The Workshop Programme

Start	End	Title	Who
9:15	9:30	*Introduction*	Steven Krauwer and Uwe Quasthoff
9:30	10:20	*What is quality*	Chris Cieri (invited talk)
10:20	10:40	*Validation of third party Spoken and Written Language Resources – Methods for performing Quick Quality Checks*	Hanne Fersøe, Henk van den Heuvel, Sussi Olsen
10:40	11:00	*Improving the Quality of FrameNet*	J. Scheffczyk, M. Ellsworth
11:00	11:30	*BREAK*	
11:30	12:10	*Valid Validations: Bare Basics and Proven Procedures*	Henk van den Heuvel (invited talk), in collaboration with Eric Sanders
12:10	12:50	*The Notion of Quality in Language Resources -- Validation of the Spoken Dutch Corpus*	Hanne Fersøe (invited talk), in collaboration with Bart Jongejan and Sussi Olsen
12:50	13:10	*Quality control of treebanks: documenting, converting, patching*	Sabine Buchholz, Darren Green
13:10	13:30	*Evaluation of a diachronic text corpus*	Mikko Lounela
13:30	14:30	*LUNCH*	
14:30	14:50	*Measuring Monolinguality*	Uwe Quasthoff, Chris Biemann
14:50	15:10	*JTaCo & SProUTomat: Automatic Evaluation and Testing of Multilingual Language Technology Resources and Components*	Christian Bering and Ulrich Schäfer
15:10	16:20	*Panel session*	Chris Cieri (LDC), Chu-Ren Huang (Acad Sin.), Takenobu Tokunaga (TIT), Khalid Choukri (ELDA)
16:20	16:30	*Winding up & Closing*	Steven Krauwer and Uwe Quasthoff
16:30	17:00	*END/BREAK*	

Workshop Organiser(s)

- **Steven Krauwer (Utrecht University / ELSNET, steven.krauwer@elsnet.org)**
- **Uwe Quasthoff (University of Leipzig, quasthoff@informatik.uni-leipzig.de)**

Workshop Programme Committee

- **Simo Goddijn (INL, goddijn@inl.nl)**
- **Jan Odijk (ELRA/Nuance/Utrecht University, jan.odijk@nuance.com)**
- **Khalid Choukri (ELDA, choukri@elda.org)**
- **Nicoletta Calzolari (ILC-CNR/WRITE, glottolo@ilc.cnr.it)**
- **Bente Maegaard (CST, bente@cst.dk)**
- **Chris Cieri (LDC, ccieri@ldc.upenn.edu)**
- **Chu-ren Huang (Ac Sin, churen@gate.sinica.edu.tw)**
- **Takenobu Tokunaga (TIT, take@cl.cs.titech.ac.jp)**
- **Harald Hoege (Siemens, harald.hoege@siemens.com)**
- **Henk van den Heuvel (CLST/SPEX, H.vandenHeuvel@let.ru.nl)**
- **Dafydd Gibbon (Bielefeld, gibbon@uni-bielefeld.de)**
- **Key-Sun.Choi (KORTERM, Key-Sun.Choi@kaist.ac.kr)**
- **Jorg Asmussen, (DSL, ja@dsl.dk)**

Table of Contents

Author Index

Introduction

Uwe Quasthoff and Steven Krauwer

Aims of the workshop

The workshop aims at

- bringing together experience with and insights in quality assurance and measurement for language and speech resources in a broad sense (including multimodal resources, annotations, tools, etc),
- covering both qualitative and quantitative aspects,
- identifying the main tools and strategies,
- analysing the strengths and weaknesses of current practice,
- establishing what can be seen as current best practice,
- reflecting on trends and future needs.

It can be seen as a follow-up of the workshop on speech resources that took place at LREC 2004, but the scope is wider as we include both language and speech resources. We feel that there is a lot to be gained by bringing these communities together, if only because the speech community seems to have a longer tradition in resources evaluation than the written language community.

Relevance

Quality assurance is an important concern for both the provider, the distributor and the user of language and speech resources. The concept of quality is only meaningful if both the producer and the user of the resources can rely on the same set of quality criteria, and if there are effective procedures to check whether these criteria are met. The universe of possible types of language resources is huge and evolves over time, and there is no universal set of qualitative or quantitative criteria and tests that can be applied to all sorts of resources. In this workshop we will try to investigate what sorts of criteria, tests and measures are being used by providers, users and distribution agencies such as ELRA and LDC, and we will try to distil from this current practice general recommendations for quality assurance and measurement for language and speech resources, The workshop will look at quality assurance and quality measures both from the provider, the distributor and the user point of view, and will explicitly address special problems in connection with very large corpora, including numerical measures, comparison of corpora, exchange formats, etc.

Sponsors

This workshop is supported by ELSNET and WRITE (the international coordination committee for written language resources and evaluation).

What is Quality? *(Invited talk)*

Chris Cieri

Linguistic Data Consortium
3615 Market Street, Philadelphia, PA 19104, USA
ccieri@ldc.upenn.edu

Abstract

I will talk about core issues in quality control such as how we define quality in the case of language resources, how much variation there is in the definition and what this means for implementing quality control procedures. I think this is important because I have seen many publications that seem to take the approach that quality is single dimension and that our primary task is to move ourselves -- or convince others to move – along that line always in the direction of ever higher quality. However, in the cases with which I am familiar, defining quality is much more subtle. The concept has multiple dimensions and the task of the language resources producer -- and user -- is to consider the multiple dimensions of quality, define a piece of n-dimensional space appropriate for a specific kind of research (or several) and then determine the best, most cost-effective way to occupy that space. In this reality, decisions are sometimes made that seem initially to lower quality or at least give up control (and in some cases they might actually) and yet they lead to scientific and technological advances.

Validation of third party Spoken and Written Language Resources – Methods for performing Quick Quality Checks

Hanne Fersøe[1], Henk van den Heuvel[2], Sussi Olsen[1]

[1]Center for Sprogteknologi (CST) – Københavns Universitet
Njalsgade 80, Copenhagen, Denmark
hanne, sussi@cst.dk
[2] SPEX/CLST – Radboud University Nijmegen
Erasmusplein 1, Nijmegen, Netherlands
H.vandenHeuvel@let.ru.nl

Abstract

This paper presents the experience and insights gained from developing and applying methodologies for quick quality checks (QQC) of third party language resources based on the existing methodologies for full validation, which were documented in validation manuals under contract for ELRA during 2003-2004. The types of resources are Spoken Language Resources (SLR) and Written Language Resources (WLR). The experience gained from applying the QQC methodologies to a number of the resources in ELRA's catalogue is described and on the basis of this, recommendations to the producers of language resources are given. The authors point to the strengths and weaknesses of the current practices, and the similarities and differences between the QQC method and its usefulness for SLR and WLR, respectively, are discussed. Finally a short account of future work is given.

1. Full Validation versus Quick Quality Checks

1.1. Background

The ever increasing importance of easily available language resources for industrial and research purposes is a well established fact, and so is the key importance of the quality of such resources. A validation report resulting from a commonly accepted and standardized validation procedure adds value to a resource as a safeguard of quality, and supports sharing, interchange, availability and reusability of resources.

About a decade ago, ELRA, having as its paramount objective to promote and distribute high quality language resources, found itself in the situation that validation reports were only provided for a part of the SLRs in its resource catalogue, namely those produced in the SpeechDat context (Höge et al, 1999). Validation procedures were not in place for the other resources they distributed. For that reason, ELRA's board decided to help drive and support the creation of quality measures for language resources by setting up a validation committee to handle generic validation issues and to select operational units, validation centres, to be in charge of validation of spoken and written resources, respectively. SPEX, SPeech EXpertise centre in Nijmegen is responsible for spoken resources, while CST, Center for Sprogteknologi in Copenhagen is responsible for written resources. A documented methodology for full validation of third party SLRs was developed first, (van den Heuvel, 2003), and applied to a number of resources (van den Heuvel et al, 2003). Later a methodology for third party WLRs, specifically lexical resources, was developed based on this approach (Fersøe, 2004; Fersøe & Monachini 2004).

ELRA's resource catalogue is available online offered by their distribution agency, ELDA, http://www.elra.info. The catalogue is organized according to type of resource, e.g. spoken, written, multimodal, terminological, and a resource is an entry with an identifier, a name, a description, a price and, possibly, a validation report or a QQC report. The description in the catalogue derives from the Description Form (DF) filled in by the owner or producer of the resource. The description form cannot be accessed online, but it can be obtained on request and is included in the package that ELDA delivers to a buyer.

1.2. The Cost of Validation

Each procedure was created in such a way that a full validation, would ideally take only about 30-40 hours. The larger the resource is, in terms of e.g. words or levels and complexity of annotation, the smaller the selection of samples for content checking is, and vice versa. This amount of hours for a full validation is indeed very low, and it should be seen as the cost a distributor allows for an external validation of a third party resource, which was not and would not be validated by the original producer. The goal of the distributor is to obtain a quality description, and they will therefore accept a certain cost, but will try to minimize it.

The idea behind a quick quality check (QQC) is to minimize the cost even more by describing only the most basic quality aspects of a resource. The goal is that a trained validator by applying tools to automate most of the checking must be able to complete a QQC report in 6-7 hours. For some potential buyers such basic quality measures will be sufficient, for others they may serve as a starting point.

Resource producers that include internal validation in their production plan followed by external validation by an independent validation centre are not very likely to adopt this kind of approach. They will usually allocate and be willing to pay for more manpower because their goal is to make sure that the resource meets the specifications. For other producers, the QQC paradigm offered by and at the cost of ELRA presents a valuable alternative quality assessment to a full external validation of the resource.

2. The QQC Method for SLR

2.1. Content of the Method

As points of departure for the QQC the following principles are adopted:

A. The QQC mainly checks the database contents against a number of minimal requirements. These requirements are of a formal nature which enables a quick, i.e. automatic, check. Content checks are not included because this would involve substantial language-dependent effort.

B. Generally, a QQC should take about 6-7 hours work at maximum.

For each SLR two QQC reports are produced: One for the provider and users on the quality of the database proper (QQC_DB); one for ELDA on the quality of the information on the description forms (QQC_DF). For the templates of the QQC_DF the division as made by ELDA into Speech and Lexicon is maintained.

2.1.1. QQC_DB

The QQC report contains a quality assessment of the resource with respect to a number of minimum formal requirements to specific parts of the resource for example documentation format, transcriptions, lexicon. A star notation is used for this.

Meaning of the quality stars:

* : The minimal criteria for this part of the resource are not met.

**: The minimal criteria for this part of the resource are not completely met.

***: The minimal criteria for this part of the resource are all met.

Other values:

Not Included: This part is not relevant for this resource and not included in the QQC.

Missing: This part is missing in the resource, but relevant.

The QQC_DB checks the documentation regarding completeness and correctness of the SLR description, along similar lines as explained in 3.1.1. for WLR. Further, the QQC concentrates on a series of formal checks regarding:

- directory structure, file names and data integrity
- design in terms of types and tokens of materials contained in the database
- acoustic quality of the speech signals
- formal quality of transcriptions and other annotations (incl. meta-data)

The QQC_DB report is intended for ELRA's database users if the database is already in the ELRA catalogue and for the database providers if the database is new and not in the catalogue yet. Prior to publication, ELDA forwards QQC reports to providers for comments. The final QQC report is made available via ELRA's web pages (catalogue).

2.1.2. QQC_DF

Each database at ELRA is accompanied by one or two description forms: a general description form and/or a specific description form. These description forms contain the basic information about a database according to ELRA. The description forms are filled out by ELDA in cooperation with the LR provider. The form is used to inform potential customers about the database. The information provided on the description form should be correct. The general description form contains information about e.g. the provider (coordinates), price and availability, information on documentation and validation of the resource, and the distribution media. The specific description form contains more detailed information, e.g. for SLR, about the number of speakers and their distribution in terms of gender, age, accent, about included annotation layers and data encoding, and so on.

The QQC_DF report contains a quality assessment of the correctness of the information on the description forms. A star notation is used for this as well.

Meaning of the quality stars:

* : The information provided is insufficient/incorrect.

**: The information provided is close to sufficient/correct.

***: The information provided is complete and correct.

Other values:

Not Included: This information is not relevant for this resource and not included in the QQC.

Missing: This part is missing in the resource, but relevant.

2.2. Applying the QQC Method: Experience gained

SPEX experiences with the QQC method for SLR can be summarized as follows:

- Data collections with many and/or voluminous speech files pose administrative difficulties in the sense that copying the material to hard disk may take a large proportion of the allocated time.
- There is no sensible way to define minimal QQC validation criteria that apply to all kinds of SLR. Currently, SPEX has developed different QQC templates. There are templates for different application domains: ASR, phonetic lexicons, TTS. Templates for multimodal LR are planned.
- The star assessment system needs a good explanation to producers. The QQC departs from the idea that a three star product (highest quality) is provided. Less stars are only provided for serious deviations of the minimal requirements. Small deviations are reported but not penalized in the star assessment.
- An action point is to complete the description forms for the resources in the catalogue. ELDA is currently working on a new procedure to fill in missing information on existing resources.

3. The QQC Method for WLR

3.1. Content of the Method

The QQC method for WLR makes use of the same star notation as the SLR method. A score of one, two or three stars is given for documentation suitability and completeness, formal properties, and reliability of content, respectively. So a few content checks are included here as opposed to the SLR method.

One QQC report only is produced for each written resource and not two, as described in section 2.1. The existing resources in the catalogue targeted by this method in most cases do not have description forms, partly because there is a stronger tradition for metadata in the SLR area. Spoken resources constituted the main focus for ELRA's resource distribution for a long time, both because many resources of this kind were available for distribution, and because they were more in demand than written resources. The routines and procedures developed for SLRs could not simply be copied, they had to be redeveloped or at least adapted first. This process is complete now, so new WLRs offered for distribution do have description forms, and in a foreseeable future QQC reports for new WLRs will include an assessment of the DF.

3.1.1. Documentation

The documentation is checked manually for suitability, i.e. whether it is clear and to the point and whether it is written in either the language of the resource or in English, the only two possibilities accepted. It is checked for completeness of the information regarding

- copyright and contact persons
- format and character set of the resource files, naming conventions and how to handle them
- languages of the lexical resource, mono-, bi- or multilingual
- type and structure of the entries, lists of legal attributes with mutual dependencies

Ideally, the documentation should specify coverage of the resource, of the domain type, and of the specific information types in the resource. Information on intended applications should also be part of the documentation.

3.1.2. Formal properties

The formal properties concern the usability of the lexical resource. Here the conformance with the specifications is checked, mostly automatically but partly manually, too. Even properties left undocumented can be checked, like e.g. size of the resource, structure of entries etc. These are checked and reported, leading to an added value of the resource.

3.1.3. Content

Finally, a few manual checks on the reliability of the resource content are performed. This is where a QQC differs most from a full validation. About 30 entries are sampled randomly, keeping in mind that different word classes and the different information types must be represented. The sample is checked for correctness of the information types present in the resource in question, be it PoS tag, morphological, syntactic, semantic information or translational equivalents.

3.2. Applying the QQC Method: Experience gained

A summary of CST's experience with the QQC method for WLR is given below.

Documentation may vary a lot in size from one page to several hundred pages. Very short documentation with little information complicates the validation process. Reading very long and detailed documentation takes up a rather large proportion of the time allocated. The extraction of the relevant parts of such documentation is not always straightforward.

Resources are of quite different size and structure, and for large resources or resources with annotation layers in separate files the handling and manipulation of the data is very time consuming.

Lack of conformance with the specifications is a general problem. In nearly all cases the inconsistencies concern the structure of the entries, the attributes and the values allowed. In the worst cases we have checked multilingual resources with two sets of specifications, a general, very detailed and comprehensive one and a language specific one, where the data turned out to be annotated with a combination of the attributes and values from both specifications mixed with other values not documented at all. For other resources we have seen inadequate documentations full of errors where data, if documented at all, do not correspond to the documentation. These examples are of course extreme but very few of the resources checked so far can claim to be fully conformant with their specifications.

Very few content errors are found in the QQCs due to the small number of entries checked but sometimes general and systematic errors are in fact detected. Lexical resources can be of very different nature, ranging from full form wordlists with PoS and morphological information through multilingual resources with semantic information to bilingual collocational resources, and it is indeed important for the credibility of the QQC to check the reliability of the content information stated in the documentation, i.e. to check that the lexical resource is what it claims to be and to give future users an impression of the quality of the resource. The discrepancy between the desire to check the content and the limited time available is quite a dilemma. The credibility of a QQC of a smaller resource is higher than for a larger resource since the percentage of the content checked is higher. Here the methodology still needs further development and a point of revision could be the discussion of whether the star notation should be used for content checks. It is hardly fair to give three stars to a resource of half a million words or more based on a sample of 30 words while this would be far more reasonable for a smaller resource.

4. Comparison of Methods

4.1. Strengths

The strength of the QQC for SLR is that it allows for a good impression of the quality of a SLR, at least at the formal level. A QQC constitutes a good test bed to assess the directory and file structure of a SLR, and it allows for testing of technical completeness and consistency of annotations at various levels. A QQC also gives a good idea of completeness and correctness of the documentation. Further, the procedure provides a general impression of the quality of the signal files by applying a series of acoustic measures on the data.

The strength of the QQC for WLR is that it gives a first quality impression of the basic properties of a resource. It gives some insight into the documentation and the formal properties together with a hint of what problems or shortcomings may exist.

4.2. Weaknesses

An inherent weakness in the SLR QQC is that content correctness is not checked. Within the objectives and time limitations of a QQC the correctness of annotations, e.g. transcriptions, cannot be checked. Especially when hand-crafted annotations are the main part of the LR (such as in phonetic lexicons), the limitations of the QQC approach are felt stronger. However, the alternative of appropriate content checks would lead to substantial amounts of labour by relatively expensive experts, which would exceed the very objective of a QQC.

For WLRs the sparse content checks represents a main weakness both because the quality of linguistic annotations, i.e. the content itself, is frequently the core concern of for the buyer and because the method does not reveal but a small part of the content errors unless these are systematic, in which case they may be detected. Furthermore the content checks are less representative for large resources than for small since the samples, due to the time limit, have to be of the same size.

Another weakness concerns the differing sizes and the differing complexity of the resources, which result in QQCs of varying quality. For some resources the QQC assessments are sound because it was possible to check thoroughly every aspect involved within the limit of the allocated time. But for very large or complex resources the manual checks can only be performed on a rather superficial level. Lists of discrepancies produced automatically are useful for the producer of a resource, but are of less value to a future user.

4.3. Similarities

The QQC approaches are to a large extent parallel and similar, because the underlying assumption is that, regardless of the classification into types such as spoken or written, language resources as such have many features in common, and both the validation and the QQC methodologies should reflect this.

They both build on the same basic assumption that a QQC report provides a valuable quality assessment with a high level of credibility because it is provided by an organization independent of the producer. They use the same simple star notation system to grade the quality and the same criteria for applying the stars. They also proceed through the same steps of checking documentation and formal aspects. Further, the procedures hardly require any language-dependent knowledge from experts, this reducing validation time and costs considerably. Finally, the same amount of time is allocated to QQC a resource, whether spoken or written.

4.4. Differences

The differences are to a high degree, although not completely, motivated by the longer tradition for resources evaluation in the speech community.

There are two QQC templates for each spoken resource, one for the resource itself and one for the description form, and there are variants of the resource template depending on the intended application areas. For written resources only one template exists. There are no variants of this template along the lines of SLR, because it is seldom declared what the intended application area is. The experience up till now shows that variants for mono-, bi-, and multilingual resources are likely to be more useful for written resources than application oriented variants. The linguistic properties of the annotations of bi- and multilingual written resources differ a lot from the monolingual ones, and splitting up the template in two variants would make it possible to skip issues irrelevant to one kind of resource perhaps making it possible to go more into some other issues. But this improvement of the methodology will concern content only, emphasizing the importance of this aspect. For written resources, furthermore, the QQC methodology applies to lexical resources only, while corpora still have to be included or rather have their own variant or their own template altogether.

Ideally most of the QQC work should be done automatically using tools, but currently this is much more the case for spoken than for written resources where more manual checks are made. However tools for WLR are under development.

Along the dimension from SLR to WLR, with phonetic lexicons residing somewhere in between, the proportional contribution (and thus value) of manually encoded annotations increases. Since content checks are not part of the QQC methodology at all for SLR and only absolutely sporadically for WLR, the limitations of the approach as true ‘validation’ of the data manifest themselves stronger for WLR than for SLR. Nonetheless, also for WLR, the QQC approach offers sufficient means to test the consistency, completeness and formal correctness of the linguistic annotations to acknowledge it as a valuable contribution to data quality assessment.

4.5. Recommendations for Producers

4.5.1. SLRs

In order to maximize the information provision to (potential) customers, SPEX recommends producers:

- To put some effort in completing the LR documentation where required, since complete and correct information substantially enhance the usability of a LR against relatively little costs.
- to complete the description forms for the resources they offer through ELRA
- to provide feedback to QQC reports that are offered. This is to the benefit of the quality of the QQC-report. In addition a good QQC report is a recommendation for the database as a product.

4.5.2. WLRs

All the observations documented in the QQC reports should be taken into account by the producers. To future producers CST has these general recommendations:

It is extremely important that a resource has a good and suitable documentation, not too detailed but clear and to the point.

The coverage of the vocabulary is indeed of interest to a potential user and it should be documented. Very few of the resources which have been QQCed, document the principles for coverage, neither the coverage of domain type, nor the coverage of different word classes or other categories. This is a weakness in quality.

For a potential user lack of conformance between the data and the specifications is a major flaw. Producers are recommended to establish internal quality procedures during production to prevent this kind of problems. And it is of great importance for both users and producers that such inconsistencies once indicated in a QQC report are repaired, resulting in a more correct resource and subsequently a better QQC.

5. Future Directions

5.1. Consolidation

In the course of the next year the work already done will be consolidated through QQC-work on more spoken and written resources on the one hand, and through subsequent fine tuning of methodologies and templates on the other hand. This will happen in areas where QQC experience reveals that fine tuning is needed, and the methodologies will be extended with more templates where necessary. This paper shows that there are still questions left unanswered, particularly about template variation according to resource type, about the degree of automation of the QQC task, and about the role of content checks at least for lexical resources.

In ELDA's catalogue of LRs a validation report column with links to the reports has been introduced for all SLRs. The values listed in the column are N for no validation, Y for a full validation, and QQC for a quick quality check. A validation report column will also be created for WLRs giving access to existing reports. It is also expected that more resources will have description forms and it will be investigated if and how the WLR template should include or treat these.

5.2. New areas

The major new areas that will be the object of attention in the future are on the one hand the development of a methodology for validation of multimodal resources and on the other hand the creation of a methodology and a QQC template for written corpora.

For written corpora the major challenge is the total size of the data and the number of files. File handling alone may well take more than 5-6 hours, and inspection of the documentation just to get an overview may also be relatively complex. Other new aspects are for example the sources, the selection of texts, the metadata, the principles for transcription and organization of spoken text corpora, the principles of alignment for multilingual corpora, multiple levels of annotation, e.g. text, chapter, paragraph, sentence, word level.

For multimodal resources the objective is to have a closer look at resources produced in the context of the CHIL project[1]. For various modalities, quick checks will be formulated to assess the formal correctness of annotations in individual modalities and mutual consistency between modalities.

Both of these areas will build on the previous work done for ELRA as described above and on other work done lately where both SPEX and CST have acquired experience with these areas.

6. References

Fersøe, H., S. Olsen (2005): Methodology for a Quick Quality Check for WLR-Lexica. Report submitted to ELRA under the ELRA/0209/VAL-1 contract.

Fersøe, H., M. Monachini (2004): ELRA Validation Methodology and Standard Promotion for Linguistic Resources. In *Proceedings LREC 2004, International Conference on Language resources and Evaluation*, Lisboa 2004, page 941-944.

Fersøe, H. (2004). Validation Manual for Lexica. Report submitted to ELRA under the ELRA/0209/VAL-1 contract.

Höge, H., Draxler, Chr., Van den Heuvel, H., Johansen, F.T., Sanders, E., Tropf, H. (1999): Speechdat multilingual speech databases for teleservices: across the finish line. In *Proceedings EUROSPEECH'99*, Budapest, pp. 2699-2702.

van den Heuvel, H. (2003): Methodology for a Quick Quality Check of SLR and Phonetic Lexicons. Report submitted to ELRA under the ELRA/0201/VAL-1 contract.

van den Heuvel, H., K. Choukri, H. Hoege, B. Maegaard, J. Odijk & V. Mapelli (2003): Quality Control of Language Resources at ELRA. *Proceedings Eurospeech*, Geneva, Switzerland, pp. 1541-1544.

van den Heuvel, H., D.J. Iskra, E. Sanders & F. de Vriend (2004): SLR Validation: Current Trends & Developments. In *Proceedings LREC 2004*, Lisbon, Portugal, pp. 571-574.

