ReCALL **19**(2): 137-161. 2007 © European Association for Computer Assisted Language Learning
DOI: 10.1017/S0958344007000420 Printed in the United Kingdom

FipsOrtho: A spell checker for learners of French

SÉBASTIEN L'HAIRE
University of Geneva, Department of Linguistics, 2, rue de Candolle, 1211 Geneva 4, Switzerland
(*email: sebastien.lhaire@lettres.unige.ch*)

Abstract

This paper presents FipsOrtho, a spell checker targeted at learners of French, and a corpus of learners' errors which has been gathered to test the system and to get a sample of specific language learners' errors. Spell checkers are a standard feature of many software products, however they are not designed for specific language learners' errors. After a brief review of the state of the art, we describe the system's architecture and interfaces. Then we describe our error typology and detail the techniques used to retrieve words and to order proposals appropriately: alphacode, phoneticization, ad-hoc, capitalization, apostrophe, and word separation error methods. Proposals are sorted by a score depending on the method(s) used to retrieve them, on the expected lexical category, gender, number and person, and on the string proximity with the unknown word. Then the test results are presented: a list of individual words containing errors was submitted to the alphacode and phoneticization methods; a corpus of authentic learners' errors was gathered and analyzed. Finally we conclude the paper with some limitations of the system and ideas for future research.

Keywords: spell checker, error diagnosis, corpus, French, evaluation

1 Introduction

Spell checkers are a standard feature of many software products (word processors, editors, email readers, CALL products). However, commercial products are generally designed for native speakers and do not deal with specific language learners' errors: for instance, learners often rely on phonetic approximation, they do not use the correct phoneme-grapheme conversion rule, they confuse some near phonemes such as nasal vowels and incorrectly apply rules from their mother tongue. They also make typographical errors such as character transposition, insertion, omission and substitution

and many morphological errors. A spell checker is useful to help learners master this part of the written code and gradually to improve their performance.

Moreover, in French, spelling is a key element in the written code. This code is very difficult to master, even for native speakers. Many characters are not pronounced (Catach, 1978). Words must be learned individually (Blanche-Benveniste & Chervel, 1978). Rules are not always logical and there are many exceptions. The word *femme* (wife or woman) is pronounced [fam] while it should be [fɛm] with regular rules.

In this paper, we present FipsOrtho, a spell checker tailored for learners of French, which targets the following error types: phonetic spelling, agglutination, diacritics, morphology and missing apostrophe. This work stems from the previous work of Ndiaye and Vandeventer Faltin (2003, 2004). FipsOrtho is accessible as a web application. Learners' productions are analyzed, XML-tagged and stored into a corpus, in order to test the system and to gather information about learners' frequent mistakes.

Section 2 presents an overview of the state of the art. Section 3 introduces our error typology. The system architecture and interface of FipsOrtho are described in section 4. In section 5, we present the spell checking techniques used in our system. Section 6 overviews the XML output. In section 7 we present the tests we have run on our system and the error corpus. Section 8 gives some future plans and section 9 concludes this article.

2 Related work

In this section we briefly describe the state of the art of spell checking, particularly in CALL. The first spell checker application was developed in Stanford in 1971 (Peterson, 1980). Since then, spell checking is an almost indispensable component of word processors, email readers and of some text editors. In the field of CALL, spell checking is rarely considered in the literature. Most of the time existing products, commercial or not, are integrated into systems without adaptation. However, the Basque corrector *XUXEN* (Agirre *et al*, 1992), and the spell checker for Turkish of Oflazer (1996), use a two-level morphological analyzer adapted from Koskenniemi (1994). *XUXEN-II* (Aldezabal et al, 1999) is based on finite-state transducers. *SPELLER* (de Haan & Oppenhuizen, 1994) is an intelligent tutoring system which uses phoneme-grapheme conversion rules to help Dutch-speaking learners of English to solve orthographic problems. *SANTY* (Rimrott, 2003) is a spell checker for German which uses regular expressions in order to correct morphological errors.

For a conjugation tool, Pijls, Daelemans and Kempen (1987) use morphological and orthographic rules; if no error is detected in rule choosing, they determine whether rules have been applied incorrectly. Bos (1994) extends this system with mal-rules and treats overgeneralization, incorrect application of rules, etc. Vosse (1992) uses both triphones and trigrams[1] to select appropriate candidates; candidate words are then ordered by a scoring and ranking mechanism. Kempen (1992) uses only triphones and calculates a similarity index between proposals and the incorrect string. His algorithm also takes into

1. Trigrams are sequences of three characters which compose a word. Spaces are also included in trigrams since they are delimiters. The word "word" is composed by the following trigrams: [space wo] [wor] [ord] [rd space]. Similarly, triphones are sequences of three phonemes.

account the word length and the order of the triphones in the string. Menzel (2004) describes a theoretical method to eliminate inadequate proposals: only the words of relevant part-of-speech should be kept in the list and then semantics and even world knowledge could be used. Finally, Doll and Coulombes (2004) propose to use word frequencies in order to eliminate inadequate proposals, without giving further details.

Other works do not deal directly with CALL but are worth mentioning: in their expert system for spell and grammar checking for French, Emirkanian and Bouchard (1988) use morphological techniques to correct erroneous words. Finite state transducers are used by Courtin *et al* (1991) for a spell checker of French not targeted for second language learners. For the Russian grammar checker *Skryba*, Nicholas, Debski and Lagerberg (1994) use morphological and phonetic rules. Morphological techniques are

Table 1 *Error codes with examples*

Code	**Designation**	**Comment**	**Example**
INS	Insertion	Superfluous character	cherval* → cheval
OMI	Omission	Missing character	a_bre* → arbre
SUB	Substitution	Neighbour key on keyboard	progrqme* → programme
INV	Inversion		agneda* → agenda
LEX	Lexical	Existing inappropriate word	fonds → fondé
PHG	Phonogrammatical	Non-existing word but correct pronunciation	fonétique* → phonétique
PHO	Phonetical	Non-existing word and incorrect pronunciation	londi* → lundi, macasin* → magasin
HPO	(quasi-)Homophone	Existing inappropriate word	prémisses → prémices, est → et
MOR	Morphological	Morphological error on conjugation, word formation, plurals etc.	rapident* → rapides
LNF	Non-functional characters	Non-pronounced characters in word for historical and etymological reasons	toujour* → toujours
AGR	Agreement		les enfants sage*
CPL	Complementation		J'attends sur* Anne
AUX	Auxiliary		les invités sont* dansé
TPS	Tense	Verbal tense	joué → a joué
MOD	Mode		Je veux que tu viens*
MAN	Missing word		colonie _ tabac
CAS	Case	Incorrect upper/lower case	Français (language) → français
PNC	Punctuation		
DIA	Diacritics	accentuation	meme → même
NPR	Proper noun/ adjective	Existing, correct word, unknown by the lexicon	Plymouth
INC	Unknown noun/verb/ adjective/ adverb	Existing, correct word unknown by lexicon	dravidien
SUP	Superfluous word	Redundancy	les pêcheurs entre* avec les Amériidiens
SPC	Separation by space	Missing or superfluous space	fauxsauniers* → faux sauniers
SEP	Separation by other sign		sinstaller* → s'installer
EMP	Borrowing	Borrowing from mother tongue	trade → commerce
BRU	Noisy proposal	False detection	Inadequate upper case
ORD	Word order		arrives-tu → tu arrives

described by Monsoon *et al*. (2004) for a spell checker for the Chilean indigenous language Mapundungun and by Enguehard and Mbodj (2004) for different African languages. Yarowsky (1994) and Simard and Deslauriers (2001) describe statistical reaccentuation methods for French. Jones and Martin (1997) use the statistical method Latent Semantic Analysis (LSA) to correct words confused with other existing words. And finally, Ben Othmane Zribi and Zribi (1999) deal with specific morphological errors in Arabic.

3 Error typology

In this section, we briefly introduce the error typology used in our corpus and listed in Table 1. We mostly follow Catach, Gruaz and Duprez (1986), Cordier-Gauthier and Dion (2003) and Vandeventer Faltin (2003). Mistakes can be tagged by several types. Some error types are detected exclusively by humans, others only by the computer, but most of them are detected by both, as section 7 will show.

