TOWARDS BETTER NLP SYSTEM EVALUATION

Karen Sparck Jones
Computer Laboratory, University of Cambridge
New Museums Site, Pembroke Street, Cambridge CB2 3QG
sparckjones@cl.cam.ac.uk

March, 1994

Proceedings of the Human Language Technology Workshop, 1994,
(ARPA), San Francisco: Morgan Kaufmann, 1994, 102-107.

Abstract

This paper considers key elements of evaluation methodology, indicating the many points involved and advocating an unpacking approach in specifying an evaluation remit and design. Recognising the importance of both environment variables and system parameters leads to a grid organisation for tests. The paper illustrates the application of these notions through two examples.

1 Introduction

There is a manifest need for a good evaluation methodology for (S&)NLP systems. This paper presents such a methodology, indicating its key elements under the headings of performance factors, evaluation gauges, test data, and assessment strategy, and illustrating its application. The paper is based on Galliers and Sparck Jones (1993), which offers a comprehensive analysis of what is involved and needed in NLP evaluation, supplemented by an extensive review of methodologies that have already been proposed or applied, and also refers to evaluation experience in the neighbouring information retrieval (IR) field.

2 Background

There is generally growing interest and activity in NLP system evaluation, stimulated partly by the fact that NLP technology is now solid enough to be of interest for real applications and partly by the (D)ARPA initiatives (cf Thompson, 1992; HLT1; MUC; S&NLW; TREC). But evaluation practice is still uneven, and sound methodologies need to be developed, both for laboratory experiments and working system investigations. There are many difficult matters to be considered in NLP evaluation, and some significant points seem not to be sufficiently recognised. Thus there are dangers in seeking to apply 'closed problem' approaches, exemplified by the use of 'word error rate', outside their legitimate bounds; in attaching prime importance to internal objects like parse trees; or in reifying important but ultimately arbitrary processing devices, for instance specific sets of case labels. The IR community's hard-won experience shows there are no easy ways of evaluating systems, no magic numbers encapsulating performance, no 'core' functions that can be pursued far in isolation, no fixed meaning-representation devices any system must have. Further, the lesson of IR is that discourse processing (in this case indexing) is only as good as the use made of it, which is necessarily on individual occasions, so it is only possible to infer, over many such occasions, what average performance will or can be designed to be. The motivation for a better evaluation methodology is thus more reliable inference, implying both a diligent attempt to lay out the premises but also, given the many factors involved in NLP systems and their resistance to precise specification, that the inferences drawn may still not be very reliable.

The essential problem of NLP evaluation is that the evaluator has to advance between Scylla and Charybdis. Evaluation may be end-to-end for systems devoted to NLP, end-to-end for hybrid ones with both NLP and non-NLP subsystems, or non end-to-end, focused on some component of an NL processor; and there is a tradeoff, from the NLP evaluation point of view, between least problematic and also least analytic, and most analytic but also most problematic. End-to-end NLP system evaluation clearly demonstrates NLP function. Hybrid system evaluation is ambiguous in this respect. NL component evaluation is just this, at best within an artificial boundary but at worst imposing inappropriate standards. However with end-to-end NLP systems the situation of use becomes important, showing that in fact in all cases the context surrounding the NLP subject being evaluated matters.

In the rest of this paper I shall consider the four clusters of concepts that need to be recognised and applied in evaluation, namely evaluation subjects and performance factors, kinds and levels of definition for performance assessment, sorts of test and evaluation data, and strategies for evaluation design and conduct. It is clear that there is no single correct way to evaluate an NLP system, and that very different evaluations are required in different circumstances, as I show in the subsequent illustration for two different cases involving the same NLP system. The important point is that evaluation has to be approached comprehensively and systematically, so the presumptions on which is based are laid bare. The methodology I outline, summarising Galliers and Sparck Jones (1993), is aimed at achieving this, leading to soundly-based tests that take account of the properties of both evaluation subjects and their contexts within a regular comparative framework. The principles involved in this are not novel; but they are not sufficiently applied in NLP evaluation, as noted in my concluding comments on the ARPA evaluations, and deserve reiteration.

