Spelling and Grammar Correction for Danish in SCARRIE

Patrizia Paggio
Center for Sprogteknologi
Copenhagen (DK)
patrizia@cst.ku.dk

Abstract

This paper reports on work carried out to develop a spelling and grammar corrector for Danish, addressing in particular the issue of how a form of shallow parsing is combined with error detection and correction for the treatment of context-dependent spelling errors. The syntactic grammar for Danish used by the system has been developed with the aim of dealing with the most frequent error types found in a parallel corpus of unedited and proofread texts specifically collected by the project's end users. By focussing on certain grammatical constructions and certain error types, it has been possible to exploit the linguistic 'intelligence' provided by syntactic parsing and yet keep the system robust and efficient. The system described is thus superior to other existing spelling checkers for Danish in its ability to deal with context-dependent errors.

1 Introduction

In her much-quoted and still relevant review of technologies for automatic word correction (Kukich, 1992), Kukich observes that "research in context-dependent spelling correction is in its infancy" (p. 429), and that the task of treating context-dependent errors is still an elusive one due to the complexity of the linguistic knowledge often necessary to analyse the context in sufficient depth to find and correct such errors. But progress in parsing technology and the growing speed of computers seem to have made the task less of a chimera. The '90s have in fact seen a renewed interest in grammar checking, and proposals have been made for systems covering English (Bernth, 1997) and other languages such as Italian (Bolioli et al., 1992), Spanish and Greek (Bustamante and Léon, 1996), Czech (Holan et al., 1997) and Swedish (Hein, 1998).

This paper describes the prototype of a spelling and grammar corrector for Danish which combines traditional spelling checking functionalities with the ability to carry out compound analysis and to detect and correct certain types of context-dependent spelling errors (hereafter simply "grammar errors"). Grammar correction is carried out by parsing the text, making use of feature overriding and error weights to accommodate the errors. Although a full parse of each sentence is attempted, the grammar has been developed with the aim of dealing only with the most frequent error types found in a parallel corpus of unedited and proofread texts specifically collected by the project's end users. By focussing on certain grammatical constructions and certain error types, it has been possible to exploit the linguistic 'intelligence' provided by syntactic parsing and yet keep the system robust and efficient. The system described is thus superior to other existing spelling checkers for Danish in its ability to deal with certain types of grammar errors.

We begin by giving an overview of the system's components in Section 2. In Section 3 we describe the error types we want to deal with. Section 4 gives an overview of the grammar: in particular, the methods adopted for treating feature mismatches and structural errors are explained. Finally, in Section 5 evaluation results are presented and a conclusion is drawn.

2 The prototype

The prototype is a system for high-quality proofreading for Danish which has been developed in the context of a collaborative EU-project[1]. Together with the Danish prototype,

[1]Main contractors in the consortium were: WordFinder Software AB (Sweden), Center for

the project has also produced similar systems for Swedish and Norwegian, all of them tailored to meet the specific needs of the Scandinavian publishing industry. They all provide writing support in the form of word and grammar checking.

The Danish version of the system[2] constitutes a further development of the CORRie prototype (Vosse, 1992) (Vosse, 1994), adapted to deal with the Danish language, and to the needs of the project's end users. The system processes text 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.

Invalid words are identified on the basis of dictionary lookup. The dictionary presently consists of 251,000 domain-relevant word forms extracted from a collection of 68,000 newspaper articles. A separate idiom list allowing for the identification of multi-word expressions is also available. Among the words not found in the dictionary or the idiom list, those occurring most frequently in the text (where frequency is assessed relative to the length of the text) are taken to be new words or proper names[3]. The remaining unknown words are passed on to the compound analysis grammar, which is 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 still unknown at this point 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 and 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.

Then the system tries to identify context-dependent spelling errors. This is done by parsing the text. 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. If the error is due to a feature mismatch, the offending feature is overridden, and if a dictionary entry satisfying the grammar constraints expressed by the context is found in the dictionary, it is offered as a replacement. If the structure is incomplete, on the other hand, an error message is generated. Finally, if the system identifies an error as a split-up or a run-on, it will suggest either a possible concatenation, or a sequence of valid words into which the misspelt word can be split up.