4 System overview

In this section, we present the system architecture and interface of FipsOrtho. This system is available and freely testable on the Web at http://latlcui.unige.ch/ spellchecker/. Users must log in and provide information on age, gender and mother tongue. Teachers can also enrol classes and have free access to their students' productions.[2] Figure 1

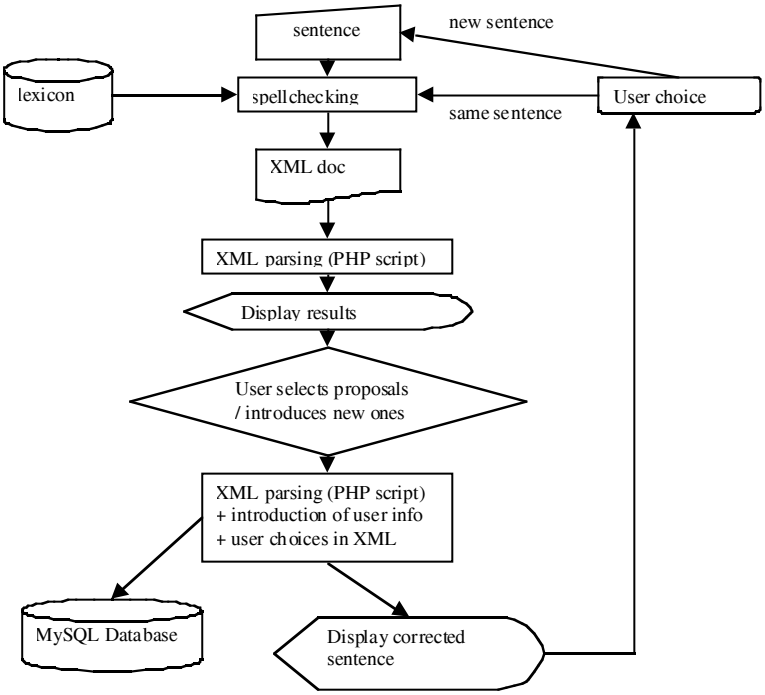


Fig. 1. System overview: usage of spell checker by learners

overviews the usage of the spell checker by learners. Sentences are tagged into a XML document which will be detailed in section 6. Then, a human expert selects sentences for the corpus and tags errors with the typology described in section 3.

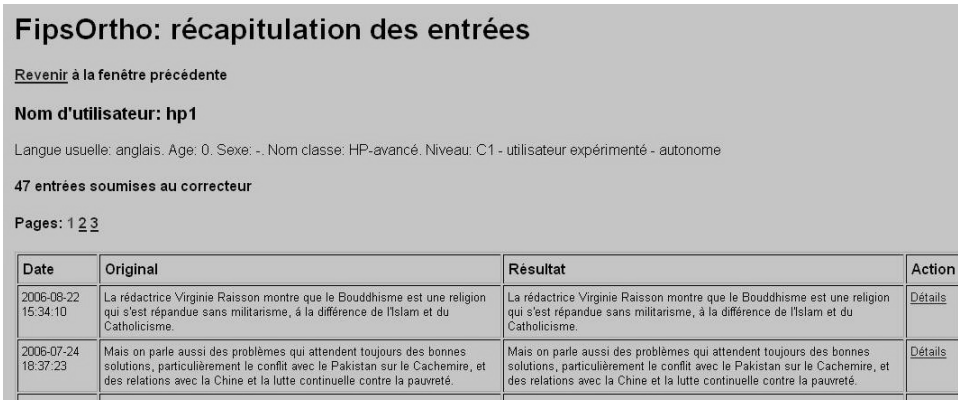
FipsOrtho uses PHP scripts in order to manage information about learners and classes, to gather the corpus and to parse the XML document. We also use some Javascript for word highlighting and form pre-processing. The spell checker is called by a CGI program which outputs an XML file.

Now let us describe the spell checker interface. Figure 2 shows the results after a learner has sent a sentence. Unknown words are displayed in red. Then for each unknown word, a combo box contains (if available) the proposals the spell checker returned. Alternatively, learners can enter their own proposal. We also provide learners with a conjugation tool which can give all verbal forms and a bilingual dictionary with speech synthesis output.

Fig. 2. Example of spell checking

2. In Switzerland, the learners' consent is not formally required in order to analyze their writing for research purposes. However, a clear statement warns learners and teachers that sentences sent to the spell checker might be stored in a corpus for research purposes and that their using of the system implies their agreement with this approach.

Learners can also access all their productions at any time, as in Figures 3 and 4. Teachers have the same access to the sentences of all their students.Every input sent to the spell checker (phrase(s) or sentence(s)) is reviewed by an expert and possibly included in the corpus. For each error found by the spell checker, the expert validates the learner's choice and tags the error given a specific typology. The expert can also tag undetected errors. Each error is stored in a table in the database and can be retrieved by the corpus users. Figure 5 describes the corpus gathering and consultation process.



FipsOrtho: récapitulation des entrées

Revenir à la fenêtre précédente

Nom d'utilisateur: hp1

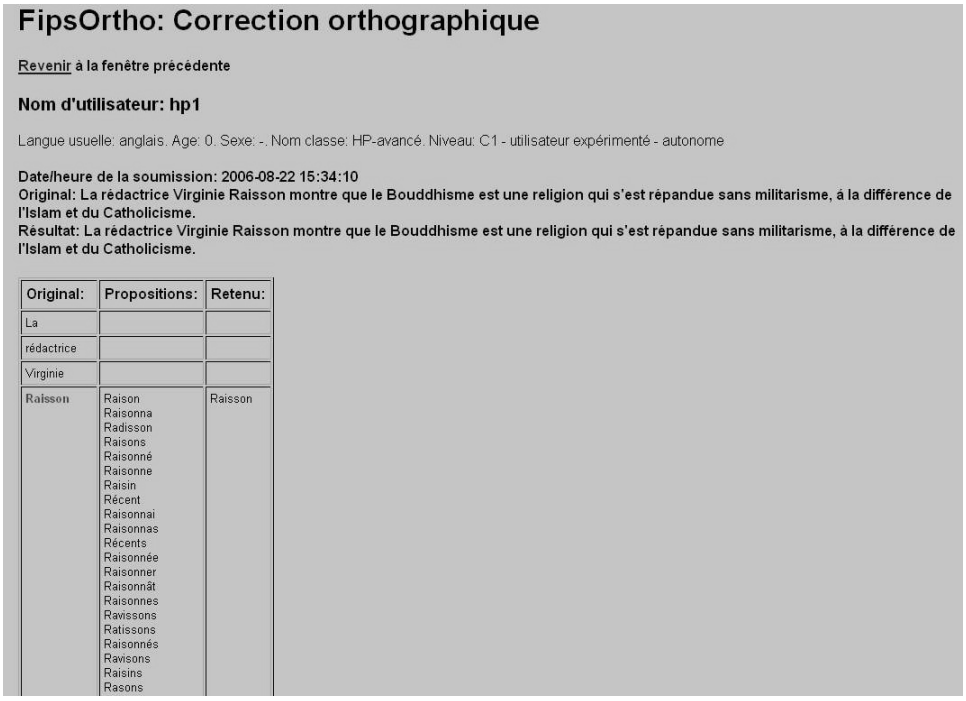
Langue usuelle: anglais. Age: 0. Sexe: -. Nom classe: HP-avancé. Niveau: C1 - utilisateur expérimenté - autonome

47 entrées soumises au correcteur

Pages: 1 2 3

Date	Original	Résultat	Action
2006-08-22 15:34:10	La rédactrice Virginie Raisson montre que le Bouddhisme est une religion qui s'est répandue sans militarisme, à la différence de l'Islam et du Catholicisme.	La rédactrice Virginie Raisson montre que le Bouddhisme est une religion qui s'est répandue sans militarisme, à la différence de l'Islam et du Catholicisme.	Détails
2006-07-24 18:37:23	Mais on parle aussi des problèmes qui attendent toujours des bonnes solutions, particulièrement le conflit avec le Pakistan sur le Cachemire, et des relations avec la Chine et la lutte continuelle contre la pauvreté.	Mais on parle aussi des problèmes qui attendent toujours des bonnes solutions, particulièrement le conflit avec le Pakistan sur le Cachemire, et des relations avec la Chine et la lutte continuelle contre la pauvreté.	Détails

Fig. 3. List of learners' sentences



FipsOrtho: Correction orthographique

Revenir à la fenêtre précédente

Nom d'utilisateur: hp1

Langue usuelle: anglais. Age: 0. Sexe: -. Nom classe: HP-avancé. Niveau: C1 - utilisateur expérimenté - autonome

Date/heure de la soumission: 2006-08-22 15:34:10
Original: La rédactrice Virginie Raisson montre que le Bouddhisme est une religion qui s'est répandue sans militarisme, à la différence de l'Islam et du Catholicisme.
Résultat: La rédactrice Virginie Raisson montre que le Bouddhisme est une religion qui s'est répandue sans militarisme, à la différence de l'Islam et du Catholicisme.