[1] http://chil.server.de/servlet/is/101/

Improving the Quality of FrameNet

Jan Scheffczyk[*], **Michael Ellsworth**[*]

[*]International Computer Science Institute
1947 Center St., Suite 600, Berkeley, CA, 94704
{jan,infinity}@icsi.berkeley.edu

Abstract

Lexical resources include large amounts of data and complex interactions between these. Usually, lexical resources are edited by hand by many authors mostly having different backgrounds. Therefore, quality control is crucial, becoming even more important if lexical resources are used in NLP and AI. We have improved the quality of FrameNet – a lexical resource for English – by two tolerant semi-automatic quality management approaches that combine quality control with a maximum level of flexibility for maintaining data: (1) Imperative data checking programs check the quality of hand-made annotations to sentences. (2) A general-purpose declarative consistency management approach (CDET) is used to improve the quality of the other parts of FrameNet and its documentation.

1. Introduction

FrameNet is a lexical resource for English, based on frame semantics (Fillmore and Baker, 2001; Ruppenhofer et al., 2005). Put roughly, a semantic frame (hereafter simply frame) represents the common semantic background for a group of words. The particular sense of each word, which is associated with a frame, is called a Lexical Unit (LU). Frames define Frame Elements (FEs), which express the semantic roles available for these LUs. FEs and LUs allow us to annotate natural-language sentences.[1] The FrameNet data and documentation are continuously maintained by the FrameNet team. Nevertheless, inconsistencies can easily arise through changes in frames or their FEs, since these have complex, non-local consequences for the huge mass of data connected to other frames, annotations, and documentation. The current FrameNet 1.3 data release contains more than 780 frames, 6,800 FEs, 10,000 LUs, and 135,000 annotated sentences. Given the quantity of data in FrameNet, strictly manual quality control measures are too labor intensive. Instead, we aim at more efficient semi-automatic consistency management approaches.

Maintaining FrameNet shares similarities with collaborative document management, where many authors edit a common document base in order to produce, e.g., technical documentation or a software specification. As with document management, both ignoring inconsistency and implementing strict control of consistency are infeasible (Spanoudakis and Zisman, 2001).

The goals of our current efforts are to formalize and automate the checking of the FrameNet data sufficiently to:

1. Provide our users with a high-quality lexical resource that is well-suited for automated NLP, where consistency, completeness, and documentation are major concerns;

2. reduce the maintenance cost of FrameNet;

3. be aware of our quality requirements, reason about them, detect violations automatically, and also generate suggestions to resolve violations;

4. integrate semi-automated quality management into the work process in order to detect and resolve violations as early as possible.

Apart from storing data in relational databases (which avoids many data integrity problems), we have introduced two complementary, tolerant approaches to quality assurance. One approach uses *imperative programs*, the other approach employs general-purpose quality-checking software with *declarative consistency rules*. Besides pinpointing violations to quality requirements, both approaches can propose repairs. We use the two distinct approaches depending on their strengths, since the FrameNet data vary in quantity and kind. Our software has verified that the current FrameNet 1.3 data release fulfills the most important quality requirements. Violations to other quality requirements are explicitly documented. These are major improvements compared to previous FrameNet data releases.

This paper proceeds as follows: In Sect. 2. we illustrate the FrameNet database architecture. We sketch the possible techniques for managing quality in Sect. 3. Sect. 4. introduces our measures for *tolerant* quality management. We give an overview of our quality requirements in Sect. 5. and an example in Sect. 6. In Sect. 7. we evaluate our approach and describe how it can be generalized to other lexicographic and ontological projects. Sect. 8. concludes and outlines directions for further research.

2. The FrameNet Architecture

The center of Fig. 1 illustrates the technical basis of FrameNet, which conceptually consists of three databases: The Lexical Database contains the relationships of word forms, lexemes, lemmas,[2] and their parts of speech. The Frame Database defines and interconnects frames and their FEs. The Annotation Database contains annotations and sentences, which comprise the majority of the FrameNet data.

All of these data are connected via the LU table, which associates lemmas to frames and is referred to by the annotation sets. There are many reasons to keep these databases distinct for our purposes:

[1]For further information consult the FrameNet website: http://framenet.icsi.berkeley.edu.

[2]A lemma may consist of multiple lexemes.

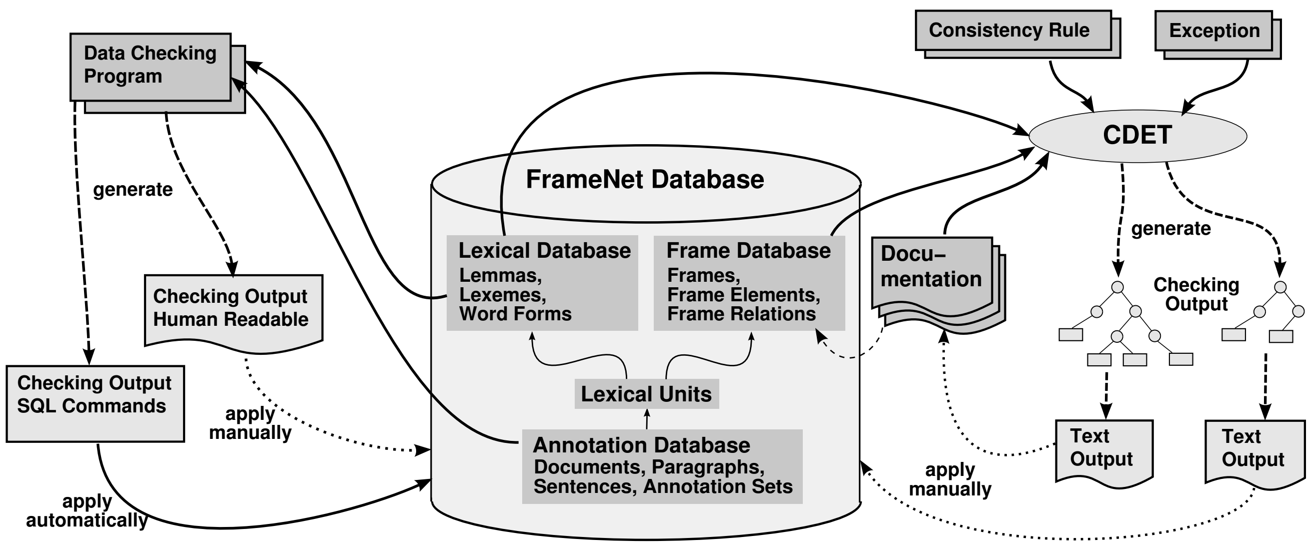


Figure 1: FrameNet architecture and quality management approaches

- Whereas the data in the Frame Database are readily formalizable, most data in the other two databases are less regular since they connect directly with natural language.
- The amount of data in the Annotation Database is far greater than in the other two databases.
- Whereas the data in the Frame Database are to a fair degree language independent, data in the other two databases are language dependent.[3]

Each of the three databases consists of several tables that are connected to each other. For example, we have separate tables for frames and FEs, where the FE table is linked to the frame table.

3. Techniques for Managing Quality

We implement several techniques for quality management:

1. Prevent errors through database structure and native database constraints.
2. Prevent errors through restricted database access via a high-level interface.
3. Tolerate and document errors by external software tools.

The third measure is particularly important because in our experience violations to quality requirements are inherent.
FrameNet data are maintained through a high-level interface, which takes care of many consistency problems. But this interface cannot take care of all problems by forbidding inconsistencies: changing or deleting data often violates quality requirements. For example, the descriptions of frames as they appear in the frame report are stored as (XML) text fields in the Frame Database. Within these descriptions, FEs may be referenced. For the description of a specific frame, the interface lets you mark up only FEs that are really defined and also belong to this frame. If, however, a referenced FE is deleted or its name changes, these references become invalid.
A violation of a quality requirement might not be an error but an exception to the requirement. In linguistics it is not always possible to fully specify why some data are an exception to a quality requirement. Therefore, we have implemented approaches to deal with *exceptions*. Common exceptions are test cases (e.g., frames having a name starting with “Test”), which are excepted from consistency checking completely.[4] Finally, we want our team members to decide whether and how they resolve violations.

4. Tolerant Quality Management

Fig. 1 illustrates our two quality management approaches, which are motivated by the characteristics of the databases. A number of data checking programs check annotations in the *Annotation Database* for correctness, completeness, and style. Each program generates a specific error report showing violations of a quality requirement. Thus, the program provides an *algorithmic* (imperative) definition of quality. Human-readable outputs are subject to further inspection as the actions to be taken are not clear in advance. Machine-readable output (native database commands) can be applied to the Annotation Database for automatic repair. The chief advantages of imperative quality assurance are fast performance, a very specific output, and the possibility of automatically performing repair actions. Due especially to speed concerns, programs have so far proven the only practical way to check annotations for quality.
For the *Lexical Database*, the *Frame Database*, and *documentation*, we employ CDET (Scheffczyk et al., 2004b;

[3]Although the Lexical Database and the Annotation Database will be very different for different languages, different linguistic concepts remain constant over multiple languages, e.g., the concept of polysemy.

[4]These test cases exist in the FrameNet databases only and are not part of a FrameNet data release.

Scheffczyk et al., 2004a).[5] Here, *declarative* consistency rules define quality in formal logic. For many purposes, first-order logic has proven a good means of balancing expressivity and checking complexity. Major extensions over classic predicate logic are: (1) *hints*, which specify violation handling strategies, (2) *exceptions*, which identify data that should be excluded from consistency checking. CDET can check databases and arbitrary documents (preferably XML or LaTeX) against the formalized consistency rules at user request or automatically. The CDET consistency management approach has successfully been applied to various fields such as version control,[6] document management, software engineering, and mathematical knowledge management.

For each consistency rule, CDET generates a formal description of violations and possible repairs, which can then be transformed to other output formats. This formal output format is visualized as a directed acyclic graph (DAG) and is common to all rules. The concrete structure of a DAG resembles the structure of its rule. The size of a DAG generally depends on the number of violations. For larger numbers of violations it is, therefore, useful to employ the text output instead. FrameNet developers can use either output in order to carry out appropriate repairs.

Defining quality declaratively has a number of advantages: A general-purpose specification language improves the understanding and formalization of quality requirements. It also allows for reasoning about consistency rules. CDET's fairly simple consistency rule language supports incremental consistency checking – a key to tight process integration. Since the Lexical Database and the Frame Database are sufficiently "relational" and hold a limited amount of data, we can employ declarative consistency management. Due to its advantages, we would prefer to use this approach for the Annotation Database also. Unfortunately, besides the performance issues, declarative consistency checking is almost precluded by the natural-language complexity of the annotated sentences. This requires the use of complex parsing algorithms, which we cannot express directly in CDET's rule language. In other words, we would lose many advantages of CDET when applying it to the Annotation Database.

5. Structure of Our Quality Requirements

In this section we give an overview of our quality requirements and provide background information.

For the Annotation Database we have implemented more than 30 data checking programs. They usually check annotated sentences for errors of omission or mislabelings. Also, they check the number of annotations per LU in order to provide sufficient training data for applications. Checking for duplicate data tends to be complex because it involves certain variations such as white spaces, punctuation, or letter case. Such duplicates may happen, for example, if two people add the same data (possibly coming from different sources). The complexity of natural language requires the expressive power of a full programming language, ruling out declarative approaches.

For the Frame Database, the Lexical Database, and FrameNet documentation we have formalized more than 65 quality requirements via CDET consistency rules. Rules vary in their complexity, the impact of violations, their purpose, and their importance for the quality of FrameNet data releases. Due to limited resources, we concentrate on fulfilling the most important consistency rules; so we tolerate violations of less important rules.

Below, we list briefly six categories of rules. Rules in the categories (1) and (2) must be fulfilled in a FrameNet data release. Here we find basic quality requirements that directly impact serious use of FrameNet. For rules in category (3) and below we minimize violations but accept them in a data release. In any case, violations are explicitly documented by CDET – a significant improvement over previous data releases. Our consistency rule categories are as follows:

1. Rules about frames in general concern individual frames, their FEs, and LUs. We require, e.g., that each frame has at least one core FE and at least one LU (unless the frame is non-lexical). Frames without FEs are non-sensical; frames without LUs are never evoked and, therefore, should be marked with the semantic type "Non-lexical Frame."

2. Rules about frame relations in general include data integrity checks on frame and FE relations without further inspection of the participating frames and FEs. We require, e.g., valid targets for frame relations or that each frame is connected to another frame.

3. Frame relations in detail: More than half of the rules are in this category. Most rules concern the mapping of FEs along frame relations. The following subdivision is based on the formal strength of a frame relation, where Inheritance is the most formal relation and SeeAlso is the least formal relation.

   (a) *Inheritance* is the most rigorous frame relation in FrameNet. Therefore, we place the most formal restrictions on this relation. Proper inheritance demands that the child frame is more specific than its parent frames, which induces many formal requirements on these two frames. For example, all core FEs of the parent frame should be inherited and no new FEs should be introduced into the inherited coresets from the parent frame.[7] Moreover, certain FE properties should also be inherited, e.g., Requires relations and Excludes relations.

   (b) *Inchoative_of*, *Causative_of*: In this category we find rules similar to those for the Inheritance relation. The rules are, however, less important because Inchoative_of and Causative_of relations

[5] Consistent Document Engineering Toolkit (CDET), see http://www.icsi.berkeley.edu/~jan/projects/CDET/.

[6] CDET supports linear temporal first-order logic.

[7] FEs in a coreset are understood to be disjunctive, in contrast to other FEs, which are conjunctive. Therefore, a new member in an inherited coreset would make the child frame *less* specific, thus violating the inheritance principles.

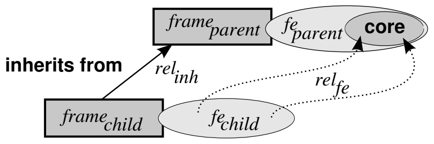


Figure 2: Example: Core FEs are inherited

convey looser relationships between frames than the Inheritance relation does. For example, inchoative frames have a different temporal point of view, which inherited frames do not. Therefore, the Frame Database contains more violations and exceptions for these. Also, errors are less harmful to NLP tasks.

(c) Since *Subframe*, *Using*, and *Perspective_on* relations are less complete than the Inheritance relation we find similar but more relaxed rules in this category. For example, we require that at least one core FE is mapped along these relations (as opposed to the frame relations above where we require the same for *all* core FEs).

(d) The *SeeAlso* relation picks out a representative from a group of similar frames. We, therefore, require that this representative distinguishes all frames in the group from one another.

4. Rules about the Lexical Database include checks on the agreement of the part of speech (PoS) between lemmas and lexemes. For example, we require that the PoS of a single-word lemma corresponds to the PoS of its lexeme; for a multi-word lemma the PoS of the head word should correspond to the lemma's PoS.

5. Rules about Descriptions of frames and FEs that appear in the frame report require, e.g., a sufficient number of examples for core FEs and that referenced FEs really exist in the FrameNet database within the appropriate frame.

6. FrameNet documentation makes references to the FrameNet databases, in order to generate special layout and links in the HTML output. First, we want to make sure that each referenced entity actually exists in the database. Second, we want database entities that are mentioned in the documentation to be marked as such.

   The latter actually requires semantic parsing of the documentation. To date, we can parse the FrameNet documentation syntactically only, which results in a lot of apparent violations, many of them are not actually errors. In addition, many potential candidates for proper markup are not identified yet. We, therefore, regard checking the documentation as a challenging field for future research involving semantic parsing instead of taking the onerous approach of additional markup.

6. Consistency Rule Example

As an example quality requirement consider that all core FEs should be inherited, which is shown in Fig. 2 (where frames are marked by rectangles, FEs by ellipses). If a frame $frame_{child}$ inherits from a frame $frame_{parent}$ then for each parent core FE fe_{parent} there should exist an FE mapping to a child FE fe_{child}. We call the inheritance relation between the frames rel_{inh} and the FE mapping rel_{fe}. Each relation can be referred to by its domain (dom) and range (rng). Formally, our rule is as follows:

```
$\forall\ frame_{parent} \in \texttt{allFrames}(\texttt{fnDB})\ \bullet$
 $\forall\ fe_{parent} \in \texttt{fesOfFrame}(\texttt{fnDB}, frame_{parent})\ \bullet$
  $\texttt{feType}(fe_{parent}) = \texttt{Core}\ \Rightarrow$
   $\forall\ rel_{inh} \in \texttt{inhRelsByRng}(\texttt{fnDB}, frame_{parent})\ \bullet$
    $\forall\ frame_{child} \in \texttt{frameByID}(\texttt{fnDB}, \texttt{dom}(rel_{inh}))\ \bullet$
     $\exists\ rel_{fe} \in \texttt{feRels}(\texttt{fnDB}, rel_{inh})\ \bullet$
      $\exists\ fe_{child} \in \texttt{fesOfFrame}(\texttt{fnDB}, frame_{child})\ \bullet$
      $\texttt{rng}(rel_{fe}) = fe_{parent} \land \texttt{dom}(rel_{fe}) = fe_{child}$
```

First, we get all frames in the FrameNet database fnDB by allFrames(fnDB) and assign them to the variable $frame_{parent}$. For each frame $frame_{parent}$ we obtain its FEs fe_{parent}. If fe_{parent} is core then we retrieve all inheritance relations rel_{inh} with range $frame_{parent}$. There should exist a corresponding FE relation rel_{fe} with range (rng) fe_{parent} and that has as domain (dom) an FE of the child Frame, i.e., fe_{child}.
CDET optimizes rules prior to checking: Implication is expressed via disjunction; negation and quantifiers are pushed inward. Thus, our example rule becomes:

```
$\forall\ frame_{parent} \in \texttt{allFrames}(\texttt{fnDB})\ \bullet$
 $\forall\ fe_{parent} \in \texttt{fesOfFrame}(\texttt{fnDB}, frame_{parent})\ \bullet$
  $\neg\ \texttt{feType}(fe_{parent}) = \texttt{Core}\ \lor$
   $\forall\ rel_{inh} \in \texttt{inhRelsByRng}(\texttt{fnDB}, frame_{parent})\ \bullet$
    $\forall\ frame_{child} \in \texttt{frameByID}(\texttt{fnDB}, \texttt{dom}(rel_{inh}))\ \bullet$
     $\exists\ rel_{fe} \in \texttt{feRels}(\texttt{fnDB}, rel_{inh})\ \bullet$
     $\texttt{rng}(rel_{fe}) = fe_{parent} \land$
      $\exists\ fe_{child} \in \texttt{fesOfFrame}(\texttt{fnDB}, frame_{child})\ \bullet$
      $\texttt{dom}(rel_{fe}) = fe_{child}$
```

From the optimized rule, CDET can generate a graphical violation description in the form of a directed acyclic graph (DAG). The DAG structure follows the structure of the corresponding consistency rule. Nodes represent logical connectives or atomic formulas; edges target the subformulas of a connective. Edges of quantification nodes (marked by ∀, ∃) carry value bindings to the quantified variable. A value represents an entity in the database, a document, or document content that is blamed for one or more violations. Conjunction nodes (marked by ∧) stand for conjunctions; disjunction nodes (marked by ∨) stand for disjunctions; negation nodes (marked by ¬) stand for negations and appear exclusively as direct ancestors of predicate leafs. A predicate leaf contains an atomic formula ϕ that causes a violation and the truth value of ϕ.
Fig. 3 includes the DAG for our optimized example rule. It shows that the rule is violated for the frame State and its core FE State. The variable $frame_{parent}$ is bound to the frame State, the variable fe_{parent} is bound to the core FE

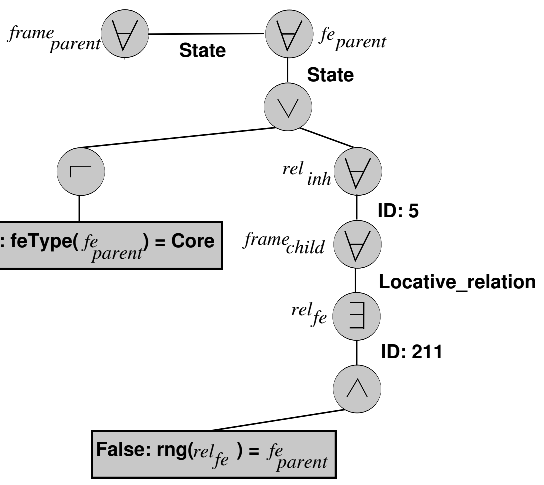


Figure 3: Graphical violation report for example in Fig. 2

State. We also see that the frame Locative_relation inherits from State. There is, however, no FE mapping that binds the core FE State to an FE of Locative_relation. The only candidate FE mapping has the ID 211, but it does not bind the FE State.

The textual representation of the DAG in Fig. 3 is reduced for fast lookup; it omits atomic formulas and shows the variable bindings only.

```
frame_parent ->
 {ID = 150, name = State}
  fe_parent ->
   {ID = 1185, name = State}
    OR 1
       2 rel_inh ->
          {ID = 5}
           frame_child ->
            {ID = 199,
             name = Locative_relation}
             rel_fe ->
              {ID = 211}
               1
```

A proper solution would be to add an appropriate FE mapping for the State FE or to change its coreness type. In the above case we changed the coreness type for the State FE. CDET can generate such repair suggestions from appropriate hints (Scheffczyk et al., 2004a).

7. Evaluation

By applying formal and rigorous quality management we were able to increase the quality of FrameNet significantly and at the same time decrease the effort needed for maintaining and cleaning up our data. Of course, formal quality management incurs the cost of *defining quality*, which is a major effort in itself. Earlier, quality requirements were described in the FrameNet manual (Ruppenhofer et al., 2005), but only in a very shallow form and scattered throughout the manual. Most of the time we spent on determining and categorizing these requirements. Formalizing consistency rules from these informal requirements requires some technical effort. This is, however, straightforward for an expert in formal logic. Most consistency rules needed further adaptation or exceptions had to be added. This is because the informal requirements did not really reflect what we wanted from our data. But we realized this only after the rules had been checked rigorously and we had looked at CDET's violation reports. In our experience such discussions give precious insight about what quality actually means, which leads to a good understanding of data and work flows beyond pure quality management. Exceptions proved an important feature, particularly in for quality management of a natural-language resource like FrameNet. Notice that the effort of rule formalization, adaptation, and adding exceptions has to be done only once. Since the consistency rules formalize general concepts and concern the language-independent part of FrameNet, we can apply them without adaptation to FrameNets in other languages.

Other lexicographic or ontological projects may not use our quality requirements directly. They can, however, benefit from our general, easily customizable, approach to formal quality management. Our general approach makes no assumptions to particular data formats. The approach only requires that data storage is supported by one of CDET's storage interfaces, which currently include version control systems (DARCS (Roundy, 2005), subversion (Collins-Sussman et al., 2004)), SQL databases (MySQL), and the normal file system. Moreover, CDET supports consistency checking across different data storage types, e.g., LaTeX documentation in a subversion repository and production data in a MySQL database.

For applying formal quality management, we suggest the following steps:

1. Identify entities and documents that are part of the lexicographic or ontological project. What are their goals and scopes? Notice that in the first step we neglect data formats and document structures.

2. Explore informal quality requirements. Investigate data and document structures.

3. Formalize rules, and define data formats and document structures.

4. Revise rules as necessary and add exceptions.

For most projects, data formats and document structures are defined already such that these tasks may be skipped. It might, however, be worthwhile to improve formats and structures, which in any case will benefit more than just quality management. We expect step (1) and (2) to take a long time; they will, however, give precious insight into the quality requirements actually needed, which leads to a good understanding of work flows beyond quality management. Step (3) includes technical details, which is subject to experts in formal logic and data formats. Step (4) requires discussion between formal logic and domain experts but, again, contributes to a firm grasp and *awareness* of quality requirements. In our experience this awareness of quality requirements prevents many typical errors made earlier and may also result in work process optimization.

8. Conclusion and Outlook

We have successfully applied two tolerant quality management approaches, each tailored to the different kinds of FrameNet data: (1) The Annotation Database is checked by imperative checking programs, defining quality algorithmically. (2) The Lexical Database, the Frame Database, and documentation are checked by CDET – a declarative approach to consistency management. Declarative consistency checking offers many advantages, some of which we have only begun to bring to realization: (1) Checking the consistency rules themselves for satisfiability and implication, easing formalization. (2) Combining repairs from some rules depending on their characteristics. (3) Satisfied consistency rules are *axioms* that characterize our data; therefore, we plan to incorporate these axioms into our ontological representation of FrameNet (Scheffczyk et al., 2006).

Although we have not yet reached to full potential of consistency management and many quality requirements are still violated in our FrameNet 1.3 data release, it is a clear improvement over previous data releases. (1) We managed to fulfill the most important quality requirements. (2) We and our users have precise knowledge about violations of less important requirements. Thus, we have a much better overview of the quality of FrameNet. Besides these improvements, continuous automatic consistency checking significantly decreases the work spent on quality management. We expect that our approaches to quality management will be of value not only for us and our users but also to other lexicographic and ontological projects.