3 The errors

To ensure the coverage of relevant error types, a set of parallel unedited and proofread texts provided by the Danish end users has been collected. This text collection 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 construct test suites for progress evaluation, and also to guide grammar development. The aim set 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 were: Munksgaard International Publishers, Berlingske Tidende, Det Danske Sprog- og Litteraturselskab, and Institut for Almen og Anvendt Sprogvidenskab at the University of Copenhagen.

[2]In addition to the author of the present paper, the Danish SCARRIE team at CST consisted of Claus Povlsen, Bart Kongejan and Bradley Music.

[3]The system also checks whether a closely matching alternative can be found in the dictionary, to avoid mistaking a consistently misspelt word for a new word.

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

grammar development was then to enable the system to identify and analyse the grammatical constructions in which errors typically occur, whilst to some extent disregarding the remainder of the text.

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: they can then be viewed as either feature mismatches, or as structural errors. Agreement errors are typical examples of feature mismatches. In the following nominal phrase, for example:

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

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 bears some other grammatical feature that is incorrect in the context. Examples are given in (3) and (4) below, which like the preceding examples are taken from the project's corpus. In both cases, the error would be a valid word in a different context. More specifically, *rigtignok* (indeed) is an adverb, whilst *rigtig nok* (actually correct) is a modified adjective; and *inden for* (inside) is a preposition, whilst *indenfor* (indoors) is an adverb. In both examples the correct alternative is indicated in parentheses.

(3) ... studerede min gruppe *rigtig nok (rigtignok) under temaoverskrifter
(studied my group indeed on the basis of topic headings)

(4) *indenfor (inden for) de gule mure
(inside the yellow walls)

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).

4 The grammar

The grammar is an augmented context-free grammar consisting of rewrite rules where symbols are associated with features. Error weights and error messages can also be attached to either rules or single features. The rules are applied by unification, but in cases where one or more features do not unify, the offending features will be overridden.

In the current version of the grammar, only the structures relevant to the error types we want the system to deal with – in other words 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.

For example, adjective phrases can be analysed as fragments, as shown in the following rule:

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

To indicate that the fragment analysis is not optimal, it is associated with an error weight, as well as an error message to be used for debugging purposes (the message is not visible to the end user). The weight penalises parse trees built by applying the rule. The rule is used e.g. to analyse an AP following a copula verb as in:

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

The main motivation for implementing a grammar based on the idea of fragments was efficiency. Furthermore, the fragment strategy could be implemented very quickly. However, as will be clarified in Section 5, this strategy is sometimes responsible for bad flags.

4.1 Feature mismatches

As an alternative to the fragment analysis, APs can be attached as daughters in NPs. This is of course necessary for the treatment of agreement in NPs, one of the error types targeted in our application. This is shown in the following rule:

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

The rule will parse a correct definite NP such as:

(6) de interessante projekter
(the interesting projects)

but also

(7) de *interessant projekter
(8) de interessante *projekterne

The feature overriding mechanism makes it possible for the system to suggest *interessante* as the correct replacement in (7), and *projekter* in (8). Let us see how this is done in more detail for example (7). The parser tries to apply the NP rule to the input string. The rule states that the adjective phrase must be definite (AP (def _ _)). But the dictionary entry corresponding to *interessant* bears the feature 'indef'. The parser will override this feature and build an NP according to the constraints expressed by the rule. At this point, a new dictionary lookup is performed, and the definite form of the adjective can be suggested as a replacement.

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 rationale behind this is the fact 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 if necessary override the gender feature either on the adjective or the determiner.

4.2 Capturing structural errors in grammar rules

To capture structural errors, the formalism allows the grammar writer to write so-called error rules. The syntax of error rules is very similar to that used in 'normal' rules, the only difference being that an error rule must have an error weight and an error message attached to it. The purpose of the weight is to ensure that error rules are applied only if 'normal' rules are not applicable. The error message can serve two purposes. Depending on whether it is stated as an implicit or an explicit message (i.e. whether it is preceded by a question mark or not), it will appear in the log file where it can be used for debugging purposes, or in the output text as a message to the end user.

The following is an error rule example.

