In: Proceedings of Language Engineering Convention - Paris 6-7 July 1994,
pp. 17-22, ELSNET, Edinburgh, 1994

TSNLP
Test Suites for Natural Language Processing

Lorna Balkan∗ Klaus Netter‡ Doug Arnold∗ Siety Meijer∗

∗ University of Essex, United Kingdom
‡DFKI GmbH, Saarbrücken, Germany

ABSTRACT

This paper describes the LRE project TSNLP (Test Suites for Natural Language Processing), which is concerned with some central issues in the design and use of test suites. The project combines theoretical research with practical implementations, aiming to provide generally usable tools and test data together with reports discussing the theoretical background. The paper begins by setting out the motivation, aims, and present state of the project, then examines the methodological issues behind it[1].

Motivation

In a Natural Language Processing context, a *test suite* is a more or less systematic collection of specially constructed linguistic expressions (test items, e.g. sentences), perhaps with associated annotations and descriptions. Test suites have long been accepted in the NLP community as a useful tool for diagnostic evaluation. However most of the existing test suites have been written for specific systems or simply contain numbers of "interesting examples". This does not meet the demand for large, systematic, well documented and annotated collections of linguistic material, required by a growing number of NLP applications. Collections of this type are not only useful as diagnostic tools, but can also support other kinds of evaluation. Last but not least, large data collections could also serve as repositories of linguistic phenomena for developers.

Project Aims

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:

1. Provide Guidelines for test suite construction

Guidelines will help the test suite constructor to write coherent and systematic test data. They address issues like the identification of linguistic phenomena and sub-phenomena to be tested, the choice of the vocabulary used in test data, how can one derive ill-formed data and to what extent the interaction between different phenomena should be tested.

2. Construct substantial test fragments

Test items will be written for parsers, grammar checkers and controlled language checkers. They will contain a number of pre-selected phenomena, and will become available in three languages: English, French and German. Essential is that the test data will be annotated in a way which considerably improves the informativity of the test data.

3. Develop a database

The test items will be stored in a database, where the annotations on the test items allow for ease of access to and manipulation of data. Work done at DFKI Saarbrücken [2] will serve as the starting point.

4. Investigate test suite construction tools

The project aims to develop methods for automating some of the processes involved in test suite construction. The tools considered here are a test suite generation tool (from grammars and corpora), and an automatic lexicon replacement tool.

The guidelines and test data will be validated by a testing phase.

The results of the project will become public domain and it is expected that the availability of validated test suites for a range of applications, together with the tools for their manipulation and use will be of great value to the NLP community.

Research to Date

During the first phase of the project a study of existing, publicly available test suites was performed[1]. The study revealed that test suites range in evaluation purpose, intended application (parsers, MT systems, etc.), depth and breadth of coverage, etc.

Existing test suites display a degree of systematicity, where test items generally only contain one linguistic phenomenon more than those previously tested. They are primarily concerned with the coverage of syntactic phenomena, often include (some) ill-formed data and have restricted lexical coverage. Some general shortcomings of existing test suites are:

- Lack of morphological, semantic and extra-grammatical phenomena.
- Lack of systematicity in testing ill-formed constructions and the co-occurrence of different phenomena.
- Lack of documentation and annotation. This is

[1] We would like to thank our partners in the TSNLP project: Dominique Estival, Kirsten Falkedal and Sabine Lehmann at ISSCO, Geneva and Sylvie Regnier-Prost and Eva Dauphin at Aerospatiale, Paris

a great drawback as it reduces the informativity and reusability of the test suite.

- They are too specific to a certain system or application to be reusable.
- It can be difficult to interpret the results of test suite testing.

The outcome of the study of existing test suites has served as input to the second (and current) phase of the project, which concentrates on the elaboration of guidelines for test suite construction and an appropriate annotation scheme. This work involves a number of theoretical and practical issues, ranging from the usefulness and purpose of test suites to the implementation of guidelines and annotations.

As this work is central to the project and its results, the rest of this paper will give an overview of the main issues and methodological challenges that are addressed in the project.

Test Suites vs. Test Corpora

Of course, test suites are not the only source of reference data for testing and evaluating NLP applications. An important issue that will arise in this context is the relation between test suites and test corpora. Clearly test suites, consisting of systematically constructed individual test items, cannot replace test corpora drawn from naturally occurring texts, but rather have to be seen as serving different purposes and achieving different results. The different aspects involved include among others the following.