Original:	Propositions:	Retenu:
La		
rédactrice		
Virginie		
Raisson	Raison Raisonna Radisson Raisons Raisonné Raisonne Raisin Récent Raisonnai Raisonnas Récents Raisonnée Raisonner Raisonnât Raisonnes Ravissons Ratissons Raisonnés Ravisons Raisins Rasons	Raisson

Fig. 4 Details of a submission

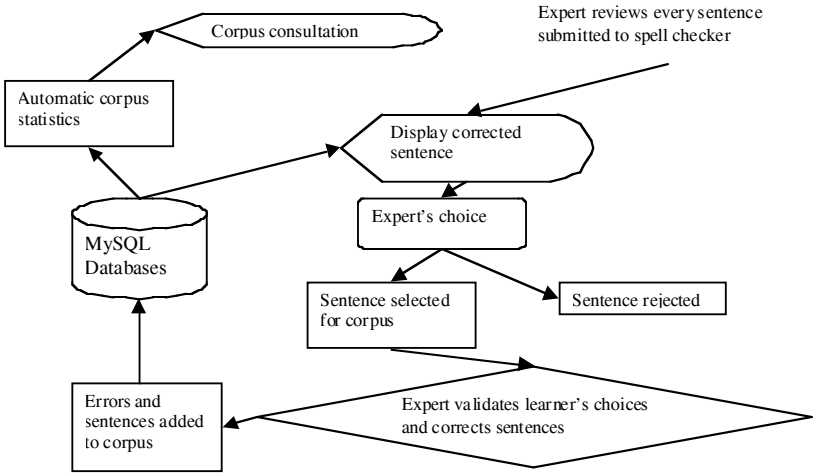


Fig. 5. Corpus gathering and consultation

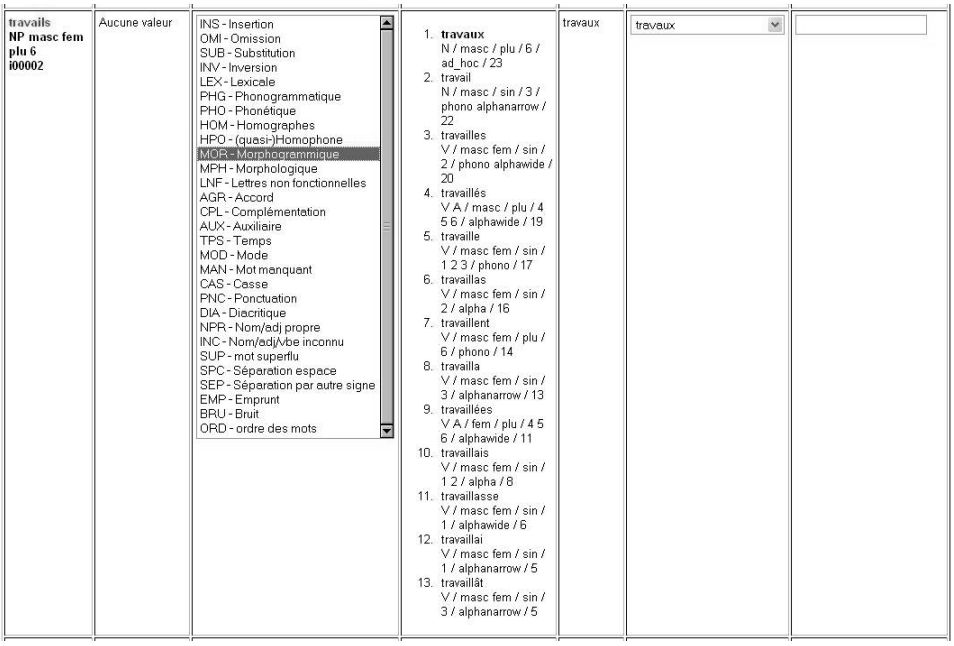


Fig. 6. Expert tagging interface

Now let us consider the expert interface. When they log into the interface, experts get a list of sentences that need validation. Figure 6 shows a part of the tagging interface. When applicable, experts get a list of proposals and see the learner's choice in boldface. They can validate it, choose another proposal or propose a new one. They also have to determine the error type. If a word has not been tagged as erroneous, the expert can tag it as well. Experts can also correct corpus entries at any time. Besides, they can also

access a compact representation of each entry, as in Figure 7.

This time-consuming process gives us an accurate measure of our spell checker's results: how many mistakes are detected; how accurate is the sorting algorithm; can we detect other kinds of mistake? The corpus also lets us see how learners write texts, how they use help tools and which kinds of mistake they make. We are particularly interested in learners' choice of proposals. We also hope to gather enough data to analyze the influence of the learners' native language. Therefore validation by an expert is crucial.

Finally, everyone on the Internet can freely access the corpus. Users get error statistics and can get details from the error database. Figure 8 shows the list for a particular error tag. If users click on the entry number, they get the compact view of the sentence in Figure 7 with the specific error highlighted.

Date/heure de la soumission: 2006-07-03 16:49:57
Original: Samuel de Champlain joué le rôle du fondateur de la ville de Québec.
Résultat: Samuel de Champlain joué le rôle du fondateur de la ville de Québec.
Expert: Samuel de Champlain a joué le rôle du fondateur de la ville de Québec.

Nb phrases: 1 (auto: 1). Nb mots: 13 (auto: 13)
Nb mots inconnus: 2. Nb. non-erreurs: 2. Nb non-détectés: 1

Commentaire: joué -> jouait, -> erreur morpho

[EDITER] [Affichage XML] [CORPUS] [SUIVANTE]

Original:	Tag(s) erreur actuel(s)	Propositions:	Retenu:
Samuel **NP** **i00001**	**NPR** - Nom/adj propre	1. Saule (10) N / masc / sin / 3 / alphanarrow 0.1818181818181818 (0.2090909090909091) 2. Salue (7) V / masc fem / sin / 1 2 3 / alphanarrow 0.1818181818181818 (0.2090909090909091) 3. Salué (7) V / masc / sin / 1 2 3 / alphanarrow 0.1909090909090909 (0.2090909090909091) 4. Salues (6) V / masc fem / sin / 2 / alphanarrow 0.1666666666666667 (0.1916666666666667) 5. Saluée (5) V / fem / sin / 1 2 3 / alphanarrow 0.175 (0.1916666666666667) 6. Salués (4) V / masc / plu / 4 5 6 / alphanarrow 0.175 (0.1916666666666667)	Samuel
de **PP** **i00002**	Aucune valeur		
Champlain **DP** **i00003**	**NPR** - Nom/adj propre	**pas de prop**	Champlain
joué **VP** **i00004**	**PHG** - Phonogrammatique **TPS** - Temps **MAN** - Mot manquant	corr manuelle ***a joué***	a joué
le **DP** **i00005**	Aucune valeur		

Fig. 7. Compact view of corpus

FipsOrtho: validation du corpus. Liste d'erreurs

Revenir à la fenêtre des statistiques

Résultats: code: OMI (54 résultats retournés)

Pages: 1 2 Tous

N°	Ent	It. Idx	Err. idx	Man	Inconnu	Sel	Cor	Nb p.	Ord sel.	Methodes	Catégories
18	7	i00002	p00001	n	origin	origine	origine	2	1	alphawide	LNF PHO OMI
19	7	i00009	p00003	n	fourures	fourrures	fourrures	23	1	alpha phono	OMI
32	14	i00007	h1	n	poque		époque	3	0		OMI

Fig. 8. List of errors

5 Spell checking

Spell checking is not a trivial task (Peterson, 1980; Kukich, 1982; Vandeventer Faltin, 2003). Words must first be identified by finding potential delimiters such as spaces, hyphens, and apostrophes. Then each word is searched in a list to check whether it belongs to the relevant language. If need be, the spell checker provides a list of possible corrections.

Let us consider the sentence: *Les travails* sont difficiles* (*The works are difficult*). This sentence contains only one error on *travails**:[3] this is an incorrect plural of *travail* instead of *travaux* . In the next sub-sections, we use this example to describe the methods used by our spell checker to propose words and the algorithms which order them by likelihood. Figure 9 shows how these methods are applied to unknown words.