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

Error type	*Total*	*Flagged*		*Hits*		*Misses*		*No suggs*	
Errors in single words	81	69	(85.2%)	35	(50.8%)	11	(15.9%)	23	(33.3%)
Errors in compounds	42	28	(66.7%)	8	(28.6%)	8	(28.6%)	12	(42.8%)
NP agreement errors	18	15	(83.3%)	15	(100.0%)	0	(0.0%)	0	(0.0%)
VP errors	13	8	(61.6%)	8	(100.0%)	0	(0.0%)	0	(0.0%)
Total	154	120	(78.0%)	66	(55.0%)	19	(15.8%)	35	(29.2%)

Figure 2: Error coverage and suggestion adequacy evaluated on test suites

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:

(9) 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".

Error rules can thus be used to explicitly describe an error 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. In fact, they are consistently used only to identify incorrect sequences of finite verbal forms or sentences missing a finite verb. 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.

5 Evaluation and Conclusion

The project's access to a set of parallel unedited and proofread texts has made it possible to automate the evaluation of the system's linguistic functionality. A tool has been implemented to compare the results obtained by the system with the corrections suggested by the publisher's human proofreaders in order to derive measures telling us how well the system performed on *recall* (lexical coverage as well as coverage of errors), *precision* (percentage of correct flaggings), as well as *suggestion adequacy* (hits, misses and no suggestions offered). The reader is referred to (Paggio and Music, 1998) for more details on the evaluation methodology. The automatic procedure was used to evaluate the system during development, and in connection with the user validation. Testing was done on constructed test suites displaying examples of the errors targeted in the project and with text excerpts from the parallel corpora.

Figure 2 shows error recall and suggestion adequacy figures for the various error types represented in the test suites. These figures are very positive, especially with regard to the treatment of grammar errors. To make a comparison with a commercial product, the Danish version of the spelling and grammar checker provided by Microsoft Word does not flag any of the grammar errors.

Figure 3 shows how the system performed on one of the test corpora. The corpus was assembled by mixing short excerpts containing relevant grammar errors and randomly chosen text. Since unlike test suites, the corpus also contains correct text, the figure this time also shows lexical coverage and precision figures. The corpus consists of 278 sentences, with an average length of 15.18 words per sentence. It may be surprising to see that it contains a limited number of errors, but it must be remembered that the texts targeted in the project are written by experienced journalists.

The corpus was processed in 58 cpu-seconds on an HP 9000/B160. As expected, the system performs less well than on the test suites, and in general precision is clearly too low. However, we still consider these results encouraging given the relatively small resources the project has been able to spend on grammar development, and we

Recall
4220 total words
4157 valid words
3996 (96.1%) valid words accepted
161 (3.9%) valid words rejected
63 invalid words (real errors)
36 (57.1%) real errors spotted (good flags)
27 (42.9%) real errors missed
Precision
175 flaggings
36 (20.6%) good flags
139 (79.4%) bad flags (false positives)
Suggestion adequacy
36 good flags
15 (41.7%) hits (initial suggestion correct)
8 (22.2%) misses (suggestions offered, none correct)
13 (36.1%) with no suggestions offered

Figure 3: Test corpus evaluation

believe they can be improved.

We regard error coverage as quite satisfactory for a research prototype. In a comparative test made on a similar (slightly smaller) corpus, SCARRIE obtained 58.1% error coverage, and Word 53.5%. To quote a figure from another recently published test (Martins et al., 1998), the ReGra system is reported to miss 48.1% real errors. It is worth noting that ReGra has much more extensive linguistic resources available than SCARRIE, i.e. a dictionary of 1.5 million words and a grammar of 600 production rules. Most of the errors not found by SCARRIE in the test have to do with punctuation and other stylistic matters not treated in the project. There are also, however, agreement errors which go unnoticed. These failures are due to one of two reasons: either that no parse has been produced for the sentence in question, or that the grammar has produced a wrong analysis.

The precision obtained is at least at first sight much too low. On the same test corpus, however, Word only reached 15.9% precision. On closer inspection, 72 of the bad flags produced by SCARRIE turned out to be due to unrecognised proper names. Disregarding those, precision goes up to 34.9%. As was mentioned early, SCARRIE has a facility for guessing unknown proper names on the basis of their frequency of occurrence in the text. But since the test corpus consists of short unrelated excerpts, a large number of proper names only occur once or twice. To get an impression of how the system would perform in a situation where the same proper names and unknown words had a higher frequency of occurrence, we doubled the test corpus by simply repeating the same text twice. As expected, precision increased. The system produced 178 flags, 60 of which were correct (39.7%). This compares well with the 40% precision reported for instance for ReGra.

