Computational Linguistics in the Netherlands Journal 1 (2011) 1-20 Submitted 7/2011; Published 12/2011

Automation of dictation exercises. A working combination of CALL and NLP

Richard Beaufort RICHARD.BEAUFORT@UCLOUVAIN.BE
Sophie Roekhaut SOPHIE.ROEKHAUT@UCLOUVAIN.BE
CENTAL, Université catholique de Louvain

Abstract

This article is in the context of the Computer-Assisted Language Learning (CALL) framework, and addresses more specifically the automation of dictation exercises. It presents a method for correcting learners' copies. Based around Natural Language Processing (NLP) tools, this method is original in two respects. First, it exploits the composition of finite-state machines, to both detect and delimit the errors. Second, it uses automatic morpho-syntactic analysis of the original dictation, which makes it easier to produce superficial and in-depth linguistic feedback. The system has been evaluated on a corpus of 115 copies including 1,532 copy errors. The accuracy of the error detection is 99%. The superficial feedback is 97.2% correct, the in-depth feedback 96%, and the morpho-syntactic analysis 87.7%.

1. Introduction

For some years now, the standard of native French speakers' spelling has been dropping significantly (Manesse 2007). This involves all levels in the society, including students in higher and university education. And yet the experts note that a proper command of the language is vital, both for successful completion of studies and for successful integration into the socio-professional environment (Didier and Fairon 2006). In order to make up for these shortcomings, special classes are being offered in certain disciplines. But overall, teachers consider that few high-quality tools are available to them, and above all that they have little time to devote to this specific subject, which very often does not form part of their basic teaching (Didier and Seron 2006).

In order to address this problem, we are currently developing the PlatON platform[1], which comes within the scope of Computer-Assisted Language Learning (CALL). Accessible on line[2], this platform is dedicated to spelling and is aimed at both native and non-native speakers, provided the latter already have an advanced command of the language, both oral and written[3]. PlatON differs in this respect from the other CALL platforms, which are primarily devoted to second language learners.

PlatON has been thought of as a collaborative platform. On the one side, teachers create courses and add exercises to them. On the other side, learners register to a course and do

[1] PlatON stands for *PLATeforme d'aide à l'enseignement et à l'apprentissage de l'Orthographe sur le Net* ("a spelling-dedicated online platform to help teachers and learners").

[2] Address: www.normalink.com/platon. At the time of writing, the website is still under development.

[3] The system is intended for foreign learners of levels C1 and C2 defined by the Common European Framework of Reference for Languages (CEFRL, Council of Europe and Education 2001).

the exercises. PlatON is currently concentrating on French, but will gradually open up to other languages, including most likely Dutch and English first.

In traditional teaching, three types of exercises are proposed to the learner: closed, open and semi-open. A closed exercise restricts the learner's choice of answers. This is the case of cloze[4] and multiple choice tests. An open exercise allows the learner to produce a free answer, hardly predictable. A good example of this is the essay question. In between, semi-open exercises allow several answers, but restrict the answer to a finite set of possibilities. We must draw the attention to the fact that this set is unknown to the learner, and even to the teacher himself. Its size greatly depends on lexical and syntactic variations implicitly allowed by the question. A well-known semi-open exercise is guided translation.

Nowadays, in commercial CALL applications, exercises are automated: the system itself asks the learner, checks the correctness of the answer and returns feedback to the learner, without any human intervention. Because of this, exercises are limited to the closed type, which is seen by suppliers as a means of ensuring the quality of the feedback offered and, in doing so, respecting a basic principle of teaching and learning: to avoid distracting the learner by offering incorrect explanations (Tschichold 2006).

The experts, however, consider that it may be very productive to offer semi-open exercises, which encourage spontaneity of answers because they avoid indicating the location of the difficulty too explicitly (Desmet 2006). Meeting this expectation is one of PlatON's primary objectives. Our initial assumption was that answers to semi-open exercises must be manageable by automated systems, because they belong to a finite set of possibilities which, somehow, must be predictable.

In the specific area of spelling teaching and learning, one possible semi-open exercise is dictation. Dictation is an educational exercise, frequently used in the French-speaking world, the objective of which is to assess the standard quality of a learner's spelling. In this exercise, a person reads aloud a text to a learner, who copies it down on paper. This copy is then manually corrected, typically by a teacher. Paradoxically, this exercise that is traditional in the teaching of French as a native language has been disparaged for a long time by numerous educationalists: they regard it as a means of checking rather than as a learning exercise, and criticize its artificiality, detached from the real use of the language (Jaffré 1992).

However, a number of studies have highlighted the pertinence of dictation as a means of assessment and improvement of the language proficiency level. Thus, in terms of second language learning, Irvine et al. (1974) observed a high degree of correlation between the results obtained in the TOEFL (Test Of English as Foreign Language) and the results obtained in dictation. Rahimi (2008), for his part, has compared the progress of two groups of Iranians learning English. In this study, only one of the groups performed a dictation at each session, the main difficulties of which were discussed after correction at the next session. The results obtained have revealed that even though both groups did make some progress, the standard achieved in grammar, reading, vocabulary, and aural comprehension by the group who had performed dictations was significantly higher.

[4] A cloze test consists of a portion of text with certain words removed. The learner is then asked to supply the missing words.

In the same line of thinking, we may speculate that metalinguistic reflection by the learner on his own errors and the reorientation of the teaching offered in accordance with the learner's errors is likely to increase the standard achieved at the end of a course. Naturally, applying this sort of approach manually, for a group of learners of a certain size, would soon become unmanageable for the teacher, who would have great difficulty adapting to each learner's difficulties. It is here that the interest of automating the exercise within a CALL platform ought to be seen: depending on the learner's actual difficulties, the system would be able to direct them towards reading sheets relevant to them, and offer them tailored exercises, maximizing the emphasis on the specific difficulty to be addressed within a text. This is the context within which we are developing PlatON: the idea is to develop the exercise of dictation from the 'punishment' tool it used to be into an effective means of targeting the areas to be revised and practising self-training.

Within the context of a platform that seeks to be comprehensive, automating an exercise like dictation requires three distinct phases to be taken into account: (1) support for the introduction of a new dictation by the teacher; (2) automation of the exercise as performed by the students, during which the dictation must be read aloud to them and their copy must be saved; (3) correction of the learner's copy and displaying the result.

In a previous paper (Beaufort and Roekhaut 2011), we concentrated on the third phase, presenting the main aspects of an original algorithm for automatic dictation correction. Here, we propose to present this algorithm in context, explaining how it integrates into the PlatON platform. We also give an in-depth description of the algorithm and a complete evaluation of its performance.

Section 2 situates automatic dictation and the notion of feedback within CALL. Section 3 presents the tools we use, as a prerequisite for understanding the correction algorithm detailed in section 4. This algorithm is organized schematically into three phases: detecting errors, tagging errors and establishing feedback. We then evaluate the system in section 5 before concluding in section 6.