3 Performance factors

In NLP the *subject* of assessment may be a language processing component, e.g. a syntax analyser; a language subsystem of a larger computational system, e.g. an interface to a fault diagnosis package; a complete computational system which may be wholly devoted to NLP, as in translation, or only partially an NLP system, as in the previous case; or a 'setup', i.e. a computational entity plus some body of people involved with it in some way. As this suggests, the onion metaphor applies, with the subject of the evaluation complemented by its *environment*. It is convenient to refer to any of the kinds of evaluation subject mentioned in the abstract as a *system*: I shall therefore use computational system, or 'c-system', specifically for the third case. In evaluation it is essential to take both system and environment into account as supplying the *factors* affecting performance. Thus attributing meaning to, or explaining, system performance depends on determining both system *parameters* and their potential *settings*, e.g. grammar or lexicon used, and environment *variables* and their *values*, e.g. text type or post-editor level of skill, in as thorough a way as possible.

It is easy to pay too much attention to the system and not enough to its environment as influencing evaluation, so relevant environment variables are not taken into account. Characterisation of both variables and parameters is needed even when an evaluation is an *investigation* of some given system to establish its performance level, but even more so when *experiments* are conducted to compare system performance when parameter settings and/or environment variables are changed. Thus with an automatic summarising system as subject, for instance, different sets of semantic categories, or discourse relations, might be considered as parameter settings, and different document types, or lengths, as variable values. There can be very many performance factors that are hard to capture so alternatives can be compared and assessment of their influence made. Environment variables include both input and output ones, referring to the data for processing and the requirements on outputs imposed by their uses, e.g. for summaries that their intended users are able to read them quickly. With environment variables, it may be hard to determine the appropriate level of *granularity*, e.g. of length differences for input or output texts in summarising, or input genres or reader classes; and it may be hard to keep system parameters separate.

The relation between a system and its environment thus has many implications for assessment practice, notably with respect to the view taken of system *ends* in evaluation, and in relation to the assessment of *generic* NLP systems. It is necessary to establish whether assessment of a system's functional performance refers to its ability to meet its internal design *objectives* or its external *function* requirements, e.g. for a translation c-system, either providing complete parses or delivering texts that don't confuse their readers. While the contrast between glass box and black box evaluation is commonplace, it is useful to interpret this in relation to the system-environment distinction and to a specific view of system function. With generic systems (such as SRI Cambridge's Core Language Engine), the issue is that establishing their merits by *intrinsic* evaluations is of limited value, so extensive *extrinsic* evaluation in a variety of environments is required. This also applies where a generic process, without any explicit generic data resources, is used, as in statistically-based text passage extraction.

4 Evaluation gauges

Apart from the distinctions between kinds of evaluation just mentioned, it is helpful to categorise performance evaluation according to whether it is concerned with *effectiveness*, *efficiency*, or *acceptability* (to humans). Such distinctions help to guide an appropriate choice of performance *criteria*. For instance a text categorisation system might be evaluated for effectiveness by its accuracy in classing (say intrinsically against independently assigned categories for test material, or extrinsically with reference to output dumping by its recipients), or for acceptability in ordering and grouping presented material. It is clearly necessary both to contextualise a notion like effectiveness for a given evaluation, and to motivate performance criteria via their evaluative role. This implies further refinement, for instance of the human class for acceptability. However it is also essential, for sound evaluation, to recognise that as performance criteria are general, e.g. quality for translation output, a criterion requires further definition first by a particular performance *measure*, e.g. proportion of sentences edited, and second, for any measure, by a specific application *method*, e.g. for choosing actual texts.

