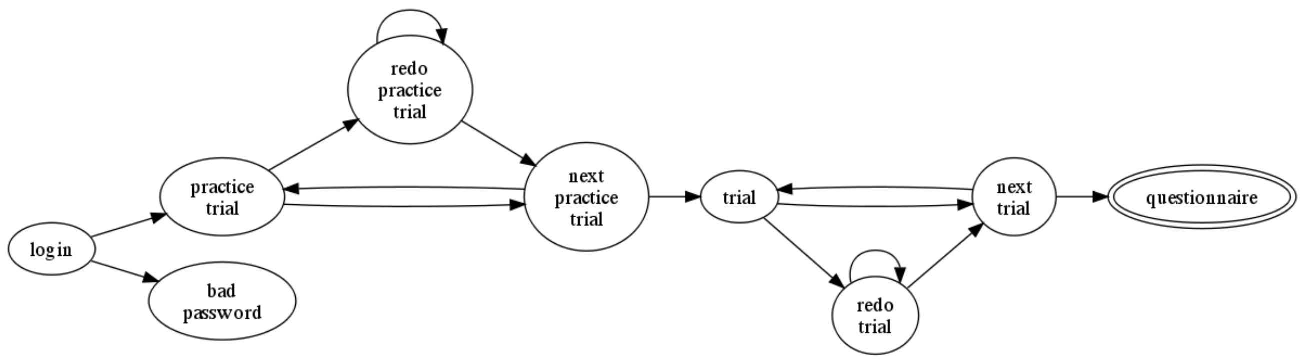


Figure 3: A simplified representation of the state machine implemented by the LG-eval toolkit.

3.4.2. Documentation

To reduce the startup cost of running evaluation experiments with LSEDs, the toolkit comes with a recipe for designing Latin squares, written with the aim of being usable as a sort of worksheet which the experimenter can plug values into. It also provides a handful of example configurations taken from past experiments. In our experience, the ability of revisit the worksheet rather than recall how to make use of the Latin squares has been invaluable.

3.5. Putting it all together

The LG-eval toolkit comes with step-by-step instructions regarding which components the experimenter needs to provide and how to name them and/or where to place them so that they will work properly with the Perl and Haskell modules described above. For example, the data to be evaluated needs to be placed in a given directory, the HTML file that forms the home page of the experiment has to be created (including instructions for evaluators), as does the HTML code that the evaluators see for each trial, including the chosen rating tool (although much of that can be copied and adapted from the example experiments provided with the toolkit). An exit questionnaire can also be added if required.

4. Concluding Remarks

In this paper, we presented the LG-eval toolkit for designing and implementing language evaluation experiments. LG-eval is the result of our work on numerous language evaluation experiments both in the context of GenChal shared tasks and in other contexts. In addition to the code itself, a thorough walk-through introduction with many examples can be found online (http://www.nltg.brighton.ac.uk/research/lg-eval). We are making the tool freely available and hope it will contribute to increasing the number of human evaluations and in particular meta-evaluations of evaluation methodology that the NLG field would benefit from.

5. References

Valerio Basile and Johan Bos. 2011. Towards generating text from discourse representation structures. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG'11)*.

A. Belz and A. Gatt. 2012. A repository of data and evaluation resources for natural language generation. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*.

Adriane Boyd and Detmar Meurers. 2011. Data-driven correction of functionwords in non-native english. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG'11)*.

E. Reiter, S. Sripada, J. Hunter, and J. Yu. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169.

J. Scott and E. C. Huskisson. 2003. Vertical or horizontal visual analogue scales. *Annals of the rheumatic diseases*, (38):560.

Elisabeth Svensson. 2000. Comparison of the quality of assessments using continuous and discrete ordinal rating scales. *Biometrical Journal*, 42(4):417–434.