Acknowledgments

The first author enjoys funding from the German Academic Exchange Service (DAAD). The FrameNet project is funded by the AQUINAS project of the AQUAINT program.

9. References

B. Collins-Sussman, B. W. Fitzpatrick, and C. M. Pilato. 2004. *Version Control with Subversion*. O'Reilly and Associates.

C. J. Fillmore and C. F. Baker. 2001. Frame semantics for text understanding. In *Proceedings of WordNet and Other Lexical Resources Workshop, NAACL*, Pittsburgh.

David Roundy, 2005. *Darcs: David's advanced revision control system*.

J. Ruppenhofer, M. Ellsworth, M. R. Petruck, and C. R. Johnson, 2005. *FrameNet: Theory and Practice*. ICSI Berkeley. www.icsi.berkeley.edu/˜framenet/book/book.html.

J. Scheffczyk, U. M. Borghoff, P. Rödig, and L. Schmitz. 2004a. Managing inconsistent repositories via prioritized repairs. In *Proc. of the 2004 ACM Symp. on Document Engineering*, pages 137–146, Milwaukee, WI. ACM Press.

J. Scheffczyk, U. M. Borghoff, P. Rödig, and L. Schmitz. 2004b. Towards efficient consistency management for informal applications. *Int. Journal of Computer & Information Science*, 5(2):109–121.

J. Scheffczyk, C. F. Baker, and S. Narayanan. 2006. Ontology-based reasoning about lexical resources. In *Proc. of OntoLex 2006: Interfacing Ontologies and Lexical Resources for Semantic Web Technologies*, Genoa, Italy. to appear.

G. Spanoudakis and A. Zisman. 2001. Inconsistency management in software engineering: Survey and open research issues. In *Handbook of Software Engineering and Knowledge Engineering*, pages 24–29. World Scientific.

VALID VALIDATIONS: BARE BASICS AND PROVEN PROCEDURES

Henk van den Heuvel, Eric Sanders
CLST/SPEX, Radboud University Nijmegen, Netherlands

CLST, Radboud University, Erasmusplein 1, 6525 HT Nijmegen, Netherlands

E-mail: {H.vandenHeuvel|Eric}@let.ru.nl

Abstract

Language resources (LRs) are essential for research and application development. In this article we outline relevant principles for LR validation. We argue that the best way to validate LR is to implement it all along the way of LR production and have it carried out by an external and experienced institute, so that this institute can help define the validation criteria in terms of LR specifications and tolerance margins. We address which tasks should be carried out by the validation institute, and which not. Further, a standard validation protocol is shown, illustrating how validation can prove its value all along the production phase in terms of prevalidation, full validation and pre-release validation.

1. INTRODUCTION

This paper deals with the validation of LRs, more specifically of spoken language resources (SLRs). SLRs are annotated collections of speech data. The difference between a mere collection of speech and an actual SLR is "the fact that the latter is augmented with linguistic annotation (i.e. a symbolic representation of the speech)", as is attested in the EAGLES handbook (Gibbon, Moore& Winski, 1997, p. 146). On the other hand, collections of annotations without accompanying speech data cannot strictly be called SLRs, even when these annotations clearly refer to spoken versions of the database entries, as is the case for e.g. phonemic transcriptions.

By validation of a Language Resource (LR) we refer to a quality assessment of the resource by way of a systematic comparison with its specifications, augmented by a set of tolerance margins for these specifications (e.g. 50% of the speakers should be male, with a permitted deviation of 5%). The specifications (the full set or a subset) and the corresponding tolerance margins are the validation criteria for an LR. The criteria may also come from a set of minimal requirements set by a validation centre which are not explicitly part of the specifications. Output of a validation is a report that lists all checks performed together with an account of the results of the checks.

The relevance of validation of large SLRs emerged when the SpeechDat project (Höge, et al., 1997) was started around 1995. The SLRs within this project were produced in a European framework according to design and recording specifications similar to the American-English Macrophone corpus (Bernstein, Taussig & Godfrey, 1994) and the Dutch Polyphone corpus (Den Os, et al., 1995). The SpeechDat SLRs were, however, produced by a large consortium, the idea being that each consortium member would produce from one to three SLRs and obtain the SLRs produced by the other partners at the end of the project. Because of its experience in the production of Polyphone, and because SPEX was not involved in the production of SpeechDat SLRs, SPEX was included in the consortium as the validation centre with the task to monitor the quality of data and to ensure that all databases would be of comparable quality. The other objective of SpeechDat was that the SLRs become available to third parties after the end of the project. This was another reason for the involvement of an independent validation centre to monitor and ascertain data quality.

Since SpeechDat, SPEX has been involved as validation centre in many projects, particularly in data collections supported by the EU, such as SpeechDat Car, SpeeCon, and OrienTel. The experience on SLR validation gained over the years has been reported at conferences, tutorials and summer schools. This paper presents a comprehensive and up–to-date overview of our experience in the field. Although the paper focuses mainly on the validation of SLRs of the SpeechDat type, experience in validations of other SLRs and pronunciation lexicons will be touched upon where considered appropriate.

In this paper we will address basic principles of validation (section 2) and proven procedures (section 3) and we conclude with lessons learnt from our experiences (section 4).

2. VALIDATION PRINCIPLES

Basic aspects of SLR validation have been addressed in Van den Heuvel, Boves & Sanders 2000), Schiel & Draxler (2003), Van den Heuvel, Iskra, Sanders, De Vriend (2004). A brief overview of SLR validation is also presented by Maegaard, et al. (2005).

2.1 Purposes

Result of a SLR validation is a validation report. This report presents a systematic survey of the validation criteria and the degree in which they were met by the SLR. The report can be used for a variety of purposes:

1. Quality assurance: in this case the validation report attests that the SLR meets the minimum of required specifications and is therefore approved;
2. Quality improvement: the validation report shows to what extent the specifications are achieved. Even if the minimum required criteria are met, the validation report can still be used to improve the SLR to meet the full specifications.
3. Quality assessment: since the validation report describes the extent to which the SLR meets the specifications, it can be added as an appendix to the SLR itself, even if remaining errors have not been corrected.

2.2 Strategies

SLR validation can be performed in two fundamentally different ways: (a) Quality assessment issues are already addressed in the specification phase of the SLR. That is, during the definition of the specifications the validation criteria are already formulated. (b) A SLR is created, and based on the specifications the validation criteria and validation procedure are defined afterwards. In this way the risk is increased that the validation of some parts of the specification may become infeasible, because in retrospect there is no meaningful way to check these specifications.
Furthermore, validation can be done in house (internal validation) or by another organisation (external validation). The two dimensions thus identified are shown in Table I.

Validator	**Validation scheduling**	
	During production	After production
Internal	(1)	(2)
External	(3)	(4)

Table I: Four types of validation strategies

(1) in this table is in fact essential for proper database production. Each LR producer is responsible for the database quality during the collection and processing of the data in order to ascertain that the specifications are met. A final check (2) should be an obvious, be it ideally superfluous, part of this procedure. These principles are employed by the Linguistic Data Consortium (LDC) (Cieri, Liberman, 2000; Strassel, et al., 2003). Alternatively (or additionally) an external organisation can be contracted to carry out the validation of an SLR. This is important if the production of database is (sub)contracted or if LR-production is carried out in a consortium where an independent validation institute has to monitor that all SLR are of sufficient quality. In fact, this strategy was adopted by many EU-funded projects, where all producers performed internal quality checks, whilst SPEX served as an independent external validation centre, being closely involved in the specifications and performing intermediate and final quality assessments. An overview of these projects is presented in Table II (see final page). In that context, all four validation activities shown in Table I are carried out.

This two-dimensional view of the SLR validation process is obviously valid for other types of LRs as well, cf. Fersøe (2004) for lexica.

2.3 The role of validation institute

Validation is just one element in the process of quality control of SLRs. Validation is an instrument to make a diagnosis about the quality of a SLR. It is important to distinguish between the validation and correction of a SLR. The two tasks should not be performed by one and the same institute; a conflict of interest may arise when the validation institute is, in the end, checking its own corrections. The appropriate procedure is that the producer corrects the deviations found and that the validation institute again checks the correctness of the adjustments.

The best position for a validation institute is when it is involved from the very beginning of the design of SLRs. Throughout the design phase, it can contribute its expertise to defining and fine-tuning specifications. It can also make clear from the start which of these specifications can be reliably validated by the institute. During the specification phase the validation institute is responsible for addressing the definition of the tolerance margins for deviations of the specifications.

When the specifications have been agreed upon, the contribution of the validation institute can be of great value by carrying out quality checks at strategic moments during the production process (see section 3 below).

It is important that the validation institute provides efficient feedback on data submissions, and keeps all communication channels open for consultation and feedback on the results found. In practice, this means that:

- The arrival of a data set at the validation office is reported to the producer instantaneously
- The data set is immediately checked for readability and completeness in terms of required files. This is of major importance if the SLR cannot be validated straight away. Readability and completeness issues can be resolved by the provider while the SLR is awaiting its turn in the validation queue.
- If possible in a reasonable time frame, the producer should be allowed to resubmit defective files on the fly during validation.
- The validation report is first reviewed by the producer before it is disclosed to anyone else.

This is correct diplomacy and necessary to avoid and remove any misunderstandings on the text of the report. For instance, a reported error may in fact be a lack of clarity in the documentation, and should be repaired there, not in the database itself. Furthermore, a validation institute can make errors, too! Based on the producer's comments a final report is edited which can be distributed to others.

The validation institute should thus be flexible, and open for communication. However, it must also be determined and assertive. The open communication channel is not meant to wipe out or reason away errors, but to obtain a proper view on their nature and cause.

2.4 Approval Authority

When the validation takes place internally, the approval authority is with the producer. Another situation arises when the producer is not the owner of the SLR (e.g. production is (sub)contracted), or when the SLR is produced within a consortium of partners producing similar SLRs with the aim of mutual exchange, as in SpeechDat. In these cases an external validation institute can play an important intermediary role. The institute can perform an objective test to ascertain whether a producing party has fulfilled the requirements set out by the patron/consortium. In these cases the tasks of the validation institute are typically twofold:

1. Checking a SLR against the predefined validation criteria;
2. Putting a quality stamp on a SLR as a result of the aforementioned check.

In these cases, the validation institute can obtain, as a third task, approval authority. However, this is not a desirable situation. The task of the external validation institute is to provide a comprehensive report in which the remaining deficiencies of an LR, if any, are clearly described. Based on this report the patron (resp. consortium) should decide upon the acceptability of a LR In SpeechDat like projects, the approval of a SLR is commonly arranged in another wat, viz. by a voting procedure. The arrangement and execution of the voting procedures is a task that can very well be delegated to the validation institute.

If a SLR is rejected, the owner will have to correct the deficiencies (re-annotate, or make new recordings) and have the corrected SLR validated once more.

3. VALIDATION TYPES AND PROCEDURES

Over the years SPEX has developed a standard validation protocol for SLR in SpeechDat-like projects, which is, apart from details, also applicable to other types of LR. The protocol is developed along three validation milestones: prevalidation, full validation, pre-release valdation.

3.1 Prevalidation

Prevalidation of a SLR is carried out before the stage of extensive data collection is entered. The main objectives of prevalidation is to detect design errors before serious data collection starts. Secondary objectives are:

- to enable the producer to go through the whole stage of documenting and packaging at the beginning so that missing information, ambiguity and errors in the documentation are avoided at the end
- to develop and fine-tune software for validation of the full database

At the prevalidation phase three components are assessed: prompt sheets, lexicon, mini database. The producer can deliver these components together as one package, or one-by-one, submitting a new component after the previous has been validated.

Prompt sheet validation
Before embarking on recording speakers, the producers design reading scripts. These scripts should be an ideal representation of the content of the corpus items and the number of repetitions for each item. Since in practice not all intended material is recorded due to problems with the recording platform, of speakers omitting certain items altogether, not reading them correctly, stuttering or speaking in an environment with high background noise, etc., the reading scripts contain the (theoretical) upper bounds of types and tokens of what is achievable in a database. You will not get more!

The validation of the prompt sheets comprises checks with regard to the presence of the corpus items, adherence of their design to the specifications as well as the maximum achievable number of repetitions at word or sentence level calculated for the complete database. For phonetically rich words and sentences, if included, it can also be checked if a fixed minimum number of tokens per phoneme can be collected, provided that a lexicon containing all the words and their phonemic transcriptions is delivered as well.

If at this stage the prompt sheets do not fulfil the validation criteria (the absolute minimum which is required in the end), measures can still be easily taken to repair the errors since no recordings have been made yet. Database producers indicate that they highly appreciate this part of validation which allows them to spot and repair errors in an early design stage.
The prompt sheet validation is also a test for the specifications as it uncover parts which are underspecified and need further clarification.

Lexicon validation
A formal check of the lexicon with regard to the format and the use of legal phoneme symbols is part of all the validation stages and can be carried out by the validation centre itself. However, the quality of the phonemic transcriptions has to be checked as well. Since this work needs to be done by phoneticians familiar with each language, the validation institute contracts this task to external experts. There are two conditions for the selection of these experts: they have to be native speakers of the language and must have a phonetic training. They

obtain the relevant parts of the documentation describing the principles of the phonemic transcriptions employed by the producer. The experts obtain a sample (normally 1000 entries) of the entire lexicon which they have to check manually. They are instructed to give the provided pronunciation the benefit of the doubt and only to mark transcriptions that reflect an overtly wrong pronunciation. This is in order to prevent marking as errors differences which are due to different phonetic theories or different ideas about what the 'most common' or 'best' pronunciation is.

Mini database validation
10 initial recordings are made in different environments and annotated. The data is formatted and packaged as if it were a final completed SLR, including documentation, and submitted to the validation institute. The purpose of this part of the prevalidation is to check if all items as specified in the prompt sheets are recorded and, if relevant, in the correct order. Further, the format, and the annotations are inspected, all with the aim of preventing errors during large-scale production. Since the documentation is included as well, the producers are forced to start documenting at an early stage. This may be felt as annoying at that time, but the advantages are clearly felt in the final production phase; the burden of documenting in that phase is greatly reduced to some final text editing and modifications of numeric tables.

3.2 Full validation

When all recordings are collected and annotated, the database is packaged and shipped to the validation institute for what is called full validation. The purpose of the full validation is a quality assessment of the end product. At full validation, all checks are carried out.

The validation institute may have a queue of SLRs to be validated. This queue is typically handled on a First-In First-Out (FIFO) basis. Nonetheless, a more efficient procedure is possible. Upon receiving the SLR, the validation institute can perform a so-called Quick Check: this is a quick formal test running the validation scripts to find out if all required files are included in the SLR and if they have the correct formal structure. If so, the SLR can remain in the queue as it is. If not, the producer is requested to submit updated versions of defective or missing files. Quick Checks avoid discovering, for instance, missing files a few weeks later when the SLR is at the end of the queue. Since action can be taken in the meantime, further delays for both the producer and the validation centre can be avoided. Quick Checks allow the producer and the validation institute to work efficiently in parallel.

Since the validation of the (orthographic) transcriptions is restricted to a sample of all recordings, not all speech data is needed during full validation. For large SLR such as those collected in SpeeCon, copying of all speech files onto a hard disk would use up the main part of the validation effort. For this reason, in SpeeCon and similar projects, the validation institute selected a list of 2000 items during the Quick Check, for which the producer instantly had to provide the speech files. Thus, the producer submitted only a subset of the speech files, so that these were available at the validation institute by the time the SLR reached the top of the queue. Note that all orthographic transcriptions were already delivered for the quick check and that updates of the transcriptions were not accepted at this stage. This avoids that new transcriptions were just made for the subset of files selected for validation.

In case all speech data is needed for validation (e.g. for acoustic quality measurements), submission of the database on DVDs or on a hard disk is a sensible alternative.

If substantial shortcomings are found during validation, rectification and a subsequent re-validation of an SLR may become necessary. This is decided by the owner or the consortium in charge of the SLR production. Since mostly not all parts are defective, re-validation is normally of a partial nature. Re-validations are, as rule, carried out at additional costs for the producing party, so as not to encourage sloppy behaviour. Re-validations may iterate until approval of the LR is achieved.

3.2 Pre-release validation

The validation of a complete database results in a report containing a list of errors which were found in the database. Some of them are irreparable and related to flaws in the (manual) annotation and/or the design of the database or the recordings themselves. However, a large number are usually minor and refer to the documentation, label files or other text files which are produced during post-processing. These errors can easily be repaired and the producers are willing to do that. The danger, however, is the introduction of new errors or format inconsistencies during the rectification. Therefore, a pre-release validation has been introduced so that the envisaged master disks can be checked again by the validation centre. The purpose of this validation is to make sure that the reparable errors which were found during complete validation are fixed and that no new errors have been introduced.

After full validation the documentation file is augmented with an additional section: "Modifications after validation". It is checked if all changes agreed upon are included in this section and if they have been implemented in the submitted pre-release version. The validation software is run, so that all formal checks on the data are carried out once more.

If the pre-release validation is finished with a positive result, the database is ready for distribution and the producers are not allowed to make any more changes, however minor, since these corrections can introduce new (possibly greater) errors.

Also the pre-release phase may have one or more iterations until the LR is approved for distribution.

4. CONCLUDING REMARKS

In this article we have clarified the concept of SLR validation. A standard validation protocol has been shown illustrating how validation can prove its value all along the production phase in terms of prevalidation, full validation and pre-release validation.

From our experience as validation centre in many (mainly European) projects we have learnt a number of valuable lessons:

- External validation is an important quality safeguard
- If the validation institute is involved during the specification phase of a SLR it can advise in the specification of the design and setting the validation criteria.
- The validation institute can provide important input at strategic points along the data collection and annotation, not only after the completion of the SLR. A good prevalidation procedure can avoid mistakes that would not be reparable at the end.
- The validation institute needs to keep open communication channels to the SLR provider
- Clear validation protocols help structuring the work and effective quality control
- A relevant part of the work of the validation institute is to find a proper balance between developing automatic checks by scripts and hand labour.
- The validation institute, as a rule, does not claim the approval authority for a SLR.
- The validation institute, as rule, does not perform any of the required corrections itself to avoid the situation in which it is checking its own work.

5. REFERENCES

Bernstein, J., Taussig, K., Godfrey, J. (1994). Macrophone: An American English telephone speech corpus for the Polyphone project. Proc. ICASSP-94, Adelaide, pp. 81-83.

Cieri, Chr., Liberman, M. (2000). Issues in Corpus Creation and Distribution: the Evolution of the Linguistic Data Consortium. Procedings LREC2000, Athens, pp. 49-56.

De Vriend, F., Maltese, G. (2004) Exploring XML-based Technologies and Procedures for Quality Evaluation from a Real-life Case Perspective. Proceedings ICSLP-Interspeech 2004, Jeju, Korea

Den Os, E.A. den, Boogaart, T.I., Boves, L., Klabbers E. (1995). The Dutch Polyphone corpus. Proceedings Eurospeech 1995, Madrid, Spain, pp. 825-828.

Fersøe, H. (2004). Validation Manual for lexica. http://www.elra.info

Gibbon, D., Moore, R., Winski, R. (eds) (1997) The EAGLES Handbook of Standards and Resources for Spoken Language Systems. Mouton de Gruyter.

Höge, H., Tropf, H.S., Winski, R., Van den Heuvel, H., Haeb-Umbach, R. & Choukri, K. (1997) European speech databases for telephone applications. Proc. ICASSP 97, Munich, pp. 1771-1774.

Höge, H., Draxler, C., Heuvel, H. van den, Johansen, F.T., Sanders, E., Tropf, H.S. (1999) Speechdat multilingual speech databases for teleservices: across the finish line. Proceedings EUROSPEECH'99, Budapest, Hungary, 5-9 Sep. 1999, pp. 2699-2702.

Iskra, D., Grosskopf, B., Marasek, K., Van den Heuvel, H., Diehl, F., Kiessling, A. (2002) SPEECON - Speech Databases for Consumer Devices: Database Specification and Validation. Proceedings LREC2002, pp. 329-333.

Iskra, D., Siemund, R., Jamal Borno, J., Moreno, A., Emam, O., Choukri , K., Gedge, O., Tropf, H., Nogueiras, A., Zitouni, I., Tsopanoglou, A., Fakotakis, N. (2004) OrienTel - Telephony Databases Across Northern Africa and the Middle East. Proceedings LREC 2004. Lisbon, pp.591-594.

Maegaard, B., Choukri, K., Calzolari, N., Odijk, J. (2005) ELRA – European Language Resources Association – Background, recent developments and future perspectives. Language Resources and Evaluation (39), pp. 9-23.

Moreno, A., Lindberg, B., Draxler, Chr., Richard, G., Choukri, K., Euler, S., Allen, J. (2000a) SpeechDat Car. A large speech database for automotive environments. Proceedings LREC 2000, Athens, pp. 895-900.

Moreno, A., Comeyne, R., Haslam, K., Van den Heuvel., H., Horbach, S., Micca, G. (2000b). SALA: SpeechDat across Latin America. Results of the First Phase. Proceedings LREC 2000, Athens, Greece, Vol. II, pp. 877-882

Moreno, A., Choukri, K., Hall, Ph., Van den Heuvel, H., Sanders, E., Tropf, H. (2004) Collection of SLR in the Asian-Pacific area. Proceedings LREC 2004, Lisbon, Portugal, pp. 101-104.

Schiel, F., Draxler, Chr. (2003) The production and validation of speech corpora. Bavarian Archive for Speech Signals. Bastard Verlag München.

Strassel, S., Miller, D., Walker, K., Cieri Chr. (2003). Shared Resources for Robust Speech-to-Text Technology. Proceedings EUROSPEECH 2003, Geneva, pp. 1609-1612.

Van den Heuvel, H. Boves, L., Sanders (2000) Validation of content and quality of existing SLR: Overview and Methodology. ELRA Technical report D1.1.

Van den Heuvel, H., Boudy, J., Bakcsi, Z., Cernocky, J., Galunov, V., Kochanina, J., Majewski, W., Pollak, P., Rusko, M., Sadowski, J., Staroniew, P., Tropf, H.S. (2001) SpeechDat-E: Five Eastern European Speech Databases for Voice-Operated Teleservices Completed. Proceedings EUROSPEECH 2001, Aalborg, Denmark, Vol. 3, pp. 2059-2062.

Van den Heuvel, H., Hall, Ph., Moreno, A., Rincon, A., Senia, F. (2004a). SALA II across the finish line : a large collection of mobile telephone speech databases from North & Latin America completed. Proceedings LREC 2004, Lisbon, Portugal, pp. 97-100

Van den Heuvel, H., Iskra D., Sanders, E., De Vriend F. (2004b). SLR Validation : Current Trends & Developments. Proceedings LREC 2004, Lisbon, Portugal, pp. 571-574

Van den Heuvel, H., Choukri, K., Gollan, Chr., Moreno, A., Mostefa, D. (2006) TC-STAR: New language resources for ASR and SST purposes. Proceedings LREC 2006, Genova.

Project	**Type of SLR**	**Number of SLR**	**Period**	**Ref.**
SpeechDat(M)	Fixed telephone network, for voice-driven teleservices, European languages	8	1994-1996	Höge et al. (1997)
SpeechDat(II)	Fixed and cellular telephone network, for voice-driven teleservices, European languages	28	1995-1998	Höge, et al. (1999)
Speechdat-Car	Car recordings incl. GSM channel, European languages	9	1998-2001	Moreno, et al (2000a)
SpeechDat-East	Fixed telephone network, for for voice-driven teleservices, Central and East European languages	5	1998-2000	Van den Heuvel, et al. (2001)
SALA	Fixed telephone network, for for voice-driven teleservices, Latin America	5	1998-2000	Moreno, et al. (2000b)
SALA II	Cellular telephone network, for for voice-driven teleservices, America (full continent)	16	2002-2005	Van den Heuvel, et al. (2004a)
Speecon	Broadband recordings for commanding consumer devices (major world languages)	28	1999-2002	Iskra et al. (2002)
Network-DC	Broadcast News (Arabic)	1	2000-2001	http://www.elda.org/article45.html
OrienTel	Fixed & Mobile telephone network, for for voice-driven teleservices (Oriental region)	23	2001-2003	Iskra et al. (2004)
TC-STAR	Parliamentary speeches & TTS	3	2004-2007	Van den Heuvel, et al. (2006)
LILA	Mobile telephone network, for for voice-driven teleservices (Asian & Pacific region)	6+	2005-	Moreno et al. (2004)

Table II. Overview of SLR data collection projects with an external validation component. Information about all projects can be obtained via http://www.speechdat.org. For TC-STAR see: http://www.tc-star.org.