2. State of the art

Little work has been done directly concerning automation of dictation. The most significant work in this field is probably that of Santiago-Oriola (1998). Her system, DICTOR, automates all the stages in the performance of a conventional dictation. The area of this work which we are interested in here is the proposed correction method: this is based on the fact that the link between the written form and the pronunciation is not easy to learn in French: only 80–85% of the letters in a given text represent a phoneme, which is the cause of many spelling mistakes (Catach 1995). On this basis, the correction proposed is split into two modules: the first takes care of detecting and classifying errors by performing an alignment of the original and the copy, in accordance with phonogramic[5] and morphogramic[6] transformation rules. Then, the second module simply selects pre-defined feedback, selected

[5] Phonetic equivalence of certain graphemes. For example, "eau", "au", and "o" are all pronounced as /o/ in French.

[6] Morphological variations of graphemes on the paradigmatic axis: gender and number of nouns and adjectives, persons of verbs.

according to the transformation rules applied in the first module. Hence we may consider that the correction is obtained in one go, given that all the intelligence of the system lies in the first module. The system has been tested in a CM2 class[7] on three very short texts (11, 19, and 41 words). For these three dictation exercises, the class was split into three groups having equivalent levels in spelling. The first group was a control group that performed the dictation in the traditional way; the other groups used DICTOR. The evaluation covered the ergonomics of the system and demonstrated the pupils' and their teacher's interest in using the tool, despite the difficulty associated with the use of a computer keyboard. It has not been tested on a larger scale, and the error detection and feedback performance has not been evaluated.

More recently, the dictation software *La Dictée interactive* has been presented in *ALSIC*, devoted to language learning (Ruggia 2000). This article, concentrating on presenting the tool's potential, gives only one piece of information about the error correction method: it targets common errors amongst Italian-speaking learners at false-beginner[8] level in French. Hence we claim that this method is not very generic.

In the last few years, dictation has completely disappeared from the scientific literature. However, in the field of semi-open exercises, we can note the work by Desmet and Héroguel (2005), whose foreign language learning platform allows, among other things, the correction of sentences translated from a source language into a target language. The correction principle proposed is not dissimilar to the dictation exercise in which the original is available: the idea here is to produce several expected answers (the "originals"), and to select, by approximate string matching, the formulation to which the learner's answer is closest. The system, which operates at word level, then indicates the errors to the learner by replacing a wrong word with XXX, a superfluous word by (XXX), and a missing word by (...). However, it does not offer any other form of feedback.

As far as we are aware, commercial applications do not yet incorporate NLP-based diagnostic tools, a fact that was also noted by L'haire (2004). Yet Heift (2004) has demonstrated the real impact in pedagogic terms of correction methods that indicate the location of the error and provide a linguistic explanation for it.

In the scientific world, two very interesting approaches have been suggested. The one by L'haire and Vandeventer Faltin (2003) applies to open exercises: sentences produced freely by learners in response to questions posed. In this context, the diagnostic system operates in three stages: (1) analysis of the lexical forms in order to detect forms that are not in the vocabulary, (2) production of the parse tree for the sentence, then progressive relaxation of local restrictions in order to detect agreement errors between elements, and (3) comparison of the deep semantic structures of the learner's sentence and the answer expected by the teacher. At each stage, the system presents the errors detected to the learner, who has to correct their sentence before the next stage is applied to it. It would not be possible to apply this type of approach to dictation, given the interactions required with the learner before final feedback is provided.

[7] Primary school class, approximate age 9–10 years.

[8] A false-beginner is a language learner who starts to study a language from the beginning again, although he already has a slight knowledge of it, varying between levels A1 and A2 of the CEFRL.

In the context of automatic correction of automatically-generated closed exercises, Kraif and Ponton (2007) establish feedback by comparing the expected answer and the answer given by the learner on various levels. The comparison first looks at graphical differences (capitals/lowercase, space errors). If the incorrect form is not in the lexicon, the feedback is established in accordance with the degree of difference between the two forms. The feedback then indicates accent errors (differences in diacritics), lexical spelling mistakes (slight differences between the forms) or form errors (substantial differences between the form). If the incorrect form is present in the lexicon, the feedback is established in accordance with the similarities and differences in the morpho-syntactic analysis of the two forms. In this way, if both forms belong to the same lemma but differ in gender or number, the feedback will indicate a grammatical error in gender or number. If the two forms come from different lemmas, despite having an identical grammatical analysis, the system will indicate to the user that the answer given is correct, but is different from the answer expected. The way we produce the automatic feedback in PlatON shares many similarities with this approach.

3. Prerequisites

The correction algorithm presented in section 4 is entirely implemented using finite-state machines (FSMs) and depends on a morpho-syntactic analysis of the dictation original.

Finite-state machines. Due to the brevity of this overview, we urge the reader who is not familiar with FSMs to consult the state-of-the-art literature (Mohri et al. 2000, Mohri et al. 2001, Mohri and Riley 1997, Roche and Schabes 1997). FSMs, which include finite-state automata (FSAs), finite-state transducers (FSTs) and their weighted counterparts (WFSAs and WFSTs), can be seen as defining both a class of graphs and a class of languages.

An FSM can simply be considered as an oriented graph with labels on each arc. Transitions of FSAs are labeled with symbols from a single alphabet Σ, while transitions of FSTs are labeled with both input and output symbols belonging to two different alphabets Σ_1 and Σ_2. Weighted machines put weights on transitions in addition to the symbols.

One may also consider FSMs as defining the class of regular languages. In this definition, an FSA is an *acceptor*: it represents the set of strings over Σ for which there is a path from the initial state to a final state of the graph. In contrast, an FST translates strings of a first language over Σ_1 into strings of a second language over Σ_2; hence, it defines *relations* between languages. In weighted machines, weights, which encode probabilities or distances, are accumulated along paths to compute either the overall weight of a string (in WFSAs), or the overall weight of mapping an input string to an output string (in WFSTs). WFSMs are thus a natural choice for solving the n-best-strings problem.

A few fundamental theoretical properties make FSMs very flexible, powerful and efficient. Among them, the composition ($\circ$), a generalization of automata intersection: from an FST T_1 working on Σ_1 and Σ_2, and an FST T_2 working on Σ_2 and Σ_3, the composition computes their intersection on Σ_2 and builds the FST T_3 working on Σ_1 and Σ_3. Our algorithm directly relies on this operation.

Another very important property of FSMs is their ability to model sets of rewrite rules (Johnson 1972). Rewrite rules take the following general form:

$$\phi \quad \rightarrow \quad \psi \quad :: \quad \lambda \quad \_ \quad \rho \tag{1}$$

which indicates that ϕ must be rewritten as ψ when surrounded by λ and ρ. This general form may be extended to allow for weighted optional rules:

$$\phi \quad ?\!\rightarrow \quad \psi \quad :: \quad \lambda \quad \_ \quad \rho \quad / \quad w \tag{2}$$

which expresses that the replacement $\phi \quad ?\!\rightarrow \quad \psi$ is optional, but gets the weight w when it occurs. Developed in the framework of generative phonology, these rules are now widely used in many areas of natural language processing.

We used our own finite-state tools: a finite-state machine library and its associated compiler (Beaufort 2008). In conformance with the format of the library, the compiler builds finite-state machines from weighted rewrite rules, weighted regular expressions and n-gram models.