5.1 Syntactic analysis

Although spell checking generally occurs before syntactic analysis, our spell checker first tries to assign the sentence(s) a syntactic analysis. We use the Fips parser (Wehrli, 1997, 2004), a robust analyzer which can retrieve chunks of analysis if no complete analysis is found.

Unknown words are assigned the adverb, verb, noun and adjective category, and then parsing rules determine the best category.[4] Although the precision of this technique is far from perfect, the guessed category is used to reorder words by likelihood after the spell checking process.

Our sentence is given the following structure: [$_{S}$ [$_{NP\ D}$ Les $_{N}$ travails*] [$_{VP\ V}$ sont [$_{AP\ A}$ difficiles]]].[5] The parser has assigned the noun category to the unknown word.

---

3 With the correct word, this sentence is grammatical, but awkward. The context could help to find a more precise word. However, the spell checker does not consider stylistic and semantic issues.

4 Quite a large number of proper nouns is already stored in our lexicon. We do not have a specific technique for proper nouns.

5. For simplification purposes, we do not use Fips' categories but more traditional labels. S : sentence. NP : noun phrase. D : determiner. N : noun. VP : verb phrase. V : verb. AP : adjectival phrase. A : adjective.

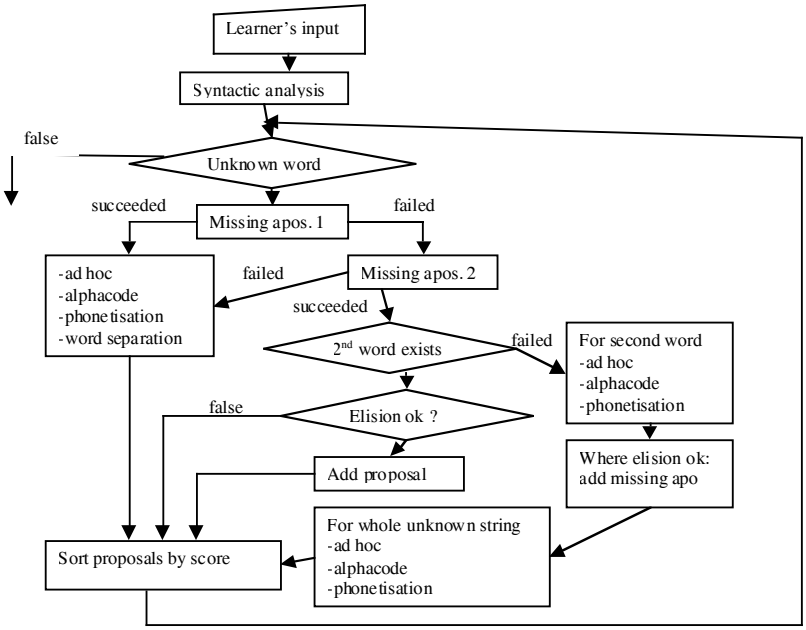


Fig. 9. Flow of spell checking techniques

5.2 Alphacode method

The alphacode method is a state-of-the-art method of retrieving words with errors of insertion, omission, inversion, reduplication and diacritics. An alphacode is a reordering of the characters composing the words. It is formed by the consonants composing the word, ordered alphabetically, and followed by the vowels, also in alphabetical order; each letter is kept only once; diacritics are removed and capital letters are considered the same as lower case letters. Words are represented by a unique alphacode but many words can share the same alphacode. Thus our unknown word *travails** has the alphacode *lrstvai*. In order to retrieve more words, we also try to add one letter at a time to the alphacode (alphawide method), which gives *blrstvai, clrstvai*, etc. and we also remove one letter at a time (alphanarrow method), which gives *rstvai, lstvai*, etc. Globally, we launch 27 queries in the lexicon for each unknown word. For *travail*, we retrieve 148 words, 6 by the alphacode (A), 93 by alphawide (W) and 49 by alphanarrow (N). Here are some words we retrieve and the method used: *travail* (N), *travailla* (N), *travaillai* (N), *travaillais* (A), *travaillas* (A), *travaillasse* (W), *travaillât* (N), *travaillées* (W), *travaillés* (W), *travailles* (W), *allitératives* (W), *ravitaillais* (A), etc. Some words are close to the original string but others are quite distant. We do not retrieve the correct form *travaux*: its alphacode *rtvxau* is too different from that of the original word. In the next section, we will show how we filter proposals to keep the most likely.

Other methods are described in the literature. The alphacode method is similar to the skeleton key of Pollock and Zamora (1984) or to the alphacode of Revuz (1991).

Pollock and Zamora (1984) also suggest using an omission key based on the frequency of letter omission. Anacodes (Zock, 2002) are formed by the letters composing the word in alphabetical order. Damerau (1964) uses a Boolean register of 28 positions (one for each letter, one for numbers and one for other symbols). Kukich (1992) suggests to use a hash table, which can retrieve inversions and, by adding or deleting letter values, omission and insertion. Finally, various trigram methods are also a very efficient solution (Peterson, 1980; de Heer, 1982; Angell, Freund and Willett, 1983; Vosse, 1992).

5.2.1 Lexicographic distance

Many of our 148 retrieved proposals are not relevant. *Allitératives* or *ravitaillais* are too remote from the original string *travails*. Therefore, it is necessary to filter out unwanted noise while keeping acceptable forms. Levenshtein (1966) proposes an algorithm, also known as edit distance, which measures the minimum number of operations (insertion, deletion and substitution) needed to transform one string into another. Insertion and deletion usually have a cost of 1 whereas substitution has a cost of 2.

Wagner and Fischer (1974) adapt the Levenshtein's distance by adding character transposition, following Damerau (1964); this algorithm is also known as the Damerau-Levenshtein distance and is detailed in Jurafsky and Martin (2000).

We have adapted this algorithm in several ways: (i) the distance is divided by the sum of the lengths of the two strings, which is a well-known weighting method. (ii) Letter case, spaces, apostrophes and hyphens are discarded (*Manger* = *manger*). (iii) Divergences on diacritics (*côté* ↔*côte*) have a cost of 0.1 because learners often discard character accentuation; therefore these errors must be less penalized. (iv) Double consonant errors (*addresse** for *adresse*) have a cost of 0.1, because this error is also frequent. Thus, between *proffesionel** and *professionnel*, the distance value is only 0.012, instead of 0.12[6] if we use the standard algorithm.

Having a distance measure, we need to set a threshold value beyond which words are eliminated. A first threshold has been fixed at 2 divided by the sum of the string lengths. However, after a test on a word list (see section 8.1), we considered it too low because it

Table 2 *List of patterns for the ad-hoc method*

Pattern	**Replacement string**	**Example**	**Pattern**	**Replacement string**	**Example**
als#	aux	chevals* → chevaux	#voir	verr	voirai* → verrai
ails#	aux	travails* → travaux	#fair	fer	fairais* → ferais
#aller	ir	allerez* → irez	age#	ment	changeage* → changement
devé	dû	Found in Mogilevski (1998)	ment#	age	repassement* → repassage
#tenir	tiendr				teniras* → tiendras

6. The distance (namely .3 and 3) must be divided by the sum of the lengths of the two strings (namely 25).

eliminated good proposals. Therefore we set a new threshold of 2.3 divided by the same sum. However, we introduced two restrictions: (i) proposals retrieved by alphawide whose distance is equal to the threshold are rejected; (ii) proposals retrieved by alphanarrow must begin with the same letter as the unknown word and the distance must be lower than the threshold. These two restrictions have proved a good compromise solution, which does not reject interesting words and does not introduce noisy proposals.

By these alphacode and edit distance algorithms, we have adapted state-of-the art processes to the French language and to common learners' errors. In our example, 10 proposals out of 148 are kept: 2 out of 6 (33%) retrieved by the alphacode method, 4 out of 93 (4,3%) by alphawide and 4 out of 49 (8.1%) by alphanarrow.

5.3 Phoneticization

The phonetic system of French is quite complex. The sound [o] can be written as *o, au, eau*, etc. There are also several nasal vowels etc. Phonetic errors are very frequent in learners' texts. They type words using phonetic approximation, also called phonetic writing. They also confuse phonemes, in particular nasal vowels. Phoneticization is a well-known method for spell checking. The system phonetizes the unknown word and provides one or more phonetic strings. Then it looks up in a phonetic lexicon for the corresponding words. We use Fips' expert system which uses about 700 rules to phonetize words (Goldman *et al.*, 2001) and returns deterministically a single phonetic string. Then we introduce some variations in the string, in order to adapt the spell checker to learners' specific needs, by swapping nasal vowels and sounds [o-ɔ] and [e-ɛ-ə-œ]. Then all the phonetic strings are looked up in the lexicon.