The Notion of Quality in Language Resources – Validation of the Spoken Dutch Corpus

Hanne Fersøe, Bart Jongejan, Sussi Olsen

Center for Sprogteknologi (CST) – Københavns Universitet
Njalsgade 80, Copenhagen, Denmark
hanne, bart, sussi@cst.dk

Abstract

This paper discusses the notion of quality on which Center for Sprogteknologi, generally bases validation of language resources. It emphasizes in particular the importance of resources having good documentation and it illustrates some of the problems that arise if the documentation, which is the formal object of the validation, turns out not to be completely sufficient and adequate to base the data validation on. The basic principles and steps in a validation are illustrated through the detailed description of the planning and preparation of the validation of the linguistic annotations of the Spoken Dutch Corpus.

1. Measuring Quality

1.1. Background

Center for Sprogteknologi (CST) has a background as ELRA's validation centre, also called operational unit, for validation of written resources since 2003, (Fersøe et al., 2006). Our work in this context is the development of methodologies for validation of lexica (Fersøe, 2004, Fersøe & Olsen, 2005) and the validation of a selection of resources in ELRA's catalogue according to these methodologies. In addition to this experience, we have also validated the linguistic aspects of the 13 lexica developed in the LC-STAR project (Fersøe et al., 2004), we have validated the Nemlar Arabic Written Corpus (Yaseen et al., 2006), and we have described other validation methodologies in the context of the ENABLER project (Calzolari et al., 2004). Future planned validation tasks include the lexica to be developed in the LC-STAR II project, and the two Dutch corpora to be developed in the projects D-Coi (Dutch Language Corpus Initiative)[1] , and DPC (Dutch Parallel Corpus), respectively. The most recently completed validation task is the validation of the linguistic layers of the Dutch Spoken Corpus, or CGN (Corpus Gesproken Nederlands)[2], which will be described in more detail in this article.

1.2. The Notion of Quality

A Google search on the string 'Definition of Quality' yielded 138 million hits. Just a few checks of the first of these revealed that definitions of quality, not surprisingly, are context dependent (e.g. product quality, service quality, process quality, etc.) and relative to specific phenomena in that context. The ISO 9000 plain English definition of quality[3] says that it is a desirable characteristic that a product must have, and that quality is achieved when the product meets needs and expectations of customers. In the validation section of ELRA's website, the concept of quality is associated with adherence to standards in the definition of validation: "The term "validation" in ELRA is used in reference to the activity of checking the adherence to standards, and the quality control of the LR product."

Our notion of quality in validation tasks is in line with these definitions. The quality of a language resource cannot be determined by measuring it against a generically defined quality level. It is neither possible nor realistic to define for instance that specific sets of attributes must be applied or that they must be defined in specific ways. The quality of a resource must always be measured against the specifications by which it was produced, and the more and better it is in conformance with its underlying specifications, the better its quality is. The question of adherence to standards then applies to the specifications and to the whole resource package as such. Here the validation methodologies developed by ELRA (Fersøe, 2004, van den Heuvel et al., 2003) offer recommendations on minimal sets of criteria that spoken and written resources must fulfill.

1.3. The Role of the Documentation

Our experience from completed validation tasks shows that the documentation of a resource is absolutely crucial for the quality measurement of the resource and thus for future users of the resource. Without a good documentation, a resource is both difficult and costly to access and consequently use.

It is our experience that, in general, the speech community has a more trained view on the role and importance of a good documentation than does the written community. For written resources it is not unusual to see well elaborated and voluminous linguistic descriptions that are cumbersome to overview and access for those not directly involved in the production process, and which turn out to represent ideal specifications which were not in the end implemented in the resource. Such discrepancies between the data and its documentation constitute major flaws in the quality of a resource.

2. Introduction to the CGN Validation

The Spoken Dutch Corpus, CGN, is well described on its own publicly available web site and in many scientific publications. It is a corpus with recordings of approx. 9 million words of contemporary Dutch as spoken by adults

[1] http://lands.let.ru.nl/projects/d-coi/
[2] http://lands.let.kun.nl/cgn/ehome.htm
[3] http://www.praxiom.com/iso-definition.htm

in Flanders and the Netherlands with all accompanying transcriptions and annotations.[4]

All rights to the corpus are held by the Dutch HLT Centre5 who has also organized the validation of the corpus by external experts. The validation of the speech aspects of the corpus (32 DVDs) was carried out by BAS Services Schiel from the Institut für Phonetik in Universität München. The validation of the linguistic annotations of the corpus (1 DVD) was carried out by CST, University of Copenhagen. The Dutch HLT Centre will publish the two validation reports on their web site.

2.1. Validation of the Speech Aspects

The validation is documented in a very thorough report of 41 pages and 25 pages of appendices. The validation was organized in three standard steps: Validation of Documentation and Metadata, Formal Validation and Manual Validation. In the summary the author explains that the corpus "has been validated against general principles of good practice and the validation specifications of the CGN consortium. The validation covered completeness, formal checks and manual checks of randomly selected samples. Data types covered by this validation are corpus structure, signal files, orthographic, phonetic, prosodic annotation, segmentation in chunks and words (manual and automatic), the single word pronunciation dictionary and all English documentation files. Manual checks were carried out by native Dutch and Flemish speakers and an experienced phonetician (for phonetic transcripts and word segmentation only)."

In the summary the author concludes that the corpus is of a far above average quality. He regrets the fact that the most important protocols are only available in Dutch, and suggests that a more detailed description of the annotation process and some missing information should be added in the next corpus release.

2.2. Validation of the Linguistic Annotations

The validation is documented in a main report of 55 pages including all appendices and sub documents. In summary the validation shows that the CGN is a carefully and skillfully elaborated language resource of a very high quality. Only few content errors were detected considering the size and complexity of the corpus: Lemmatization 74 errors (error rate 0.25%), PoS-tagging 123 errors (error rate 0.41%) MWU (Multi Word Unit) precision 98.6% (4 errors in 290 MWUs), MWU recall 86.4% (45 of 290 not found), and Syntactic annotation 73 errors (error rate 2.43%). The formal validation, which checked the technical quality and integrity of the CD and the organisation of the files, revealed that all the validation criteria were met. The validation of the documentation, which checked three specific documentation files for *availability* of specific information, *completeness* of information, and *adequacy* for future users of the corpus, identified a number of areas where the criteria were not met or inadequately met, one of these being that the protocols are only available in Dutch, and a number of shortcomings which made it impossible to decide the correctness of some annotations. The validation centre recommends that these errors and shortcomings in the documentation be repaired in the next corpus release.

3. The Validation Process

The validation process can be summarized in these steps: specification of validation criteria; unpacking and installation of all the files on the DVD; formal validation; documentation validation; creation of samples; development of a validation template; training of validators, content validation; error counting and results; validation report. The steps need not necessarily be completed in this order, and some of them were in fact performed simultaneously while others were completed in several iterations, but basically these steps represent the distinct activities that took place. The content checks, specifically, require a lot of data preparation and common understanding of the specifications in order for a group of people to be able to complete them in a consistent way.

Below we describe in more detail some of the validation steps, and in section 4 we give examples of some of the more challenging aspects of the validation of the documentation versus the implementation of these specifications in the corpus.

3.1. Specification of Validation Criteria

The first deliverable of the validation task was the definition of the specific and full set of criteria to be applied. The producer had specified which aspects of the linguistic annotations would be the object of the validation and how large samples should be, so the task was to make these specifications operational by pinning them out into more detail. This also included the sampling.

For the documentation the producers wanted us to validate three documents in Dutch of approx. 180 pages in total, and they wanted them validated with respect to quality and adequacy for future users of the corpus. We made these requirements operational by setting up checklists of specific pieces of information that would be checked for availability and completeness: administrative information, formal technical information and content information. It was not considered relevant to define a measurement scale or grading system or some other notation to express the quality based on e.g. how many pieces of information were missing and/or not complete.

For the formal technical aspects of the data there were no requirements from the producers, but we agreed to very basically just check that the DVD contained all the required files with their correct file names, and that the layout and file structure corresponded to the documentation.

For the content, the producers wanted the linguistic annotations checked on samples of certain sizes. The lemmatization and PoS tagging layers were to be checked for correctness and consistency on a sample of 30,000 words with the purpose of determining the error rate. Multi word units were to be checked for correctness and completeness with the purpose of determining precision and recall in a sample of 30,000 words. The syntactic annotation layer was to be checked for correctness on a sample of 3,000 sentences. These checks in themselves are quite straightforward and do not require long additional checklists, so only a few additional checks were defined that could be made semi-automatically and in-house.

[4] Only one million words of the corpus have been annotated with syntax.
[5] http://www.tst.inl.nl/index_en.php

The sampling was to be made randomly with a uniform distribution over the various components that constitute the CGN. The data consists of 14 components. Two thirds of the data was collected in the Netherlands and one third originated from Flanders. Moreover, each component contains a variable number of sessions, and each session contains at least one file, resulting in a quite large number of files.

3.2. Creation of Samples

The sample for the PoS, lemma and MWU checks consisted of 30,000 words selected randomly from the different corpus components. This was quite a complex task. The sample could not be created just by randomly selecting 0.00333% of the words from each corpus component since the sample should not only represent the different corpus components, but during sampling it became clear that it should also reflect the sentence length distribution in each component. A sampling algorithm involving several steps of measuring sentence length was developed to assure that the correct number of words would be extracted from each corpus component. The sampling script worked on xml annotated files. The relevant data, without the xml annotation, was subsequently imported into approx. 35 Excel files. The validation had to be made within a rather short time frame, and we had to be able to measure progress almost on a day to day basis. Several persons would work on different files at the same time out of the house, and therefore the files had to be relatively small, and the data in the samples had to be very easy to overview and understand for the validators.

A sample of 3,000 sentences was created for the syntax validation. The sampling process was more or less the same as for PoS, lemma and MWUs but here the sentence length distribution was even more relevant, since the complexity of the syntactic structure of a sentence is closely related to the sentence length. The syntax validation would be accomplished in-house by one person, and therefore there was no need to convert the sample into file sizes and formats that could be handled by e-mail.

3.3. Development of a Validation Template

In order to make the data easy to overview and access for their work with the content validation, the relevant data for validation of PoS, lemma and MWU checks were presented in Excel files in what we call the validation template. The design of the validation template was not a trivial task, so it was the subject of much consideration and discussion. The basic requirements to the template are that the validators must be able to have easy access to all the relevant information and only that, and that it must be easy to mark and correct errors and to count them. Consequently we should not create more columns than what could be displayed on the screen at the same time, and the information should be organized in such a way as to minimize scrolling and keying. In addition, related information must be shown together.

The validation template also forms the basis for processing the results afterwards. In connection with the design of the template we therefore had to make various decisions. The first question was whether all errors are of the same category or whether they should be classified into severe and minor errors. Is a PoS assignment error more severe than a value error in an attribute like e.g. number in a noun? The producers clarified that all errors were to be treated alike. Other considerations were related to the precise method of counting errors in MWUs and whether to mark all the words of an MWU or only the first one. The design decisions we made will follow from the detailed description of the template below.

The Excel file which a validator opens displays the data according to the template. The first row shows the identifier (a number) of the corpus file from which the sentence was extracted; this is for easy reference. The second row displays the entire sentence horizontally across the columns, this is for easy context overview while checking the individual words. The following rows each display one individual word of the sentence with all its associated annotations in individual columns.

The columns are divided into data columns which should not be modified and columns for validation annotation. The data columns are showed in the following order: first a column for corpus name, sentence number and the entire sentence below each other, and then followed by columns for the identifier of the MWU, the word form, the lemma, the PoS, the word reference, i.e. the position of the word in the sentence.

The columns for validation annotation needed to reflect the way the errors should be handled. The validation columns are located immediately after the first data column with the entire sentence and before the other data columns.

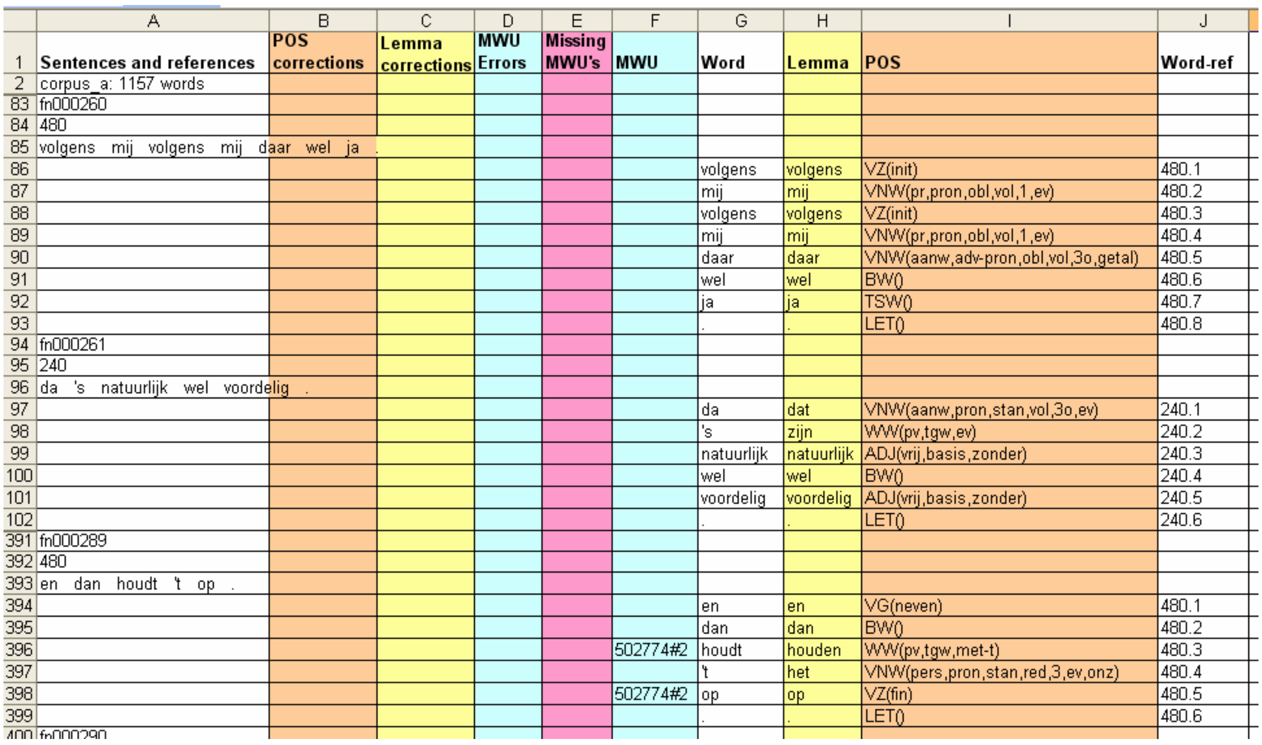
The PoS correction column allows the validator to insert a correction from a pick list of the entire tag set. This implies that an incorrect PoS tag is counted as one PoS error regardless of the number of errors within the current tag.

In the lemma correction column the validator inserts the correct lemma. An incorrect lemma counts as one error.

The MWU error column is for marking words that have been wrongly identified as part of an MWU. Since the validation did not include correction of the MWUs the validator only inserts a ‘1’ for each error found. Only the first word of a wrongly identified MWU is marked and so an MWU only counts as one error regardless of the number of wrongly identified words that occur in it.

The missing MWU column is where the validator marks the MWUs not identified. Here again the errors were not to be corrected, so the job of the validator was only to insert the mark ‘1’ in the line corresponding to the first word of the non-identified MWU. For reporting a candidate list was produced, see section 3.5.

To make the overview easier the template was designed with colour codes such that the colour of a column reflects the annotation type it displays. So the data column for PoS and the PoS correction column have the same colour while the lemma data column and its corresponding lemma correction column has a different colour. Columns like word form, the entire sentence and word reference were left white. Figure 1 illustrates a validation file.



	A	B	C	D	E	F	G	H	I	J
1	**Sentences and references**	**POS corrections**	**Lemma corrections**	**MWU Errors**	**Missing MWU's**	**MWU**	**Word**	**Lemma**	**POS**	**Word-ref**
2	corpus_a: 1157 words									
83	fn000260									
84	480									
85	volgens mij volgens mij daar wel ja .									
86							volgens	volgens	VZ(init)	480.1
87							mij	mij	VNW(pr,pron,obl,vol,1,ev)	480.2
88							volgens	volgens	VZ(init)	480.3
89							mij	mij	VNW(pr,pron,obl,vol,1,ev)	480.4
90							daar	daar	VNW(aanw,adv-pron,obl,vol,3o,getal)	480.5
91							wel	wel	BW()	480.6
92							ja	ja	TSW()	480.7
93							.	.	LET()	480.8
94	fn000261									
95	240									
96	da 's natuurlijk wel voordelig .									
97							da	dat	VNW(aanw,pron,stan,vol,3o,ev)	240.1
98							's	zijn	WW(pv,tgw,ev)	240.2
99							natuurlijk	natuurlijk	ADJ(vrij,basis,zonder)	240.3
100							wel	wel	BW()	240.4
101							voordelig	voordelig	ADJ(vrij,basis,zonder)	240.5
102							.	.	LET()	240.6
391	fn000289									
392	480									
393	en dan houdt 't op .									
394							en	en	VG(neven)	480.1
395							dan	dan	BW()	480.2
396						502774#2	houdt	houden	WW(pv,tgw,met-t)	480.3
397							't	het	VNW(pers,pron,stan,red,3,ev,onz)	480.4
398						502774#2	op	op	VZ(fin)	480.5
399							.	.	LET()	480.6
400	fn000290									

Figure 1 Validation template

3.4. Training of validators

The content validation was completed by a temporary staff of junior linguists with a mother tongue or expert knowledge of Dutch. Their training and preparation for the task consisted in first reading the documentation and familiarizing themselves with it. Secondly they all participated in a workshop we organised in order to discuss and clarify in a common forum any question that came up from reading the documentation. This discussion for instance led to the creation of the pick list with tags in the validation template. In the workshop we also presented the validation template and a detailed task description documenting the template and its use and describing the file handling and file naming conventions that would be used.

During the course of validation there was an active communication between the validators. Cases of doubt were discussed via e-mail and frequent reunions where difficult cases were discussed led to a higher inter-validator agreement.

3.5. Error Counting and Results

The requirement from the producer was to deliver error rates and for MWUs recall and precision. We were not asked to correct the errors we identified. In the process, however, it turned out to be more practical for the validators to correct errors instead of simply marking them. This allowed the validators to go back to previous validation decisions and thus ensure consistency and intervalidator agreement. We also thought this would make the validation more useful for the producers.

For identified PoS and lemma errors, the errors and their corrections were listed in the validation report.

For MWUs two lists were produced: One list of wrongly identified MWUs and one list of MWU candidates for not identified MWUs. Precision and recall were calculated on the basis of the number of wrongly identified MWUs found (precision) and the number of non-identified MWUs (recall).

For syntax the list of errors with corrections was simply described in text.

The validation template was not designed with a column in which to register shortcomings or gaps in the documentation. Problems encountered were discussed between the validators in order to establish whether they were due to the documentation or whether they were e.g. systematic errors in the annotation.

4. Documentation Validation

During the content validation process the validators' judgments had to be in agreement with the documentation at hand. Formally we were asked to validate three documentation files, which consisted of four sets of guidelines - for lemmatisation, for the demarcation of multiple word units, for syntactic annotation and for part of speech tagging, respectively. Two additional sources were also used as authoritative documentation: For the identification of one type of multiple word units the guidelines refer to a dictionary, the 13th edition of the dictionary *Groot Woordenboek der Nederlandse Taal* published by Van Dale, and for the correct part of speech tagging of pronouns, the guidelines refer to the electronic CGN-lexicon.

By and large, these six sources covered the annotation in our sample very well, but we were also confronted with problems that we were not able to solve by looking at these sources of information alone. In such cases we always gave the annotation in the corpus the benefit of the doubt and we attributed the lack of clarity to the guidelines.

After the validation of the data, but before the final report was delivered, it came to our knowledge that the annotators had worked with supplementing guidelines for the syntactic annotation. It was also made clear to us that the electronic CGN-lexicon was created only partly before the creation of the corpus and that it contained many data describing, rather than prescribing, the corpus. It therefore contained both authoritative and after-the-fact data. This background information led us to the revision of some of our findings, in part cancelling some found errors in the corpus, in part adding some. At the same time remarks regarding the disagreement between the documentation and the annotation practice were added to the final validation report.

The CGN-lexicon, encompassing much more information than the aforementioned list of pronouns, was an invaluable source of information during the validation process, although we always were in doubt how much authority we really could attribute to it. The CGN-lexicon is composed of data from several sources, for example the Van Dale dictionary mentioned before. These sources do not always agree, and in those cases the creators of the lexicon had to make a choice. The way these choices were made is hidden from those validating or otherwise using the CGN-corpus. This makes the CGN-lexicon a unique, irreplaceable, and canonical linguistic resource for users of the corpus, whatever the lexicon's authoritative status compared with the supreme authoritative status of the four sets of guidelines.

Adding to the confusion was the lexicon field that indicated that the CGN-lexicon had been through a validation process, all word entries being marked with 'V' (valid), 'C' (correct), 'I' (incorrect) or 'O' (not validated). This validation process was not mentioned in the guidelines at hand and it never became clear to us how it had been done. Nevertheless, because of these quality stamps, but even more because there was no other dictionary that we could trust more, we occasionally based our validation decisions on the CGN-lexicon even in cases were the guidelines did not explicitly attribute authority to this lexicon.

The reason we sometimes took refuge to a lexicon was that some annotation decisions were very word specific and could therefore not be based on the documentation, they had to be based on some other source. Native-speaker intuition was an option we could have chosen to solve such issues, but the CGN-lexicon seemed to be the best of all choices, combining all the lexical resources (including native speaker intuition) that had played a role during the construction of the corpus.

4.1. PoS-tagging

The following is an example that illustrates how the validation process sometimes critically depended on the CGN-lexicon. In this example, the validity of the annotation of word *A* was depending on the mere existence of word *B*. In the corpus, *beneden* in the sentence *hij ging naar beneden* (*down*, as in *he went down*) is annotated as a postpositional preposition. We thought this was wrong, because according to the Van Dale dictionary *beneden* in *naar beneden* is an adverb and this was in agreement with the CGN documentation, which says: 'We count only those words as postpositional prepositions which can be preceded by an adverbial pronoun; accordingly, *heen* and *af* are regarded as prepositions, but not *terug*, *weg* and *geleden* (**erterug*, **hierweg*, **waargeleden*); the latter we count as adverbs.' In other words, if *beneden* can be preceded by an adverbial pronoun (*hierbeneden*), then *beneden* is a postpositional preposition, otherwise it is an adverb. The word *hierbeneden* is not in the Van Dale dictionary, and therefore we concluded that *beneden* is an adverb and not a postpositional preposition. But we were wrong: the word *hierbeneden* is a word in the CGN lexicon, and it has a 'V'-marked (validated) reading, so we accepted that *beneden* in *naar beneden* is a postpositional preposition after all.

It was only after seeing that almost 100% of the occurrences of *naar beneden* were annotated as postpositional prepositions that we began to doubt our decision to count this annotation as an error in the corpus. So, in fact, this example not only shows that the required fairness of the validation process forced us to use a lexical resource with a dubious authoritative standing, it also illustrates that we sometimes faced an overwhelming majority of instances that cast doubt on the rightness of the judgment and that we therefore decided to revise the judgment.

4.2. Lemmatisation

In one case we were so much overwhelmed by a wrong annotation practice that we were misled to give all the occurrences of the wrong annotation the benefit of the doubt. This was the case with the lemmatisation of dialect words. According to the guidelines, the lemma of dialect words always is identical with the word itself. However, for most dialect words that seemed to be (or are) inflected forms of non-dialect words the uninflected wordform was taken as the lemma (*diejen – die*, *nen – een*, etc.). This practice was fully supported by the CGN-lexicon, which has 'validated' entries for such words. We assumed that the guidelines were wrong, not the corpus and the CGN-lexicon. However, a representative of the CGN-team told us that the guidelines were right and that the corpus and lexicon were wrong. Luckily it was very easy to spot all dialect words in our sample and we could report an additional 50 lemmatisation errors, tripling the total number of lemmatisation errors.

4.3. Multiple Word Units

There was no friction between the guidelines for the identification of multiple word units on the one hand and other lexical resources on the other. The guidelines were very clear. We found one case that was difficult to decide: the words *hot shots* (both loans from English). According to the guidelines we had to check two circumstances: First, if each of the words occurs in the Van Dale dictionary, then the sequence is not a multiple word unit. Secondly, if the sequence as a whole occurs in the Van Dale dictionary, then the sequence is a multiple word unit, even if each word also is in the dictionary by its own. Now, both *hot* and *shot* are in the Van Dale dictionary and *hot shot* as a sequence is not. So, formally, *hot shots* is not a multiple word unit. On the other hand, *shot* in *hot shots* has a different meaning than *shot* in the dictionary and the compound word *hotshot* occurs in the dictionary. So, in the spirit of the guidelines, *hot shots* is a multiple word unit, but it would have been better if the expression had been spelled *hotshots* (thereby again losing the status of multiple word unit). However, the spelling of the corpus was not to be validated and therefore the only solution was to just accept whatever the corpus annotator had chosen, which was not to regard *hot shots* as a multiple word unit.