Morpho-syntactic analysis. The analysis of the dictation original is produced by the eLite system (Beaufort and Ruelle 2006, Beaufort 2008). This analyzer stores its analysis of a text in the layered structure depicted in Figure 1. The goal of the *Token* layer is to identify sequences forming one unit, like urls, phones or currencies. At this level, a lexical form is considered as an *Alphabetical Token*. The *Unit* layer gathers lexical forms like compound nouns (*pomme de terre*, 'potato') or verbs (*a mangé*, 'has eaten'), to make the contextual disambiguation easier, by attributing a single category to the whole unit (*pomme de terre*, noun), (*a mangé*, verb). The *Word* layer is the smallest sequence of characters considered to form one unit. In an *Alphabetical Token*, a *Word* is an inflected form. In a *Url Token*, a *Word* is a part of the url, like the protocol, the hostname or the domain. In a *Punctuation Token* made of several punctuation marks (for instance, a double quote followed by a period), a *Word* is a single punctuation mark. Spaces are not stored in the structure. When the text in the data structure is printed out, a SmartPrint function regenerates the spaces required in accordance with the typographic conventions of the language concerned.

eLite first pre-processes the whole text to detect paragraphs, sentences and tokens. Then, it carries out, sentence by sentence, a morphological analysis and a contextual disambiguation. The morphological analysis performs, at the *Word* level, a lexicon look-up to determine the set of possible categories for each word given its token kind. Then, it creates the *Unit* level by detecting compounds. The contextual disambiguation works at the *Unit* level, and applies a statistical language model (Beaufort et al. 2002) to reduce the set of categories of each unit to the most likely given the context.

Integration into PlatON. When a new text is added by a teacher on the platform, the system prepares it for dictation: first, it produces its morpho-syntactic analysis, then it generates a vocal version of it[9].

Each time a learner starts an exercise, the system loads the associated analysis and sound files. Guided by the analysis, the system lets the learner listen and type down only

Paragraph → *Sentence* → *Token* → *Unit* → *Word*

Figure 1: Layered structure of eLite

[9] The platform allows the teacher to choose between synthetic and natural speech. The synthetic speech is automatically generated by eLite, which is a text-to-speech synthesizer. The natural speech is simply recorded using a dedicated flash recorder.

one sentence at a time. This allows the system to unambiguously identify the sentence boundaries[10] in the learner's copy, even if the learner makes mistakes and involuntarily deletes (parts of) sentences.

At the end of the exercise, original, copy and morpho-syntactic analysis of the original are sent to the correction module, which performs its work. The morpho-syntactic analysis supports the correction module on two levels:

1. the presence of sentence boundaries allows the correction module to be applied sentence by sentence, considerably reducing the complexity of the process;

2. the category associated to each *Word* of the text contributes to the third part of the correction, the feedback establishment.

When the correction is finished, the result is stored in the databases of the system and linked to the learner's account, which makes it available to both the learner and the teacher. The result is presented to the user as an HTML page in which errors are highlighted in red (see Figure 6). When the user passes the mouse pointer over an error, the corresponding feedback appears in a pop-up window, which lasts until the user takes the mouse off the error (see Figure 7).

4. Correction algorithm

Figure 2 shows an example of an original and a copy. This artificial text is used as the basis for Figures 3, 4 and 5, which illustrate the various steps in the algorithm. Figures 6 and 7 show the result of the whole algorithm applied on this example. The correction algorithm involves three stages:

1. detection of error positions and boundaries;

2. for each error, assignment of tags allowing the type of error to be characterized;

3. for each error, generation of feedback on the basis of the tags assigned in (2). At this stage, if necessary, a comparison of the morpho-syntactic analyses of the correct and incorrect forms is performed.

Detection. Detection of the learner's errors is based on alignment of the original and copy sentences. This principle is the sole point of comparison with the algorithm by Santiago-Oriola (1998). Our alignment is based on the theoretical foundations of the standard edit distance and its enhanced version in the form of finite-state machines.

Conventionally, the alignment of two sequences is calculated by approximate string matching via their edit distance (Damerau 1964, Levenshtein 1966). Now, standard edit distance allows only basic operations: substitution, insertion and deletion of a character, and transposition of two adjacent characters. In our case, this poses a problem, because learner errors often correspond to substitutions of the type "n-m", where n characters are replaced by m characters: "-es‘" ↔ "-ent", "ait" ↔ "-aient", "-er" ↔ "-ées", etc. In the context

[10] An invisible "Sentence Boundary" marker is inserted after each sentence.

of standard edit distance, n-m substitution is modelled in the form of several edit operations; this tends to move it away from pertinent solutions, given that the distance attributed to it ends up being the sum of several operations. In order to circumvent this limitation, we have resorted to using finite-state machines and a method that we have described in (Beaufort 2010): given two sequences x and y represented in the form of finite-state automata $\mathcal{X}$ and $\mathcal{Y}$, we construct the weighted transducer $\mathcal{E}$ corresponding to the set E of possible alignments between x and y. This set is obtained through the cascade of composition:

$$\mathcal{E} = \mathcal{X} \circ \mathcal{F} \circ \mathcal{Y} \tag{3}$$

where $\mathcal{F}$ is a weighted transducer that models the accepted edit operations. The best alignment between x and y corresponds to the best path in $\mathcal{E}$, obtained by calculation of the shortest path of a graph. The method is called ‘filtered composition’, because the weighted transducer $\mathcal{F}$ can be regarded as a filter that determines the size of the intersection between x and y. The filter F is compiled into the form of a transducer $\mathcal{F}$ using a set of context-free weighted optional rewrite rules:

$$\begin{array}{lll}
\text{ais} & ?\rightarrow & \text{ait} \;/\; 1 \\
\text{ais} & ?\rightarrow & \text{aient} \;/\; 1 \\
\text{ait} & ?\rightarrow & \text{ais} \;/\; 1 \\
\text{a} & ?\rightarrow & \text{e} \;/\; 1.5 \\
\text{a} & ?\rightarrow & \text{b} \;/\; 2 \\
\text{a} & ?\rightarrow & \text{" "} \;/\; 7 \\
\ldots & &
\end{array}$$

These rules were designed manually. The complete set counts 25,962 rules. A significant part of them describes how morphological suffixes of French may be confused because of phonetic equivalences, while the other focuses on substitutions of single characters. Weights were tuned without any try on the test set used for the evaluation (see Section 5). The idea was just to give preference to phonetically-equivalent suffixes (*ais* ↔ *ait*), then to substitutions of characters belonging to the same class (*a* and *e* are both vowels) and finally, to rewrites between letters and separators (*a* ↔ “ ”).

The error detection algorithm is entirely built around this principle and is split into three stages:

Original	**Copy**
Le plus jeune pourrait demander un entretien au conseil, quoi qu'en disent ses aînés et quelles qu'en soient les conséquences.	*Le jeune pourraient demandé run n'entretien au conseil quoiqu'en dise ces aîners et qu'elles quen soient les conséquences.*

Figure 2: An example of a dictation and learner's copy. Translation: “The youngest might request an interview to the council, no matter what his elders say and whatever the consequences.”