All three levels of characterisation are required for assessment, even for non-numerical approaches to evaluation. A given criterion can typically be given a range of specific interpretations, quantitative or qualitative, as individual performance measures: for instance translation quality by the proportion of sentences requiring post editing. Any measure in turn has to be operationalised, say for the precise way in which a random data sample is gathered, or how a measure is normalised. Individual choices of significance test would ordinarily be made at this level. Significance testing is required but is hard in NLP because normally only weak, non-parametric tests are applicable. They also only show the least required real difference: this has to be interpreted in practical terms in relation to operational system behaviour, and also needs to be supplemented by some view of what larger differences may actually be attainable. Again, IR experience has emphasised the relevance of all these layers, for example in their application to recall/precision figures. Thus while recall and precision may be used as criteria, thay can be more specifically defined in various ways, and also computed differently in averaging, with non-trivial consequences for what performance 'looks' like. Equally, statistical performance differences may difficult to translate into statements about getting noticeably more good documents in initial search output, while what level of performance is attainable in a given environment is usually unclear.

In many cases there are either existing or natural comparative reference points for evaluation, which may be taken as defining *baseline* or *benchmark* performance. The former provides a rock-bottom view of performance, e.g. as in word-for-word translation or simple word indexing, the latter the current best state of the art, e.g. sentence-by-sentence literal translation. Both allow grounded performance comparisons either for alternative system philosophies or for payoffs from extra effort.

5 Data sorts

The characterisation of the *test data* used provides the other support for evaluation. *Evaluation data* is not just *working data* in the ordinary sense of material to which a system is applied, e.g.

a body of texts to be translated, but also includes *answer* data, e.g. correct translations for key terms supplied by humans. [1] The particular character of the evaluation data naturally follows from the goal and manner of the evaluation, as well as from the nature of the system and its environment. Thus while in some cases concrete answers may be available e.g. replies to database queries, in others this is not feasible: there are no correct outputs for a summarising system, only utility judgements. But it is useful to recognise, for NLP evaluation, some important properties of test and evaluation data and major *sorts* of such data: it is all too easy to assume that any sufficiently large and miscellanous assemblage of material can be used simply because it is large and miscellanous.

Thus it is necessary to consider requirements for data as *realistic*, *representative*, and *legitimate*. For an evaluation of a system for dealing with newspaper material, for instance, a set of newspaper stories is realistic, but it should also be a representative sample of the larger universe of stories with which the system may deal. Representative data should, further, be so in *distribution*, e.g. reflecting the relative frequency of newspaper topic domains in the larger universe. The need for legitimacy, even where data is already realistic and representative, seems superfluous but matters because evaluation data may be so expensive to acquire it is natural to seek to exploit it for other tests than those for which it was originally acquired, for example test paragraphs retrieved for specific queries regardless of other user constraints.

The foregoing defines the most strongly constrained evaluation data. Other forms are proper and useful, but their limitations need to be accepted. Thus natural corpus test data may be selected for *coverage* of linguistic phenomena rather than to reflect statistical frequency; and artificially constructed bodies of test data or *test suites*, for example read speech material, may be provided without explicit accompanying answers (though these may be assumed as 'obvious', as in the speech case). However the most important distinction is between natural and artificial evaluation data, i.e. between *test collections* and *checking collections*. For example, in the IR case a test collection is properly only constituted from the relevance assessments made by the user submitting the query, while the agreed assessments of a set of librarians constitutes a checking collection. There are clear problems on the one hand where the same user cannot be 'reused', for instance offered alternative explanations by different systems for the same original, uninformed question, and on the other where surrogate users offer their answers, especially normative ones. Different test and evaluation data *sorts* serve quite distinct functions and need to be correctly related to an evaluation's goal to avoid illegitimate inferences from the assessment results. Further, the task and situation complexity characterisic of many NLP applications, for instance translating repair instructions or summarising reports for 'non-reusable' users, implies large data sets.

6 Assessment strategy

It is evident that serious evaluation requires a well-understood decomposition of the whole, in terms of the evaluation aims, or *remit*, and *design*, and the precise definition of the evaluation subject. This applies even in the case where all that is wanted is some snapshot of the performance of an operational system. It applies much more where comparisons, which to be useful must be systematic, are wanted, whether in the large grain e.g. for two whole NLP c-systems in some environment or in the small grain, e.g. seeking the reasons for different performance for the two systems. The former naturally leads to a greater emphasis, in testing, on the data variable values to see whether different systems are impervious to changes in these; the latter to changes in the system parameter settings, to see if these really matter. However well-founded tests require changes on both dimensions, implying a *grid* design with individual *runs* representing specific combinations of variable values and parameter settings. Clearly when (as in the ARPA cases) whole systems with very different properties are involved, changes on the system dimension have to be relativised, or taken at several levels. The implication is nevertheless that for reliable conclusions about comparative performance, very many runs are required (as has been found in IR, where long term projects have carried out literally thousands of them).