Control over test data While test suites allow the construction of focused test data illustrating specific phenomena in isolation or controlled combinations, corpora will always comprise arbitrary combinations of different phenomena. Of course, the combinations found in corpora are empirically relevant in that they show which combinations typically occur. However, for evaluation purposes, corpora primarily provide overall results of system performance, but do not allow the degree of focus that is possible with a test suite.

Systematic coverage of phenomena 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 automatically related to each other. One could argue again that only those variations which actually occur "in real life" are interesting. However, corpora never represent "real life" in its full wealth, but only a more or less representative sample.

Non-redundant representation of phenomena While well constructed test suites will list every phenomenon only once, corpora are clearly redundant, in that certain phenomena will occur over and over again.

Negative or ungrammatical examples Ungrammatical examples do not occur naturally (or occur unsystematically) in corpora. However, for diagnostic purposes testing for negative data may be relevant. This is obviously the case for grammar checkers or controlled language checkers, which crucially have to make a distinction between acceptable and non-acceptable input.

Annotation of test items Items in test suites can be annotated with different types of useful information, as discussed below. With few exceptions such annotations are hard to find in combination with corpora.

To summarize: It would be rather foolish to argue against corpora as a means for testing and evaluation. However, for certain purposes test suites could provide a more suitable and direct tool. Ideally, test suites and corpora should stand in a complementary relation, with the former building on the latter wherever possible and necessary.

Test Suites and Evaluation Type

A number of different factors relating to the type of evaluation to be undertaken are relevant to the design and use of test suites (see e.g. [4, 3]). These include the *purpose* of the evaluation (diagnostic, progress, adequacy evaluation), the *type of evaluator* (user vs. developer), the *evaluation basis* (test suites vs. corpora), the *type of system* (module vs. compound system), the *evaluation method* (black box vs. glass box evaluation), etc.

At first sight one may be inclined to constrain the usage of test suite to diagnostic evaluation performed by a developer on a single module in a glass box situation. While this is a quite straightforward proto-typical use of test suites, we want to discuss below to what degree test suites could or should be involved in other types of evaluation.

Purpose of evaluation The purpose of an evaluation has an impact on the utility of test suites. As suggested by (inter alia) the EAGLES subgroup on evaluation [3], the following purposes can be distinguished:

- *diagnostic evaluation*, which aims at localizing deficiencies in the system;
- *progress evaluation*, for a comparison between successive stages of development of a system over a period of time;
- *adequacy evaluation*, which determines whether and to what extent a particular system meets some pre-specified requirements.

These types are not necessarily disjoint. For example, adequacy evaluation attempts to find out whether a system is suitable for a specific application. This may amongst others involve the identification of the deficiencies of a system, which requires diagnostic evaluation. And since many NLP products are customized on user demands, progress evaluation could also be an essential part of adequacy evaluation.

It is therefore not possible simply to identify user evaluation with adequacy evaluation and developers' evaluation with diagnostic evaluation. For users this trivially follows from the above, if they are interested in more than just the overall performance of a system. For developers it can be derived from the fact that they have to develop products which can be sold on the market, i.e., adequacy evaluation will eventually be as important for a developer as it is for a user, even if diagnostic evaluation may play a more important role at earlier stages of development.

Test suite and corpus evaluation There is often assumed to be a link between diagnostic evaluation and test suites, and between adequacy evaluation and corpus based evaluation. This is correct in the sense that a properly chosen corpus is likely to provide a representative sample of inputs, and thus forms a good basis for adequacy evaluation. However, if a test suite has been properly designed, taking frequency and relevance of phenomena into account, it may well be almost as useful as a corresponding corpus.

Glass box and black box evaluation A frequently made, although in our view hypothetical, correlation is the one between user, adequacy and black box evaluation on the one hand and developer, diagnostic and glass box evaluation on the other. What is commonly understood by black and glass box evaluation is that in the former case the evaluator does not have access to anything but the input and output of a system, whereas in the latter he can have insight into internal states or generally the flow of information between modules. at all. A developer doing diagnostic evaluation of a module or component is usually as much in a black box situation about the internal workings of a component as is a user with respect to an entire system. After all, the developer will not trace every internal step of a module during the evaluation process, but rather tries to locate potential deficiencies by looking at the output pattern that emerges from a series of systematic test items. (Removing a deficiency will of course involve access to the code, but this is neither the purpose of evaluation nor of diagnostics proper.)

