Syntactic Analysis and Error Correction for Danish in the SCARRIE project

Patrizia Paggio
Center for Sprogteknologi
patrizia@cst.ku.dk

May 10, 2000

Abstract

This paper reports on work carried out at CST in Copenhagen to develop the Danish version of the SCARRIE prototype, addressing in particular the issue of how a form of shallow parsing is combined with error detection and correction to treat context-dependent spelling errors. The paper describes the corpora used to develop the system, and shows some preliminary evaluation results.

1 The SCARRIE project

SCARRIE was a EU-funded collaborative project, the purpose of which was to develop a high-quality proof-reading tool for the Scandinavian publishing industry. The consortium consisted of partners from Sweden, Norway, and Denmark[1]. The project terminated in the spring of 1999, and resulted in the development of three prototypes covering Norwegian, Swedish and Danish. Although the three prototypes all address the same basic issue, the functionality they provide differs slightly to reflect language specific needs as well as different research interests and expertise in the groups. In this paper, we shall deal with the Danish SCARRIE, and focus on its grammar checking component. For a description of the Norwegian prototype, see (de Smedt and Rosen, this volume).

SCARRIE builds on CORRie (Vosse 1992) (Vosse 1994), a proof-reading system originally developed for Dutch and distributed by Stichting Cognitieve Technologie (the Netherlands), who acted as a subcontractor in the project. The project has adapted the system to comply with the language specific features of the languages covered, and with the requirements of the project's end users. The system processes text

Proceedings of NODALIDA 1999, pages 152-161

in batch mode and produces an annotated output text where errors are flagged and replacements suggested where possible. Text correction is performed in two steps: first the system deals with spelling errors and typos resulting in invalid words, and then with grammar errors.

2 The Danish prototype

Localisation of the system to the Danish language has mainly consisted in the development of a set of lexical and grammar resources. These include first of all a dictionary of 251,000 domain-relevant word forms that have been derived from a collection of 68,000 newspaper articles. From the same text collection we have also extracted a separate list of 717 idioms. The list is used to identify multi-word expressions such as complex prepositions, or idioms including words that would be invalid in isolation (e.g. *carte blanche*). Both dictionary and idiom list were developed through a cooperation between CST and the Danish language and literature society (det Danske Sprog- og Litteraturselskab).

Another important component is the compound analysis grammar, a set of regular expressions covering the most common types of compound nominals in Danish. This is an important feature, as in Danish compounding is very productive, and compounds are written as single words.

Words which the system cannot find in the dictionary or the idiom list, or analyse as compound forms, or assign the label of proper name, are taken to be spelling errors. The system flags them as such and tries to suggest a replacement. The algorithm used is based on *trigram and triphone analysis* (van Berkel & de Smedt 1988), and takes into account the orthographic strings corresponding to the invalid word under consideration and its possible replacement, as well as the phonetic representations of the same two words. Phonetic representations are generated by a set of grapheme-to-phoneme rules (Hansen 1999) the aim of which is to assign phonetically motivated misspellings and their correct counterparts, identical or similar phonetic representations.

The last lingware component developed for the Danish prototype is the phrase structure grammar, which is used by the parser to identify context-dependent spelling errors (from hereafter grammar errors). Parsing results are passed on to a corrector to find replacements for the errors found. The parser is an implementation of the Tomita algorithm with a component for error recognition whose job is to keep track of error weights and feature mismatches as described in (Vosse 1991). Each input sentence is assigned the analysis with the lowest error weight. As we shall see in more detail below, the system can treat grammar errors of

two different kinds, i.e. feature mismatches and structural errors. In the case of a feature mismatch, the system tries to find the correct form of the misspelt word by overriding the offending feature, and by looking for an alternative word form in the dictionary. In the case of a structural error, on the other hand, specific error rules are applied to parse the incorrect input and an error message is generated.

3 The errors

To define the coverage of the system, the project has assembled corpora of parallel raw and proofread texts for the three languages involved. The Danish corpus consists of newspaper and magazine articles published in 1997 for a total of 270,805 running words. The articles have been collected in their raw version, as well as in the edited version provided by the publisher's own proofreaders. Although not very large in number of words, the corpus consists of excerpts from 450 different articles to ensure a good spread of lexical domains and error types. The corpus has been used to define the coverage of the grammar and to extract test data.