[1] Training vs test data is a different contrast.

Overall, therefore, the methodology required for an NLP evaluation is an *unpacking* one, designed to address the very many distinctions involved and make properly-related choices on each. The outcome of this combination of interdependent choices will be an individual evaluation which may differ substantially, for the same system, from others with different motivations. There is no one way to evaluate NLP c-systems, primarily because these are not autonomous entities: assuming that there is is a version of the naturalistic fallacy which supposes that NLP aspires towards human LP capabilities without allowing for the fact that humans have different capabilities that are differently deployed in different circumstances.

The decompositional approach is naturally followed by answering a series of questions, gradually working into the fine detail of the evaluation in one or more cycles of specification, to emerge with the *scenario* for the whole evaluation. Thus the questions on the evaluation remit establish the *motivation* for the evaluation, and its specific *goal*, along with the *perspective* from which it is undertaken (e.g. task-specific, financial), what *interest*, individual or group, is prompting the evaluation and what *consumers* are envisaged for its findings. Then given the stated goal, what *orientation*, intrinsic or extrinsic; what *kind* of test, i.e. investigation or experiment; what *type* of evaluation, black box or glass box; what *form of yardstick* (e.g. baseline, benchmark); what *style* of evaluation (e.g. indicative or exhaustive); and what *mode* (e.g. quantitative or qualitative)?

The answers to this first set of questions on the evaluation remit lead to those for the second set, on the evaluation design. Initial questions here define the evaluation subject at the necessary level of detail, by addresssing its ends, i.e. objective or function, the subject's *context*, and its *constitution*. The definition of ends validates the remit and also evaluation gauges; with the context and constitution stated, the first specific subset of questions about performance factors, i.e. environment variables and system parameters, can be answered. The second subset addresses the choices of criteria, measures and methods, and the third the choice of evaluation data, its sort e.g. test suite and *status*, i.e. representativeness etc. The fourth subset determines the evaluation *procedure*.

7 Example

To illustrate all the notions outlined, consider two different evaluations concerned with the same (notional) c-system, PlanS, a system used for tearching achitecture students about house design problems, with which they interact by an NL interface and a graphical one. The essential features of PlanS are that teachers give the students the requirements specification for a house e.g. four bedrooms, tile roof, cost below 100K dollars, etc.; the students plan the house in a graphics window, with form filling for features such as roof material. During design they may query the system in NL, e.g. for the prices of types of bricks. When the design is finished the system assesses the student's plan and feature specification, makes comments e.g. the plan has only three bedrooms, and offers its own version(s). The student may use the NL interface to query the system's comments or plan(s), after which a new design cycle may begin.

The two evaluations for PlanS, showing how the methodology I have described is applied, are for 'Case U', the setup involving PlanS and its student user body, and 'Case L', where the subject is just the NL interface. These examples can only be sketched here: they are more fully described, with other cases, in Galliers and Sparck Jones (1993); the aim here is to show answering the questions increasingly constrains the evaluation.

7.1 Case U

The motivation here is educational effectiveness interpreted for the evaluation goal as finding whether PlanS does, as claimed, enable students to grasp planning concepts rapidly. The perspective is therefore task oriented. The interest behind the evaluation is the department chair, its consumers all the teaching staff. We have an intrinsic orientation, concerned with PlanS's own function of superior training. Since various factors e.g. problem hardness or problem sequence can affect learning speed, an experimental kind of evaluation seems called for, though this can be a black box study, not changing the setup itself, only its inputs. The appropriate form of yardstick appears to be performance for similar students without PlanS, but this is (we suppose) impractical, so a notion of attainable speed has to be defined. The evident challenge of evaluating U suggests an initial indicative rather than exhaustive evaluation style, but as one dealing with learning times, with a quantitative mode.