4.4. Syntactical Annotation

In general, the guidelines for the syntactic annotation were sufficient to be able to validate the syntactic annotation. There were, however, also some white spots and constructions that needed worked-out examples. The most notable lacuna was the correct treatment of constructs like *van A naar B* (from A to B). The corpus annotators had chosen to annotate these constructs as prepositional phrases with two complements *A* and *B* of the same type OBJ1. However, according to the guidelines this was not allowed. We decided to count the practiced annotation as erroneous. However, the supplementing guidelines for the syntactic annotation, the existence of which we became aware of after the validation, nicely filled out the white spots and also presented many worked-out examples for difficult constructs. These supplementing guidelines supported the coding practice for prepositional phrases with two OBJ1 complements and one or two heads.

5. References

Calzolari, N., K. Choukri, M. Gavrilidou, B. Maegaard, P. Baroni, H. Fersøe, A. Lenci, V. Mapelli, M. Monachini, S. Peperidis (2004): ENABLER Thematic Network of National Projects: Technical, Strategic and Political Issues of LRs. In *Proceedings of LREC 2004, International Conference on Language resources and Evaluation,* Lisboa 2004, side 937-940.

Fersøe, H., H. van den Heuvel, S. Olsen (2006): Validation of third party Spoken and Written Language Resources – Methods for performing Quick Quality Checks. To appear in *Proceedings of LREC 2006, International Conference on Language resources and Evaluation*, Genoa 2006.

Fersøe, H., S. Olsen (2005): Methodology for a Quick Quality Check for WLR-Lexica. Report submitted to ELRA under the ELRA/0209/VAL-1 contract.

Fersøe, H., E. Hartikainen, H. van den Heuvel, G. Maltese, A. Moreno, S. Shammass, U. Ziegenhain (2004): Creation and Validation of Large Lexica for Speech-to-Speech Translation Purposes. In *Proceedings LREC 2004, International Conference on Language resources and Evaluation,* Lisboa 2004, page 1431-1434.

Fersøe, H. (2004). Validation Manual for Lexica. Report submitted to ELRA under the ELRA/0209/VAL-1 contract. See under Validation Standards in http://www.elra.info

van den Heuvel, H., D.J. Iskra, E. Sanders & F. de Vriend (2004): SLR Validation: Current Trends & Developments. In *Proceedings LREC 2004*, Lisbon, Portugal, pp. 571-574.

Yaseen, M., M.Atiyya, C. Bendahman, B. Maegaard, K. Choukri, N. Paulsson, S. Haamid, H. Fersøe, S. Krauwer, M. Rashwan, B. Haddad, C. Mukbel, A. Mouradi, A. Ali, M. Shahin, A. Ragheb, N. Chenfour: Building Annotated Written and Spoken Arabic LRs in the NEMLAR Project. To appear in *Proceedings of LREC 2006, International Conference on Language resources and Evaluation*, Genoa 2006.

Quality control of treebanks: documenting, converting, patching

Sabine Buchholz, Darren Green

Speech Technology Group, Cambridge Research Lab, Toshiba Research Europe Ltd
St George House, 1 Guildhall St, Cambridge CB2 3NH, United Kingdom
{sabine.buchholz, darren.green}@crl.toshiba.co.uk

Abstract

We report about our experiences with using many different syntactically annotated corpora (treebanks). We list various types of format and annotation errors we have noticed and propose common sense as well as novel ways to prevent, detect and handle these. We show how the quality of a treebank's annotation and its documentation are related and how the concepts of patching and versioning that come from the software community can be applied to treebanks in order to improve quality.

1. Introduction

The first author is one of the organizers[1] of the shared task on multilingual dependency parsing for the 2006 Conference on Computational Natural Language Learning, CoNLL-X.[2] For that task, treebanks for 13 different languages were converted to a common format (Buchholz et al., 2006). A further three (Sampson, 1995; Sima'an et al., 2001; Aduriz et al., 2003) were studied or partially converted but in the end not included in the task. So the shared task is easily the biggest "user" of treebanks to date. The converted treebanks have been made available to 28 interested groups and some bugs have already been reported. Through our work at Toshiba Research Europe we have experience with a further four treebanks that offer commercial licenses.

While the majority of treebanks are of high quality, we have noticed problems with some. As many of the treebanks are for languages we do not speak, we tend to notice issues with the form rather than with the content of the annotation. Such problems can often be detected by relatively simple means and are therefore **easily avoidable**. In this paper, we discuss the issues we encountered and suggest methods to detect them. In addition to quality of annotation, we want to draw attention to two other aspects of quality control. One is the quality of documentation of resources and the other is the handling of defects once they are noticed. We will show that these three aspects are related and how improvements in one can benefit the others.

2. Issues with treebanks

The issues we noticed range from pure format to more linguistic problems. While linguists using a treebank are normally only bothered by the latter, computational linguists should be concerned about the former as well, as they can directly affect the outcome of any experiment on the data.

2.1. End-of-line convention

Different operating systems use different control characters to terminate a line. We have encountered one treebank in which some files followed the Macintosh convention of carriage returns and others the UNIX convention of line feeds.

[1]Many thanks to the other organizers: Erwin Marsi, Amit Dubey and Yuval Krymolowski.

[2]http://ilps.science.uva.nl/ erikt/signll/conll/

2.2. Whitespace

People who look at treebank annotations, including annotators, normally do not care much about whitespace. However, when faced with the task of converting a treebank from one format to another, whitespace becomes relevant. We cannot recall any treebank documentation that defined explicitly how whitespace is used. However, when looking at treebank files, it often becomes apparent that whitespace is used in a certain way. Our first version of a script for converting one of the treebanks to the shared task format made many assumptions about where spaces, tabs or blank lines were required or forbidden, based on our study of several treebank sentences. However, every single one of these assumptions was violated by at least one other sentence. From this we draw two conclusions: First, scripts taking a treebank as input should never make such assumptions, as they would introduce errors in the output. Second, treebank providers should explicitly mention any rules about whitespace that should hold, and then ensure that they do. We have noticed spurious whitespace in at least two other treebanks.

2.3. Other delimiting characters

Sometimes, characters other than whitespace are used to separate one field from another in a treebank. Typically colons or semicolons, single or double quotation marks and round, square, curly or angle brackets fill this role. We have encountered four treebanks where for various sentences one of these characters was missing, doubled, misplaced, or replaced by another character.

These errors are particularly serious if they occur with XML delimiters, such as the quotation marks around attribute values or the angle brackets around XML tags. In one treebank, several files were not valid XML due to this kind of error.

2.4. Character encoding

We have encountered one treebank in which some files were encoded in UTF-8 (Unicode) and others in the Windows-1254 encoding.

2.5. Encoding of special characters

In spite of Unicode, which can encode practically all languages, language group-specific encodings remain in use. In fact, only one treebank we encountered uses

Unicode (Hajič et al., 2004). Sometimes, an encoding (e.g. ASCII or one from the ISO-8859 family) is used for a treebank because it is the standard encoding for the treebank's language, but the texts on which the treebank is based originally contained a few characters that are not covered by the encoding. A common solution is then to encode these characters as named character entities. For example, the Prague Dependency Treebank (Böhmová et al., 2003) uses `&agrave;` for à, the British component of the International Corpus of English (ICE-GB) (Greenbaum, 1996) uses `&degree;` for ° and SUSANNE (Sampson, 1995) uses `<deg>` for the same symbol. In ICE, we have encountered some inconsistencies in these entity names, e.g. both `&much-less-than;` and `&much-smaller-than;` and both `&Beta;` and `&BETA;` are used. In addition, some special characters that should have been encoded were not.

Certain characters have a special meaning in most annotation schemes (see Section 2.3.). Therefore, they should be encoded or escaped when they are meant to represent the original character. In XML, either single or double quotation marks must enclose attribute values.[3] If the attribute value is meant to contain the same quotation mark as is used for the enclosure, it has to be encoded as `&apos;` or `&quot;` respectively.[4] In addition, & has to be encoded as `&amp;`.[5] We have encountered two treebanks that failed to do that and therefore contained files that were not valid XML.

Actually, the need to encode quotation marks can be avoided by putting information such as the stem or lemma in an element of its own instead of some element's attribute.[6] However, this annotation style was not chosen in any of the four XML-encoded treebank which we encountered that contain lemma information[7] (although it is used in the SGML version of the Prague Dependency Treebank). Three treebanks even encode the word forms as attributes in their XML versions.

2.6. Presence and order of all fields

We have encountered two treebanks where the POS tag for a word was missing, another one where the lemma information was missing in part of the treebank, a fourth one where one word token had two conflicting PoS attributes and a fifth one where one lemma was empty, four constituent labels were missing and one constituent had two different function labels. Two of these treebanks also had at least one case where pieces of information (such as lemma, POS and other features) were in a different order than in the rest of the treebank.

2.7. Typos in labels

We have encountered five treebanks that had typos in labels, i.e. in the names of POS, constituents, grammatical functions or additional features. There was also at least one case of the lemma being for a completely different word than the token itself and one treebank where a few XML labels had inconsistent case (`<W>` ended by `</w>` and vice versa).

2.8. Typos in structure

Tree structure can be expressed in various ways: for example, phrase structure by indentation or paired brackets or tags, dependency structure by a dependency tree with indexes on the leaves indicating linear order or by a linearly ordered list where each token has a field containing the index of its head. In all these cases, typos in the structure can lead to incorrect annotation. If the result is incorrect attachment, it is very hard to detect automatically. However, we have encountered four types of problems that can be spotted automatically. One XML-encoded phrase structure treebank contained more than 40 cases of XML tags that were not properly nested. One dependency treebank contained some tokens for which the index of the head was larger than the highest index in the sentence. Two dependency treebanks contained sentences with dependency cycles, i.e. where either a token directly links to itself or it links to another token that links to another token, etc., that links back to the first. One treebank encoded phrase structure but allowed for discontinuous constituents. The beginning and end of a discontinuity in a constituent *xp* is marked by a mirrored pair of symbols *xp*– and –*xp*. We have encountered cases where there were beginnings without ends and vice versa and at least one case where the depth of the beginning and end was different (although it must be identical). In theory, the same problem can occur with normal phrase structure brackets, although we have not encountered that.

2.9. Potential errors in the linguistic annotation

It is rare that one notices errors in the linguistic annotation for a language one does not speak. However, sometimes a general linguistic understanding of a treebank's annotation scheme is enough to at least suspect that something is wrong (and consequently report it to the treebank's authors). For example, one treebank has separate labels for finite, non-finite and averbal (sub)clauses. Given these labels, we suspect an error when we encounter, for example, a constituent without a verb being labelled a finite clause or a constituent with a verb labelled an averbal clause. Sometimes a dependency treebank annotation scheme contains a designated label for the root token of the whole sentence or clause. For example, in the Metu-Sabancı treebank (Oflazer et al., 2003; Atalay et al., 2003), this is the `SENTENCE` label. So we suspect an error when we encounter a sentence that does not contain this label.

3. Preventing and detecting problems

In the previous section, we listed many types of problems that we noticed in various treebanks. In this section, we propose ways to prevent or detect these problems.

[3]`<token lemma="O'Neill"> O'Neill </token>`

[4]`<token form='O&apos;Neill'> O'Neill </token>`

[5]See the W3C Recommendation (`http://www.w3.org/`) on XML 1.1, Section 2.4 "Character Data and Markup"

[6]`<token> <lemma> O'Neill </lemma> <form> O'Neill </form> </token>`

[7]Possibly because it was felt that only the original word forms are "data" and everything else is meta-data.

3.1. Explicit documentation

We think that the first step towards ensuring that a treebank does not exhibit any of the problems discussed in Sections 2.1. to 2.7. is to explicitly document the conventions used for a treebank. This will raise awareness of these conventions in everybody working on or with the treebank. In particular, machine-readable lists of all special character encodings and labels used in the treebank would be very helpful. Creating such lists encourages treebank providers to check for mistakes, and having such lists readily available encourages script writers to actually check whether the data adheres to it. The majority of the treebanks do not provide such lists in easily machine-readable form (although sometimes they can be copy-pasted from web pages or PDF files).

3.2. Format checker

An explicit documentation of conventions can be used as a specification for an automatic format checker, i.e. software that reads in treebank files and flags cases that violate the specification. Such software could automatically detect the problems described in Sections 2.1. to 2.8. Some treebank projects use tools, such as Annotate (Plaehn, 2002), for treebank creation/editing that directly enforce the format, for example by providing a pull-down menu of labels instead of having annotators type in labels by hand. However, creating such a tool from scratch or even adapting an existing one to a different annotation scheme can be too big an overhead for a treebank project. The next best thing then is to write a format checker, and use it at regular intervals, ideally after each editing session. Once explicit documentation is available, writing such a format checker should be relatively easy for somebody with experience in scripting languages such as Perl or Python. If the treebank authors do not have this experience themselves, they should encourage users to write and submit such software, as it is in everybody's interest that it is used.

Some people might think that some treebank formats, such as XML, are inherently less susceptible to format errors. In our experience, however, this is not true. We have encountered at least two treebanks that use the XML format but apparently have been created or modified with a simple text editor and without validating the XML afterwards (see Sections 2.3. and 2.5.). In fact, only three out of seven XML-encoded treebanks came with a DTD or a reference to an XML Schema. We have also seen several scripts for converting treebanks to or from XML that parse or construct XML "by hand", i.e. through regular expressions and print statements instead of by using available XML library functions. This method is error-prone and should be avoided. In general we think that writing a format checker for a non-XML format is not more difficult than writing a DTD/Schema for an XML-based format. Note also that some restrictions, e.g. on cycles cannot be expressed in a DTD/Schema.

In general, we have not found one format better than another. We have, however, noticed two practices which we would discourage as they make the introduction of errors easier and their automatic detection harder. The first one is using XML attribute values with internal structure, e.g. attribute values (encoded as strings) that are lists or sets of more atomic values, such as morphological feature values. Such an annotation style means that annotators have to deal with two different ways to encode structure, one that uses XML tags and one that uses for example brackets. It also means that these attributes cannot easily be validated through a DTD/Schema. The second practice that we would like to discourage is to keep different levels of annotation in separate files, e.g. one file with the part-of-speech annotation and another with the syntactic annotation of the same text. This approach has several disadvantages:

- Annotators of one level do not see the other level, so there is a higher chance of inconsistent annotation and a missed chance for detecting errors in the lower levels (e.g. the Penn treebank (Marcus et al., 1994) contains some VPs whose only verb is tagged as a noun).
- Some information, such as the words themselves, has to be repeated. This can lead to inconsistencies between the two versions if any corrections (e.g. of typos or tokenization errors) or changes (e.g. to the multiword policy) are carried out in only one version.
- Format checkers and especially conversion scripts would have to read in and parse more than one file at a time and establish the correspondence between tokens before being able to check or convert the content. This makes these tools more complicated and therefore less likely to be developed and used.

We have encountered at least one treebank that used the separate-file approach and suffered from not being format checked.

3.3. Conversion by head table

The errors in clause labelling described in Section 2.9. were detected during the conversion from the original phrase structure to the shared task dependency format. Jelinek et al. (1994) introduced the idea of a head table to automatically determine the head child of each constituent in a phrase structure treebank and Magerman (1995) used the first version for the Penn Treebank (Marcus et al., 1994). Collins (1996) slightly modified that table and used it to convert the Penn Treebank phrase structures into a collection of bilexical dependencies. Yamada and Matsumoto (2003) and Nivre and Scholz (2004) built on that idea and used a slightly modified version of Collins' head table to convert the Penn Treebank into an unlabelled or labelled dependency treebank respectively, on which a dependency parser can be trained and tested. A similar approach was used to convert the Alpino (van der Beek et al., 2002), BulTreebank (Simov et al., 2005; Marinov and Nivre, 2005), Bosque (Afonso et al., 2002), Cast3LB (Civit Torruella and Martí Antonín, 2002), Sinica (Chen et al., 2003), Talbanken05 (Nilsson et al., 2005), TIGER (Brants et al., 2002) and Japanese Verbmobil (Kawata and Bartels, 2000) treebanks to labelled dependency treebanks (or partially labelled in the case of Cast3LB).

If one looks at Magerman's and especially Collins' head table (Collins, 1999, p.240), one notices a number of linguistically implausible potential head children, e.g. ADJP, NN, NNS[8] or NP as head children of VP. For anyone who has worked with the Penn Treebank, it is clear that these try to compensate for annotation errors in the treebank. Although this makes sense for the researcher only interested in converting a messy treebank as well as possible, we propose to actually use the conversion by head table as an instrument of quality control. This can be achieved by three additions to the head table format.

Firstly, a flag for each element in a rule stating whether this is a linguistically sound or rather a "heuristic" potential head child. Whenever a "heuristic" rule part has to be used, the head finding algorithm should output a warning about this fact. Secondly, the direction is not specified per line (parent constituent) but per element (head child). This was implicitly already needed for Collins (1999)'s special rules for NPs and explicitly realized in Bikel (2002)'s reimplementation of Collins' parser. Thirdly, in addition to the directions "left" and "right" that state that the head child is the leftmost, respectively rightmost, matching child, we propose a requirement of "exactly one".[9] If for example one thinks that an adjective is a potential head child of an ADJP, and that at most one adjective should occur as a direct child of an ADJP, one would write a rule `ADJP: "only" A`. The algorithm that applies the head table should then output a warning whenever it tries to apply an "only" rule in a context where there are several matching elements. "Only" rules will be especially frequent for treebanks such as Alpino, Bosque, Sinica, Talbanken05, TIGER and Verbmobil, which explicitly mark heads, as the linguistic definition entails that there should be only one head (although multi-words can sometimes be an exception). The proposed method for detecting errors, which involves linguistic understanding, can be complemented by fully automatic methods such as Ule and Simov (2004) and Dickinson and Meurers (2005).

4. Handling problems

Magerman's original head table dates from 1995 and the much-cited work by Collins that also uses it dates from 1996 and 1997. There has been a new release of the Penn Treebank since then (Treebank-3, 1999). However, many of the errors that necessitated the "heuristic" head rules are still in the treebank, see e.g. (Dickinson and Meurers, 2005). There are two possible explanations for this: Either the treebank users did not tell the treebank authors about these errors[10] or the treebank authors did not do anything with this information.

Whenever we or one of the shared task participants[11] noticed errors in the shared task data that were due to errors in the original treebank (and not to conversion bugs), we reported these back to the treebank providers, and we are very pleased to report that, whenever feasible, they were corrected, sometimes within days. As a consequence, some of the errors reported in this paper are no longer present in the respective treebanks.

4.1. The challenges

We think that this feedback loop is an important part of quality control and should be encouraged explicitly. This can be done by clearly stating in the treebank's documentation that bug reports are welcome and by providing a contact address for reporting them. Although a quick fix of reported problems is the best encouragement for users actively working with the treebank to report further problems, should they find any, this might not always be feasible. In addition, treebank providers might be reluctant to officially release new versions of a treebank at very short intervals or for very minor changes. Also, funding for the project might have ended, and the original authors might have moved on to other jobs. In that case, even if users notice errors and fix them in their own copies of the treebank, there is currently no easy way for future users to profit from those fixes, as license restrictions often prohibit one user from passing on modified versions to another. Finally, when treebanks are used to train and test systems such as taggers and parsers, and when one wants to be able to compare results by different systems at different times, it is vital that one compares against exactly the same version of the treebank.

4.2. Patching treebanks

We think that we can learn from software developers, especially the open source community, how to achieve these goals and overcome these problems. We introduce the concept of "patches" to treebanks. In essence, treebanks are text files, just as source code is. After a treebank author or user has made changes to a treebank in order to fix reported or noticed problems, the Unix `diff` utility allows them to list all changes in a very concise manner. If produced with the options `-c` or `-C`, the output of `diff` can be used as the input of the Unix `patch` utility to apply the same changes to somebody else's copy of the original treebank. This means that

- if users think they know how to fix a problem, they can make fixes to their copy and then have a very easy and foolproof way to report back to the treebank authors how they propose to change the treebank.
- treebank authors then simply have to decide whether to accept or reject the proposed patch; in either case they can simply post it to the treebank's web site with an explanation. This directly makes accepted patches available to other users without the need for a new release of the treebank. Also, authors can automatically apply patches accepted since the last release for the next release, thereby keeping their own effort to a minimum.
- if a treebank is not maintained any longer by the license holders, users can probably share patches among themselves without violating the treebank's license, as patches only contain fragments of the treebank texts and are useless without the original treebank. This would allow even the Penn Treebank to be

[8]NN and NNS are the Penn Treebank POS tags for singular and plural nouns, respectively.

[9]Based on an idea by Montserrat Civit (p.c.), although with a slightly different interpretation.

[10]Note that Michael Collins was a PhD student at the University of Pennsylvania itself at that time.

[11]Thanks to Ryan McDonald, Svetoslav Marinov and Masayuki Asahara for reporting bugs.

patched. Obviously there is a risk of diverging or disputed fixes, as noted in Blaheta (2002), and there is also the problem of new users not knowing about or not having access to older fixes.[12]

4.3. Versioning

Out of the twenty-one treebanks we studied, four used numbers for major releases/versions only (e.g. TIGER Version 1 and 2), nine used two-part numbers for the major and minor release (e.g. Bosque 7.3) and eight did not use any apparent version numbers. We propose that all treebanks follow a versioning scheme similar to software, with major and minor releases and numbered patches. This will allow researchers to state precisely which version of a treebank their reported results pertain to. Ideally, all old versions should be kept available, so that if some researchers wants to replicate or compare to somebody else's results years later, they can still get the appropriate version. They can then state something like "applying our new method to the same treebank version as X, we get y% while X got only z%; applying our method to the latest version ($i.j.k$), we get u%". As treebanks can be big, treebank authors might decide to keep only major releases and allow users to reconstruct in-between versions by applying a number of patches to the preceding major release.

Obviously the boundary between a major and a minor release or between a minor release and a patch is a fuzzy one but the term "release" seems to suggest some important improvement has been made. Therefore treebank providers are probably reluctant to make a new "release" if the changes "only" fix problems such as the ones discussed in Sections 2.5. to 2.7. and prefer to wait for the next release, which might be years in the future. It is for the quick fixes to these problems in particular that we advocate the use of patches.

Finally, we can only recommend the use of versioning software such as CVS (Concurrent Versions System)[13] or Subversion[14], both of which are available for free, to help keep track of versions.

5. Summary

We have described the quality problems we encountered during our work with a large number of treebanks. We propose preventing and detecting these problems through the explicit documentation of format conventions, in particular machine-readable lists of labels, through the use of format checkers and, for phrase-structure treebanks, through additions to the head table mechanism. We also plead for a structured approach to handling reported problems and propose to adopt the software concepts of patching and versioning. We hope that this paper will help to improve existing and future treebanks.

Acknowledgements

Many thanks to the treebanks providers who kindly allowed the use of their treebanks for the shared task, answered questions about the annotation and fixed errors, to the other organizers and some treebank providers who converted the treebanks and to the shared task participants who asked useful questions and noticed bugs. Without the help of all these people the work on which this paper is based would not have been possible.

[12]For example, the URL mentioned in Blaheta (2002) for his list of corrections to the Penn Treebank does not exist anymore.

[13]See e.g. `http://www.nongnu.org/cvs/cvs.html`

[14]See `http://subversion.tigris.org/`

6. References

I. Aduriz, M. Aranzabe, J. Arriola, A. Atutxa, A. Daz de Ilarraza, A. Garmendia, and M. Oronoz. 2003. Construction of a Basque dependency treebank. In *Proc. of the Second Workshop on Treebanks and Linguistic Theories (TLT)*.

S. Afonso, E. Bick, R. Haber, and D. Santos. 2002. "Floresta sintá(c)tica": a treebank for Portuguese. In *Proc. of the Third Intern. Conf. on Language Resources and Evaluation (LREC)*. ELRA.

N. Atalay, K. Oflazer, and B. Say. 2003. The annotation process in the Turkish treebank. In *Proc. of the 4th Intern. Workshop on Linguistically Interpreteted Corpora (LINC)*.

D. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proc. of the Human Language Technology Conf. (HLT)*.

D. Blaheta. 2002. Handling noisy training and testing data. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 111–116.

A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká. 2003. The PDT: a 3-level annotation scenario. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*.

S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. 2002. The TIGER treebank. In *Proc. of the First Workshop on Treebanks and Linguistic Theories (TLT)*.