In our example, the word *travails* is phonetized as [tʀavaj] and the lexical lookup retrieves travail, ***travaille, travaillent*** and *travailles*. Words in boldface are new proposals; others have also been retrieved by alphacodes.

Some other techniques are worth mentioning: *SOUNDEX* (Odell & Russell, 1918, 1922) is a very old method which returns a letter followed by a numeric value of characters which depends on phonetic proximity (in English). Tanaka & Kojima (1987) propose a complex method based on a hierarchical file which classifies words on three different classifications and four depth levels, the deepest the finest. Thus, an unknown word is first converted to a sequence of phonemes and then the closest matchings are found in the dictionary. Van Berkel and De Smedt (1987) and Vosse (1992) propose a method based on triphones. Véronis (1988) relies on a phoneme proximity table. Finally, for their spell and grammar checker, Courtin *et. al.* (1991) describe a phonetizer based on transducers.

5.4 Ad-hoc rules

Morphological errors are frequent on irregular forms (singular/plural, declension, conjugation etc.). Incorrect endings are added to an existing root. Learners are prone to making such mistakes. Due to lack of time, we could not adapt Fips' morphological analyzer to deal with unknown words. Therefore, we developed an ad-hoc method, which deals with frequent errors. We set a list of strings which can be word beginnings, word endings or entire words. Table 2 shows the current list. The # sign marks a word beginning when it is on the left of the string, and a word ending when on the right;

Table 3 *Score values of methods*

Method	**Value**	**Method**	**Value**
Ad-hoc	12	Missing apostrophe	10
Separation	9	Phoneticization	6
Alphacode	5	Alphawide	3
Alphanarrow	2	First capital letter	0

Table 4 *Score values for feature matching*

Feature	**Value**	**Feature**	**Value**
Category	3	Number	3
Gender	3	Person	2

strings without # represent a whole unknown word.

Since this method could build non-words, proposals are looked up in the lexicon and, if the word exists, it is put in the proposal list. In our example, this method retrieves the correct plural form *travaux*.

5.5 Apostrophe

Apostrophes are a particular character often ignored by learners. They replace it with a space or simply glue words together. There are a few words which incorporate an apostrophe: *aujourd'hui, prud'homme, prud'hommal, presqu'île, entr'aide, s'entr'aider*, etc. In French, after a space or at the beginning of a sentence, the following characters can be followed by an apostrophe: *c, d, j, l, m, n, s* and *t*, where vowels *e* or *a* are elided before a word beginning with a vowel. Also, with words like *que, jusque, lorsque,* etc., the final e can be elided and replaced by an apostrophe.

We use two different methods to address this problem: (i) the first method treats lexical analysis failure, where a single unknown lexical item contains a space[7]; in this case, we first replace the space with an apostrophe and check in the lexicon to see whether we retrieve a known word; (ii) a second procedure detects if the first characters of the unknown word belong to the above list of characters that can be followed by an apostrophe; if so, we look up in the lexicon to see if the remainder of the string is there; if we retrieve a known word, we insert the correct string with the apostrophe in the proposal list; if we do not retrieve a known word, we launch the alphacode, phonetic and ad-hoc methods to retrieve proposals (only proposals before which words must be elided are kept).

7. The lexical analyser of our parser sometimes considers two words separated by a space as an unknown word. This happens when it recognizes the beginning of a special compound word. Strings like "prud home*" or "aujourd hui*" are considered a single lexical item and an unknown word.

Table 5 *List of proposals for word "travails"* . AH : Ad-hoc. P : phoneticization. A : alphacode. W: alphawide. N: alphanarrow.

Proposal	Cat.	Gen.	Num.	Person	Method(s)	Distance	Thresh.	Score
travaux	N	m	P	3	AH	0.2	0.15333	23
travail	N	m	S	3	P, N	0.06666	0.15333	22
travailles	V	m, f	S	2	P, W	0.06111	0.12777	20
travaillés	V, A	m	P	1-3	W	0.06111	0.12777	19
travaille	V	m, f	S	1-3	P	0.06471	0.13529	17
travaillas	V	m, f	S	2	A	0.06111	0.12777	16
travaillent	V	m, f	P	3	P	0.16315	0.12105	14
travailla	V	m, f	S	3	N	0.06471	0.13529	13
travaillées	V, A	f	P	1-3	W	0.11052	0.12105	11
travaillais	V	m, f	S	1-2	A	0.11052	0.12105	8
travaillasse	V	m, f	S	1	W	0.11	0.115	6
travaillai	V	m, f	S	1	N	0.11666	0.12777	5
travaillât	V	m, f	S	3	N	0.11666	0.12777	5

We did not find any reference to the problem of apostrophe in the literature. Our method is based on typical error observation and is adapted to our system's functioning. For our example, this method is not relevant and retrieves no proposal.

5.6 Capitalization

Capitalization errors are detected in a very trivial way. We rely on Fips' lexical analysis and simply check if the first word of the sentence begins with a capital letter. If not, we insert a proposal. This method is not sound enough: Fips' lexical analysis does not rely on capital letters to set sentence boundaries, since in informal texts capital letters are

```
<LATLCORR xml:lang="fr">
    <SUBMISSION>
        <SENTENCE sentenceId="1">
        <ITEM index="i00001" pos="1" projcat="DP" gender="masc fem" number="plu" pers="6">
            <ORIGINAL itemTag="i00001">Les</ORIGINAL></ITEM>
        <PUNC key="space"/>
        <ITEM index="i00002" pos="2" projcat="NP" gender="masc fem" number="plu" pers="6">
            <ORIGINAL itemTag="i00002">travails</ORIGINAL>
            <PROPS itemTag="i00002">
            <PROPOSAL index="p00001" itemTag="i00002" cat="N" gender="masc"
        number="plu" pers="6" method="ad_hoc" dist="0.2" thresh="0.1533333333333333"
        score="23">travaux</PROPOSAL> ... </PROPS>
        </ITEM>
        <PUNC key="space"/>
        <ITEM index="i00003" pos="3" projcat="TP" gender="masc fem" number="plu" pers="6">
            <ORIGINAL itemTag="i00003">sont</ORIGINAL></ITEM>
        <PUNC key="space"/>
        <ITEM index="i00004" pos="4" projcat="AP" gender="masc fem" number="plu" pers="4 5
    6">
            <ORIGINAL itemTag="i00004">difficiles</ORIGINAL></ITEM>
            <PUNC pos="5">.</PUNC>
    </SENTENCE>
</SUBMISSION></LATLCORR>
```

Fig. 10. Sample of XMl output

Table 6 *Score of methods on words list*

Methods	**Score**	**Percent**
Alpha + phono	45	27.61
Alphawide + phono	13	7.98
Alphanarrow + phono	7	4.29
Alpha	36	22.09
Alphawide	6	3.68
Alphanarrow	7	4.29
Phono	25	15.34
No proposal	13	7.98
No correct proposal	11	6.75
TOTAL:	163	

dropped out; like most parsers, Fips is targeted to grammatical texts and is less accurate with learners' texts containing mistakes. Therefore, we cannot rely on Fips' analysis to delimit sentences and consequently words that must begin with capital letters. However, developing new algorithms to treat specific learner inputs would be too demanding. Consequently this superficial method has been considered better than nothing and we should draw learners' attention to this point.

For our example, this method did not find an error, since the sentence begins with a capital letter and, despite the error, Fips' output is a complete sentence.