Now working out the evaluation design given this remit, we validate the motivation against the setup's internal purpose, which is promoting fast learning; characterising U's context includes the contribution of the teachers both in assigning design problems and in marking eventual exam results, and the other inputs in the students' course specifically bearing on the development of design skills. The context also includes properties of the students, e.g. they are first year ones. The constitution of U includes the students' various activities within U relating to PlanS, e.g. how often they use it, and all aspects of PlanS itself, non-linguistic and linguistic. The environment variables for U include levels of exposure to planning concepts, and general experience with computing etc; the system parameters include both e.g. 'workover' habits of the students exploring different designs, and lengths of session, convenience of the graphics, helpfulness of PlanS's critiques. Seeking appropriate performance gauges for U's effectiveness suggests both absolute speed to problem solution and increase of speed, for some attained design level, as criteria: so e.g. a measure could be cycle time for acceptable solutions, defined as ones not breaching planning regulations, not having obvious faults like rooms with no doors. The method averages times for date sets on plans determined acceptable for both benchmark reference and new result sets, with significance tests on comparisons. The test data would ideally be session log files, or if not available select records of input problems and output critiques, along with times. The evaluation requires reference data for good times, which could be obtained by taking last year's best students, getting their times for new problems, and using these as benchmarks. The procedure involves assembling a set of problems (of similar difficulty), getting the benchmark times, and administering two random subsets to the students at two date intervals.

For Case U, the evaluation is rather basic, directed towards obtaining a first view of PlanS's performance, probably leading towards further studies. The test grid has only one proper run, since the second, benchmark, one is poorly controlled for differences of user set. From the NLP point of view the evaluation is one of a system offering NLP in use, but does not throw any specific light on the particular contribution made by the NL interface.

7.2 Case L

This evaluation is designed specifically to focus on the contribution made by the NL interface. The motivation is to establish whether NL interaction is best for PlanS, so the goal is to establish whether NL is better than the obvious competitor, a menu interface, The perspective here again is task oriented, with (we say) teachers both as interested and consuming parties; however the orientation is extrinsic, since the emphasis is on the interface's stimulus to student thinking, with menu prompting deemed to force a more comprehensive view of design. An experimental kind of evaluation is again appropriate, comparing NL and menu interfaces, again of black box type but, here, in exhaustive style and with direct comparison since there is no clear need for or obvious yardstick; however hybrid mode, with qualitative as well as quantitive criteria, seems called for.

Now filling out the design, the evaluation is concerned with the L-system's external function, contributing to the support of training; its context, however, is not just the rest of the c-system as a whole, but also the body of student users; the subject's constitution consists of the L-system's (or analogously the competing menu interface's) various components, e.g. parser, lexicon. The environment variables for this evaluation are therefore both the properties of design problems and students, as before, and those of the rest of the system, for instance the knowledge base and critiquing capabilities of the design sub-system. The system parameters are the interfaces' external communication elements: e.g. degree of freedom through sentences versus headed boxes, and linguistic resources in vocabulary and sentence types for NL or, for the menu, words or phrases used as slot labels or allowed as fillers. These are the relevant parameters, which we assume are supported by common other parameters like the same information transfer capacity with respect to the underlying design system.

The evaluation criteria, addressing effectiveness, are on the one hand plan quality and on the other interface utility to the user. The measure for the plans is quantitative, via ratings on a scale of three by competent judges, with the method averaging over all completed plans (perhaps supplemented by distribution data). The qualitative user view is obtained by questionnaire rating interfaces positive, neutral or negative for ease of expression for inputs, of clarity for outputs, and general appropriateness. The measures are relative distributions of the ratings for the features, and the method here is by self-completed questionnaire at session end. However the data gathering is complex. The evaluation presupposes a menu interface, which has be strictly comparable in concept coverage with the NL one and will therefore have to be specially provided, as part of the test enterprise as a whole. But it is evident that while given design problems may be the same, this is the only common element, and there is no concrete answer data in the form of target plans. The only strategy, given (we assume) the real students cannot be disrupted, is to train some other students up to the relevant architecture level, and then supply them with the real students' problems and one interface or the other, used over a period of time. The evaluation procedure is thus an elaborate and expensive one which depends on obtaining the menu interface, and then training the substitute students, giving them problems and assessing the resulting plans, and organising and processing the questionnaires.