(1) the original sentence x and the copy y are converted into finite-state automata $\mathcal{X}$ and $\mathcal{Y}$. The two automata are composed via the filter $\mathcal{F}$. The result of this composition, the transducer $\mathcal{E}$ of the possible alignments between x and y, is then reduced to its best path $\mathcal{E}'$, corresponding to the best alignment of x and y;
(2) the transducer $\mathcal{E}'$ is converted into three vectors: one for the original sentence, one for the copy, and one for the weighting associated with the operations performed. In the weighting vector, a positive weighting indicates the start of an edit operation (Figure 3 a);
(3) the three vectors are run through in parallel in order to enclose the errors within markers [and] which indicate their boundaries. The system considers that an error starts when the weighting is other than 0, and ends when the weighting becomes 0 again and both letter vectors represent identical characters (Figure 3 b).

At this stage, the proper detection is finished, and the tagging phase can begin.

Error tagging. The objective of tagging is to perform the best possible classification of the errors, in order to make feedback easier. To this end, we identify:
(1) errors within a word, whose position is noted: at the beginning (*BeginWord*), at the end (*EndWord*), or within the word (*InWord*). These tags are not mutually exclusive: one word may contain several errors located in different positions;
(2) errors that span across several words (*MultiWord*);
(3) errors between two words (*BetWord*);
(4) Missing (*WordMiss*) or spurious (*WordExtra*);
(5) errors consisting exclusively of separators (*OnlySep*), whether punctuation or spaces.

We recall the reader that a word here refers to an element in the *Word* layer, which can be a lexical form or a punctuation mark. Because spaces are not stored in the data structure, a space error (spurious or missing) will be tagged *BetWord*. On the other hand, a lexical form or a punctuation mark added between two words will be tagged *WordExtra*, and an omitted lexical form or punctuation mark will be tagged *WordMiss*.

Hence it is necessary to identify the word boundaries in the dictation and copy vectors. To achieve this, we insert markers: { at the start of a word, and } at the end of a word. These will make it easier to subsequently calculate the tags to be assigned to the errors.

The word boundary marker insertion algorithm is guided by the linguistic data structure. Going through the elements in the *Word* layer makes it possible to identify the boundaries of the current word in the dictation vector, and to insert the markers in the positions identified in the two vectors (Figure 4 a). Then the marker positions are adjusted where necessary, in order to include within the current word the contiguous errors corresponding to insertions of alphabetic characters (Figure 4 b). This is the case with the ‘r’ at the start of the form ‘run’, which will be included into the word and tagged *BeginWord*. On the other hand, the ‘n’ at the start of the form ‘n’entretien’ will not be included in the word, because of the presence of a separator, the apostrophe. This insertion, left outside the word, will be tagged *WordExtra*.

Once the calculation of the tags to be assigned to the errors surrounding a word is finished, the errors and their tags are entered into the data structure. When the error is an inserted word (*WordExtra*) or involves a space omitted or inserted (*BetWord*), an element

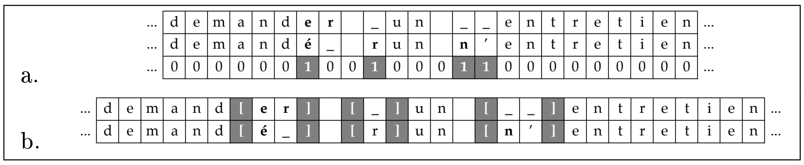

a.
... d e m a n d e r _ u n _ _ e n t r e t i e n ...
... d e m a n d é _ r u n n ′ e n t r e t i e n ...
... 0 0 0 0 0 0 1 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 ...
b.
... d e m a n d [e r] [_] u n [_ _] e n t r e t i e n ...
... d e m a n d [é _] [r] u n [n ′] e n t r e t i e n ...

Figure 3: Detection of error positions and boundaries

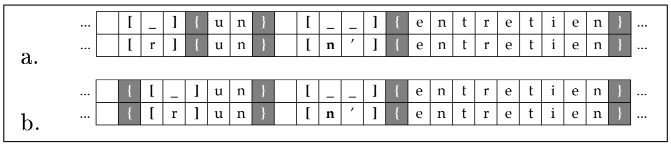

a.
... [_] { u n } [_ _] { e n t r e t i e n } ...
... [r] { u n } [n ′] { e n t r e t i e n } ...
b.
... { [_] u n } [_ _] { e n t r e t i e n } ...
... { [r] u n } [n ′] { e n t r e t i e n } ...

Figure 4: Detection of word boundaries

is created at the appropriate point in the *Word* layer and receives the error and the relevant tags.

Feedback. At this stage, all the information available is saved in the *Word* elements of the data structure: the morpho-syntactic analysis of the correct form and, if there has been an error, the incorrect form and the tagging generated. Feedback is only triggered for *Word* elements containing an error. Overall, the errors concern one of these three categories: (1) a separator, (2) a word or (3) a sequence of several adjacent forms (words/spaces/punctuation marks).

Whatever the category, two tags directly decide on the feedback: *WordMiss* and *WordExtra*. The word, punctuation or sequence is labelled as "missing" if the tag is *WordMiss* (Figure 7 a and e), and "spurious" if the tag is *WordExtra* (Figure 7 c). Other feedback is only proposed when none of these two tags is associated with the error.

The *BetWord* tag indicates that a separator is missing, or spurious: a missing separator may indicate a merging into a form that is in the lexicon (*quoi que* → *quoique*, *qu'elles* → *quelles*); a spurious separator, a split into several forms that are in the lexicon (*quoique* → *quoi que*, *quelles* → *qu'elles*). Thus if the tags lead it to do so, the feedback algorithm starts by testing an error across several words, and only offers other feedback if this test has failed. However, for clarity, we are going to start by detailing the operation of the feedback on separators and on lexical forms.

1) When the tag associated with a separator error (punctuation or space) is neither *WordMiss*, *WordExtra* nor *BetWord*, the feedback is very simple to produce. The separator is incorrect, and the feedback simply indicates that a different separator was expected.

2) A word error may be lexical and/or grammatical. An error is lexical if the incorrect form is out-of-vocabulary (*run*, Figure 7 b, does not belong to the French lexicon) or belongs to the same category as the correct form, but has a different lemma (*sceptique* ↔ *septique*). An error is grammatical if the incorrect form exhibits grammatical features that differ from those of the correct form (*parle* ↔ *parles* differ in terms of the person). An incorrect form may of course include both lexical and grammatical errors (*différent* ↔ *différant* includes a lemma error and a category error). In order to offer one of these elements of feedback, we start by looking up the incorrect form in the lexicon. If the form is not present there, it is deemed to be out-of-vocabulary. Otherwise, the idea is to compare the linguistic analysis

adopted when the dictation was prepared to the lattice of possible analyses proposed by the lexicon for the incorrect form, and to adopt the incorrect form analysis that is closest to that of the correct form. Whether considering a form that is correct or incorrect, a linguistic analysis is always made up of a lemma and the following grammatical features: tense/mood, gender, number, person. The method we are using for comparing analyses is very similar to the alignment method we have presented above. It is illustrated in Figure 5: the correct form analysis (a_1) and the lattice of incorrect form analyses (a_2) are compiled into finite-state automata ($\mathcal{A}_1$ and $\mathcal{A}_2$ respectively). On this basis, the best analysis to be retained for the incorrect form ($\mathcal{A}'_2$) corresponds to the best path from the composition of these two automata via a filter $\mathcal{F}_t$:

$$\mathcal{A}'_2 = \text{Best}(\mathcal{A}_1 \circ \mathcal{F}_t \circ \mathcal{A}_2) \tag{4}$$

where the filter allows weighted conversions between grammatical features. For example, an infinitive may be converted into a past participle at a cost of 1 and into a noun at a cost of 5. Hence the best path between the two analyses is the one that produces the least-costly feature conversions.