In addition to the problem of unkown proper names, false flags are related to unrecognised acronyms and compounds (typically forms containing acronyms or dashes), and a not very precise treatment of capitalisation. Only 13 false flags are due to wrong grammar analyses caused either by the fragment approach or by the grammar's limited coverage. In particular, genitive phrases, which are not treated at the moment, are responsible for most of these false alarms.

In conclusion, we consider the results obtained so far promising, and the problems revealed by the evaluation tractable within the current system design. In particular, future development should focus on treating stylistic matters such as capitalisation and punctuation which have not been in focus in the current prototype. The coverage of the grammar, in particular the treatment of genitive phrases, should also be further developed. The data pro-

vided by the evaluation reported on in this paper, however, are much too limited to base further development on. Therefore, more extensive testing and debugging should also be carried out.

In addition, two aspects of the system that have only be touched on in this paper would be worth further attention: one is the mechanism for the treatment of split-ups and run-ons, which as mentioned earlier is not well-integrated at the moment; the other is the weight adjustment process, which is done manually at the moment, and for which the adoption of a semi-automatic tool could be considered.

References

A. Bernth. 1997. EasyEnglish: a tool for improving document quality. In *Proceedings of the Fifth Conference on Applied Natural Language Processing.*

A. Bolioli, L. Dini, and G. Malnati. 1992. JDII: Parsing Italian with a robust constraint grammar. In *Proceedings of COLING-92*, pages 1003–1007.

Flora Ramírez Bustamante and Fernando Sánchez Léon. 1996. GramCheck: A grammar and style checker. In *Proceedings of COLING-96*, pages 175–181, Copenhagen, Denmark.

Peter Molbæk Hansen. 1999. Grapheme-to-phoneme rules for the Danish component of the SCARRIE project. In Hanne E. Thomsen and Sabine Kirchmeier-Andersen, editors, *Datalingvistisk Forenings årsmøde 1998 i København, Proceedings*, number 25 in LAMBDA, pages 79–91. Institut for datalingvistik, Handelshøjskolen i København.

Anna Sågvall Hein. 1998. A chart-based framework for grammar checking: Initial studies. In *Proceedings of Nodalida-98.*

Tomáš Holan, Vladislav Kuboň, and Martin Plátek. 1997. A prototype of a grammar checker for Czech. In *Proceedings of ANLP'97.*

Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439.

Ronaldo Teixeira Martins, Ricardo Hasegawa, Maria Volpe Nunes, Gisele Monthila, and Osvaldo Novais De Oliveira Jr. 1998. Linguistic issues in the development ofReGra: a grammar checker for Brazilian Portuguese. *Natural Language Engineering*, 4(4):287–307, December.

Patrizia Paggio and Bradley Music. 1998. Evaluation in the SCARRIE project. In *Proceedings of the First International Conference on Language Resources & Evaluation*, pages 277–282. Granada, Spain.

Patrizia Paggio. 1999. Treatment of grammatical errors and evaluation in SCARRIE. In Hanne E. Thomsen and Sabine Kirchmeier-Andersen, editors, *Datalingvistisk Forenings årsmøde 1998 i København, Proceedings*, number 25 in LAMBDA, pages 65–78. Institut for datalingvistik, Handelshøjskolen i København.

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

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

Brigitte van Berkel and Koenraad De Smedt. 1988. Triphone analysis: a combined method for the correction of orthographical and typographical errors. In *Proceedings of the 2nd conference on Applied Natural Language Processing*, pages 77–83. ACL, Austin.

Theo Vosse. 1991. Detection and correction of morpho-syntactic errors in shift-reduce parsing. In R. Heemels, A. Nijholt, and K. Sikkel, editors, *Tomita's Algorithm: Extensions and Applications*, number 91-68 in Memoranda Informatica, pages 69–78. University of Twente.

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

Theo G. Vosse. 1994. *The Word Connection - Grammar-based Spelling Error Correction in Dutch.* Ph.D. thesis, Rijksuniversiteit at Leiden: the Netherlands. ISBN 90-75296-01-0.