Module and system evaluation An important factor influencing the utility of test suite evaluation appears to be the complexity and predictability of the tested system. One of the main advantages of test suites is that they allow test items to be constructed on the basis of a systematic variation of certain parameters while other parameters remain stable. As just noted, unexpected irregularities in the input/output associations, can be used to detect system internal deviations from an expected behaviour. However, this is only possible to the extent that the expected output can be predicted from the input. If the system output is influenced by parameters that cannot be controlled by the setup of test items, at least the (semi-)automatic evaluation by means of a test suite becomes less reliable. For this reason, test suites appear most suitable for the testing and evaluation of systems (or components) which behave in a relatively predictable and deterministic manner.

To summarize: Test suites are not only suitable for diagnostic and progress evaluation, but can also be used to support adequacy evaluation if the evaluator is interested also in more detailed information about deficiencies and customizability of a system. They can complement the function of corpora in adequacy evaluation if they are tuned towards the relevant domain, in the sense of containing information about frequency and relevance. They are suitable for black box evaluation on condition that the systematic paradigmatic patterns in the test suite in correlation with the expected system output allow inferences to be drawn about internal irregularities of a system. This presupposes that the parameters influencing the output of a system or component are controlled or are systematically dependent on the input test item.

Test Suites and Application Type

Much of what has been said in the previous section is of relevance for determining the types of application for which a test suite is appropriate. However, we also have to be aware of the kinds of information that can be encoded in a test suite.

The most basic information that can be associated with a test item is whether it belongs to a language or not. This distinction may coincide with the notion of grammaticality, if the language in question is a natural language, or it may have another basis in the case of sublanguages (as is appropriate for evaluation of controlled language checkers, e.g.).

Since a test suite cannot enumerate all sentences of a language, it has to simplify certain phenomena, such as recursion, it may have to generalize over certain phenomena, such as lexical distinctions, and it may have to abstract from possible combinations of phenomena occurring in natural language, such that an appropriate degree of minimality is achieved.

The question is how to establish which phenomena are to be included in a test suite and whether they are represented by just one or two examples or by means of a larger set of items. This will relate to whether the breadth or depth of coverage of an application is being tested.

Encoding of information Information can be carried either by a test item or a sequence of test items forming a systematic pattern, or it can be provided by annotations to test items. In the latter case one has to keep in mind that different applications or systems require different types of annotations.

Clearly, in many cases information about the expected output will have to go into the annotations. However, it must be borne in mind that that such annotations only make sense if they can be matched with the system output. This will be possible only if the range of possible output is sufficiently narrowed that it is predictable from the input. Whether or not this is possible will strongly depend on the kind of application chosen as well as what parts of an application are to be tested.

Consider, for example, the testing of grammar checkers and controlled language checkers. Here the most basic annotation of "grammaticality" with respect to some (sub-)language is crucial. One is testing whether such systems accept all grammatical sentences (i.e. all sentences belonging to the relevant (sub-)language) and reject or mark all ungrammatical sentences. If the kind of error report expected from the system in the case of ungrammatical sentences is known, this information may be encoded in the annotations of a test suite. The same may hold for suggestions for corrections, if the system provides a fixed protocol for errors. To account for the error report and corrections one should control the ill-formed input, i.e. the number of errors in test items should be as minimal as possible.

On the other hand, information about grammaticality will be less important in other applications. For example machine translation systems (or any system which involves parsing). This is because one expects such systems to be "robust", i.e. to tolerate a degree of ill-formedness in the input, treating such input in much the same way as well-formed input. (Of course, ungrammatical test items, and hence such annotations, are useful for testing this).

Problems in the Construction of Test Suites

When constructing a test suite from scratch or by building on an existing test suite a number of design issues have to be taken into account, not all of which can be given clear cut prescriptions.

