Developing a new grammar checker for English as a second language

Cornelia Tschichold, Franck Bodmer, Etienne Cornu, François Grosjean,
Lysiane Grosjean, Natalie Kübler, Nicolas Léwy & Corinne Tschumi

Laboratoire de traitement du langage et de la parole
Université de Neuchâtel, Avenue du Premier-Mars 26
CH - 2000 Neuchâtel

Abstract

In this paper we describe the prototype of a new grammar checker specifically geared to the needs of French speakers writing in English. Most commercial grammar checkers on the market today are meant to be used by native speakers of a language who have good intuitions about their own language competence. Non-native speakers of a language, however, have different intuitions and are very easily confused by false alarms, i.e. error messages given by the grammar checker when there is in fact no error in the text. In our project aimed at developing a complete writing tool for the non-native speaker, we concentrated on building a grammar checker that keeps the rate of over-flagging down and on developing a user-friendly writing environment which contains, among other things, a series of on-line helps. The grammar checking component, which is the focus of this paper, uses island processing (or chunking) rather than a full parse. This approach is both rapid and appropriate when a text contains many errors. We explain how we use automata to identify multi-word units, detect errors (which we first isolated in a corpus of errors) and interact with the user. We end with a short evaluation of our prototype and compare it to three currently available commercial grammar checkers.

1 Introduction

Many word-processing systems today include grammar checkers which can be used to locate various grammatical problems in a text. These tools are clearly aimed at native speakers even if they can be of some help to non-native speakers as well. However, non-native speakers make more errors than native speakers and their errors are quite different (Corder 1981). Because of this, they require grammar checkers designed for their specific needs (Granger & Meunier 1994).

The prototype we have developed is aimed at French native speakers writing in English. From the very start, we worked on the idea that our prototype would be commercialized. In order to find out users' real needs, we first conducted a survey among potential users and experts in the field concerning such issues as coverage and the interface. In addition, we studied which errors needed to be dealt with. To do this, we integrated the information on errors found in published lists of typical learner errors, e.g. Fitikides (1963), with our own corpus of errors obtained from English texts written by French native speakers. Some 27'000 words of text produced over 2'800 errors, which were classified and sorted. We have used this corpus to decide which errors to concentrate on and to evaluate the correction procedures developed. The following two tables give the percentages of errors found in our corpus, broken down by the major categories, followed by the subcategories pertaining to the verb.

Error type	percentage
spelling & pronunciation	10.3 %
adjectives	5.3 %
adverbs	4.4 %
nouns	19.6 %
verbs	24.5 %
word combinations	8.3 %
sentence	27.6 %

Table 1: Percentage of errors by major categories

Error type	percentage
morphology	1.2 %
agreement	1.4 %
tenses	8.8 %
lexicon	8.0 %
phrase following the verb	5.1 %

Table 2: Percentage of verb errors by subcategories

The prototype includes a set of writing aids, a problem word highlighter and a grammar checker. The writing aids include two monolingual and a bilingual dictionary (simulated), a verb conjugator, a small translating module for certain fixed expressions, and a comprehensive on-line grammar. The problem word highlighter is used to show all the potential lexical errors in a text. While we hoped that the grammar checker would cover as many different types of errors as possible, it quickly became clear that certain errors could not be handled satisfactorily with the checker, e.g. using *library* (based on French *librairie*) instead of *bookstore* in a sentence like *I need to go to the library in order to buy a present for my father*. Instead of flagging every instance of *library* in a text, something other grammar checkers often do, we developed the problem word highlighter. It allows the user to view all the problematic words at one glance and it offers help, i.e. explanations and examples for each word, on request. Potential errors such as false friends, confusions, foreign loans, etc. are tackled by the highlighter. The heart of the prototype is the grammar checker, which we describe below. Further details about the writing aids can be found in Tschumi et al. (1996).