5.7 Word separation errors

The last method deals with separation errors. It is used after other methods. The string is split into two parts at every possible location; for each solution, we insert a hyphen (*portemonnaie** → *porte-monnaie*) and an apostrophe (*prudhomme** → *prud'homme*) between the two parts and look up in the lexicon to see if the word exists; we also look up the first part of the string and, if it exists, look up the second part; if the two parts are retrieved, we insert a proposal with the two parts separated by a space (*veuxpas** → *veux pas*). If the second part is unknown, we run the alphacode, phoneticization and ad-

Table 7 *Error statistics of corpus*

Number of errors in corpus: 1188		
Detected unknown words: 407 (of 6656 word, 6.115%) Undetected error corrected by expert: 781		
Correct proposal by spell checker: 198 (48.649%)	Non-errors: 161 (39.558%)	Manually corrected error: 48 (11.794%
By one method: 118 (59.596%) By >1 method: 80 (40.404%)	W/ proposal: 88 (54.658%) W/o proposal: 73 (45.342%)	W/ proposal: 31 (64.583%) W/o proposal: 17 (35.417%)

Table 8 *Results of methods on corpus and of error categories involved*

Method	Score	1 meth. alone	>1 meth.	% alone	% combined	Avg. number of props	Max number of props	Avg range of correct prop.	Err categories
Ad-hoc	3	1	2	33.3	66.6	12.333	13	3	MOR, MOD
Alpha	144	76	68	52.778	47.222	14.431	54	1.243	DIA, INS, PHG, PHO, OMI, etc.
Alphanarrow	18	12	6	66.6	33.3	5.389	15	2	INS, NPR, PHG, MOR, TPS, etc.
Alphawide	24	15	9	62.5	37.5	7.292	26	2.375	OMI, PHO, PHG, DIA, EMP, etc.
Upper case	3	3	0	100	0	1	1	1	CAS
Missing apostrophe	1	1	0	100	0	1	1	1	SEP
Phono	87	7	80	8.046	91.954	12.828	49	1.161	DIA, PHG, INS, OMI, PHO, MOR, LEX, LNF, AGR, NPR, etc.
Separation	4	3	1	75	25	3.5	9	1	SPC

hoc methods on it and make proposals. We found no references in the literature for this issue either. For our example, this method is not relevant.

5.8 Ordering of proposals

After applying these methods, we have to reorder proposals by descending order of likelihood. Therefore, we calculate a score for each proposal. Each method has a score value given in Table 3. If a proposal is retrieved by several methods, the scores are added. If the lexicographic distance between the unknown word and the proposal is less than 0.1, the score is incremented by 8. Then we use Fips' analysis in order to present first proposals that fit better into the sentence. The unknown word *travails** has the features noun, gender both masculine and feminine, number plural and person third. For each corresponding value of the proposal, the score is increased following Table 4.

Finally, all the proposals are reordered by decreasing score and then by increasing lexicographic distance. To summarize, for our example, we found 13 proposals, which are ordered as shown in Table 5.

Although most of the words in the list do not belong to the guessed category of the unknown word, we keep them, in order to balance the lack of accuracy in guessing categories. After this survey of spell checking techniques, we give a sample of the XML output.

6 XML output

In this section, we shortly describe the XML files output by the spell checker, which are then modified by the corpus application before they are stored in the database. Figure 10 shows a sample file

We store Fips' analysis or expert system's prediction in the tag *ITEM*. Each *ITEM* gets a unique index value. Each proposal also gets a unique index value and contains information about category, number, gender, person, method(s) involved, lexicographic distance, distance threshold value and score. If the learner chooses one of the proposals, the corresponding tag *PROPOSAL* is updated with an attribute *selected="yes*". If the learner enters his/her own proposal, it is stored in tag *HUMAN_CORR*, which also gets an index value. If the learner chooses to keep the original word, an attribute *selected="yes*" is added to the tag *ORIGINAL*. When the learner validates his/her corrections, information about age, mother tongue, country, learner's level, and submission date and time are added to the XML file in the tag *SUBMISSION* and his/her choice among proposals is added to the XML code.

If the submission is selected for the corpus by the expert, the attribute *correctchoice*="*yes*" is added to the correct proposal. If s/he manually adds a proposal or tags a word which has not been detected as incorrect, the tag *HUMAN_CORR* is added with the attribute *expert*="*yes*".

7 Test results

In this section, we show the results of the first tests on our spell checker. The first test

was a word list which was used to test the alphacode and phoneticization methods. The second test was on authentic sentences.

7.1 Word list

For our first test, we took the same word list used by Ndiaye and Vandeventer Faltin (2003, 2004), which comes from Dinnematin, Sanz and Bonnet (1990) and Burston (1998). We have also introduced some variation in spelling. The list of 162 words is given in appendix A. Table 6 lists the results by method. One word results in two equally good proposals and is counted for the phono and alphawide + phono methods. Some words did not get a correct proposal: either the correct word was not in the lexicon (rare words) or the incorrect word was too far from the correct one.[8]

On average, 152.2 proposals per unknown word were retrieved and 6.025 proposals were selected by filtering. About six proposals per unknown word provide good results, since learners cannot rely on their intuition to determine if they need more proposals. On average, the alphacode method retrieves 12.6 proposals, alphanarrow 57.3 and alphawide 81.7.

7.2 Corpus gathering

Our second test was run on authentic productions from various sources. We gathered 296 entries, from:

- Authentic sentences provided by teachers (native speakers of English from Jamaica, Australia and Canada);
- Articles on CALL (Cordier-Gauthier & Dion, 2003 and Mogilevski, 1998);
- Sentences from a benchmark of grammar checkers (Sanz, 1992), constituted from the text of a famous French TV dictation contest (Dictée de Bernard Pivot) filled with errors;
- Email from a native speaker.

This corpus contains information about the learner's age, gender, mother tongue, country and level of French. When we get sufficient and representative enough data, this information will be useful in defining the parameters of the spell checker, depending on the learner's characteristics, in order to get better results earlier earlier in the list of proposals. The corpus contains 6656 words, or 559 sentences, on average 22.487 words per entry. Table 7 summarizes the statistics of the corpus.

Table 8 summarizes the results of each method. Globally, the average number of proposals is 7.7568, which seems reasonable. The average range of correct proposals is 1.4646, which is a good result. Error categories retrieved by each method are listed by decreasing order.

Not surprisingly, the ad-hoc method is associated with morphological errors. The alphacode methods deal mostly with diacritics, insertion and omission errors; substitution is rare and suppression is not even represented in our data. Phonological

8. Some incorrect words were artefacts with phonetic spellings

errors refer mostly to diacritics, phonological and phonogrammatic errors, and also omission and insertion errors, since they do not usually modify pronunciation. Finally, other methods deal with specific error categories.

Non-errors are words left unchanged, mostly proper names, unknown words or false detections, namely in the upper case method. The manually corrected words are predominantly proper names combined with other errors (punctuation etc.).

We also tagged manually 703 errors which were not (and cannot be) detected by the spell checker. Most of them are lexical (242), agreement (178), complementation (101), superfluous word (89) and missing word errors (63). Some of these errors could be found by other techniques, like the strategies we developed in the FreeText project (L'haire & Vandeventer Faltin, 2003).

8 Future plans

In this section, we talk about some future plans to improve the spell checker. Above all, we should gather more data to refine our techniques. Unfortunately, we do not have direct access to learners and do not have a complete CALL environment either, where learners' productions are elicited through learning activities. After a call on a mailing list, only one teacher kindly gave us files with learners' data and another colleague contacted personally did the same. It would also be interesting to gather learners' opinions about our tools.

Also, the phoneticization method is not totally satisfying. The phoneticization rules are deterministic and our technique of vowel replacement is not accurate. Relying on large corpus data, we could develop a phonetizer which can return several proposals. Another way of improvement is to activate and deactivate rules depending on the learner's mother tongue (nasal vowels confusion deactivated for Portuguese native speakers, [R-l] confusion for Asian learners, etc.).

A morphological analyzer would also be useful. The ad-hoc method is not computationally efficient and has minimal coverage. This analyzer could also be made available to learners as a learning tool. We could also develop an interface to the lexicon so that learners can check the subcategorization frame of verbs, adjectives and nouns; they could also read the phonetic transcription of words and listen to a speech synthetizer.

We could also improve the results of proposal selection by lexical distance. If no proposal is found, the threshold level could be increased. We should rely more on syntactic analysis to select lexemes. Scoring values used to order results seem reliable, but this has still to be confirmed by more data.

The maximum input length is too limited, due to the short processing time inherent in Web applications. Therefore, we could provide a standalone application and/or some batch treatment of input that could be returned back to learners via email or by other means.

Finally, the user interface could be adapted to the learners' level. Some syntactic information could be provided for more advanced learners. Lexical analysis of words could also be available. Learners and teachers could also choose to activate / deactivate some methods or to discard new orthographic rules (*aout* instead of classical orthography *août, aigüe* instead of *aiguë*, etc.).

9 Conclusions

Our goal was to develop a spell checker targeted at learners of French. Using some state-of-the-art methods, we adapted these techniques to learners' specific mistakes. Our results are quite reliable and encouraging. Our spell checker could be a useful tool for learners, though it must be regarded as a help rather than a teaching medium. Since the written code must be learned word by word, we can expect learners to gradually increase their vocabulary, since specific feedback reinforces language awareness.

Although the data gathered in the corpus is sparse, the techniques involved in Fips Ortho seem to give exploitable results.