When the lemmas for the two forms differ (*sceptique* ↔ *septique*), the composition of the automata fails. In this event, the error is at least lexical (Figure 7 g). However, we still have to select the analysis of the incorrect form and to test for a possible grammatical error. The same calculation is for this reason reproduced on two new automata, containing only the grammatical features of the two forms to be compared. This composition always yields a result.

3) In principle, the analysis of a sequence error is performed in the same way as for a word: the incorrect sequence is looked up in a lexicon. However, if no result is returned, the sequence is not considered to be out-of-vocabulary; the feedback is simply oriented towards one of the other two error types.

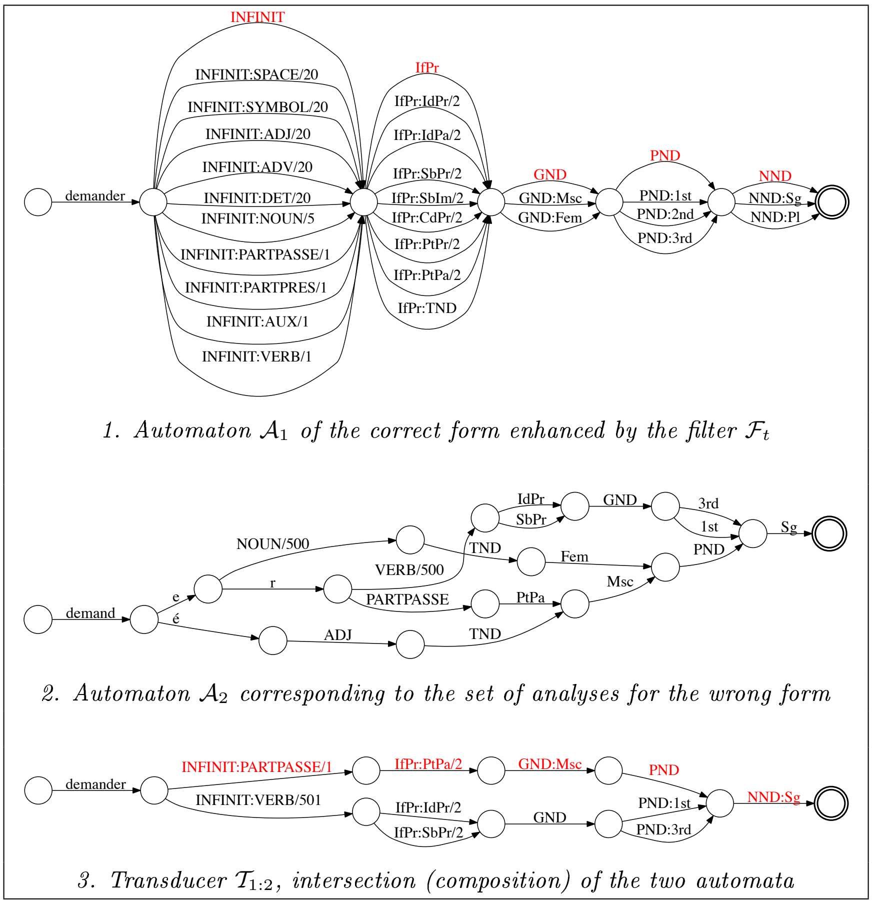
Looking up the incorrect sequence in the lexicon differs in two respects from looking up a word: the sequence to be looked up has to be *constructed*, and an appropriate lexicon has to be *selected*.

In the case of a missing separator (*quoique* for *quoi que*, *quelles* for *qu'elles*), it is assumed that the incorrect form is a word. Correct words (for example, *quoi* and *que*) are in this case concatenated without a separator (*quoique*) and looked up in the same lexicon as that used for analysing word errors (Figure 7 f).

In the case of a spurious separator (*quoi que* for *quoique*, *qu'elles* for *quelles*), it is assumed that the incorrect form contains several individual correct forms. In this case, the fragments of words (for example, *qu* and *elles*) are concatenated around the spurious separator (*qu'elles*) and looked up in a lexicon corresponding to the following regular expression (Figure 7 h):

$$(WordApo \quad | \quad (Word \quad Sep))^{+} \quad Word \tag{5}$$

where $WordApo$ is a word ending in an apostrophe (*d'*, *qu'*, etc.) and *Sep* is a space or hyphen. Thus this expression simply permits a sequence of words that obey the typographic conventions for French.



1. *Automaton $\mathcal{A}_1$ of the correct form enhanced by the filter $\mathcal{F}_t$*

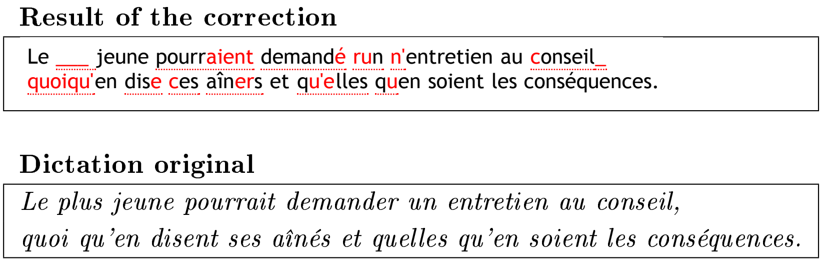
2. *Automaton $\mathcal{A}_2$ corresponding to the set of analyses for the wrong form*

3. *Transducer $\mathcal{T}_{1:2}$, intersection (composition) of the two automata*

Figure 5: Feedback. Computation of morphological differences between expected and wrong forms, through automata composition. The expected form is "*demander, verb, infinitive*" ($\mathcal{A}_1$). The wrong form is "*demandé, verb, past participle, masculine, singular*" ($\mathcal{A}_2$). They differ in terms of mood/tense, gender and number ($\mathcal{T}_{1:2}$). Table 1 presents the list of symbols used in these finite-state machines.