2 The grammar checker

Texts written by non-native speakers are more difficult to parse than texts written by native speakers because of the number and types of errors they contain. It is doubtful whether a complete parse of a sentence containing these kinds of errors can be achieved with today's technology (Thurmair 1990). To get around this, we chose an approach using island processing. Such a method, similar to the chunking approach described in Abney (1991), makes it possible to extract from the text most of the information needed to detect errors without wasting time and resources on trying to parse an ungrammatical sentence fully. Chunking can be seen as an intermediary step between tagging and parsing, but it can also be used to get around the problem of a full parse when dealing with ill-formed text.

Once the grammar checker has been actived, the text which is to be checked goes through a number of stages. It is first segmented into sentences and words. The individual words are then looked up in the dictionary, which is an extract of CELEX (see Burnage 1990). It includes all the words that occur in our corpus of texts, with all their possible readings. For example, the word *the* is listed in our dictionary as a determiner and as an adverb; *table* has an entry both as a noun and as a verb. In this sense, our dictionary is not a simplified and scaled-down version of a full dictionary, but simply a shorter version. In the next stage, an algorithm based on neural networks (see Bodmer 1994) disambiguates all words which belong to more than one syntactic category. Furthermore, some multi-word units are identified and labeled with a single syntactic category, e.g. *a lot of* (PRON)[1]. After this stage, island parsing can begin. A first step consists in identifying simple noun phrases. On the basis of these NPs, preprocessing automata assemble complex noun phrases and assign features to them whenever possible. In a second step, other preprocessing automata identify the verb group and assign tense, voice and aspect features to it. Finally, in a third step, error detection automata are run, some of which involve interacting with the user. Each of these three steps will be described below.

3 The preprocessing stage

The noun phrase parser identifies simple non-recursive noun phrases such as Det+Adj+N or N+N. The method used for this process involves an algorithm of the type described in Church (1988) which was trained on a manually marked part of our corpus. The module is thus geared to the particular type of second language text the checker needs to deal with. The resulting information is passed on to a preprocessing module consisting of a number of automata groups. The automata used here (as well as in subsequent modules) are finite-state automata similar to those described in Allen (1987) or Silberztein (1993). This type of automata is well-known for its efficiency and versatility.

In the preprocessing module, a first set of automata scan the text for noun phrases, identify the head of each NP and assign the features for person and number to it. Other sets of automata then mark temporal noun phrases, e.g. *this past week* or *six months ago*. In a similar fashion, some prepositional phrases are given specific features if they denote time or place, e.g. *during that period, at the office*. Still within the same preprocessing module, some recursive NPs are then assembled into more complex NPs, e.g. *the arrival of the first group*. Finally, human NPs are identified and given a special feature. This is illustrated in the following automaton:

Automaton 1

```
<NP @[CAT = N, +HUMAN]
    (NP_TYPE => NP_HUM) >
```

Every automaton starts by looking for its anchor, an arc marked by "@" (not necessarily the first arc in the automaton). The above automaton first looks for a noun (inside a noun phrase) which occurs in the list of nouns denoting human beings. If it finds one, it puts the value NP_HUMAN in the register called NP_TYPE, a register which is associated with the NP as a whole.

The second group of preprocessing automata deals with the verb group. Periphrastic verb forms are analyzed for tense, aspect, and phase, and the result stored in registers. The content of these registers can later be called upon by the detection automata. After these two preprocessing stages, the error detection automata can begin their work.

4 Error detection

Once important islands in the sentence have been identified, it becomes much easier to write detection automata which look for specific types of errors. Because no overall parse is attempted, error detection has to rely on well-described contexts. Such an approach, which reduces overflagging, also has the advantage of describing errors precisely. Errors can thus not only be identified but can also be explained to the user. Suggestions can also be made as to how errors can be corrected.

One of the detection automata that make use of the NPs which have been previously identified is the automaton used for certain cases of subject-noun agreement (e.g. **He never eat cucumber sandwiches*):

[1] This part of speech corresponds to CELEX's use of 'PRON'.