The errors occurring in the corpus have been analysed according to the taxonomy in (Rambell 1997). Figure 1 shows the distribution of the various error types into the five top-level categories of the taxonomy. As can be seen, grammar errors account for 30% of the errors. Of these, 70% fall into one of the following categories (Povlsen 1998):

- Too many finite verbal forms or missing finite verb
- Errors in nominal phrases:
  - agreement errors,
  - wrong determination,
  - genitive errors,
  - errors concerning pronouns;
- Split-ups and run-ons.

Another way of grouping the errors is by the kind of parsing failure they generate. As mentioned earlier, we can make a distinction between feature mismatches and structural errors. Agreement errors are typical examples of feature mismatches. In the following nominal phrase, for example:

(1) de *interessant projekter
(the interesting projects)

Error type	No.	%
Context independent errors	386	38
Context dependent errors	308	30
Punctuation problems	212	21
Style problems	89	9
Graphical problems	24	2
Total	1019	100

Figure 1: Error distribution in the Danish corpus

the error can be formalised as a mismatch between the definiteness of the determiner *de* (the) and the indefiniteness of the adjective *interessant* (interesting). Adjectives have in fact both an indefinite and a definite form in Danish.

The sentence below, on the other hand, is an example of structural error.

(2) i sin tid *skabet han skulpturer over atomkraften
(during his time wardrobe/created he sculptures about nuclear power)

Since the finite verb *skabte* (created) has been misspelt as *skabet* (the wardrobe), the syntactic structure corresponding to the sentence is missing a verbal head.

Run-ons and split-ups are structural errors of a particular kind, having to do with leaves in the syntactic tree. In some cases they can only be detected on the basis of the context, because the misspelt word has the wrong category or carries some other grammatical feature that is incorrect in the context. Although the system has a facility for identifying and correcting split-ups and run-ons based on a complex interaction between the dictionary, the idiom list, the compound grammar and the syntactic grammar, this facility has not been fully developed yet, and will therefore not be described any further here. More details can be found in (Paggio 1999).

The next section describes the way in which agreement errors in NPs and structural errors in verb groups are dealt with in the grammar, and explaines how the treatment of these errors fits in with the general analysis strategy adopted in the grammar.

4 The grammar

The grammar is expressed in an augmented context-free grammar formalism consisting of rewrite rules where symbols are associated with

features. It is also possible to add error weights to both rules and individual features, and to specify error messages. The rules are applied by unification, but in cases where one or more features of a given word do not unify with relevant features in a grammar rule, the offending features can be overridden.

Two kinds of rules may be used, "normal" rules describing the valid structures of the language, and "error" rules describing invalid structures. Thanks to the feature overriding mechanism, however, normal rules can also analyse sentences containing feature mismatch errors.

4.1 Feature mismatches

For example, consider the following rule, which is intended to account for definite nominal phrases:

```
NP(def Gender PersNumber) ->
        Det(def Gender PersNumber)
        AP(def _ _)
        N(indef Gender:2 PersNumber)
```

The rule is used to analyse NPs consisting of a definite determiner, an adjective phrase and a noun. Determiner and adjective bear the feature "def" for *definite*, whereas the noun bears the feature "indef" for *indefinite*. This reflects the fact that determiners, adjectives and nouns all inflect for definiteness in Danish, but the noun has to be in the indefinite form if preceded by a determiner. Furthermore, the three nodes must share values for *gender* and *person/number*, as indicated by the capitalised variables "Gender" and "PersNumber".

The rule will parse a correct definite NP such as:

(3) de interessante projekter
(the interesting projects)

but also

(4) de *interessant projekter
(5) de interessante *projekterne

Both (4) and (5) violate the definiteness constraints, since the adjective is indefinite in (4), and the noun is definite in (5). The feature overriding mechanism makes it possible for the system to suggest *interessante* as the correct replacement in the former case, and *projekter* in the latter. What happens is that the parser selects the rule as applicable because the syntactic backbone matches the input, but detects a violation in one of the features. It then overrides the violating feature on the incorrect word and looks for a suitable replacement by searching

for alternative forms of the same lemma in the dictionary. The resulting analysis carries an error weight generated by the overriding operation.

Weights are used to control rule interaction as well as to establish priorities among features that may have to be overridden. For example in our NP rule, a weight has been attached to the Gender feature in the N node. The weight expresses the fact that it costs more to override gender on the head noun than on the determiner or adjective. The reason is that if there is a gender mismatch, the parser should not try to find an alternative form of the noun (which does not exist), but rather override the gender feature either on the adjective or the determiner.

4.2 Structural errors

Error rules are very similar to normal rules, the only difference being that they have to be associated with an error weight and an error message.