The aim of this evaluation makes it very different from that for U and, with its controlled comparisons, very elaborate and costly. The test grid even so has only two runs: it needs filling out with checks on environment variables like student quality and training, problems set, plan criteria or system critiquing capabilities; and, because variables and parameters interact, parameter settings for e.g. inquiry flexibility or critique phrasing need testing. The parameter testing would move towards glass box interface evaluation. The evaluation described does no more than assess the NL interface as a whole: it does not analyse its behaviour in detail, or attribute this to underlying properties of its data resources or processors. That would require other, different evaluation with many runs, though it would still be very hard to link individual design characteristics of the interface with its perceived merits and demerits as a communication device. Other evaluation gauges also need consideration, e.g. design closure measured by number of dialogue turns.

These brief illustrations also show both how the detailed design choices in an evaluation, e.g. of data or measure, have to be clearly motivated from outside, and how the performance results for an evaluation subject have always to be interpreted in relation to the subject's environment. Even when environment variables are not explicitly manipulated, they need to be specified in order to flag their possible implications, for instance problem and student characteristics for PlanS.

8 Conclusion

As noted, while there have been individual evaluations of interest, and attacks on methodology (cf Thompson, 1992), the (D)ARPA/NIST evaluation initiatives are major efforts in terms of their goals, scale, and hard labour. Relating these to my analysis and methodology their major feature is that they are laboratory experiments, and as such are naturally distanced from detailed operational influences, Moreover the desire for control, to be achieved not only by blind testing but more materially via highly elaborated and polished answer data, further emphasises their detachment. This is particularly noticeable in the MUC and SLS (ATIS) cases, where the assessment data is not realistic or representative in any strong, or at any rate demonstrated, sense. There is an assumption that the nature of the evaluation data reflects real needs, and that the relative scores obtained correctly predict relative operational utility. While SLS assessment via logfiles refers more to systems in use, current MUC evaluation concerns with 'internal' NLP products e.g. predicate-argument structures, reflect researchers' interests in fine-grained explanatory evaluation concentrating on system parameters, properly viewed heuristically, not absolutely. These are legitimate interests, but there is not enough of the necessary complementary concern, even for the laboratory approach, with environment variables and their interaction with parameters and impact on performance. TREC has the advantage of more realistic (and also more simply specified) evaluation, but there are still concerns here about legitimacy and generalisation; these instructively include what operationally relevant inferences can be drawn from test results even when these use such intuitively plausible and universal measures as recall and precision.

One of the main requirements for future NLP evaluations is thus to approach these in a comprehensive as well as systematic way, so that the specific tests done are properly situated, especially in relation to the ends the evaluation subject is intended to serve, and the properties of the context in which it does this.

9 References

Galliers, J.R. and Sparck Jones, K. *Evaluating natural language processing systems*, TR 291, Computer Laboratory, University of Cambridge, 1993, (187pp). A gzipped copy of TR 291 (needing binary transfer) is available from the anonymous FTP server ftp.cl.cam.ac.uk, as the file TR291-ksj-jrg-evaluating-nl-systems.ps.gz in the directory reports.

HLT1: *Proceedings of the ARPA Workshop on Human Language Technology, 1993*, Morgan Kaufmann, San Mateo, CA.

MUC: *Proceedings of the Message Understanding Conferences, 1991; 1992; 1993*, Morgan Kaufmann, San Mateo, CA.

S&NLW: *Proceedings of the Speech and Natural Language Workshops, 1991; 1992*, Morgan Kaufmann, San Mateo, CA.

Thompson, H. (ed) *The strategic role of evaluation in natural language processing and speech technology; Workshop record*, HCRC, University of Edinburgh, 1992.

TREC: *Proceedings of the Text Retrieval Conferences, 1992; 1993*, NIST, Gaithersburg, MD.