S. Buchholz, E. Marsi, A. Dubey, and Y. Krymolowski. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of the Tenth Conf. on Computational Natural Language Learning (CoNLL-X)*. SIGNLL. To appear.

K. Chen, C. Luo, M. Chang, F. Chen, C. Chen, C. Huang, and Z. Gao. 2003. Sinica treebank: Design criteria, representational issues and implementation. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*.

M. Civit Torruella and M$^{\text{a}}$ A. Martí Antonín. 2002. Design principles for a Spanish treebank. In *Proc. of the First Workshop on Treebanks and Linguistic Theories (TLT)*.

M. Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proc. of the 34th Annual Meeting of the ACL*.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proc. of the 35th Annual Meeting of the ACL*.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

M. Dickinson and W. D. Meurers. 2005. Prune diseased branches to get healthy trees! In *Proc. of the Fourth Workshop on Treebanks and Linguistic Theories (TLT)*, pages 41–52.

S. Džeroski, T. Erjavec, N. Ledinek, P. Pajas, Z. Žabokrtsky, and A. Žele. 2006. Towards a Slovene dependency treebank. In *Proc. of the Fifth Intern. Conf. on Language Resources and Evaluation (LREC)*.

S. Greenbaum, editor. 1996. *Comparing English Worldwide: The International Corpus of English*. Clarendon Press, Oxford.

J. Hajič, O. Smrž, P. Zemánek, J. Šnaidauf, and E. Beška. 2004. Prague Arabic dependency treebank: Development in data and tools. In *Proc. of the NEMLAR Intern. Conf. on Arabic Language Resources and Tools*.

E. Jelinek, J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi, and S. Roukos. 1994. Decision tree parsing using a hidden derivation model. In *Proc. of the Workshop on Human Language Technology*.

Y. Kawata and J. Bartels. 2000. Stylebook for the Japanese treebank in VERBMOBIL. Verbmobil-Report 240, Seminar für Sprachwissenschaft, Universität Tübingen.

M. T. Kromann. 2003. The Danish dependency treebank and the underlying linguistic theory. In *Proc. of the Second Workshop on Treebanks and Linguistic Theories (TLT)*.

D. Magerman. 1995. Statistical decision-tree models for parsing. In *Proc. of the 33rd Annual Meeting of the ACL*.

M. Marcus, G. Kim, M. Marcinkiewicz, R. Mac-Intyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn treebank: Annotating predicate argument structure. In *Proc. of the Workshop on Human Language Technology*.

S. Marinov and J. Nivre. 2005. A data-driven dependency parser for Bulgarian. In *Proc. of the Fourth Workshop on Treebanks and Linguistic Theories (TLT)*, pages 89–100.

J. Nilsson, J. Hall, and J. Nivre. 2005. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proc. of the NODALIDA Special Session on Treebanks*.

J. Nivre and M. Scholz. 2004. Deterministic dependency parsing of English text. In *Proc. of the 20th Intern. Conf. on Computational Linguistics (COLING)*.

K. Oflazer, B. Say, D. Zeynep Hakkani-Tür, and G. Tür. 2003. Building a Turkish treebank. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*.

O. Plaehn. 2002. Annotate Bedienungsanleitung. NEGRA project report, Universität des Saarlandes, Germany.

G. Sampson. 1995. *English for the Computer: The SUSANNE Corpus and analytic scheme*. Clarendon Press.

K. Sima'an, A. Itai, Y. Winter, A. Altman, and N. Nativ. 2001. Building a tree-bank of modern Hebrew text. In B. Daille and L. Romary, editors, *Journal Traitement Automatique des Langues (t.a.l.) — Special Issue on Natural Language Processing and Corpus Linguistics*.

K. Simov, P. Osenova, A. Simov, and M. Kouylekov. 2005. Design and implementation of the Bulgarian HPSG-based treebank. In *Journal of Research on Language and Computation – Special Issue*, pages 495–522. Kluwer Academic Publishers.

T. Ule and K. Simov. 2004. Unexpected productions may well be errors. In *Proc. of the Fourth Intern. Conf. on Language Resources and Evaluation (LREC)*, pages 1795–1798.

L. van der Beek, G. Bouma, R. Malouf, and G. van Noord. 2002. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN)*.

H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proc. of the 8th Intern. Workshop on Parsing Technologies (IWPT)*.

Evaluation of a diachronic text corpus

Mikko Lounela

Kotimaisten kielten tutkimuskeskus
Sörnäisten rantatie 25
00510 Helsinki
Finland
mikko.lounela@kotus.fi

Abstract

This paper describes an evaluation procedure of a two-part diachronic corpus of 20^{th} century periodicals, and the results of the evaluation. The corpus was composed by scanning the original papers, running them through OCR, and automatically re-structuring the text to TEI-based XML format, with original images linked to the text. A small sub-corpus was extracted and its structure was enhanced and corrected manually. The testing procedure was based on manual evaluation of random samples of the corpus, from which both the markup and the OCR quality were tested. In addition to the manual evaluation, the OCR quality was tested semi-automatically, using a morphological analysis tool. The results show that the composing method leaves a considerable amount of errors both in the markup and the actual text, and suggest directions for correcting actions. The testing procedure itself brings up some questions about how this kind of internally varying, structurally complex corpus should be tested, and what would be the best base for composing the test material.

1. Introduction

In year 2003, The Academy of Finland allocated 300 000 euros for a one-year project composing a diachronic research corpus of Finnish periodicals published in twentieth century. Three institutions took part in the project. The Research Institute for the Langauges of Finland (RILF) coordinated the project, and accounted for composing and evaluating the final corpus. The Microfilming- and Conservation Centre of the Helsinki University Library was responsible for collecting the periodicals, and scanning and optical character recognition (OCR) of the text material. The Copyright Society Kopiosto worked out the leagal issues considering (limited) re-publisihing of the texts for research purposes.

When composing the corpus, text was re-structured from layout-based WordML to text-structure-based TEI-compatible XML, see TEI (2006). A small sub-corpus was extracted from the whole, and its markup and meta-information were enhanced by hand. The original images of the periodical pages were linked to corresponding parts of the text. No morphological or syntactic analysis was included in either part of the corpus. Henceforth, the main corpus will be called base corpus, and the subcorpus will be called core corpus. For overview of our data model, see Lounela (2002) and Lehtinen and Lounela (2004). Note that the current corpus was only structured to paragraph level.

2. Goal of the paper

In this paper, I describe the effort of evaluating the result corpus of the project, concentrating on three problems: sampling a varying, annotated corpus, evaluating the XML and OCR quality of the corpus manaually, and using a morphological analyser for evaluation of OCR. First, I have to explain briefly the structure of the compiled corpus. Then I go through the methods and results of evaluating the XML-structure and OCR quality. Finally, I make some points about collecting and testing this kind of a diachronic text corpus.

3. The Corpus

The size of the base corpus is about 8.1 million words and 3.4 million markup items (XML tags or attribute-value pairs). It consists of material of 26 annual volumes of 4 periodicals, totaling 385 issues. The publishing time of the periodicals in the corpus is between years 1917 and 1972. Suomen Kuvalehti (5 461 557 words, 343 issues, 7 annual volumes) is a general weekly periodical. Lakimies (1 362 392 words, 15 issues, 8 volumes) is a periodical of a juridical association. Suomi (728 589 words, 7 issues, 5 volumes from 1917 to 1938) is a national romantic scholarly periodical from the beginning of the century. Historiallinen aikakauskirja (590 985 words, 20 issues, 5 volumes from 1917 to 1945) is a periodical of a historical society.

The base corpus is organised in such a way, that each file includes one issue of a periodical. The automatic markup process aimed at recognising feature borders, headers, paragraphs, captions, line-, column-, and page breaks, and such non-editorial elements as page numbers. Captions, page numbers, and other text elements disturbing the regular text flow were moved to the end of the files, and linked to their original positions. An automatically created meta-information file was attached to each issue, including publishing information about the text, along with links to the images of the original paper issue. The meta-information system follows the guidelines of expressing Dublin Core elements in RDF format, see Kokkelink and Schwänzl (2002).

The core corpus consists almost solely of Suomen Kuvalehti (SK), a periodical that also forms the backbone of the base corpus. It includes four issues of each annual volume of SK, plus one issue of each of the other periodicals, including about 667 000 words and 359 000 markup items. The texts are re-structured so that each XML-file consists of one feature, and each feature has its own meta-information file. The markup of the texts in the core corpus is based on the automatic markup of the base corpus, but it is (partly) corrected manually: the feature border elements and text displacing decisions were checked, along with some markup considering headers and paragraphs. The meta-information consists of publishing information, augmented manually with key words according to VESA key word thesaurus, see HUL (2000), and feature type codes. For a metadata-oriented introduction to the corpus, see Heikkinen et al. (2005) (in Finnish).

In addition to the XML-texts and the meta-information, the corpus includes images of the originals of each page in the corpus. The page break and feature border tags in the XML-markup act as links to the corresponding image. The images are also linked to XML-files through the metadata-files. One of the purposes of this linking is to provide the interested researchers the look and feel of the original text, other is to make it possible to check the correctness of the OCR:ed text in suspicious cases.

4. The Corpus Evaluation Scheme

The corpus validation was carried on using random samples of the XML files. The sample amount and size were based on a validation manual released by Oxford University Computing Services, (OUCS, 2003). The samples were collected, and the markup made visible in HTML-format with a self-written program. We used a test material of 70 samples from the base corpus. Each sample included 500 meaningful items (elements, attributes, or word parts). The test material included 24 201 words (after words divided by line breaks were united), 6 315 tags and 1 843 attribute-value pairs (1 196 of which were rendering information that was not tested). The core corpus was tested with 50 such samples, totaling 14 585 words, 3847 tags, and 1 257 attribute-value pairs. The proportion of text (words) vs. markup (tags and attribute-value pairs) in test material was 74.8 / 25.2 % for the base corpus, and 74.1 / 25.9 % for the core corpus.

The corpus was tested by proofreading the sample sets, comparing them to the linked images. The actual work was carried out using a self-designed, HTML-based test bed that shows the test material and linked images side by side. A regular WWW-browser acted as user interface for the work. We found different kinds of XML and OCR errors in the digitised text, depending on varying quality and layout of the original periodicals. We created classifications of the failures (for OCR and XML separately), taking into account the effect on usability of the corpus. This was done previous to the actual evaluation, with a preliminary sample set. The precision of the markup was reduced somewhat after the preliminary classification. We noticed that the layout-based recognition of the text elements disturbing the text flow did not work properly. About half of the text transferring decisions were mistaken. So, we decided to return the transferred elements (other than page numbers) to their original places before the final test round. This applies to the base corpus. For the core corpus, these failures were corrected manually. Also the distinction between line- and column breaks worked so badly that we decided to reduce the column breaks to line breaks in both parts of the corpus.

For the actual evaluation, we produced another pair of sample sets. The base corpus test material was read through twice - first for evaluating the markup, then for evaluating the OCR quality. For the core corpus, only markup was tested, as the same text content appears in the base corpus. The detected errors were counted and divided into classes.

After manual evaluation, a morphological analyser for Finnish, Fintwol, was used to re-test the OCR quality of the base corpus. About the morphological analyser, see LINGSOFT (2006), Koskenniemi (1983). The text contents were run through Fintwol, and unrecognised words were captured. These were again checked against the images, and classified with another classification. The results of the two OCR quality tests were compared.

The test results were produced by people who were not directly involved in producing the data they were testing. The meta-information files of the corpus were not tested.

The results of the OCR test lead to a decision of manually proofreading the text content of the core corpus. This work is in progress. That way we will have a small corpus with manually corrected markup and manually corrected text, along with the larger corpus of worse quality.

5. Evaluating the markup

The failure types of the XML-structure, along with their frequencies in the base and core corpus tests are expressed in table 1. The table shows also the size of the markup item sets in which the failures can occur (applicable elements). The failures are classified according to their level in the text structure. The classes are: feature-level failures (errros in the distribution of the feature boundary elements), paragraph-level failures (errors in selection and distribution of the paragraph and header elements), line-level failures (superfluous or missing line break elements), discontinuous text recognition failures (errors in detecting and displacing discontinuous text), and image linking failures (errors in links to original images from feature boundary and page break elements).

When we relate the figures of the classified failures to the sizes of the relevant tag sets, we may come to some conclusions about the structural evaluation of the corpus.

* The figures show that the automatic feature boundary heuristics worked badly - more than half (66.7 %) of the decisions made by it failed. This was corrected by hand to the core corpus, resulting in an error rate of 2.9 per cent.

* The failure rate in paragraph-level was 6.9 % in base corpus, and 7.6 % in core corpus. Many of these errors origin from the advertising material with somewhat lively layout. This material is more frequent in SK, which is better represented in the core corpus.

* The line level errors give rates of 2.7 / 2.1 %. The markup of this level is copied directly from the WordML markup. Most of these errors can be explained by the fact that super- and subscript caused extra line breaks int the OCR of the scholarly periodicals.

Failure	Base Corpus	Applicable elements	Core Corpus	Applicable elements
Feature-level failures	48	72	2	69
Paragraph-level failures	83	1199	41	538
Line-level failures	122	4528	54	2616
Discontinuous text recognition failures	16	153	3	187
Linking failure	2	144	4	125
Total failures	**271**	**6196**	**104**	**3535**

Table 1: Text structure errors

* Discontinuous text recognition failures (including page numbers etc. and in core corpus also captions and other elements moved aside from disturbing the main text flow) give 10.5 % / 1.6 % failure rate. These numbers are not mutually comparable, because most of the text diplacings were cancelled in the base corpus, and the displacing decisions were checked by hand for the core corpus.

* Linking failure rates are 1.4 / 3.2 %, a number that indicate higher error rate in the core corpus. The amounts of failures are only 2 and 4, though. Overall, it seems that the linking is working quite well in both parts of the corpus.

The total failure rate is 4.4 % of the applicable elements (the element set in which these errors can occur) in the base corpus, and 2.9 % in the core corpus. Corresponding numbers when compared to the amount of all the markup items are 3.3 % and 2.5 %.
To evaluate the adequacy of sampling, we produced two extra sample sets from the base corpus. These were produced with the same specifiactions as the original sample set. The total amounts of the markup items in these sets are 6 248 (4 150 elements) and 6 738 (4 078 elements).

* Extra set 1 (4 150 elements):

  - 3228 line breaks (77.8 % of elements),
  - 647 paragraphs (15.6 %),
  - 118 divisions (28.4 %),
  - 111 gaps (2.7 %),
  - 56 headers (1.3 %).

* Extra set 2 (4 078 elements):

  - 2943 line breaks (72.2 %),
  - 784 paragraphs (19.2 %),
  - 187 gaps (4.6 %),
  - 155 divisions (3.8 %),
  - 78 headers (1.9 %).

A statistical evaluation of the distribution of the most common elements in the sets suggest that the sample set of this size is adequate for testing the structure of this kind of material. This is indicated by a t-test, calculated from the absolute numbers of the elementsd in the extra sets. The test shows that the significance values are well below the 0,05 border value[1] (see table 2, below).

6. Evaluating the text recognition

The quality of the text recognition was only tested from the base corpus. Forr this, we used the same test material that was used for evaluating the XML-structure. The text was read through, and all the discovered differences between the corpus text and the digitised image of the original page were classified. The OCR errors that would lead to acceptable words in morphological analysis were counted separately for further purposes.
Word-level errors were separated from punctuation-level errors (strings not including alphanumeric characters). The word-level failures were classified to misrecognised, attached, split, suoperfluous, missing and unrecognisable words. The last category sources from phonetic transcription in one of the original periodicals. Such errors as sub- or superscripts separated from words were not considered text level errors as they are included in the line break recognising failures in the XML structure test. The Suomi-magasine includes a lot of linguistic articles containing phonetic transcription, which is not recognisable by OCR, let alone by morphological analyser. This error type was counted separately (and included in the overall error rate). Table 3 shows results of this test.
The overall failure rate of the test material is 5.6 %. Some of these failures can be seen as more severe than others, eg. an extra punctuation mark is less harmful than a misrecognised word. The punctuation recognition errors are mostly such that they can't be noticed using a morphological analyser. Also, a fair amount of OCR errors lead to words that the morphological analyser will recognise as a normal word.
If we divide the word-level OCR failures to ones that are detectable with Fintwol, and ones that are not, we get figures that are better comparable with validation using morphological analyser. Two figures interest us, the amount of word-level failures (total failures excluding failures conserning non-alphanumeric strings), and the word-level failures that Fintwol interprets as correctly spelled words. Total word-level failure rate is 4.7 % of tested words. The errors that resemble some real word, abbreviation or other entity so that Fintwol would not capture the error are included in this number. These constitute 0.5 % tested words.

[1]The satistical test was performed by Tuula Kähkönen, of which I want to express my gratitude.

Test value = 0						
	t	df	Significance (2-tailed)	Mean difference	95 % Confidence interval of the difference	
					Lower	Upper
Set 1	1.367	4	.243	832.000	-857.63	2521.63
Set 2	1.527	4	.202	829.400	-878.78	2337.58

Table 2: The t-test results of the extra sample sets.

Failure type	**Number of failures**	**% of failures (1346)**	**% of words (24 201)**
Words with misrecognised letters	505	37.5 %	2.1%
Attached words	51	3.8 %	0.2 %
Split words (number of extra parts)	146	10.8 %	0.6 %
Superfluous words	24	1.8 %	0.1 %
Missing words	159	11.8 %	0.7 %
Unrecognisable words (phonetic transcription)	262	19.5 %	1.1 %
Total word-level OCR failures	**1147**	**85.2 %**	**4.7** %
Failures conserning non-alphanumeric strings	199	14.8 %	0.8 %
Total OCR failures	**1346**	**100 %**	**5.6** %

Table 3: OCR errors

7. Evaluating the text recognition using a morphological analyser

The OCR quality was re-tested using the morphological analyser Fintwol. The XML codes were stripped off the test material, and the text was run through the analyser. The words not recognised by Fintwol were captured and compared to same text in digitised image originals. Table 4 shows results of this test.

The rate of words and names, that were correctly recognised by OCR, but not recognised by Fintwol is 4.5 % of the text content (43.5 % of Fintwol alarms). This comes mainly from three sources: change of spelling and punctuation conventions during the 20^{th} century, the amount of foreign words (mainly Swedish and English) in the corpus, and the fact that Fintwol does not recognise all proper nouns.

Earlier we noticed that Fintwol leaves 0.5 % of misrecognised words of the test material undetected. The amount of misrecognised and unrecognisable words in the Fintwol test (table 5) is 963, which is 4.0 % of all the words. Assuming that Fintwol lets through 0.5 % of the misrecognised words would lead us to overall OCR error rate of 4.5 %. This is quite close to the rate of all word-level errors in manual test (4.7 %, see table 3).

Finally, to relate Fintwol results of the test to the whole base corpus, we made still another classification for the words not recognised by Fintwol. This classification can be applied automatically to large material. The words are divided to those consisting of character entities, those including character entities, and those consisting of alphanumeric characters (and possible final punctuation mark). Fintwol is not XML-compliant, so it cannot tackle character entities. This means that all the strings containing entities end up as unrecognised. This classification is not exhaustive, but it covers most of the unrecognised words. The frequencies of these types among words unrecognised by Fintwol in the test material and in the base corpus are expressed in tables 5 and 6.

The all-automatic Fintwol-error classification test supports the reliability of the original Fintwol test. In all the proportional figures in tables 5 and 6, the difference between the test material and the base corpus is less than 1.5 %.

To better understand the errors made by Fintwol, we can have a look at two lists. The lists show 15 most common words that were not recognised by the analyser. Table 7 shows the words extracted from the test material, and table 8 the same list extracted from the core corpus. Here we see, that the most common unrecognised words are names, abbreviations, foreign words and old-fashioned or colloquial/poetic words. The names dominate the list extracted from the test material, while the old-fashioned and foreign words dominate the list extracted from the whole core corpus. This is quite natural, as the latter are less dependent on the subject of the text.

Using the error rate of the manual evaluation or corrected error rate of the analyser-based evaluation, we can estimate that from 4.5 to 4.7 % of the words in the corpus are misrecognised. This would lead us to total amount of 367 00 to 383 000 misrecognised words in the whole corpus. This would include about 80 000 words (1 % of word mass) of phonetic transcription from the Suomi-magasine.

8. discussion

I have described a way of testing the XML structure and the OCR quality of a corpus with one steady sample set (per corpus), using different methods of exploring the set both manually and semi-automatically. This seems to be an economical way of evaluating a corpus, and it tells us a lot about its overall quality and typical flaws. However, some questions arise about the sampling and testing procedure, and also about the corpus composing process itself. Finally, I present a list of questions and comments considering these subjects.

Error type	**Words**	**% of errors (2505)**	**% of words (24 201)**
OCR misrecognition or unrecognisable word	963	38.5 %	4.0 %
Unrecognised words (names incl.)	1090	43.5 %	4.5 %
A non-word character string	452	18.0 %	1.9 %
Total reported	**2505**	**100** %	**10.4** %

Table 4: Words not recognised by Fintwol, manually classified

Classification	**Amount**	**% of Fintwol errors (2 505)**	**% of words (24 201)**
Include character entities	404	16.1 %	1.7 %
Consist of character entities	333	13.3 %	1.4 %
Consist of word characters	1591	63.5 %	6.6 %

Table 5: Automatic classification of non-recognised words in the test material

Word	**Amount**	**Type**
Brenner	7	Name
VValdheim	6	OCR error (name)
Vanishing	5	Foreign
Stählbergin	5	OCR error (name)
Pond	5	Name
mä	5	Colloquial/poetic
Freisler	5	Name
aˆömä	5	OCR error
Stählberg	4	OCR error (name)
Ondreiko	4	Name
KirjanpitoL:n	4	Abbreviation
Keder	4	Name
it	4	Foreign
ECONOMIC	4	Foreign
Bastian	4	Name

Table 7: Test material: top 15 words unrecognised by Fintwol

Word	Amount	Type
niinkuin	1563	Old-fashioned
och	1385	Foreign
Valok	1299	Abbreviation
den	866	Foreign
ennenkuin	865	Old-fashioned
N:ot	786	Abbreviation
Mr	688	(Foreign) abbreviation
und	640	Foreign
mä	606	Colloquial/poetic
ko	596	Fragment
att	588	Foreign
om	584	Foreign
ikäänkuin	572	Old-fashioned
rahap	564	Abbreviation
ta	527	Fragment

Table 8: Core corpus: top 15 words unrecognised by Fintwol

* The different tests seem to support the view that 70 samples of 500 meaningful items may give somewhat reliable picture of the overall error rate of OCR quality of a corpus of this kind and size. Note, that 10 % of different material with special error sources may add considerable uncertainty. This was the case with the phonetic transcriptions in the Suomi magasine. Also, the vocabulary and the quality of the originals vary according to the publishing time. To handle this kind of variation, the material from different sources and times should probably be tested separately. Ensuring the adequacy of the sampling for testing the markup would probably require some more work, especially for such varying material.

* When evaluating the testing scheme, the representativeness of the test material for the XML structure does not proof to be sufficient for evaluating all the qualities we tried to test. It seems that more accurate results could have been achieved, if we had chosen some basic qualities of the markup, calculated a statistically significant sample set for each quality, and tested accordingly. However, it is not guaranteed that the results would have described the qualities and typical errors of the markup better than the current approach.

* In evaluating the errors, their classification to different levels (structural levels in XML evaluation and word and character levels in OCR evaluation) seems to be necessary for different reasons. In OCR evaluation, it is necessary to to distinguish punctuation-level failures from word-level failures, and in XML evaluation superfluous line breaks are of different importance than mistaken links. Suitable classifications can also help relating the tests to each other, as in the case of the manual and analyser-based OCR tests above.

* Using a morphological analyser for evaluating the text contents of a corpus is possible, but our experience shows that in real-life text, about half of the words that the analyser does not recognise may be false alarms. Also, the analyser may leave considerable part of the misrecognised words undetected. This means that the analyser behaviour should be carefully examined, and

Classification	Amount	% of Fintwol errors (816 569)	% of words (8 148 189)
Include character entities	143 115	17.5 %	1.8 %
Concist of character entities	114 240	14.0 %	1.4 %
Concist of word characters	513 921	62.9 %	6.3 %

Table 6: Automatic classification of non-recognised words in the base corpus

corrected with two estimates: the error slip-through rate and the proportion of false alarms. A preliminary manual evaluation seems to be necessary to get this done.

* In practise, the evaluation proved to be useful, as it showed how to improve the quality of the corpus. Based on the testing, we decided to reduce the column breaks to line breaks in both parts of the corpus, cancel the displacements of (most of) the discontinuous elements in the base corpus, and manually correct the OCR results of the core corpus.