The purpose of the weight is to ensure that error rules are applied only if normal rules are not applicable. Error messages serve two different purposes. If they are preceded by a question mark, they only appear in the log file for the developer to trace the analysis process. Otherwise, they are shown to the end user when the rule they are associated with has been applied. In other words if an error rule is applied to analyse a sentence, the system will not look for a replacement, but present the user with an error message indicating the kind of grammatical failure that has been observed.

The following is an error rule example.

```
VGroup(_ finite Tense) ->
       V(_ finite:4 Tense)
       V(_ finite:4 _)
       "Sequence of two finite verbs":4
```

A weight of 4 is attached to the rule as a whole, but there are also weights attached to the "finiteness" feature on the daughters: their function is to make it costly for the system to apply the rule to non-finite forms. In other words, the feature specification "finite" is made difficult to override to ensure that it is indeed a sequence of *finite* verbal forms the rule applies to and flags.

The rule will for example parse the verbal sequence in the following sentence:

(6) Jeg vil *bevarer (bevare) min frihed.
(*I want keep my freedom)

As a result of parsing, the system in this case will not attempt to correct the wrong verbal form, but issue the error message "Sequence of two finite verbs".

In many cases, in fact, it may be quite difficult to suggest a correction for a wrong verb sequence, since there may be several reasonable ways of amending it, some of which imply more than just replacing one form with another.

To sum up, error rules can be used to describe an error explicitly and to issue error messages. However, so far we have made very limited use of them, as controlling their interaction with normal rules and with the feature overriding mechanism is not entirely easy. To this sparse use of error rules corresponds, on the other hand, an extensive exploitation of the feature overriding mechanism. This strategy allows us to keep the number of rules in the grammar relatively low, but relies on a careful manual adjustment of the weights attached to the various features in the rules.

4.3 Shallow analysis

In the current version of the grammar, only the structures relevant to the error types we wanted the system to deal with – nominal phrases and verbal groups – are accounted for in detail. The analysis produced is thus a kind of shallow syntactic analysis where the various sentence constituents are attached under the topmost S node as fragments. This choice was made for two reasons. Firstly, we wanted the system to target the error types represented in our corpus to tailor its functionality to the needs of our end users. Secondly, we did not want to impair the system's efficiency by striving for too complex a model of syntactic analysis.

Below, we show the rules (the features attached to the various categories have been removed here for the sake of exposition) implementing the fragment strategy just mentioned:

```
S -> Fragments VGroup Fragments
S -> Fragments VGroup
S -> InvVGroup Fragments
Fragments -> Fragment
Fragments -> Fragment Fragments
Fragment -> NP
Fragment -> PP
Fragment -> AdvP
Fragment -> ...
```

As can be seen, a sentence is built up of a verb group possibly preceded and followed by one or more fragments, in turn analysed as either NPs, PPs and so on. The internal structure of verb groups - including possible wrong structures - is specified in a number of rules rewrit-

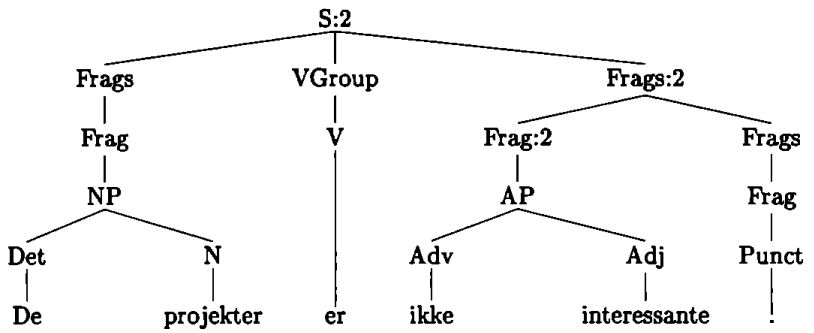

Figure 2: A parse tree

ing VGroup. InvVGroup in the third sentence rule above, accounts for subject-main verb inversion in interrogative sentences.

There are cases, of course, in which attaching a constituent directly under the S node does not enable the system to spot an error for which we would expect a flag. Adjective phrases are an example. Since agreement errors in nominal phrases as we saw are rather frequent in the SCARRIE database, we wanted the system to be able to identify and correct them. Therefore, APs can also be analysed as nominal modifiers by the NP rules. To indicate then that the fragment analysis is not optimal (it should only be resorted to when the adjective is not part of a nominal phrase), it is associated with an error weight, as well as a system-internal error message (invisible to the end user):

```
Fragment -> AP "?Fragment AP rule":2
```

The weight penalises parse trees built by applying the rule. However, in lack of a better solution, the rule is triggered e.g. to analyse the AP in the following sentence:

(7) De projekter er ikke interessante.
(Those projects are not interesting)

The parse tree produced by the system is shown in Figure 2. Note that the error weight introduced by the Fragment AP rule is percolated up to the top S node.