Acknowledgement

The author wants to thank Gabriela Soare and Eric Wehrli for their careful reading and fruitful comments on this paper. He thanks also warmly the colleagues who kindly gave him files of learners' authentic productions and the reviewers for their useful comments.

References

Agirre, E., Alegria, I., Arregi, X., Artola, X., Diaz de Ilarraza, A., Maritxalar, M., Sarasola, K. and Urkia, M. (1992) XUXEN: a spelling checker / corrector for Basque based on two-level morphology. In: *Third conference on Applied Natural Language Processing: Proceedings of the conference*. Trento, Italy: Association for Computational Linguistics, 119–125.

Aldezabal, I., Alegria, I., Ansa, O., Arriola, J. M., Ezeiza, N., Aduriz, I. and Da Costa, A. (1999) Designing spelling correctors for inflected languages using lexical transducers. In: *Proceedings of EACL'99: Ninth Conference of the European Chapter of the Association for Computational Linguistics*. Bergen, Norway: Association for Computational Linguistics, 265–266.

Angell, R. C., Freund, G. E. and Willett, P. (1983) Automatic spelling correction using a trigram similarity measure. *Information Processing and Management*, **19** (4): 255–261.

Blanche-Benveniste, C. and Chervel, A. (1978) *L'orthographe*. Paris: Maspéro, 3rd ed.

Ben Othmane Zribi, C. and Zribi, A. (1999) Algorithmes pour la correction des erreurs orthographiques en arabe. In: *TALN 99. 6e conférence annuelle sur le Traitement Automatique des Langues Naturelles: Actes*. Cargèse, Corsica: ATALA, 223–232.

Bos, E. (1994) Error Diagnosis in a Tutoring System for the Conjugation and Spelling of Dutch Verbs. *Computers in Human Behavior*, **10** (1), 33–49.

Burston, J. (1998) Antidote 98. *CALICO Journal*, **16** (2): 197–212.

Catach, N. (1978) *L'orthographe. Que sais-je?* Paris: Presses Universitaires de France.

Catach, N., Gruaz, C. and Duprez, D. (1986) *L'orthographe française. Traité théorique et pratique*. Paris: Nathan, 2nd ed.

Cordier-Gauthier, C. and Dion, C. (2003) La correction et la révision de l'écrit en français langue seconde : médiation humaine, médiation informatique. *Alsic*, **6** (1): 29–43.

Courtin, J., Dujardin, D., Kowarski, I., Genthial, D. and de Lima, V. L. (1991) Towards a complete detection/correction system. In: I*nternational Conference on Current Issues in Computational Linguistics*. Penang, Malaysia, 158–173.

Damerau, F. J. (1964) A Technique for Computer Detection and Correction of Spelling Errors. *Communications of the Association for Computing Machinery*, **7** (3): 171–176.

de Haan, A. and Oppenhuizen, T. (1994) SPELLER: A Reflexive ITS to Support the Learning of Second Language Spelling. *Computers in Human Behavior*, **10** (1):21–31.

de Heer, T. (1982) The application of the concept of homeosemy to natural language information retrieval. *Information Processing and Management*, **18** (5): 229–236.

Dinnematin, S., Sanz, D. and Bonnet, A. (1990) Sept correcteurs pour l'orthographe et la grammaire. *Science et Vie Micro*, **78**:118–130.

Doll, F. and Coulombe, C. (2004) L'avenir des correcteurs grammaticaux: un point de vue industriel. *BULAG*, **29**: 33–50.

Emirkanian, L. and Bouchard, L. H. (1988) Knowledge integration in a robust and efficient morphosyntactic analyzer for French. In: *Coling Budapest: proceedings of the 12th International Conference on Computational Linguistics,* 22-27 August 1988. Budapest: J. von Neumann Society for Computing Science, vol. 2, 166–171.

Enguehard, C. and Mbodj, C. (2004) Des correcteurs orthographiques pour les langues africaines. *BULAG,* **29**: 51–68.

Goldman, J.-P., Gaudinat, A., Nerima, L. and Wehrli, E. (2001) FipsVox: a French TTS based on a syntactic parser. In: *4th. ISCA international Workshop on speech synthesis (SSW4)*. Edinburgh

Jones, M. P. and Martin, J. H. (1997) Contextual Spelling Correction Using Latent Semantic Analysis. In: *Fifth Conference on Applied Natural Language Processing, Proceedings of the Conference*. Washington Marriott Hotel, Washington, DC, USA: ACL, 166–173.

Jurafsky, D. and Martin, J. H. (2000) *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River N.J: Prentice Hall, cop.

Kempen, G. (1992) Language Technology and Language Instruction: Computational Diagnosis of Word Level Errors. In: Swartz, M. L. and Yazdani, M. (eds.), *Intelligent Tutoring Systems for Foreign Language Learning. The Bridge to International Communication*. Berlin: Springer Verlag, 191–198.

Koskenniemi, K. (1994) A General Computational Model for Word-form Recognition and Production. In: *Proceedings of the 15th International Conference on Computational Linguistics (COLING '94)*. Kyoto, Japan, 178–181.

Kukich, K. (1992) Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys*, **24** (4): 377–439.

Levenshtein, V. I. (1966) Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, **10** (8): 707–710.

L'haire, S. and Vandeventer Faltin, A. (2003) Error Diagnosis in the FreeText Project. *Calico Journal*, **20** (3): 481–495.

Menzel, W. (2004) Errors, Intentions, and Explanations: Feedback Generation for Language Tutoring Systems. In: *Proceedings of InSTIL/ICALL2004: NLP and Speech Technologies in Advanced Language Learning Systems*, Venice.

Mogilevski, E. (1998) Le Correcteur 101 (A Comparative Evaluation of Version 2.2 and Version 3.5 Pro). *Calico Journal*, **16** (2): 183–196.

Monson, C., Levin, L., Vega, R., Brown, R., Font Llitjos, A., Lavie, A., Carbonell, J., Cañulef, E. and Huisca, R. (2004) Data Collection and Analysis of Mapundungun Morphology for Spelling Correction. In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*. Lisbon, Portugal: ELRA - European Language Resources Association, vol. **5**, 1629–1632.

Ndiaye, M. and Vandeventer Faltin, A. (2003) A Spell Checker Tailored to Language Users. *Computer Assisted Language Learning*, **16** (2–3): 213–232.

Ndiaye, M. and Vandeventer Faltin, A. (2004) Correcteur orthographique adapté à l'apprentissage du français. *BULAG*, **29**: 117–134.

Nicholas, N., Debski, R. and Lagerberg, R. (2004) Skryba: An Online Orthography Teaching Tool

for Learners from Bilingual Backgrounds. *Computer Assisted Language Learning*, **17** (3–4): 441–458.

Odell, M. K. and Russell, R. C. (1918, 1922) *Patent Numbers 1,261,167 (1918) and 1,435,663 (1922)*. U.S. Patent Number, U.S. Patent Office.

Oflazer, K. (1996) Error-tolerant Finite-state Recognition with Application to Morphological Analysis and Spelling Correction. *Computational Linguistics*, **22** (1): 73–89.

Peterson, J. L. (1980) Computer Programs for Detecting and Correcting Spelling Errors. *Communications of the Association for Computing Machinery*, **23** (12): 676–687.

Pijls, F., Daelemans, W. and Kempen, G. (1987) Artificial Intelligence Tools for Grammar and Spelling Instruction. *Instructional Science*, **16**, 319–336.

Pollock, J. L. and Zamora, A. (1984) Computer Programs for Detecting and Correcting Spelling Errors. *Communications of the Association for Computing Machinery*, **27** (4): 358–368.

Revuz, D. (1991) *Dictionnaires et Lexiques, Méthodes et Algorithmes*. PhD Dissertation: Université Paris VII.

Rimrott, A. (2003) SANTY: *A Spell Checking Algorithm for Treating Predictable Verb Inflection Mistakes Made by Non-Native Writers of German*. Term Paper for LING 807 Computational Linguistics, Simon Fraser University.

Sanz, D. (1992) Grammaire: quatre ténors à l'épreuve. *Science et Vie Micro*, **90**:100–108.

Tanaka, Eiichi and Kojima, Yurie (1987) A High Speed String Correction Method Using a Hierarchical File. *IEEE transactions on pattern analysis and Machine Intelligence*, **9** (6): 806–815.

van Berkelt, B. and De Smedt, K. (1988) Triphone Analysis: A Combined Method for the Correction of Orthographical and Typographical Errors. In: *Second Conference on Applied Natural Language Processing: Proceedings of the Conference.Austin*, Texas, USA, 77–83.