1st	first person	IdPa	past indicative	Pl	plural
2nd	second person	IdPr	present indicative	PtPr	present participle
3rd	third person	IfPr	present infinitive	PtPa	past participle
ADJ	adjective	INFINIT	infinitive	SbIm	imperfect subjunctive
ADV	adverb	Msc	masculine	SbPr	present subjunctive
CdPr	present conditional	NND	number undefined	Sg	singular
DET	determinant	NOUN	noun	SPACE	space, blank
Fem	female	PARTPASSE	past participle	SYMBOL	symbol
GND	gender undefined	PND	person undefined	TND	tense undefined

Table 1: List of symbols used in Figure 5



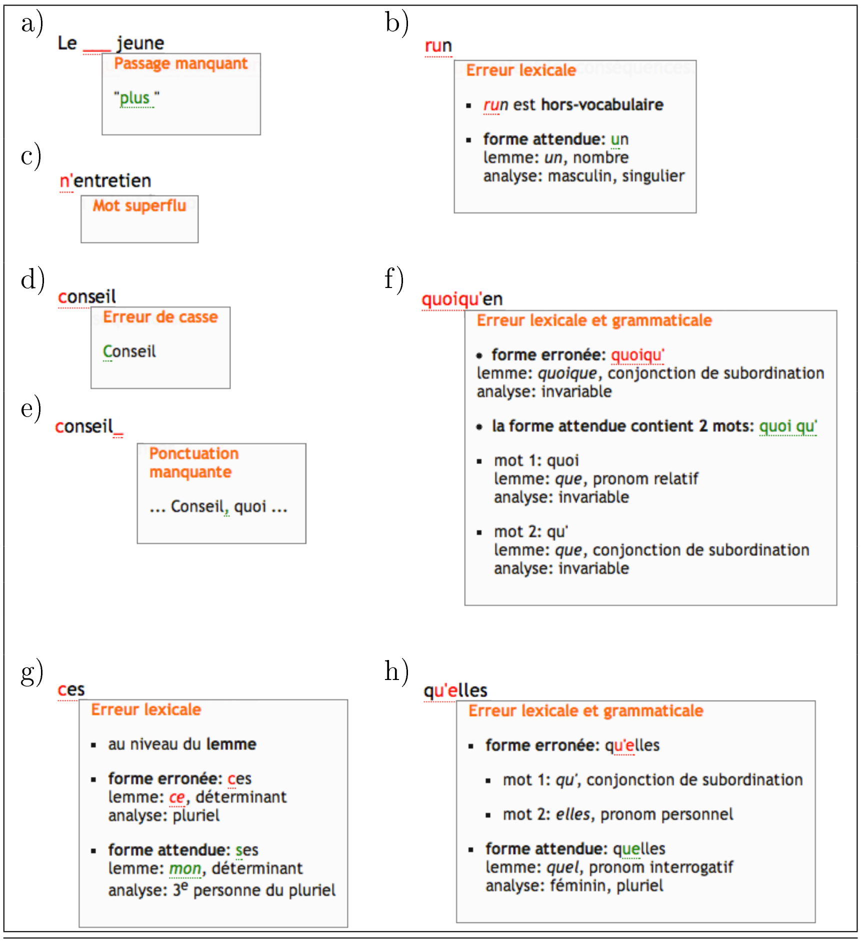
Result of the correction

Le ___ jeune pourraient demandé run n'entretien au conseil_
quoiqu'en dise ces aîners et qu'elles quen soient les conséquences.

Dictation original

Le plus jeune pourrait demander un entretien au conseil,
quoi qu'en disent ses aînés et quelles qu'en soient les conséquences.

Figure 6: Result of the correction compared with the dictation original. In the correction, a wrong word is underlined in red and inside it, the error itself is printed in red.



a) Le ___ jeune

Passage manquant

"plus "

b) run

Erreur lexicale

- *run* est hors-vocabulaire
- forme attendue: un
  lemme: *un*, nombre
  analyse: masculin, singulier

c) n'entretien

Mot superflu

d) conseil

Erreur de casse

Conseil

e) conseil_

Ponctuation manquante

... Conseil, quoi ...

f) quoiqu'en

Erreur lexicale et grammaticale

- forme erronée: quoiqu'
  lemme: *quoique*, conjonction de subordination
  analyse: invariable
- la forme attendue contient 2 mots: quoi qu'
- mot 1: quoi
  lemme: *que*, pronom relatif
  analyse: invariable
- mot 2: qu'
  lemme: *que*, conjonction de subordination
  analyse: invariable

g) ces

Erreur lexicale

- au niveau du lemme
- forme erronée: ces
  lemme: *ce*, déterminant
  analyse: pluriel
- forme attendue: ses
  lemme: *mon*, déterminant
  analyse: 3^e personne du pluriel

h) qu'elles

Erreur lexicale et grammaticale

- forme erronée: qu'elles
  - mot 1: *qu'*, conjonction de subordination
  - mot 2: *elles*, pronom personnel
- forme attendue: quelles
  lemme: *quel*, pronom interrogatif
  analyse: féminin, pluriel

Figure 7: Examples of feedback, illustrating some errors of Figure 6. Feedback is shown in pop-up windows, initially hidden to the user. A given pop-up appears when the user passes the mouse pointer over the corresponding error.

5. Evaluation

For 40 years (1969–2008), Prof. Michèle Lenoble-Pinson at the Facultés Universitaires Saint-Louis (Brussels, Belgium) gave her 1$^{\text{st}}$ and 2$^{\text{nd}}$ year degree students a dictation examination. In all, the corresponding corpus contains 40 dictations, with a total of 1,300 copies, which have recently been digitized (Fairon and Simon 2009). This digitization provided the occasion to classify the 13,255 errors in the copies into various types, which we shall be presenting later. In the remainder of this article, the digitized version of this corpus is referred to as *Digitized Lenoble-Pinson* (DLP).

Our evaluation was carried out on 5 years of the DLP (one year in five from 1969 onwards), *i.e.* 5 different dictations, resulting in 115 copies that include 1,532 errors. We evaluated the correction system in terms of efficiency and performance. The performance tested involved the detection module and the feedback module.

Efficiency. The evaluation was carried out on an HP Compaq 8000 Elite, Intel Core Quad CPU 2,66GHz 64 bits, 4 GB RAM, running Ubuntu 10.04.2. The application was compiled in such a way as to use only a single CPU.

The execution time corresponds to the interval between the moment the system receives a file for processing and the moment the response is completely generated in the form of an HTML file capable of being displayed in a browser. On average, the system processes a character in 0.17 ms (σ 0.04 ms) and a word in 1.05 ms (σ 0.23 ms). Thus overall the system is efficient and fairly consistent. For instance, a dictation of 56 sentences, containing 3,559 characters in 566 words, is processed in 590 ms (0.16 ms per character, 1.04 ms per word).

Evaluation of the detection. The evaluation was semi-automatic. First, a script performed an "improved diff", parsing the DLP and the output of our system in parallel. This allowed to automatically point out the differences, in terms of error location, between the two corpora. Then, the differences were compared manually. The successful detection rate is 99%: out of 1,532 errors, 11 were incorrectly aligned by the system. These mistakes are all of the same type: they concern spurious or missing passages that should have been considered as a whole, but were split into pieces. In order to understand this phenomenon, here is an example of a missing passage aligned incorrectly:

```
...  français____________________e_______________________________.
...  française, quels qu'en puissent être la gravité et le nombre.
```

Contrary to all expectations, the system has aligned the 'e' of 'française' to the 'ent' of 'puissent'. After analysing the weighting of the different possible alignments, this error is in fact due to the filter, which gives preference to the substitution $e \leftrightarrow ent$ (cost = 1) and by the same token, avoids two deletions (n and t, at a cost of 3 per deletion).

