

Methods, Requirements and Licenses for Shared NLG Resources

David Reitter and Charles Callaway

School of Informatics

University of Edinburgh

United Kingdom

dreitter | ccallawa @ inf.ed.ac.uk

Abstract

Tools and data that can be shared in the Natural Language Generation community require common standards for data collection, documentation, implementation and licensing as well as a central place to find such resources. We argue for open, free, well-documented and simply-structured resources, and introduce a free and open online repository of NLG resources.

1 Shared tasks and shared resources

Work in Natural Language Processing has come to depend on automated evaluation schemes which provide empirical measures of success. This has led to constructive competition between various groups, for instance in syntactic parsing, information extraction, and machine translation. It has also led to rapid improvements in localized problems, but not yet to large end-to-end systems. Such schemes need large quantities of common, annotated data to deduce statistical relationships between produced and desired output. They need common and reusable, thus controlled test components or tasks.

Meanwhile, NLG groups have used hand-crafted data to demonstrate and qualitatively evaluate their systems. The NLG community has yet to produce common, standardized datasets, although this has slowly been changing. For instance, participants at the 2005 European Workshop on NLG expressed their desire to establish shared data, consisting of structured databases with domain-specific content and “gold standard” human-written results. Shared resources could allow for a centralized and coordinated evaluation of

systems performing a *shared task*. Similarly, new systems could be evaluated against older competing ones.

In this context, the idea of reusable software tools becomes very attractive. Any new NLG module needs to interface with existing ones before it can be sensibly evaluated by humans. For instance, a new adaptive realizer for a dialogue system will need a backend to supply selected content and a user model. Unfortunately, many systems created are domain-specific demos. Here, the underlying, novel principles are meant to be reusable, but their implementations are not.

In the following, we propose some requirements for sharable data which should make implementations subsequently more reusable. We announce a common directory of resources, which is already available to the community.

On the downside, we will want to avoid a situation where researchers produce code to beat the automatic score, rather than make progress on solving the important challenges. For instance, summarization shies away from systems that aggregate and paraphrase because they wouldn’t score well in the standard ROUGE metric. Automatic evaluation must be used with caution.

2 Other Fields and their Ethical and Organizational Considerations

Fields differ greatly in their adoption of data-sharing. Genetics researchers, for instance, adopted the idea early on, and the US-American National Institute of Health mandated it in 2003.

Even without being pressured by funding bodies, we are under a moral obligation to share data, once it has been collected with the help of public funds. To address concerns about the validity of studies carried out with these data, *the exact col-*

lection method should be documented, so that resulting constraints on the analysis are evident.

Ethical considerations are important in any setting. Carrying out NLG data collection experiments will commonly pose little difficulties. However, we suggest that *participants are asked to permit the free dissemination of the recorded data, even if a distribution is not immediately planned. Any anonymization, where necessary for privacy reasons, should be planned and agreed beforehand.*

When it comes to sharing analytical data rather than just descriptive datasets such as genomes, a common language becomes even more important. The need for a common classification standard for observed phenomena is common to NLG and medicine.¹

3 Development of resources

In the following, we will leave aside the difficult questions regarding a reference architecture. It should not be surprising that a generic generation architecture may consist of a pipeline with components such as content and sentence planners, a realizer with a lexicon including lexical choice algorithms and a grammar ((e.g., Cahill et al., 2001), implement a generation system within a generic framework.) The lower-level components may be expected to be less domain-specific. In this section, we address technical requirements and discuss distribution methods.

3.1 Data exchange for corpora and tool APIs

A corpus format can be designed to support just one type of data. For example, syntactic trees can be encoded in the Penn Treebank style. Part-of-speech tagged or lemmatized text may be represented via inline tags: `la/DT pasta/N`. On the other end of the spectrum, we find XML-based stand-off annotations, where several layers of information are distributed over many files, linked via XPath expressions.

If designed well, sequential rather than hierarchical corpus-specific syntax in a text file can be easily read in a terminal and parsed via regular expressions with *grep* or Perl. XML, on the other hand, often requires the use of parser libraries which come at a computational cost and increase implementation effort greatly. XML toolkits such as the NITE XML toolkit (NXT) (Carletta et al.,

¹<http://www.nlm.nih.gov/research/umls/>

in press) require an additional abstraction layer. However, NXT provides a framework to create graphical user interfaces which visualize and support data creation.

XML technologies building on a corpus encoding framework should be used if and only if their features are essential. Simple formats, parsable with regular expressions, should be preferred when encoding non-hierarchical ASCII data.

3.2 Dependencies

Tools can be implemented with a variety of methods. While the particular implementation language, formalisms and libraries are a matter of research methodology, the resulting requirements have knock-on effects for who and for how many people can make use of the software.