Automaton 2

```
<NP (PERS_NP = P3,
    NBR_NP = SING)>
[CAT = ADV]*1
@[CAT = V, TNS = PRES, MORPH ≠ P3]
```

This automaton (simplified here) first looks for the content of its last arc, a verb which is not in the third person singular form of the present tense. From there it proceeds leftwards. The arc before the anchoring arc optionally matches an adverb. The next arc, the second from the top, matches an NP which has been found to have the features third person singular. If the whole of this detection automaton succeeds, a message appears which suggests that the verb be changed to the third person singular form.

5 User interaction

It is sometimes possible to detect an error without being completely sure about its identity or how to correct it. To deal with some of these cases, we have developed an interactive mode where the user is asked for additional information on the problem in question. The first step is to detect the structure where there might be an error. Here again, the preprocessing automata are put to use. For example, the following automaton looks for the erroneous order of constituents in a sentence containing a transitive verb (as in **We saw on the planet all the little Martians*).

Automaton 3

```
@ [CAT = V, +TRANS]
<PP (PP_TYPE = PP_TIME |
    PP_TYPE = PP_PLACE)>
<NP (NP_TYPE ≠ NP_TIME)>
```

If this automaton finds a sequence that contains a transitive verb followed by a prepositional phrase with either the feature PP_TIME or PP_PLACE and that ends in a noun phrase which does not have the feature NP_TIME, then the following question is put to the user:

"La séquence "all the little Martians" est-elle l'objet direct du verbe "saw"?"

("Is the sequence "all the little Martians" the direct object of the verb "saw"?")

If the user presses on the Yes-button, a reordering of the predicate is suggested so as to give V - NP - PP (this can be done automatically). If the No-button is pressed, a thank-you message appears instead and the checker moves on to the next error.

Interaction is also used in cases where one of two possible corrections needs to be chosen. If a French native speaker writes **the more great*, it is not clear whether s/he intended to use a comparative or a superlative. This can be determined by interacting with the user and an appropriate correction can then be proposed if need be.

While developing the automata which include interaction, we took great care not to ask too much of the user. So far we have used less than ten different question patterns and the vocabulary has been restricted to terminology familiar to potential users. In addition we only ask one question per interaction.

Interacting with the user can thus be a valuable device during error detection if it is used with caution. Too many interactions, especially if they do not lead to actual error correction, can be annoying. They should therefore be restricted to cases where there is a fair chance of detecting an error and should not be used to flag problematic lexical items such as all the tokens of the word *library* in a text.

6 Evaluation

In a first evaluation using the approach described in Tschichold (1994), we compared our prototype to three commercial grammar checkers: Correct Grammar for Macintosh (version 2.0 of the monolingual English grammar checker developed by Lifetree Software and Houghton Mifflin), Grammatik for Macintosh (the English for

Measure	Prototype	Correct Grammar	Grammatik	WinProof
Error detection	14.5%	10.5%	12.5%	34%
Error overflagging	43.5%	74%	75%	76%

Table 3. Error detection and error overflagging scores for the prototype and three commercial grammar checkers

French users 4.2 version of the Reference Software grammar checker) and WinProof for PC (version 4.0 of Lexpertise Linguistic Software's English grammar checker for French speakers). While the overall percentage of correctly detected errors is still rather low for all of the tested checkers, Table 3 clearly shows that our prototype does the best at keeping the overflagging down while doing relatively well on real errors.[2] A more extensive evaluation can be found in Cornu et al. (forthcoming).

Another way of evaluating our prototype is to see how closely it follows the guidelines for an efficient EFL/ESL grammar checker proposed by Granger & Meunier (1994). They describe four different criteria which should apply to grammar checkers used by non-natives. According to them, a good second language grammar checker should:

- be based on authentic learner errors:

This first criterion is met with the error corpus we collected to help us identify the types of errors that need to be dealt with.

- have a core program and bilingual 'add-on' components:

The general island processing approach we have adopted can be employed both for monolingual and bilingual grammar checkers. Full parse approaches, however, are not as easy to use on non-native speaker texts which contain many errors. 'Add-on' components will include the kinds of writing aids we have integrated into the prototype.

- be customizable:

The grammar checker can be customized by turning on or off certain groups of automata relating to grammatical areas such as "subject-verb agreement" or "tense".