Vandeventer Faltin, A. (2003) *Syntactic Error Diagnosis in the context of Computer Assisted Language Learning*. PhD Dissertation, University of Geneva, Faculty of Arts, Geneva.

Véronis, J. (1988) Computerized Correction of Phonographic Errors. *Computers and the Humanities*, **22**:43–56.

Vosse, T. (1992) Detecting and Correcting Morpho-syntactic Errors in Real Texts. In: Third Conference on Applied Natural Language Processing: *Proceedings of the Conference*. *Trento, Italy: Association for Computational Linguistics*, 111–118

Wagner, R. A. and Fischer, M. J. (1974) The String-to-String Correction Problem. *Journal of the Association for Computing Machinery*, **21** (1): 168–173.

Wehrli, E. (1997) *L'analyse syntaxique des langues naturelles: problèmes et méthodes*. Paris: Masson.

Wehrli, E. (2004) Un modèle multilingue d'analyse syntaxique. In : Auchlin, A., Burger, M., Filliettaz, L., Grobet, A., Moeschler, J., Perrin, L., Rossari, C. and de Saussure, L. (eds.). *Structures et discours: mélanges offerts à Eddy Roulet*. Québec: Nota Bene, Langues et pratiques discursives, 311–329.

Yarowsky, D. (1994). Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French. In: *32nd Annual Meeting of the Association for Computational Linguistics: Proceedings of the Conference*. Las Cruces, New Mexico: New Mexico State University, 88–95.

Zock, M. (2002) Sorry, but what was your name again, or, how to overcome the tip of the tongue problem with the help of a computer? *COLING-02: SEMANET: Building and Using Semantic Networks*. *Proceedings*. Taipei, Taiwan.

Appendix A

Unknown word	Correct word	Method(s)	Nb. proposals	Nb. Selected
absorpsion	absorption	WP	75	2
acessit	accessit	-	403	0
acceuil	accueil	A	89	4
aceuil	accueil	A	89	5
acolite	acolyte	P	106	3
addresse	adresse	AP	589	20
address	adresse	AP	589	23
aigüe	aiguë	AP	56	8
aigue	aiguë	A	55	7
algorythme	algorithme	P	2	2
appas	appât	P	86	7
appogiature	appog(g)iature	-	62	0
aéropage	aréopage	A	90	3
arome	arôme	AP	110	6
asujettir	assujettir	AP	179	24
attrapper	attraper	A	618	15
azalé	azalée	AP	35	5
azalee	azalée	A	35	4
barette	barrette	AP	174	10
barete	barrette	A	174	17
béquée	becquée	-	7	1
beckee	becquée	-	1	0
bifteek	bifteck	W	3	3
biftèque	bifteck	P	5	4
boîter	boiter	A	90	13
boursouffler	boursoufler	A	38	12
braîment	braiment	A	324	8
celà	cela	AP	204	8
charriot	chariot	AP	79	4
chariau	chariot	P	22	4
charette	charrette	AP	336	12
chrysalyde	chrysalide	WP	2	2
chrysantème	chrysanthème	AP	16	2
comparition	comparution	W	83	1
comparison	comparaison	A	104	5
comcombre	concombre	W	23	1
concurentco	ncurrent	AP	478	24
concuran	concurrent	P	40	8
congruement	congrûment	-	14	1
connection	connexion	AP	312	12
consonnant	consonant	-	333	4
contigüe	contiguë	A	18	8
controle	contrôle	AP	283	14
control	contrôle	WP	45	8
convaint	convainc	NP	112	11
convin	convainc	WP	30	17
coordonateur	coordinateur	W	135	3
courier	courrier	AP	89	18
coutumace	contumace	W	42	2
cyprés	cyprès	AP	8	2
cipre	cyprès	-	168	5
débarasser	débarrasser	A	226	17
déclancher	déclencher	N	28	4

Unknown word	Correct word	Method(s)	Nb. proposals	Nb. Selected
déguingandé	dégingandé	N	50	2
dérilection	déréliction	-	122	0
dévôt	dévot	AP	12	4
dilemne	dilemme	N	22	1
disgrâcier	disgracier	AP	145	21
disparâte	disparate	AP	646	7
drôlatique	drolatique	-	16	0
dislexie	dyslexie	WP	18	2
échaufourée	échauffourée	AP	56	2
anthropie	entropie	-	183	1
erronné	erroné	AP	235	7
éthymologique	étymologique	NP	5	2
filigramme	filigrane	-	20	0
gaité	gaieté	AP	190	23
gheto	ghetto	A	5	2
guéto	ghetto	P	63	6
gueto	ghetto	-	61	4
halucination	hallucination	AP	24	2
hypothénuse	hypoténuse	A	8	2
hipotenus	hypoténuse	-	16	0
imbécilité	imbécillité	AP	7	3
infractus	infarctus	A	53	1
infarctusse	infarctus	NP	198	1
inommé	innommé	-	77	0
insassiable	insatiable	WP	307	4
intenssemment	intensément	A	499	4
intensement	intensément	A	499	5
macchiavélique	machiavélique	AP	2	2
malaîse	malaise	AP	460	19
malapris	malappris	-	51	1
malapri	malappris	-	43	1
malgrès	malgré	NP	148	2
malgres	malgré	N	148	2
malgre	malgré	A	85	2
mapemonde	mappemonde	AP	12	2
marâsme	marasme	AP	714	9
marasm	marasme	WP	126	19
négligeamment	négligemment	NP	158	2
negligement	négligemment	A	61	2
aurenge	orange	P	176	5
occurence	occurrence	AP	234	2
ocurance	occurrence	NP	118	3
pannacée	panacée	AP	42	6
panassée	panacée	P	206	21
pantomine	pantomime	A	181	2
pécunière	pécuniaire	WP	28	2
pélerine	pèlerine	A	60	5
piqure	piqûre	A	74	10
picur	piqûre	P	6	2
picure	piqûre	P	45	5
pickure	piqûre	P	3	2
précéde	précède	AP	57	17
profesionel	professionnel	A	130	6
proffessionel	professionnel	AP	130	6
proffesionel	professionnel	A	130	5

Unknown word	**Correct word**	**Method(s)**	**Nb. proposals**	**Nb. Selected**
profesionnel	professionnel	A	130	6
protége	protège	AP	51	16
psychadélique	psychédélique	N	2	2
psiquédélique	psychédélique	-	18	0
râtisser	ratisser	A	2284	22
recoit	reçoit	A	181	4
ressoi	reçoit	P	366	5
reswa	reçoit	-	64	0
réddhibitoire	rédhibitoire	AP	3	2
redibitoir	rédhibitoire	W	51	1
remerciment	remerciement	AP	165	6
renumération	rémunération	A	333	1
shéma	schéma	WP	57	4
schema	schéma	A	137	2
séborhée	séborrhée	-	41	0
soufle	souffle	AP	95	17
soufl	souffle	WP	42	10
subbit	subit	AP	60	13
subcidiaire	subsidiaire	N	21	3
subsidière	subsidiaire	WP	55	2
substanciel	substantiel	NP	47	4
succint	succinct	AP	50	3
suxin	succinct	P	9	2
superfètatoire	superfétatoire	AP	87	2
simptomatique	symptomatique	WP	20	2
sinptomatik	symptomatique	P	11	2
sindrome	syndrome	P	462	2
syndrôme	syndrome	AP	15	2
synthése	synthèse	AP	10	4
sintèz	synthèse	P	148	6
sinthèse	synthèse	P	150	2
sizygie	syzygie	-	5	0
traditionnaliste	traditionaliste	AP	648	4
troglodite	troglodyte	P	14	2
esplication	explication	-	304	1
jud'aurenge	jus d'orange	-	7	0
quesque	qu'est-ce que	-	63	2
impère	impair	P	199	6
san	sans	AP	101	18
adissione	additionne	P	405	4
adicione	additionne	P	115	3
chifre	chiffre	AP	56	12
contien	contient	A	310	16
angletterre	Angleterre	A	280	1
presentimen	pressentiment	A	345	8
so	saut	P	49	22
premiere	première	A	196	7
videttes	vedettes	N	135	6
emploiés	employés	-	100	7
ecrire	écrire	A	203	25
raissonable	raisonnable	A	346	2
particuliaire	particulière	AP	160	2
fenaitre	fenêtre	P	959	2
puit	puis/puits	WP/P	12	8