5 Evaluation and Conclusion

The evaluation methodology adopted in the project capitalises on the fact that we had access to a set of parallel unedited and proofread texts

(see (Paggio & Music 1998)). This made it possible to develop a tool that compares the results obtained by the system with the corrections suggested by the publisher's human proofreaders. The tool derives *recall* measures (lexical coverage as well as coverage of errors), a *precision* measure (percentage of correct flaggings), as well as *suggestion adequacy* measures (hits, misses and no suggestions offered). The same automatic procedure was used to evaluate the system during development, and to validate it at the user site. Testing was done on constructed test suites displaying examples of the errors targeted in the project and with text excerpts from the parallel corpora.

The results obtained on the test suites are very positive, especially with regard to the treatment of grammar errors. More extensive testing (see (Paggio to appear) for more details) has shown that when the system is run on a text, error coverage decreases especially because of punctuation and other stylistic matters not treated in the project. There are also, however, agreement errors which go unnoticed, mainly due to the imprecision introduced by the fragment analysis approach. A large number of the false flags produced is due to the grammar's limited coverage. In particular, genitive phrases, which are not treated at the moment, are often the cause of wrong NP analyses.

Considering the fact that relatively little time was spent on grammar development in the project's lifetime, we consider the results obtained encouraging. There is, however, space for improvement, especially with regard to extending the coverage of the grammar.

Notes

[1]Main contractors were: WordFinder Software AB (Sweden), Center for Sprogteknologi (Denmark), Department of Linguistics at Uppsala University (Sweden), Institutt for lingvistikk og litteraturvitenskab at the University of Bergen (Norway), and Svenska Dagbladet (Sweden). A number of subcontractors also contributed to the project. Subcontractors in Denmark: Munksgaard International Publishers, Berlingske Tidende, Det Danske Sprog- og Litteraturselskab, and Institut for Almen og Anvendt Sprogvidenskab at the University of Copenhagen (Denmark).

References

Hansen, P. M. (1999). Grapheme-to-phoneme rules for the Danish component of the SCARRIE project. H. E. Thomsen & S. Kirchmeier-Andersen, eds, *Datalingvistisk Forenings årsmøde 1998 i Købehavn, Proceedings*, number 25 in 'LAMBDA'. Institut for datalingvistik, Handelshøjskolen i København, 79–91.

Paggio, P. (1999). Treatment of grammatical errors and evaluation in SCARRIE. H. E. Thomsen & S. Kirchmeier-Andersen, eds, *Datalingvistisk Forenings årsmøde 1998 i København, Proceedings*, number 25 in 'LAMBDA'. Institut for datalingvistik, Handelshøjskolen i København, 65–78.

Paggio, P. (to appear). Danish grammar checking in SCARRIE. *Proceedings of ANLP 2000.* Seattle, Washington.

Paggio, P. & Music, B. (1998). Evaluation in the SCARRIE project. *Proceedings of the First International Conference on Language Resources & Evaluation.* Granada, Spain, 277–282.

Povlsen, C. (1998). Three types of grammatical errors in Danish. Technical report. Copenhagen: Center for Sprogteknologi.

Rambell, O. (1997). Error typology for automatic proof-reading purposes. Technical report. Uppsala: Uppsala University.

van Berkel, B. & de Smedt, K. (1988). Triphone analysis: a combined method for the correction of orthographical and typographical errors. *Proceedings of the 2nd conference on Applied Natural Language Processing.* ACL, Austin, 77–83.

Vosse, T. (1991). Detection and correction of morpho-syntactic errors in shift-reduce parsing. R. Heemels, A. Nijholt & K. Sikkel, eds, 'Tomita's Algorithm: Extensions and Applications', number 91-68 in *Memoranda Informatica.* University of Twente, 69–78.

Vosse, T. (1992). Detecting and correcting morpho-syntactic errors in real texts. *Proceedings of the Third Conference on Applied Natural Language Processing.* Trento, Italy, 111–118.

Vosse, T. G. (1994). The Word Connection - Grammar-based Spelling Error Correction in Dutch. PhD thesis. Rijksuniversiteit at Leiden, the Netherlands. ISBN 90-75296-01-0.