Choice of Phenomena As we mentioned, a test suite cannot cover an entire language, so some effort has to be spent on selecting those aspects and phenomena for which testing will support the development or evaluation of an application. What appears to be required here is a natural balance between basic or very common phenomena and those which are more complex or possibly even marginal. A crucial question here is whether one can base the selection of phenomena on general linguistic research. On the one hand, linguistic research often provides a well-structured perspective of phenomena based on a wealth of extensively studied examples. On the other hand, such research often does not discriminate between central and marginal phenomena, but concentrates on issues which are popular at certain times. The postulation of a universal theory of grammar in which all phenomena can be reduced to the same basic may be of high theoretical interest, but cannot form the basis for choosing the relevant phenomena across different languages in building a multi-lingual test suite. A striking example are long-distance dependencies, which are clearly quite important and frequent in English, but hardly play a role a German.

There are also phenomena, which are either too complex or too "trivial" to be satisfactorily handled by linguistic theory. Examples of the former would be coordination, comparatives and different forms of ellipsis, for which to date there is no generally accepted theory which one could draw on when constructing a test suite. Examples of the latter are phenomena which have little theoretical linguistic interest, but which appear to be subject to much idiosyncrasy across languages, such as expressions for time, dates, or different forms of measurement. Even ignoring this theoretical issue, there remains the problem that often the relevance of phenomena may be application specific, so that what may be important in one domain of applications may be quite irrelevant in others.

The answer to this problem very much depends on whether one wants to construct a general pool of test items from which application specific test suites can be drawn, or whether one wants to tailor the test suite right from the beginning towards a specific purpose. The former choice appears to be quite preferable, as long as the structure of the test suite supports the extraction of sub-sets of test items. This is indeed what the TSNLP project is aiming for, although with a restricted number of applications. As it will hardly be possible to foresee all appropriate applications at the time of constructing and annotating a test suite, the possibility of augmenting the annotations has to be taken into account as well.

Systematicity of coverage Once the choice of phenomena to be covered by a test suite has been made, the question of breadth and depth of coverage arises. Very shallow coverage, which lists only one central item per phenomenon, will only give a rough indication of the coverage of a system, and will systematically test the depth of a system. In some cases (word order for example), it does of course not even make sense to have only one item. On the other hand, attempting to explore all possible variations can easily lead to a sequence of test items which comprises both very central and very marginal examples. Again, word order would be a case in point, since some variations may be grammatical but not very frequent. In this case, systematic coverage should be associated with an appropriate rating of the test items.

Control over parameters What makes test suites valuable in comparison to corpora is that they can focus on specific linguistic phenomena and present them in isolation. Of course, one cannot say that any sentence illustrates just some phenomenon, since language is inherently a combination of different words standing in specific grammatical relations. Thus, talk about presenting a single phenomenon can only be understood in the sense that as many linguistic parameters as possible are being kept under control. This holds in particular for those cases where a phenomenon is illustrated by systematic variation over the parameters making up this phenomenon, while all other parts of the test items are being kept constant.

Lexical and structural phenomena Vocabulary is one of these parameters that need to be controlled, which is not always possible for all phenomena. For some phenomena the choice of the specific vocabulary (besides the functional words) hardly makes any difference and therefore can be kept stable; this is for example true for coordination phenomena and relative clauses. However, some grammatical phenomena depend on specific lexical items (e.g. subcategorization, passive voice, preposition selection).

In general, one should look for vocabulary which is as neutral as possible. Here very simple noun phrases or even pronouns, which abstract from any specific semantic properties are useful. For some applications, such as controlled language checkers, the choice of vocabulary may be important even beyond these aspects, since many simplified languages restrict the use of certain words in specific contexts to prevent ambiguous interpretations.

Combination of phenomena The advantage of test suites over corpora, i.e. the fact that they present phenomena in isolation, can also be seen as their disadvantage, in that often the real test case for a system is whether it also covers the arbitrary combination of linguistic phenomena. Clearly, combining all different possible phenomena would lead to a combinatorial explosion and therefore the straightforward answer to this problem would be that this is why test suites and corpora should complement each other. However, up to a limited degree the combination of phenomena in a controlled manner can also be taken into account in test suites.

One way of achieving the required purpose is to concentrate on those combinations which are crucial for testing purposes. Even if a system could always fail on some arbitrary combination, it is usually the case that not any variation or combination of phenomena presents an interesting test case.

A second possibility to increase the combinatorial coverage of a test suite would be to use lexical or phrasal replacement tools. Test items can also be designed as a pattern in which substrings can be replaced by other substrings standing in a paradigmatic relation. For example, if a test suite already contains different forms of noun phrases as test items, these items could be used to replace noun phrases in other sentential test items (given that certain constraints are obeyed). A test suite designed along these lines would have the same function as a generative grammar, with the advantage that the basic building blocks are constructed by hand and thus can be arbitrarily complex.