However, we varied the costs assigned to the various operations: in spite of everything, this type of unwanted alignment recurred on other strings. This is due to the very principle of edit distance, the objective of which is first and foremost to determine the *minimum* number of operations to allow one string to be converted into another, even if this is to the detriment of their alignment. In biology, where the identification of sequences common to two sections of DNA is a real prerequisite in establishing filiations or genetic mutations, the solution adopted has been to calculate *local alignments*, making it possible to identify identical or extremely similar sequences, prior to determining the best way of aligning the divergent sequences using edit distance (Gusfield 2007). We intend to evaluate this solution, which seems entirely suitable for dictation

Feedback quality. Table 2 lists the types of errors from PlatON and from the DLP. Many of the categories and sub-categories in the two lists are identical or very similar. This is the case for the category "punctuation" and most of the "transcription" errors. However, it can be noted that the sub-categories "illegible sequence" and "hyphenation problem" do not appear in PlatON. In fact, these errors were apparent in the hand-written copies, but disappeared from the dictations when they were coded. Conversely, space errors appear under PlatON, but not for the DLP. These, too, correspond to typing mistakes, and thus to errors not present in the hand-written dictations.

Contrary to the DLP, PlatON offers a three-level feedback for the categories "usage", "grammar", and "usage/grammar":

1. a **superficial description**, which is the error's category: *lexical*, *grammatical* or *lexical and grammatical*;

2. an **in-depth description**, which highlights the morphological features of the error. For instance, the in-depth description of a grammatical error points out the gender and/or the number when the error occurs in a noun, while it points out the mood/tense and/or the person when the error occurs in a verb;

3. the **complete morphological analysis** of both correct and wrong forms.

Actually, establishing this three-level feedback starts out from the morphological analysis and ends up with the superficial description. For example, a morphological analysis that highlights a difference in terms of gender between two forms leads to the in-depth description "error in terms of gender", which in turn allows the superficial description "grammatical error" to be offered. As all this information was available to us, it seemed to us worthwhile to present it in its entirety, rather than restricting ourselves to a superficial description.

The DLP error type list does have the advantage of having a "homophony" category, unlike that of PlatON. Given that PlatON does not currently have a phonetic comparison module, we preferred to leave the homophones of the corpus in the categories "usage", "grammar" and "usage/grammar". Although less precise, this classification is however not incorrect. All the same, it should be noted that the classification in the DLP probably needs to be made more uniform. For example, the confusion *quelque* ↔ *quel que* is correctly classified under "homophony", unlike *quelquefois* ↔ *quelle que fois*, which is classified under the "usage" category.

The evaluation of the feedback only covered the 1,521 errors that were correctly detected (aligned) by the system. The evaluation was carried out in a semi-automatic manner. For the categories that are identical or similar in both systems, validation was carried out automatically. For the other categories, the PlatON feedback was validated manually. We evaluated the pertinence of the superficial and in-depth description. For the errors in the "usage", "grammar" and "usage/grammar" categories, we also evaluated the quality of the morpho-syntactic analysis.

The results obtained are given in Table 3. In its current state, the superficial description is 97.2% correct, the in-depth description 96%, and the morpho-syntactic analysis 87.7%.

	PlatON	*LPN*
Usage	**Lexical errors** + out-of-vocabulary word + *analysis of the correct form*	Usage + spelling
	Case/diacritic error	Usage + case
Grammar	**Grammatical error** + number and/or gender + *analysis of the two forms*	Grammar + adjective
		Grammar + determinant
		Grammar + noun
		Grammar + past participle
	Grammatical error + mood/tense and/or person + *analysis of the two forms*	Grammar + verb + mood/tense/person
Usage and grammar	**Lexical and grammatical error** + lemma and category + *analysis of the n/m forms*	
Homophony		Homophone
Punctuation	**Spurious punctuation**	Punctuation + spurious
	Missing punctuation	Punctuation + missing
	Punctuation error	Punctuation + error
Transcription	**Missing word**	Transcription + missing sequence
	Missing sequence	
	Missing sentence	
	Spurious word	Transcription + spurious sequence
	Spurious sequence	
	Spurious sentence	
		Transcription + illegible sequence
		Transcription + hyphenation problem
	Spurious space	
	Missing space	
	Space error	

Table 2: Correspondence between feedback. In the PlatON typology, superficial description is in bold, in-depth description, in normal and morphological analysis, in italic font. Original typologies were in French.

In all, the quality of the results decreases with the depth of the analysis. This is due to the fact that an error in an analysis at a given depth does not always have a negative effect on the quality of the analysis proposed at the level above. For example, let us assume that the learner writes “rappelle” instead of “rappellent”, and that the correct form should be a subjunctive. If the system considers that both forms are in the indicative, there is an error in the morpho-syntactic analysis. However, this error will have no influence on the in-depth description, which will specify that the error in fact lies in terms of the number of the verb. Along the same lines, if the in-depth description “error in terms of gender and number” is offered, when all that was expected was “gender error”, the superficial description will be identical all the same: “grammatical error”.

In certain cases, the morpho-syntactic analysis fails because the lexical form, correct or incorrect, is wrongly absent from our lexicon. This is the case, for example, for *bathysphère*

Type of error	Superficial feedback		In-depth feedback		Morpho-syntactic analysis	
Usage	436/441	98,9%	434/441	98,4%	244/278	87,8%
Grammar	428/453	94,5%	411/453	90,7%	401/453	87,7%
Usage and grammar	62/69	89,5%	62/69	89,5%	51/63	80,9%
Punctuation	457/457	100%	457/457	100%	–	
Transcription	96/101	95%	96/101	95%	–	
Total	1479/1521	**97,2%**	1460/1521	**96%**	696/794	**87,7%**

Table 3: Evaluation of the feedback

and *Tyrrhénienne*. However, most often, the analysis fails due to a lack of contextual information. In this case, there is an error in terms of the grammatical features attributed to the correct form (*disent*: indicative ↔ subjunctive; *testez*: indicative ↔ imperative) at the time of preparing the dictation. Currently, we are considering remedying this problem with the help of local grammar, which will verify the absence or presence of certain constraining factors in the context surrounding an ambiguous form and which ought to improve the decision-making at the level of the language model. However, to avoid producing incorrect feedback that could mislead the learner, we intend to implement the "moins-disante" strategy proposed by Kraif and Ponton (2007), which consists in only presenting a piece of information once its validity has been checked.

6. Conclusion and future work

In this article, we have described and evaluated an algorithm for the automatic correction of dictation copies. It comprises three steps: detecting the errors in the copy, assigning tags to the errors, and producing feedback that is guided, on the one hand, by the tags assigned, and on the other hand, by the morpho-syntactic analysis of the dictation original.