During the evaluation, we constantly had a feeling, that statistical methods could have been used more effeciently to solve some of our problems, especially considering sampling. We would have liked to experiment on calculating statistically sufficient sets for evaluating the text content of different subsets of the the corpus (eg. different time periods and different periodicals) and relating them to the whole. Our resources and abilitites did not suffice for this kind of work, though. A simple and focused guide to basic statistics for corpus evaluation would have been useful throughout the process.
The evaluation results may cast a doubt on the method of collecting and automatically structuring a diachronic corpus based on scanning and OCR. The facts, that the quality of the originals and also the text structure conventions vary depending on the original publishing time make this quite a challenge. This experience showed us once more, that there is no cheap lunch without a following bill. However, we have managed to collect a diachronic 8 million word corpus with bearable costs and within a relatively short time – and while the corpus certainly has its deficiences, it is a useful tool already, and a good starting point for future improvements.

9. References

Vesa Heikkinen, Tuure Hurme, Mikko Lounela, and Mikko T. Virtanen. 2005. Teksti, aihe ja laji: diakronisen korpuksen koostaminen ja käyttäminen. Forthcoming in the papers from the 32nd Finnish Conference of Linguistics; In Finnish.

HUL. 2000. Vesa - verkkosanasto/webbtesaurus. Online: http://vesa.lib.helsinki.fi/. In Finnsh / Swedish; Referred 21.3.2006.

Stefan Kokkelink and Roland Schwänzl. 2002. Expressing qualified dublin core in rdf / xml. Online: http://dublincore.org/documents/dcq-rdf-xml/. Referred 21.3.2006.

Kimmo Koskenniemi. 1983. *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production*. University of Helsinki, Department of General Linguistics.

Outi Lehtinen and Mikko Lounela. 2004. A model for composing and (re-)using text materials for linguistic research. In M. Nenonen, editor, *Papers from the 30th Finnish Conference of Linguistics*, Studies in Language, University of Joensuu, pages 73–78. University of Joensuu, Joensuu.

LINGSOFT. 2006. Fintwol: Finnish morphological analyser. Online: http://www.lingsoft.fi/doc/fintwol/. Referred 21.3.2006.

Mikko Lounela. 2002. Aiming towards best practices in xml techniques for text corpora annotation: City of helsinki public works department - a case study. In Eero Hyvönen and Mika Klemettinen, editors, *Towards the Semantic Web and Web Services. Proceedings of the XML Finland 2002 Conference*, HIIT Publications, pages 123–135. Institute for Information Technology, Helsinki.

OUCS. 2003. D1: Validation manual for written language resources. Online: http://www.oucs.ox.ac.uk/rts/elra/D1.xml. Referred 21.3.2006.

TEI. 2006. Tei: Yesterday's information tomorrow. Online: http://www.tei-c.org/. Referred 21.3.2006.

Measuring Monolinguality

Uwe Quasthoff, Chris Biemann

NLP Department,
Faculty of Mathematics and Computer Science
University of Leipzig, Germany
{quasthoff,biem}@informatik.uni-leipzig.de

Abstract

We present an approach to measuring the amount of material in a natural language text corpus that consists of text in languages other than the main language. Having a presumably monolingual corpus at hand, we ask for the amount of multilingual noise by comparing the frequency of high-frequent words in monolingual corpora of different languages to their frequency in the corpus in question. The ratio of the expected and the measured frequencies per language quantifies the amount of noise per language. The measure is very effective since it requires only the comparison of a few thousand frequency counts.

We evaluate the method by artificial mixtures of two language corpora for different noise levels and demonstrate the effect of a corpus cleaning method by measuring monolinguality before and after cleaning.

1. Introduction

When building a large corpus for a given language, one has to assure that the data are as clean as possible. This is especially important when using resources from the Web, where neither top-level-domain nor source guarantees monolinguality in any respect. See (Kilgarriff, 2001) for a discussion about the web as a corpus; multilinguality on the same web servers is even employed by (Resnik and Smith 1995) to construct aligned bilingual corpora. The definition for cleanness may vary according to the principles chosen for the corpus construction. For this paper, we want to assume the following:

- The corpus is sentence separated and the order of the sentences is not important.
- Clean means monolingual, i.e. during pre-processing, sentences belonging not to the specified language should be identified and removed.

The question whether a sentence belongs to a given language is not at all trivial because a German sentence can, for instance, contain an English movie title or a Latin medical term. The principles for corpus construction might contain hints how many foreign language objects should be tolerated. Usually one will allow such isolated foreign language items, but not foreign language sentences (which might contain some words in the corpus language). At this tolerance level (i.e. we allow only a few foreign language objects), the cleaning becomes more difficult and one has to check the quality of the cleaning process.

The aim of the paper is not to describe algorithms for cleaning procedures but to measure their result. A numerical value of monolinguality can be considered as a quality measure for corpora. This measure should also give satisfactory results if the languages considered have some words in common.

Note that the usual tests for language detection (e.g. described in (Dunning, 1994)) are not sufficient for cleaning because they allow a considerable amount of multilingual material to pass, dependent on document length. The test described here should be able to quantify this amount. As shown in the examples, foreign language material of 0.001% can be measured. The lower bound of the verifiability depends only on the corpus size.

The availability of clean monolingual resources is important for a variety of applications. To name a few, methods that construct language models from corpora (e.g. Brown et al. 1992) will be disturbed by alien language material and morphology induction (like described in (Goldsmith, 2001) inter al.) will face undesired problems. In dimensionality reduction steps (e.g. Derweester et al. 1990), some of the dimensions will be occupied by other languages than the target language, hampering performance.

2. A Measure for Monolinguality

2.1. Informal description

We propose a measure, which distinguishes between random foreign noise, and foreign language objects of a certain special kind like proper names, quotations etc. While the latter might be allowed in a corpus of language A, we will measure mainly typical text of another language B contained in the corpus. Such typical text will contain nearly all high frequency words of language B.

If the absolute amount of word of language B is large enough, their distribution will be like in an ordinary language B corpus for many of these words. Hence, many of those words will be a similar ratio of there usual relative frequency compared to their relative frequency in the language A corpus.

Of course, this is not true for all words of language B under consideration. Exceptions are words often used in typical foreign language objects like named entities or titles. And, of course in the case of words being used in both languages A and B. However, their number turns out to be surprisingly low.

Hence, we get a clear peak when counting the number of words for different frequency ratios. Moreover, the resulting peak does not depend on the number of high frequent words used. In the examples, we use always the 1000 most frequent words.

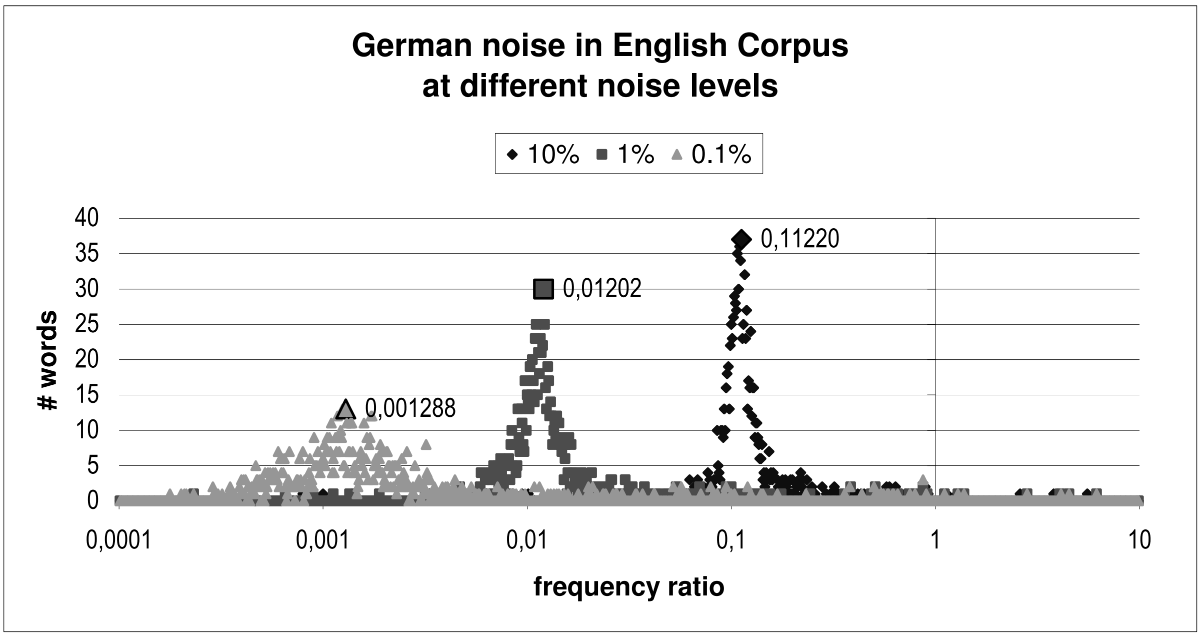


Figure 1: German noise in English corpus. The numbers attached to the peaks are the results of our measure (experiment 1a)

2.2. Comparing high frequency words

Assume a corpus of a language A contains **x**% of noise of some language B. Moreover, the corpus should be large (say, more than 1.000.000 sentences) and the noise should be typical text of language B. Then we consider the top-1000 high frequent words of language B. If such a high frequent word w is not contained in language A, it should appear in the corpus with a relative frequency of roughly **x**% of its relative frequency in language B. If w is also a valid word in language A, its relative frequency will be much higher. We define the frequency ratio of w as the relative frequency of a word w in A divided by its relative frequency in the corpus B.

There are four groups of words in the top 1000 words of language B:

- Words that do not occur in language A. Their frequency ratio will be around x%.
- Words that are also amongst the highest frequency words of language A and moreover have the same function. Their frequency ratio will be around 1.
- Words that occur in language A, but at different frequency bands. They are a random sample of words of L and distributed in a Zipf way, cf. (Zipf, 1949).
- Words of B that are often used in named entities and titles (such as capitalized stop words). They appear in the corpus of language A more frequently then the expected x% of noise.

The second group of words is only present in languages that are very similar to each other. Table 1 shows overlaps in the top 1000 words of some European languages.

	da	de	ee	en	es	fr	is	it	nl	no
de	36									
ee	11	5								
en	41	26	11							
es	18	14	7	27						
fr	33	19	10	59	52					
is	43	13	7	9	6	11				
it	31	11	9	25	98	51	9			
nl	69	56	10	52	25	40	21	30		
no	489	33	18	38	25	35	55	40	64	
se	221	23	15	27	23	32	50	32	54	257

Table 1: overlap in some European languages with regard to the most frequent 1000 words: Danish (da), German (de), Estonian (ee), English (en), Spanish (es), French (fr), Icelandic (is), Italian (it), Dutch (nl), Norwegian bokmål (no), Swedish (se)

2.3. The dominant frequency ratios

In the figures 1 and 2, we have the frequency ratios at the x-axis ranging from 10^{-4} to 10 on a logarithmic scale. After discretizing the frequency ratios it is counted, how many words fall into the corresponding intervals. We find a Gaussian shaped curve with a clear maximum at the amount of noise at x% caused by words of group 1, a similar peak near 1 due to the second group (if the languages are similar) and some uniformly distributed noise introduced by the words of group three. Words of group four are scattered between x% and 1.

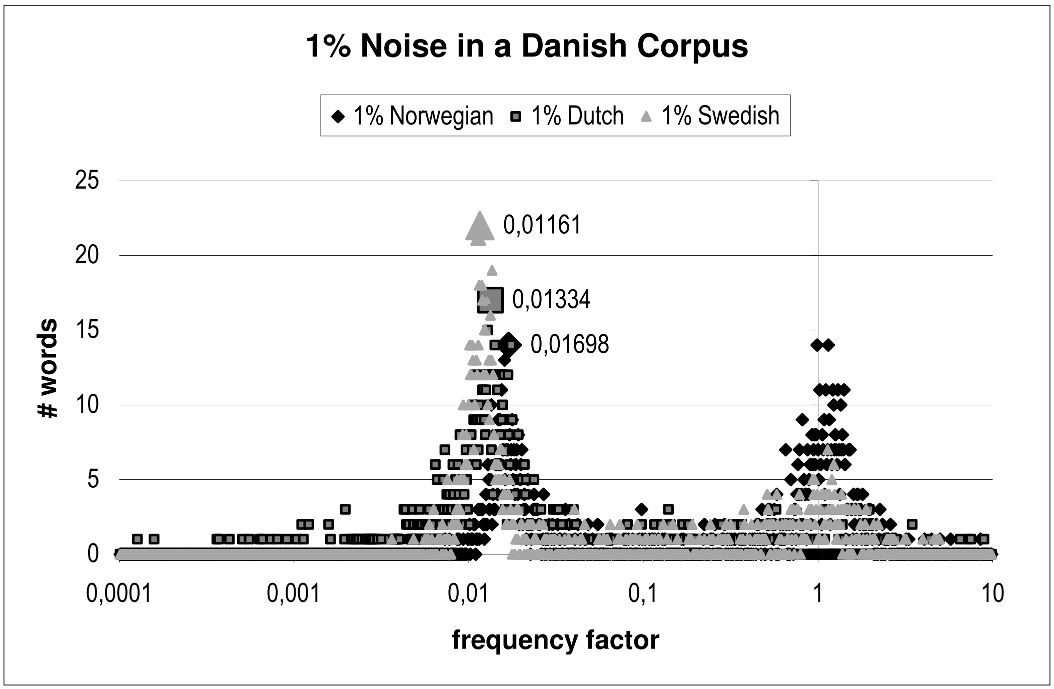


Figure 2: Norwegian, Dutch and Swedish noise in Danish corpus (experiment 1b)

3. Experiments

In the following two examples we search for foreign language noise in English and German corpora. In the first experiment, this noise is manually inserted into the British National Corpus (BNC, http://www.natcorp.ox.ac.uk/, (Leech, 1992)). As a result we should measure the exact amount of the previously inserted noise. Moreover, the effect of very similar languages is discussed using Scandinavian languages.

The second experiment uses randomly collected text from the web using only .de-domains. For the reduction to a corpus in German language, foreign language sentences are automatically removed. The amount of foreign language text is shown before and after cleaning.

3.1. Experiment 1: Artificial Noise

In order to test our measure we performed two experiments with introducing noise in monolingual corpora. In experiment 1a we aimed at finding out how well the measure captures different noise levels, in experiment 1b we tested very similar languages.

Figure 1 shows the frequency ratio interval counts for a 10%, 1% and 0.1% German noise as taken from http://www.wortschatz.uni-leipzig.de (Biemann et al., 2004) injected in a chunk of the BNC corpus. All mixtures consisted of about 20 Million tokens. The noise levels measured are slightly larger than expected, see figure 1. This is due to the fact that the BNC corpus contains German sentences (some containing errors) like e.g.

- Geschichte in Literatur und Film seit den sechziger Jahre , in : Geschichte als Literatur , ed .
- Cantatas No. 140 , Wachet auf , ruft uns die Stimme ; No. 147 , Herz und Mund and Tat und Leben .
- Prince : Hans Adam von und zu Liechtenstein II .
- Nur an den beiden Poien menschlicher Verbindung , dort , wo es noch keine oder keine Worte mehr gibt , im Blick und in der Umamung , ist eigentlich Glück zu finden , denn nur dort ist Unbedingtheit , Freiheit , Geheimnis und tiefe Rücksichglosigkeit .

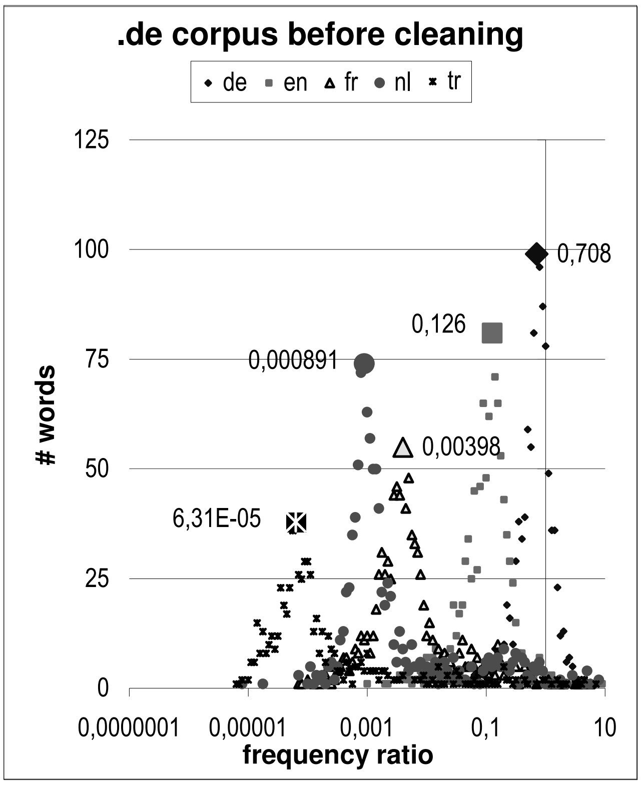
Figure 2 depicts the distribution for very similar languages (in terms of table 1). Again, the measure deviates not severely from the goal of 1% noise. Material was taken from http://corpora.informatik.uni-leipzig.de to build corpora of about 17 Million words.

3.2. Experiment 2: Web Text

Experiment 2 uses a corpus of about 40 million sentences randomly collected from .de-domains. We measured the amount of foreign languages before and after cleaning, which was carried out as outlined in (Quasthoff et al. 2006). Table 2 contains not only the main frequency ratios, but also the number of top-1000-words of a foreign language found in the corpus. As stated in 2.1,

	Before cleaning		After cleaning	
	Number of top-1000-words found	Approx. Frequency ratio	Number of top-1000-words found	Frequency ratio
German	1000	0.708	1000	0.946
English	995	0.126	987	0.0010
French	924	0.0398	906	0.00002
Dutch	995	0.000891	775	0.000006
Turkish	642	0.0000631	562	0.000006

Table 2: frequency ratios and number of top 1000 words when cleaning a German web corpus



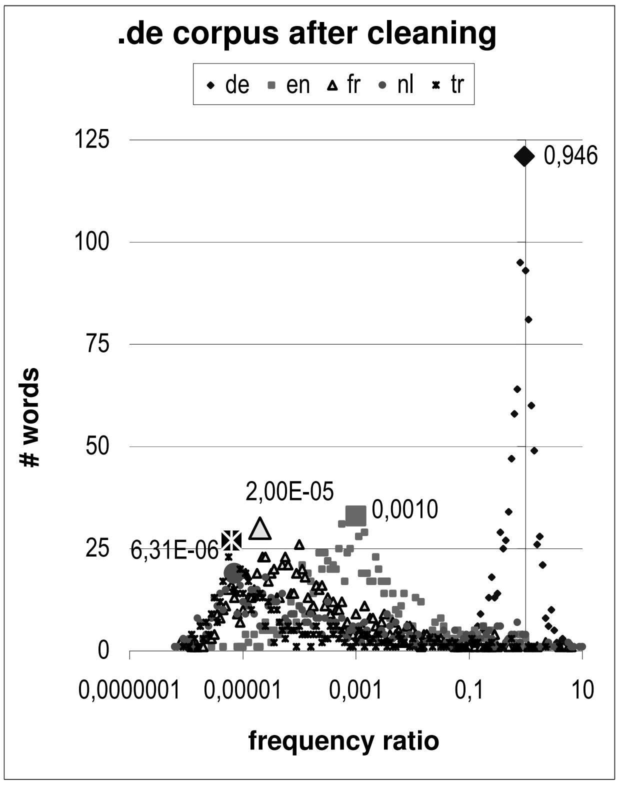


Figure 3: The effects of corpus cleaning with regard to the monolinguality measure

(nearly) all of the top-1000-words of the noise language are expected to appear in the corpus. The smaller numbers for Turkish and Danish (only in the cleaned version) indicate that the limits of the method are reached in the case where (size of top-wordlist) / (frequency ratio) has the same order of magnitude as the frequency of the most frequent word in the corpus. Figure 3 visualizes the findings of table two.

Moreover, the table shows that language cleaning can reduce the noise by a factor of at least 100. The corresponding noise can be measured down to a frequency ratio of approximately 10^{-5}.

4. Conclusion

We presented a measure for estimating the amount of multilingual noise in monolingual corpora. It can be calculated efficiently as it involves only 1000 frequency counts per noise language tested. Experiments show that the measure correlates well with artificial mixtures of monolingual corpora. For large corpora, noise will be detected down to a ratio of 10^{-5}.

A possible application in a World Wide Web context is to measure the amount of web sites that belong to a defined set of languages. This is done by querying the index of a search engine for the top 1000 words per language for frequency to produce statistics as e.g. in (Langer 2001).

5. References

Biemann, Chr., Bordag, S., Heyer, G., Quasthoff, U. and Wolff, Chr. (2004): *Language-independent Methods for Compiling Monolingual Lexical Data*. Proceedings of CicLING 2004, Seoul, Korea and Springer LNCS 2945, pp. 215-228, Springer Verlag Berlin Heidelberg

Brown, P.F., Della Pietra, V. J., deSouza, P., Lai, J.C. and Mercer, R. L. (1992): *Class-Based n-gram Models of Natural Language*. Computational Linguistics 18(4):467-479

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R. (1990): *Indexing by latent semantic analysis*. Journal of the Society for Information Science, 41(6):391-407

Dunning, T. (1994): *Statistical Identification of Language*. Technical report CRL MCCS-94-273, Computing Research Lab, New Mexico State University

Goldsmith, J. (2001): *Unsupervised learning of the morphology of a natural language*. Computational Linguistics, 27:153-198

Kilgarriff, A. (2001): *Web as corpus*. In Proceedings of Corpus Linguistics 2001, Lancaster, England.

Langer, S. (2001): *Natural languages on the Word Wide Web*. In: Bulag. Revue annuelle. Presses Universitaires Franc-Comtoises, S. 89-100

Leech, G. (1992): *100 million words of English: the British National Corpus*. Language Research 28:1, 1-13.

Quasthoff, U., Biemann, C. and Richter, M. (2006): *Corpus Portal for Search in 16 Monolingual Corpora*. Proceedings of LREC-2006, Genoa, Italy

Resnik, P. and Smith, N.A. (2003): *The Web as a Parallel Corpus*. Computational Linguistics 29(3):349-380

Zipf, G. K. (1949). *Human behaviour and the principle of least effort*. Addison-Wesley, Reading, MA.

JTaCo & SProUTomat: Automatic Evaluation and Testing of Multilingual Language Technology Resources and Components

Christian Bering[†], **Ulrich Schäfer**[*]

[†]Computational Linguistics Department, Saarland University
P.O.Box 151150, D-66041 Saarbrücken, Germany
christian.bering@acrolinx.com
[*]Language Technology Lab, German Research Center for Artificial Intelligence (DFKI) GmbH
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
ulrich.schaefer@dfki.de

Abstract

We describe JTaCo, a tool for automatic evaluation of language technology components against annotated corpora, and SProUTomat, a tool for building, testing and evaluating a complex general-purpose multilingual natural language text processor including its linguistic resources (lingware). The JTaCo tool can be used to define mappings between the markup of an annotated corpus and the markup produced by the natural language processor to be evaluated. JTaCo also generates detailed statistics and reports that help the user to inspect errors in the NLP output. SProUTomat embeds a batch version of JTaCo and runs it after compiling the complex NLP system and its multilingual resources. The resources are developed, maintained and extended in a distributed manner by multiple authors and projects, i.e., the source code stored in a version control system is modified frequently. The aim of JTaCo & SProUTomat is to warrant a high level of quality and overall stability of the system and its lingware.

1. Introduction

The development of multilingual resources for language technology components is a tedious and error-prone task. Resources (lingware) like morphologies, lexica, grammars, gazetteers, etc. for multiple languages can only be developed in a distributed manner, i.e., many people work on different resources.

However, the resulting systems are supposed to deliver the same good recognition quality for each language. Dependencies of resources and subsystems may lead to suboptimal performance, e.g., reduced recognition rates, of the overall systems in case of errors creeping in during the development process. Hence, in analogy to software engineering, testing and evaluation of the developed lingware has to be carried out on a regular basis, both for quality assurance and comparability of results in different languages. Annotated natural language corpora can be thought of as providing a rich and potentially very useful body of test material in this context. However, it is often not possible to flexibly incorporate the material at hand into the development process. The reasons are manifold: Not only may annotations in different sources be of very diverse nature, but the NLP component under development usually generates a markup in yet another format defined by the development environment.

In this paper, we describe a framework consisting of two major components, JTaCo and SProUTomat, that facilitates frequent (e.g., daily) building, testing and evaluation of multilingual language components and resources in a quality assurance and development cycle as depicted in Figure 1. We have implemented and will demonstrate the framework for the multilingual SProUT processor. However, the concepts and mechanisms described could be applied to any other resource-intensive natural language processing system.