- indicate what it can and cannot do:

This variable is dealt with both in flyers sent around and in the checker's manual. As our prototype is not yet commercialized, this last criterion does yet apply here.

7 Concluding remarks

In this paper we have presented the prototype of a second language writing tool for French speakers writing in English. It is made up of a set of writing aids, a problem word highlighter and a grammar checker. Two characteristics of the checker make it particularly interesting. First, overflagging is reduced. This is done by moving certain semantic problems to the problem word highlighter, gearing the automata to specific errors and interacting with the user from time to time. Second, the use of island processing has kept processing time down to a level that is acceptable to users.

Because we have designed a prototype with the view to commercializing it, scaling up can be achieved quite easily. The dictionary already contains all word classes needed and we do not expect the disambiguation stage to become less efficient when a larger dictionary is used. In addition, the error processing technology that we

[2] The error detection scores shown here are rather low compared to those of other evaluations (e.g. Bolt 1992, Granger & Meunier 1994). This can be explained by the fact that most of the test items used came from real texts produced by non-native speakers and were not specifically written by the evaluator to test a particular system. Real texts contain many lexical and semantic errors which cannot be corrected with today's grammar checkers.

have implemented allows for all components to be expanded without degrading the quality. Commercial development will involve increasing the size of the dictionary, adding automata to cover a wider range of errors, finishing some of the writing aids and integrating the completed product, i.e. the writing tool, into existing word processors.

As for the reusability of various components of our grammar checker for other language pairs, we believe that many of the errors in English are also made by speakers of other languages, particularly other Romance languages. Thus many of our automata could probably be reused as such. However, the writing aids and the problem word highlighter would obviously need to be adapted to the user's first language.

Acknowledgements

The work presented here was part of a project funded by the Swiss Committee for the Encouragement of Scientific Research (CERS/KWF 2054.2).

References

Steven Abney. 1991. Parsing by chunks. In R. Berwick, S. Abney & C. Tenny (Eds.) *Principle-Based Parsing*. Dordrecht: Kluwer.

Franck Bodmer. 1994. DELENE: un désambiguïsateur lexical neuronal pour textes en langue seconde. *TRANEL (Travaux neuchâtelois de linguistique)*, 21: 247-263.

Philip Bolt. 1992. An evaluation of grammar-checking programs as self-helping learning aids for learners of English as a foreign language. *CALL;* 5:49-91.

Gavin Burnage. 1990. *CELEX - A Guide for Users*. Centre for Lexical Information, University of Nijmegen, The Netherlands.

Kenneth Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. *Proceedings of the Second Conference on Applied Natural Language Processing*. Austin, Texas.

Pit S. Corder. 1981. *Error Analysis and Interlanguage*. Oxford: Oxford University Press.

Etienne Cornu, N. Kübler, F. Bodmer, F. Grosjean, L. Grosjean, N. Léwy, C. Tschichold & C. Tschumi. (forthcoming). Prototype of a second language writing tool for French speakers writing in English. *Natural Language Engineering*.

T.J. Fitikides. 1963. *Common Mistakes in English*. Harlow: Longman.

Sylviane Granger & F. Meunier. 1994. Towards a grammar checker for learners of English. In U. Fries, G. Tottie & P. Schneider (eds.), *Creating and Using English Language Corpora*. Amsterdam: Rodopi.

Max Silberztein. 1993. *Dictionnaires electroniques et analyse automatique de textes*. Paris: Masson.

Georg Thurmair. 1990. Parsing for grammar and style checking. *COLING*, 90:365-370.

Cornelia Tschichold. 1994. Evaluating second language grammar checkers.*TRANEL (Travaux neuchâtelois de linguistique)*, 21: 195-204.

Corinne Tschumi, F. Bodmer, E. Cornu, F. Grosjean, L. Grosjean, N. Kübler, N. Léwy & C. Tschichold. 1996. Un logiciel qui aide à la correction en anglais. *Les langues modernes*, 4:28-41.

Terry Winograd. 1983. *Language as a Cognitive Process: Syntax*, Reading, Mass.: Addison-Wesley.