The evaluation of the detection stage has revealed the fact that the module still does not effectively handle all missing and spurious passages, and ought to be supplemented by a prior search for identical and similar passages. The evaluation of the feedback stage has shown that it would be worthwhile augmenting the morpho-syntactic analysis currently being used, and also that the detail of the feedback offered probably ought to be varied, as part of a "moins-disante" strategy, depending on the confidence level of the information proposed by the morpho-syntactic analysis.

That said, the effectiveness of the system and the overall level of performance recorded mean we can indeed envisage real exploitation of this type of application. We are in contact with educators at several levels, from secondary to university, and we will shortly be testing the system within these various communities. One of the objectives will of course be to verify the robustness of the system. But another, much more important, objective will be to validate the process as a whole, from the ergonomics of the platform implementation to the pertinence of the feedback produced as a function of the public addressed.

References

Beaufort, Richard (2008), *Application des machines à états finis en synthèse de la parole. Sélection d'unités non uniformes et correction orthographique*, PhD thesis, Research Center in Information Systems Engineering (PReCISE), Faculté d'informatique, Facultés Universitaires Notre-Dame de la Paix (FUNDP), Namur, Belgique. 605 pages.

Beaufort, Richard (2010), Composition filtrée et marqueurs de règles de réécriture pour une distance d'édition flexible. Application à la correction des mots hors-vocabulaire, *Traitement Automatique des Langues (T.A.L.)* **51** (1), pp. 11–40, Association pour le Traitement Automatique du Langage (ATALA), Paris, France.

Beaufort, Richard and Alain Ruelle (2006), eLite: Système de synthèse de la parole à orientation linguistique, *Proc. JEP*, pp. 509–512.

Beaufort, Richard and Sophie Roekhaut (2011), Le TAL au service de l'ALAO/ELAO. L'exemple des exercices de dictée automatisés, *Actes de la 18^{e} conférence sur le Traitement Automatique des Langues Naturelles (TALN 2011)*, 27 juin–1$^{\text{er}}$ juillet, Montpellier, France. A paraître.

Beaufort, Richard, Thierry Dutoit, and Vincent Pagel (2002), Analyse syntaxique du français. Pondération par trigrammes lissés et classes d'ambiguïtés lexicales, *Proc. JEP*, pp. 133–136.

Catach, Nina (1995), *L'orthographe, Que Sais–je?*, Vol. 685, 6ème édition corrigée ed., P.U.F.

Council of Europe and Education (2001), *Common European Framework of Reference for Languages: Learning, Teaching, Assessment*, Press Syndicate of the University of Cambridge.

Damerau, Fred J. (1964), A technique for computer detection and correction of spelling errors, *Communications of the ACM* **7** (3), pp. 171–176, ACM New York, NY, USA.

Desmet, Piet (2006), L'enseignement/apprentissage des langues à l'ère du numérique: tendances récentes et défis, *Revue française de linguistique appliquée* **11** (1), pp. 119–138, Publications Linguistiques.

Desmet, Piet and Armand Héroguel (2005), Les enjeux de la création d'un environnement d'apprentissage électronique axé sur la compréhension orale à l'aide du système auteur IDIOMA-TIC, *Alsic. Apprentissage des Langues et Systèmes d'Information et de Communication* **8** (1), pp. 281–303, Revue.org.

Didier, Jean-Jacques and Cédrick Fairon (2006), Introduction, *"Le français m'a tuer": actes du Colloque L'orthographe française à l'épreuve du supérieur, Institut libre Marie Haps, Bruxelles, 27 mai 2005*, Les Cahiers du CENTAL, p. 5.

Didier, Jean-Jacques and Michel Seron (2006), Un manuel d'orthographe pour les autodidactes: l'esprit et la méthode, *"Le français m'a tuer": actes du Colloque L'orthographe française à l'épreuve du supérieur, Institut libre Marie Haps, Bruxelles, 27 mai 2005*, Les Cahiers du CENTAL, pp. 23–32.

Fairon, Cédrick and Anne Catherine Simon (2009), Informatisation d'un corpus de dictées: 40 années de pratique orthographique (1967-2008), *Pour l'amour des mots. Glanures lexicales, dictionnairiques, grammaticales et syntaxiques. Hommage à Michèle Lenoble-Pinson.*, pp. 131–154.

Gusfield, Dan (2007), *Algorithms on strings, trees, and sequences : computer science and computational biology*, Cambridge Univ. Press.

Heift, T. (2004), Corrective feedback and learner uptake in CALL, *ReCALL* **16** (2), pp. 416–431.

Irvine, Patricia, Parvin Atai, and John W. Oller Jr. (1974), Cloze, dictation, and the test of English as a foreign language, *Language Learning* **24** (2), pp. 245–252, Language Learning Research Club, University of Michigan.

Jaffré, Jean-Pierre (1992), *Didactique de l'orthographe*, Paris: Hachette.

Johnson, C. Douglas (1972), *Formal aspects of phonological description*, Mouton, The Hague.

Kraif, O. and C. Ponton (2007), Du bruit, du silence et des ambiguïtés: que faire du TAL pour l'apprentissage des langues, *Actes de la conférence sur le Traitement Automatique des Langues Naturelles (TALN 2007).*

Levenshtein, Vladimir (1966), Binary codes capable of correcting deletions, insertions and reversals, *Soviet Physics Doklady* **10**, pp. 707–710, MAIK Nauka/Interperiodika.

L'haire, S. (2004), Vers un feedback plus intelligent. Les enseignements du projet FreeText, *Journée d'étude de l'ATALA. TAL & Apprentissage des langues.*

L'haire, S. and A. Vandeventer Faltin (2003), Error diagnosis in the FreeText project, *Calico Journal. Special Issue Error Analysis and Error Correction in Computer-Assisted Language Learning* pp. 481–495.

Manesse, Danièle (2007), *Orthographe: à qui la faute ?*, ESF editor.

Mohri, Mehryar and Michael Riley (1997), Weighted determinization and minimization for large vocabulary speech recognition, *Proc. Eurospeech'97*, pp. 131–134.

Mohri, Mehryar, Fernando Pereira, and Michael Riley (2000), The design principles of a weighted finite-state transducer library, *Theoretical Computer Science* **231** (1), pp. 17–32.

Mohri, Mehryar, Fernando Pereira, and Michael Riley (2001), Generic ϵ-removal algorithm for weighted automata, *Lecture Notes in Computer Science* **2088**, pp. 230–242.

Rahimi, Mohammad (2008), Using Dictation to Improve Language Proficiency, *Asian EFL Journal* **10** (1), pp. 33–47, Asian EFL Journal.

Roche, Emmanuel and Yves Schabes, editors (1997), *Finite-state language processing*, MIT Press, Cambridge.

Ruggia, Simona (2000), La dictée interactive, *Alsic. Apprentissage des Langues et Systèmes d'Information et de Communication* **3** (1), pp. 99–108, Revue.org.

Santiago-Oriola, Conception (1998), *Système vocal interactif pour l'apprentissage des langues - la synthèse de la parole au service de la dictée*, PhD thesis, Toulouse III.

Tschichold, Cornelia (2006), Intelligent CALL: The magnitude of the task, *Verbum ex machina: actes de la 13^{e} conférence sur le traitement automatique des langues naturelles (TALN 2006).*