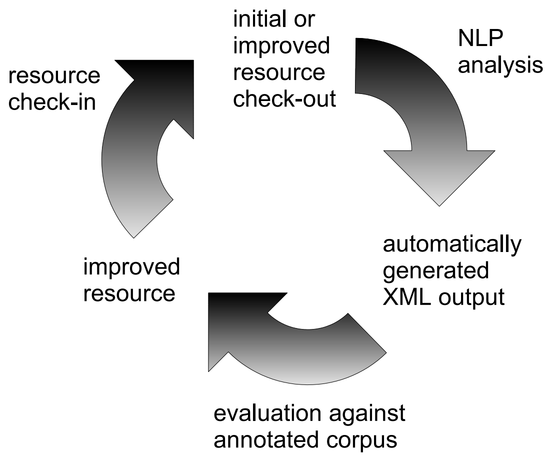

initial or improved resource check-out
NLP analysis
automatically generated XML output
evaluation against annotated corpus
improved resource
resource check-in

Figure 1: Quality assurance and development cycle for multilingual linguistic resources.

2. SProUT

SProUT is a shallow, multilingual, general-purpose natural language processor (Drozdzynski et al., 2004). SProUT comes with a powerful, declarative grammar formalism XTDL that combines finite-state techniques and typed feature structures with structure sharing and a fully-fledged, efficiently encoded type hierarchy—in contrast to systems like GATE (Cunningham et al., 2002) that support only simple attribute-value pairs.

SProUT rules consist of regular expressions over typed feature structures[1]. A rule is matched against a sequence of input feature structures which are filled by basic components like tokenisers, morphology or gazetteer lookup run-

[1]The acronym SProUT stands for Shallow Processing with Unification and Typed feature structures. SProUT's homepage is `http://sprout.dfki.de`.

ning on input text or, in more complex cases, XML output from external NLP components or even output from previous SProUT grammar stages.
The matching condition is unifiability of the input sequence with the expanded regular expression of the left hand side of a rule. In case of a match, feature structure unification is used to transport information from the matching left hand side to the output feature structure on the right hand side of the rule. The output feature structure can then, e.g., be transformed to any XML format.
The SProUT system provides basic components like tokenisers, morphologies and domain-specific gazetteers for languages such as English, German, French, Spanish, Greek, Japanese, Italian, Chinese, Polish and Czech, and comes with a user-friendly integrated development environment (IDE). The current main applications of SProUT are information extraction and named entity recognition (NER).
To illustrate the SProUT formalism, we give a short example in Figure 2 of a grammar rule that recognises river names. The rule matches either expressions consisting of an (unknown) capitalised word (via token type match), followed by a noun with stem *river* or *brook* (via the English morphology component; disjunction has a higher precedence than concatenation), or Gazetteer entries of type *gaz_river* containing English river names represented by the Gazetteer type *gaz_river*. The generated output structure of type *ne-location* contains a location type *river* and the location name transported via the coreference symbol $\boxed{\text{loc\_name}}$. To sum up, this rule recognises both unknown river names (via a pattern involving morphology lookup that tolerates morphologic variants) and known river names (via a gazetteer match), using a concise, declarative pattern and returning a structured description.
SProUT has been and is currently used in many research and industrial projects for opinion and text mining, information extraction, automatic hyperlinking, question answering and semantic web applications (Drozdzynski et al., 2004).

3. JTaCo

The aim of JTaCo (Bering, 2004; Bering et al., 2003) is to allow the developer of an NLP component or resource, e.g., of a grammar, to make unified use of variably annotated source material for testing. The component developer provides suitably, i.e., usually semi-manually or manually marked-up reference sources on the one hand, and a parser or similar NLP component on the other hand. JTaCo extracts the original annotation from the corpus, compares this annotation with the markup the component in question generates for the same input, and generates statistics and reports from the comparison results[2].
Since a focus of JTaCo lies on the integration of diverse manual annotation schemes one the one hand and differing NLP components on the other, JTaCo employs a very modular architecture in which its different processing stages allow independent adaptations to varying input and different environments. JTaCo is realised as a pluggable lightweight, mostly architecture-independent framework. Currently, there are two JTaCo plug-in realisations for usage with grammars developed in SProUT: A GUI plug-in integrated into the SProUT IDE, and a batch version integrated into SProUTomat.

[2]JTaCo stands for Java Tagging Comparator.

3.1. JTaCo's Processing Stages

JTaCo works in four separate transformational processing stages. Figure 3 gives an overview of these stages, of their input and the results they generate. The process starts from an annotated written corpus against which the NLP component or resource is to be tested. In the first step, JTaCo uses an *AnnotationParser* to separate the corpus into

- the 'raw' text contained in the corpus (i.e., the text without any annotation) and
- its *true* annotation (interchangeably also called the *reference* or *manual* annotation).

The extracted text is fed into the *Parser* (or a similar component) which the developer wants to test, yielding the annotation to compare with the manual annotation. The comparison is executed by a *TaggingComparator*. The comparator's result in turn is used by an *OutputGenerator* to select, format and output the needed information.

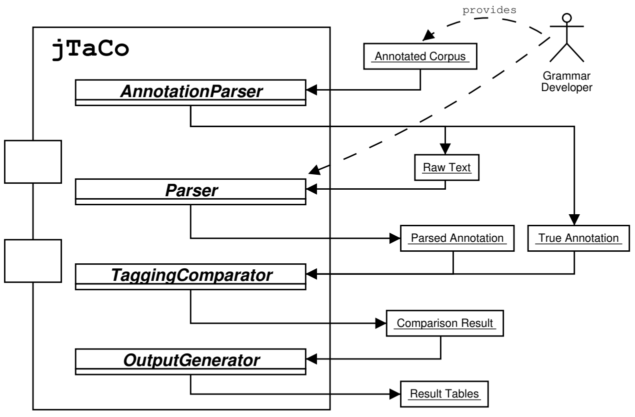


Figure 3: An overview of JTaCo's processing stages and the (intermediate) results they yield.

There are two main advantages gained from such a modular architecture: On the one hand, the abstract representations in the intermediate results hide details specific to the corpus or component used. For instance, differing types of annotations are mapped to an abstract annotation representation, for which a comparison operation – i.e., especially the notion of equality between entities in the two annotations – can be defined in an adequately flexible manner, and the underlying annotated sources as well as NLP components can be exchanged transparently. Thus, whenever a new annotation format or component makes it necessary to integrate a tailored module into JTaCo, the capabilities of the new module can readily interact with existing functionality of other modules.
The second, more practically relevant advantage is that the settings of any one stage can be changed, and the process at that stage rerun with the new settings without having to

$$\text{river} :> \left(\left[\begin{array}{ll} \textit{token} & \\ \text{TYPE} & \textit{first\_capital\_word} \\ \text{SURFACE} & \boxed{\text{loc\_name}} \end{array}\right] \bullet \left(\left[\begin{array}{ll} \textit{morph} & \\ \text{STEM} & \texttt{"river"} \\ \text{POS} & \textit{noun} \\ \text{SURFACE} & \boxed{\text{key}} \end{array}\right] \middle| \left[\begin{array}{ll} \textit{morph} & \\ \text{STEM} & \texttt{"brook"} \\ \text{POS} & \textit{noun} \\ \text{SURFACE} & \boxed{\text{key}} \end{array}\right] \right) \right)$$

$$\left| \left[\begin{array}{ll} \textit{gazetteer} & \\ \text{GTYPE} & \textit{gaz\_river} \\ \text{CONCEPT} & \boxed{\text{loc\_name}} \\ \text{DESIGNATOR} & \boxed{\text{key}} \end{array}\right] \right. \rightarrow \left[\begin{array}{ll} \textit{ne-location} & \\ \text{LOCTYPE} & \textit{river} \\ \text{LOCNAME} & \boxed{\text{loc\_name}} \\ \text{DESCRIPTOR} & \boxed{\text{key}} \end{array}\right].$$

Figure 2: A SProUT grammar rule recognizing river names. Boxed feature values denote structure sharing, type names are typeset in italics. The dot after the first token indicates concatenation, the vertical bars separate alternatives.

re-iterate the previous process stages, as long as their results are still available. This can be especially useful for the last two stages in an interactive environment (i.e., comparison and report generation), where the developer might want to experiment with different settings without repeatedly having to rerun the probably time-consuming processes of reading the corpus and parsing it.
For each of the stages, a JTaCo plug-in uses one or more processing realisations adapted to the desired representations. In what follows, we will draw upon the implementations integrated into SProUT and SProUTomat to illustrate the information flow in JTaCo.

3.2. Reading the Annotated Corpus

For use in the following processing stages, JTaCo extracts from the annotated corpus the 'raw' content, i.e., the written text without any markup, on the one hand, and the reference annotation on the other. Both the extraction of the text and of the annotation can be configured according to the specific annotation scheme. E.g., a corpus usually not only contains the annotated textual material, but also meta-information intended for, e.g., administrative purposes. Such information has to be exluded from the text extracted to be used for testing. Currently, JTaCo includes support for annotations which satisfy certain regular constraints and for XML annotations such as found in MUC corpora (Grishman and Sundheim, 1996). For use with SProUT, JTaCo transforms the XML-encoded entities into typed feature structures.
As an illustration, consider the following MUC time expression:

```
<TIMEX TYPE="DATE">07-21-96</TIMEX>
```

The textual content consists just of the date expression *07-21-96*. JTaCo transforms the tag information as well as the surface and character offsets into feature-value pairs in a feature structure:

$$\left[\begin{array}{ll} \textit{timex} & \\ \text{TYPE} & \texttt{"DATE"} \\ \text{CSTART} & \texttt{"27"} \\ \text{CEND} & \texttt{"34"} \\ \text{SURFACE} & \texttt{"07-21-96"} \end{array}\right]$$

Here, CSTART and CEND indicate the inclusive start and end character positions of the annotated element in the 'raw' text, i.e., without counting the markup. The resulting reference annotation is the collection of all feature structures generated from the corpus. More complex, embedded annotations would be translated in a similar manner.

3.3. Parsing the Extracted Text

In this second processing stage, JTaCo feeds the NLP component which the developer wants to test with the text retrieved from the previous stage, and the NLP component in turn produces some specific markup of the text. As in the previous stage, JTaCo transforms this annotation into a format which it can compare with the reference annotation. For the previously employed example expression, *07-21-96*, SProUT's named entity recognition markup delivers structured output in an XML-encoded typed feature structure[3], where CSTART and CEND indicate start and end character positions of the matched named entity in the input text:

$$\left[\begin{array}{ll} \textit{point} & \\ \text{SPEC} & \textit{temp-point} \\ \text{MUC-TYPE} & \textit{date} \\ \text{CSTART} & \texttt{"27"} \\ \text{CEND} & \texttt{"34"} \\ \text{SURFACE} & \texttt{"07-21-96"} \\ \text{YEAR} & \texttt{"1996"} \\ \text{MONTH} & \texttt{"07"} \\ \text{DOFM} & \texttt{"21"} \end{array}\right]$$

3.4. Comparing the Annotations

In this stage, the annotations obtained from the two previous tranformation processes are compared, i.e., the 'manual' annotation read directly from the corpus, and the 'parsed' annotation obtained through the NLP component. For JTaCo, an annotation is a collection of tags, where a tag consists of some linguistic information about a piece of text. Minimally, a tag contains

- some name, e.g., a linguistic label,

[3]Transformation of typed feature structures and general XML markup is discussed in the context of the upcoming ISO standard in (Lee et al., 2004). Actually, SProUT's default XML output format is very close to the proposed ISO format for typed feature structures.

- the surface string to which the label applies,
- token count information about where this string is found in the corpus.

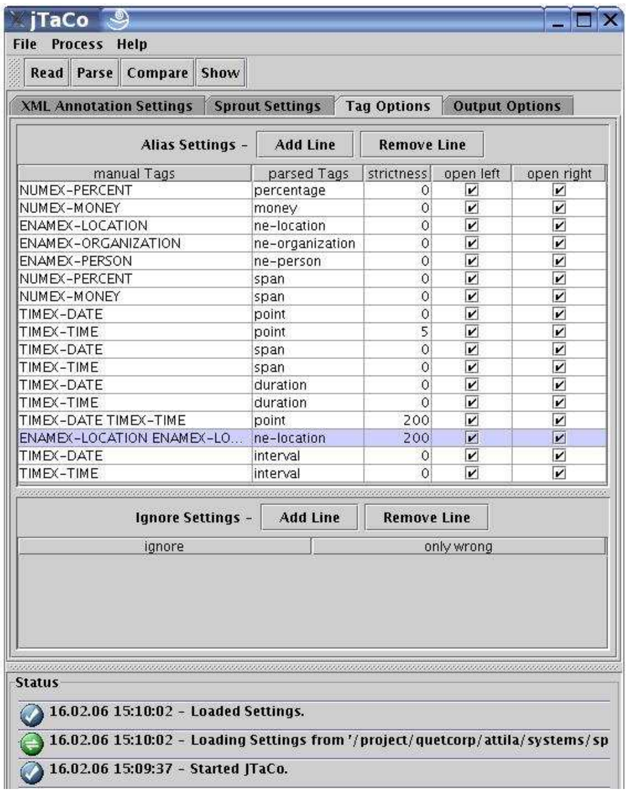
Usually, the setup uses tags which incorporate more information, and the relation used to determine entity equality between the two annotations typically depends on this information. For instance, for use with, SProUT JTaCo generates an annotation consisting of tags which are augmented with feature structure information. The equality notion of these tags is defined though unification.
An important feature of JTaCo is that the comparison can be configured to accomodate for a variety of systematic differences in annotations:

- The annotations may use different labels, differing perhaps even in granularity. E.g., one annotation might globally use the label *organisation*, while the other uses subclasses such as *university*, *government*, etc.
- The annotated entities may differ in their surface spans. E.g., one annotation might consider the expression *President Hugo Chavez* to be a named entity, while the other might exlude the title.
- One annotation may contain sequences of entities which in the other annotation correspond to one single entity. For instance, MUC will usually separate a date followed by a time into two named entities (`TIMEX-DATE` and `TIMEX-TIME`), while SProUT considers this to be one entity.

The screenshot in Figure 4 shows a part of the defined tag mappings used when comparing SProUT's annotation to the original MUC markup. Most of the mappings constitute simple entity label correspondences, e.g., a MUC `TIMEX-DATE` can correspond to a `point`, a `span`, a `duration`, or an `interval` in the annotation generated by SProUT. An entity named a `duration` by SProUT can in turn be a MUC `TIMEX-DATE` as well as a `TIMEX-TIME`. In the example settings, all of these correspondences are further 'softened' to ignore surface span discrepancies: The *open left* and *open right* switches allow for a mismatch in the `CSTART` and `CEND` features, respectively. The example settings also contain a mapping of the sequence `TIMEX-DATE` and `TIMEX-TIME` in MUC to the SProUT entity `point`. The *strictness* is a measure of how far apart these two elements are allowed to occur and still be valid elements for a sequence matched against a single `point`.

3.5. Generating a Report

Finally, JTaCo generates a report of the comparison. JTaCo can output statistical information (precision, recall, etc.) as well as detailed occurrence lists of entities that were or were not correctly identified in the parse. The settings for this processing stage determine which results are shown (e.g., for which tags) and how the information is formatted. JTaCo can export the generated reports as ASCII and as HTML tables.



manual Tags	parsed Tags	strictness	open left	open right
NUMEX-PERCENT	percentage	0	✓	✓
NUMEX-MONEY	money	0	✓	✓
ENAMEX-LOCATION	ne-location	0	✓	✓
ENAMEX-ORGANIZATION	ne-organization	0	✓	✓
ENAMEX-PERSON	ne-person	0	✓	✓
NUMEX-PERCENT	span	0	✓	✓
NUMEX-MONEY	span	0	✓	✓
TIMEX-DATE	point	0	✓	✓
TIMEX-TIME	point	5	✓	✓
TIMEX-DATE	span	0	✓	✓
TIMEX-TIME	span	0	✓	✓
TIMEX-DATE	duration	0	✓	✓
TIMEX-TIME	duration	0	✓	✓
TIMEX-DATE TIMEX-TIME	point	200	✓	✓
ENAMEX-LOCATION ENAMEX-LO...	ne-location	200	✓	✓
TIMEX-DATE	interval	0	✓	✓
TIMEX-TIME	interval	0	✓	✓

ignore	only wrong

```
Status
16.02.06 15:10:02 - Loaded Settings.
16.02.06 15:10:02 - Loading Settings from '/project/quetcorp/attila/systems/sp
16.02.06 15:09:37 - Started JTaCo.
```

Figure 4: Definition of comparison settings in JTaCo's SProUT IDE plug-in. See Section 3.4. for a detailed explanation.

4. SProUTomat

SProUTomat, described in more detail in (Schäfer and Beck, 2006), is an automatic build, testing and evaluation tool for linguistic resources and components that has been implemented for SProUT. SProUTomat is used for daily building and testing the development and runtime system from the program and lingware source code checked out from a version control system.

4.1. Build Procedure

SProUTomat is an extension of the build mechanism for language technology components and resources we have developed for the SProUT system using Apache Ant (`http://ant.apache.org`). Ant is a standard open source tool for automatic building and packaging complex software systems. On the basis of target descriptions in an XML configuration file, Ant automatically resolves a target dependency graph and executes only the necessary targets.
Before testing and evaluating, a system has to be built, i.e., compiled from the sources checked out from the source control system. The Java program code compilation of SProUT is a straightforward task best supported by Ant. The case is, however, different for lingware sources (type hierarchy[4], tokeniser, morphology, gazetteer, XTDL grammars).
While the appropriate Java code compilation tasks know

[4]The SProUT formalism uses a subset of TDL (Krieger and Schäfer, 1994) that is compiled using the *flop* compiler of the PET system (Callmeier, 2000).

what a compiled class file is and when it has to be recompiled (source code changes, dependencies), this has to be defined explicitly for lingware resources which Ant natively is not aware of. The `uptodate` task can be used to compare source files (.tdl in the following example) against their compiled version (.grm).

```
<uptodate property="tdl_input_is_uptodate"
           srcfile="${typehierarchy}.tdl"
        targetfile="${typehierarchy}.grm"/>
```

For each of the different lingware types, these source file dependencies are defined as are the calls to the dedicated SProUT compilers and parameters for their compilation.
Lingware-specific targets have common parameters and properties like `"lang"`, `"project"` or the lingware type that are used to locate, e.g., the source and compiled files in the hierarchically defined directory trees or `"charset"` to specify encodings for source files to read.

```
<!--usage : ant compile_ne -Dlang=en -->
<target name="compile_ne" depends="jar"
 description="Compile NER grammar.">
  <property name="lang" value="en"/>
  <property name="project" value=""/>
  <property name="charset" value="utf-8"/>
  <!-- compile type hierarchy -->
  <antcall target="compile_tdl"/>
  <!-- compile tokeniser -->
  <antcall target="compile_tokenclass"/>
  <!-- compile gazetteer -->
  <antcall target="compile_gazetteer"/>
  <!-- compile XTDL grammar for NER -->
  <antcall target="compile_grammar"/>
</target>
```

Figure 5: A sample target definition: named entity grammar compilation.

Dependencies between different lingware types are handled by calls to defined sub-targets. Figure 5 shows the definition of the `compile_ne` target that calls four other compilation sub-targets. Each subtarget compiles only when necessary, and the `compile_ne` target itself depends on the `jar` target that provides working and up-to-date SProUT lingware compilers.
Besides the program and lingware compilation, many other targets exist, e.g., to generate documentation, package runtime systems, start the integrated development environment, etc.
Thus, using a single command, it is possible to compile the whole system including code and all dependent available linguistic resources, or to update it after changes in the sources.

4.2. Test and Evaluation

When SProUTomat is started, it first updates all program and lingware sources from the version control system, and compiles them. For each language resource to test, a reference text is then analysed by the SProUT runtime system. This checks for consistent (re)sources. The next step is comparison of the generated named entity and information extraction annotation against a gold standard. SProUTomat uses the batch version of JTaCo for the automatic evaluation and computation of precision, recall and f-measure. For English, the annotated corpus is taken from the MUC evaluation data. For other languages for which no MUC annotations exist (e.g., German), a manually developed corpus is employed.

4.2.1. Report

Finally, a report is generated and emailed to the developers with an overall status (OK or ERROR) for quick information. The report also contains diagrams consisting of precision, recall and f-measure curves since beginning of regular measurements per language that visually give a quick overview of the resource development progress over time (cf. Figure 6). To this end, the evaluation numbers are also added to a global evaluation database.

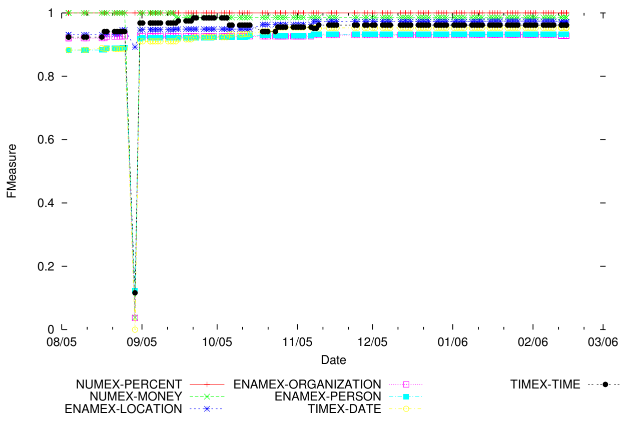


Figure 6: F-measure curves of the English MUC-compatible NER grammar collected by SProUTomat from 08/05 to 03/06. The drop in August/September was caused by a code change not followed by an immediate adaptation of the lingware.

5. Summary

We have presented a comprehensive framework for automatically testing and evaluating multilingual linguistic resources and language technology components. The system is in daily use since March 2005 and successfully helps to maintain the quality and reliability of the multilingual language processor with its various resources that are developed by many authors and used in several projects. The framework greatly helps to improve and accelerate the development - evaluation/comparison - refinement cycle, gives motivating feedback (such as raising recall and precision curves over time) and thus provides continuous quality assurance for a complex natural language processing system.

6. Acknowledgements

We would like to thank Daniel Beck for helping to implement SProUTomat, the SProUT grammar developers for their feedback, Witold Drożdżyński for extending the SProUT API to our needs and the reviewers for helpful comments.

This work has been supported by research grants from the German Federal Ministry of Education and Research in the context of the projects QUETAL (FKZ 01 IW C02) and COLLATE (FKZ 01 IN A01).

7. References

Christian Bering, Witold Drozdzyski, Gregor Erbach, Clara Guasch, Petr Homola, Sabine Lehmann, Hong Li, Hans-Ulrich Krieger, Jakub Piskorski, Ulrich Schäfer, Atsuko Shimada, Melanie Siegel, Feiyu Xu, and Dorothee Ziegler-Eisele. 2003. Corpora and evaluation tools for multilingual named entity grammar development. In *Proceedings of Multilingual Corpora Workshop at Corpus Linguistics*, pages 42–52, Lancaster, UK.

Christian Bering, 2004. *JTaCo User Guide*. Saarbrücken, Germany. Saarland University, Computational Linguistics Department.

Ulrich Callmeier. 2000. PET – A platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(1):99–108.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia, PA.

Witold Drozdzynski, Hans-Ulrich Krieger, Jakub Piskorski, Ulrich Schäfer, and Feiyu Xu. 2004. Shallow processing with unification and typed feature structures – foundations and applications. *Künstliche Intelligenz*, 2004(1):17–23. Available online.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference - 6: A brief history. In *Proceedings of COLING-96*, pages 466–471, Copenhagen, Denmark.

Hans-Ulrich Krieger and Ulrich Schäfer. 1994. TDL – a type description language for constraint-based grammars. In *Proceedings of COLING-94*, pages 893–899.

Kiyong Lee, Lou Burnard, Laurent Romary, Eric de la Clergerie, Ulrich Schäfer, Thierry Declerck, Syd Bauman, Harry Bunt, Lionel Clément, Tomaz Erjavec, Azim Roussanaly, and Claude Roux. 2004. Towards an international standard on feature structure representation (2). In *Proceedings of the LREC-2004 workshop on A Registry of Linguistic Data Categories within an Integrated Language Resources Repository Area*, pages 63–70, Lisbon, Portugal.

Ulrich Schäfer and Daniel Beck. 2006. Automatic testing and evaluation of multilingual language technology resources and components. In *Proceedings of the 5th International Conference on Language Resources and Evaluation LREC-2006*, Genoa, Italy.

Panel Session

Panelists:
Chris Cieri (LCD, USA)
Khalid Choukri (ELDA, France)
Chu-Ren Huang (Acad Sin, Taiwan)
Takenobu Tokunaga (TIT, Japan)

In this panel we will get back to the question *What is Quality*, addressed by Chris Cieri in his invited talk at the beginning of this workshop, in the light of the presentations given at this workshop. Panelists will be asked to give their own views on what the essence of quality is, and which impact this has on best practice in quality assurance and validation of language resources.