TSNLP — Test Suites for Natural Language Processing

Stephan Oepen, Klaus Netter*

German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
{oe | netter}@dfki.uni-sb.de

Abstract

We present recent results from the LRE project TSNLP (Test Suites for Natural Language Processing) which is concerned with central issues in the design and use of test suites.. The paper focusses on (i) the **motivation for test suites in comparison to (annotated) corpora**, (ii) the **theory-neutral annotation schema** developed in TSNLP; and (iii) the construction of a **linguistic database as a (virtual) meta test suite** to ease applications in diagnosis and evaluation. — A software demonstration is possible.

1 Background: The TSNLP Project

In a natural language processing context a *test suite* is a (more or less) systematic collection of linguistic expressions (test items, e.g. sentences or phrases) and often includes associated annotations or descriptions. However, most of the existing test suites have been written for specific systems or simply enumerate a set of 'interesting' examples; this clearly does not meet the demand for large, systematic, well-documented and annotated collections of linguistic material required by a growing number of NLP applications.

The TSNLP project addresses a range of issues related to the construction and use of test suites. The main goals of the project are to:

- provide guidelines for the construction of test suites to facilitate a coherent and systematic data collection;
- develop a rich annotation schema that is maximally neutral with respect to particular linguistic theories or specific evaluation and application types;
- construct substantial test fragments in English, German and French that are applicable to different NLP application types; (parsers, grammar and controlled language checkers);
- design and implement a linguistic database to store, access and manipulate the test data;
- investigate methods and tools for a (semi-) automated test suite construction.

Both the methodology and test data will be validated in a testing and application phase. The results of TSNLP will be made available to the public domain.

*Most of the work presented here is joint research of the TSNLP consortium; we acknowledge our partners Doug Arnold, Lorna Balkan and Siety Meijer (University of Essex, UK), Dominique Estival, Kirsten Falkedal and Sabine Lehmann (ISSCO, Geneva) and Eva Dauphin and Sylvie Regnier-Prost (Aerospatiale, France)

2 Test Suites vs. Text Corpora

Test suites consisting of systematically constructed individual test items are not the only source of reference data for testing and evaluating NLP applications. In particular they clearly cannot replace test corpora drawn from naturally occurring texts but rather have to be seen as serving different purposes. Among the key differences between test suites and corpora are the following (see [1] for more details):

- **Control over test data:** While test suites allow the construction of focussed test data illustrating specific phenomena in isolation or in controlled interaction, corpora will always comprise arbitrary combinations of different phenomena. Although the combinations found in corpora may be empirically relevant they do not allow for a fine-grained diagnosis of system performance.
- **Systematic coverage:** Test suites can provide test items which illustrate systematic variations over a specific phenomenon. In a corpus such variations are likely to occur only accidentally and they will not be related to each other automatically.
- **Non-redundant representation:** While well constructed test suites will list every phenomenon only once, corpora are clearly redundant, in that one and the same phenomenon can occur over and over again.
- **Inclusion of negative data:** Ungrammatical examples do not occur naturally (or occur unsystematically) in corpora. However, for diagnostic purposes testing for negative data may be relevant (e.g. with grammar and controlled language checkers).
- **Coherent annotation:** As is discussed below, constructed test data can be associated with different kinds of systematic and coherent annotations. With few exceptions such annotations are hard to find in combination with corpora.

For most of these aspects it could be argued that e.g. the selection and combination of phenomena and the degree of variation within some particular phenomenon as they are found in corpora reflect what is actually relevant in the real world. However, corpora — regardless of their size — will only give a more or less representative sample and do not allow for a focussed diagnosis. Ideally, test suites and corpora should stand in a complementary relation, with the former building on the latter wherever possible and necessary.

The TSNLP project aims to construct test suites that address all of the desiderata listed above; there is a special emphasis on the collection of minimal sets of test items reflecting the systematic variation of a single parameter while all others are controlled.

3 The TSNLP Annotation Schema and Database Model

Based on a survey of existing test suites ([4]) and the types of annotations (if any) found, a detailed annotation schema has been designed which neither presupposes some token linguistic theory nor a particular evaluation type nor a limited range of suitable applications (see [2] for a complete presentation).