Software should run on standard computing environments (not just one). It should only require free and openly available libraries or interpreters. For example, an implementation in Prolog could run on the freely available SWI-Prolog² and YAP³. Hard requirements for a good, but costly commercial system such as SICSTUS⁴ should be justified by actual features – and not, because it happened to be available to the developer.

3.3 Allow collaboration

While the free dissemination and reuse of methods has long been a standard technique in academia, actual implementations are often withheld – even though the implementation is considered less valuable academically. But practically, implementations with source code are of great value.

Leaving local policies and intellectual property agreements aside, allowing others to use, improve and redistribute the source code may result in higher-quality code in the long term. Distributing the source code initiates a community review process, which has benefited high-profile software projects such as GNU/Linux and Apache. Similarly important, iterative development will extend tools to provide the features needed by the community, with limited effort from the original author. Licensing the software liberally is a stepping stone toward that goal.

Thus, *we recommend to license publicly-funded research implementations under a non-restrictive*

²<http://www.swi-prolog.org/>

³<http://www.ncc.up.pt/~vsc/Yap/>

⁴<http://www.sics.se/sicstus/>

license such as the GNU General Public License⁵. This license requires newly modified versions to contain the original copyright notice, which usually means that authorship attribution is ensured. A reference to the original publication describing the work may be required from users.

It should be noted that these are prerequisites to collaboration. Actual, informal collaboration needs platforms – either locally, or in form of forums and online directories (see Section 4.1).

3.4 Documentation

Collaboration on the source code level as well as reuse of implemented tools *require well-documented code and APIs*. A typical researcher in NLG may want to work on content planning, but leave syntactic and lexical realization to a module produced by someone else. In this case, the researcher may not particularly care for the techniques employed in encoding the grammar, or optimizing the output.

In such a situation, a conference paper is not sufficient as documentation, and the interfaces of each particular tool need to be documented. Techniques such as Doxygen⁶ and JavaDoc allow programs to be documented in the source code. Human-readable manuals can be automatically created. Additionally, practical usage examples, possibly coming from a larger set of test cases is helpful.

4 Disseminating Resources

Two organizations have been established to facilitate the distribution of general linguistic resources. The Linguistic Data Consortium⁷ and the European Language Resources Association⁸ both disseminate corpora to its members and also make them available to third parties for a low (LDC) or a large (ELRA) fee. Distribution of resources in a scientific community does not have to depend on charges to recover the costs, in fact such a model will surely fail as funding for NLG is much less than for speech recognition or data mining.

4.1 The ACL SIGGEN's Resource Wiki

The ACL SIGGEN board has established a centralized directory to enable NLG researchers to advertise resources. The SIGGEN Resources Wiki,

hosted at University of Edinburgh, is intended to become a community-edited online bibliography with pointers to downloadable resources and data-sets. Wiki technology allows visitors to edit the content and also review other people's edits. The Resources Wiki is already online at <http://www.siggen.org/resources>. Access is free and immediate. We believe that the Wiki represents a viable long-term solution, since it requires little maintenance, as long as the community accepts it.

The Resource Wiki is a suitable venue to announce available resources, and to distribute them, if they are to be made available for free. *We recommend to announce reusable resources (data and implementations) on the SIGGEN Wiki, and all NLG systems in the Bateman/Zock online compendium*⁹.

5 Conclusion

Datasets and reusable tools may add interesting and useful dimensions to the field of natural language generation research. Open access to the source code and easy dissemination via the newly introduced SIGGEN Resources Wiki help ensure the best possible adoption and ongoing, community-based improvements – provided the funding structure behind the resources is designed to allow for free distribution. Establishing a standard that allows researchers to replicate experiments with original data such as in the biomedical sciences could be a helpful development, not only for quantitative, but also largely for qualitative studies in NLG.¹⁰

References

- Jules J Berman. 2003. A tool for sharing annotated research data: The "Category 0" UMLS (Unified Medical Language System) vocabularies. *BMC Medical Informatics and Decision Making*, 3(6).
- L. Cahill, J. Carroll, R. Evans, D. Paiva, R. Power, D. Scott, and K. van Deemter. 2001. From rags to riches: exploiting the potential of a flexible generation architecture. In *Proc. of the 39th Annual Meeting on Association for Computational Linguistics*, pages 106–113.
- J. Carletta, S. Evert, U. Heid, and J. Kilgour. in press. The NITE XML toolkit: Data model and query. *Language Resources and Evaluation Journal*.

⁵<http://www.gnu.org/licenses/gpl.html>

⁶D van Heesch - <http://www.doxygen.org>, 2004

⁷<http://www ldc.upenn.edu/>

⁸<http://www.elra.info/>

⁹John Bateman and Michael Zock's List of NLG systems, <http://www.fb10.uni-bremen.de/anglistik/langpro/NLG-table/NLG-table-root.htm>

¹⁰We would like to thank Johanna D. Moore, Abhishek Arun and James Clarke.