Ill-formed test items As mentioned above, negative examples can be interesting from the developers point of view, since the indiscriminate acceptance of an ungrammatical construction often bears the risk of misinterpreting a grammatical construction. And they can be of direct relevance to applications such as grammar checkers, which are designed to recognize and mark such constructions. However, the problem is that for any grammatical example one can construct ungrammatical examples which can go wrong in many different ways, not all of which may be interesting and not all of which can be listed.

In the ideal case, the set of ungrammatical examples falls out from the systematic variation of parameters. For example, one way to cover different coordination phenomena in language like German is to take a sentence illustrating a basic coordination construction and then perform all possible combinations of deletion and permutation that are logically possible. As a result one will cover all possible types of coordination including gapping, right or left node raising etc. Quite a number of these examples will turn out to be ungrammatical illustrating the constraints that hold for coordination. Similarly, for agreement phenomena the relevant features could be systematically varied such that grammatical and ungrammatical examples arise automatically. In these cases, the parameter variations define a logical space which constrains the set of ungrammatical examples.

Annotation Types

If the systematicity of the test data already carries enough information to make inferences about the internal structure of a system, it may play a minor role how the test items are annotated. Still, for most applications, some annotations recording the structure of the test items will be useful. Since there is hardly any general consensus on all details of phrase structure, annotations along the lines of tree banks may be too specific or even idiosyncratic. More abstract annotations, such as a semantically oriented dependency structure on the other hand, could be too unspecific to serve the purpose of testing, for example, controlled language checkers or grammar checkers.

Basically one can distinguish annotations which are of general interest and describe the structure of test items in a way which abstracts from specific theoretical assumptions, and annotations which are application specific.

Annotations describing the structure of a test item or a set of items could be of the following kind:

- classifications of the phenomena illustrated by a test item, such that it can be seen in a relation with other test items relevant for the phenomenon;
- classifications of the grammaticality, centrality and complexity of the illustrated phenomenon;
- morpho-syntactic tagging of the lexical material of the test item;
- abstract functor-argument structures, which do not impose but still can be mapped onto specific phrase structures;
- general comments on the specific linguistic issues associated with an item or phenomenon.

Annotations which are specific to the application to be tested will have to take into account the expected output or system behaviour with respect to a test item. It is difficult to foresee at this moment, whether it will be possible to design these annotations such that different systems of the same application type can be accomodated.

From Benchmarking to Evaluation: Frequency and Relevance

As we have mentioned several times, general purpose test suites normally list test items on a par without rating them in terms of frequency or even relevance with respect to an application. We emphasized that test suites without such information cannot serve as the basis of evaluation in the same way as an application, task or domain specific corpora could do, unless the human evaluator assigned his own ranking to individual test items or phenomena.

As to relevance, this will presumably always have to be assigned by hand, since a relevant phenomenon need not necessarily be a frequent phenomenon. For example, the correct coverage of constructions with negation can be very crucial to an application, even if these constructions may not be very frequent in comparison to others.

As to frequency, one of the major challenges in constructing and applying test suites for evaluation consists in the combination of the controlled environment of a test suite with the information about the frequency of individual constructions and classes of phenomena to be found in application specific corpora. One approach that we feel future research should concentrate on would be to match tagged test items against analogously tagged corpora, such that the matching abstracts away from additional material in the corpus taking into account only the relevant parts of the sentence.

References

1. Estival, Falkedal et al. (1994), *Analysis of Existing Test Suites*, Report to LRE 62-089 (D-WP1), University of Essex.
2. J. Nerbonne, K. Netter, A Kader Diagne, J. Klein, L. Dickman, *A Diagnostic Tool for German Syntax.* DFKI D-92-03, Saarbrücken. Also in: J. Neal / S. Walter. eds. (1991): *Natural Language Processing Systems Evaluation Workshop, Berkeley.* Rome Laboratory, RL-TR-91-362, New York
3. Draft Interim Report EAGLES Evaluation Group, 1994.
4. J.R. Galliers and K. Sparck Jones, *Evaluating Natural Language Processing Systems.* University of Cambridge, England, Technical Report No. 291, 1993.