Test data and annotations in TSNLP test suites basically are organized into four distinct representational levels:

- **Core data:** The core of the data collection are the individual *test items* (sentences, phrases et al.) together with all general, categorial and structural information that is independent of phenomenon, application and user parameters. Besides the actual string of words, annotations at this level include (i) bookkeeping records (date, author, origin and item id), (ii) the item format, length, category and wellformedness code, (iii) the (morpho-)syntactic categories and string positions of lexical and — where uncontroversial — phrasal elements and (iv) an abstract dependency structure. Encoding a dependency or functor-argument graph rather than a phrase structure tree avoids to impose a specific constituent structure but still can be mapped onto one.
- **Phenomenon related data:** Based on a hierarchical classification of *phenomena* (e.g. verb valency being a subtype to general complementation) each phenomenon description is identified by its phenomenon id, supertype(s), interaction with other phenomena, and presupposition of such. Moreover, the set of (syntactic) parameters that is relevant in the analysis of a token phenomenon (i.e. the number and type of complements in the case of verb valency) is determined.[1] Individual test items are then assigned to one or several phenomena and annotated according to its respective parameters.
- **Test sets:** As an optional descriptive means inbetween the test item and phenomenon levels pairs or sets of grammatical and ill-formed items can be grouped into *test sets*. Thus, it can be encoded how e.g. test items that originate from the systematic variation of a single phenomenon parameter relate to each other.
- **User and Application parameters:** Information that will typically correlate with different evaluation and application types in the use of a test suite is factored from the remainder of the data into *user and application profiles*. Currently, we have foreseen (i) a centrality measure on a per phenomenon, test set or even test item basis and (ii) judgements of relevance with respect to a specific application. To match the results obtained from a token NLP system (a parser, say) against a test suite, at the profile level a formal or informal output specification can be added.

In addition to the parts of the annotation schema that follow a more or less formal specification, there is room for textual comments at the various levels in the above hierarchy to accommodate information that cannot be formalized.

Building on previous work at DFKI ([3]) all TSNLP test data will be stored in a relational database facilitating the access and maintenance. The resulting linguistic database can be seen as a virtual or meta test suite that allows the retrieval (i.e. automated construction) of actual test suite instances according to arbitrary linguistic or extra-linguistic constraints. For an initial construction phase a graphical test data editor has been built following the rough (four-level) conceptual ontology sketched above. During the database import the data will be organized into a more finer-grained entity-relationship model where e.g. items and phenomena are first class objects and categorial or functional annotations cast in separate relations.

4 Putting it to the Test: Applications for the TSNLP Linguistic Database

To allow for a maximal degree of flexibility and portability and in order to put both data *and* software into the public domain, a small and efficient relational database engine is currently under development. The database kernel is implemented in ANSI C and organized into a library that shell or application programs can be linked against to be given access to the linguistic database.

Currently there is an ascii-based retrieval frontend that interprets an SQL-like query language and some rudiments of a graphical browser. It is envisaged that at least two additional client programs shall be linked to the database kernel during TSNLP: (i) a lexical replacement tool done at the University of Essex and (ii) the NL discourse core engine developed at DFKI to validate the TSNLP linguistic database in diagnostic and progress evaluation purposes.

[1] Although originally a plain relational database representation was envisaged (see below) the direct encoding of the phenomena hierarchy would suggest a notion of parameter inheritance similar to concepts known from taxonomic databases. It remains to be seen how much of this can and shall be imported into the TSNLP data model.

References

[1] Lorna Balkan, Klaus Netter, Doug Arnold, and Siety Meijer. Test Suites for Natural Language Processing. In *Proceedings of the Language Engineering Convention*, Paris, 1994.

[2] Dominique Estival, Kirsten Falkedal, Sabine Lehmann (principal authors), Lorna Balkan, Siety Meijer, Doug Arnold, Sylvie Regnier-Prost, Eva Dauphin, Klaus Netter, and Stephan Oepen. Test Suite Design — Annotation Scheme. Report to LRE 62-089 D-WP2.2, University of Essex, UK, 1994.

[3] John Nerbonne, Klaus Netter, Kader Diagne, Ludwig Dickmann, and Judith Klein. A Diagnostic Tool for German Syntax. *Machine Translation*, 8:85–107, 1993.

[4] Dominique Estival (principal author), Kirsten Falkedal, Lorna Balkan, Siety Meijer, Sylvie Regnier-Prost, Klaus Netter, and Stephan Oepen. Survey of Existing Test Suites. Report to LRE 62-089 D-WP1, University of Essex, UK, 1994.