論文 / 著書情報
Article / Book Information

論題(和文)	
Title(English)	Improving Automatic Phonetic Transcriptions of French Language
著者(和文)	ﾎﾟﾝﾃｽ ｼﾞｮｻﾞﾌｧ, 古井 貞熙
Authors(English)	Josafa Pontes, Sadaoki Furui
出典(和文)	日本音響学会 2007年秋季講演論文集, Vol. , No. 2-4-5, pp. 333-334
Citation(English)	, Vol. , No. 2-4-5, pp. 333-334
発行日 / Pub. date	2007, 9

Improving Automatic Phonetic Transcriptions of French Language *

◎ Josafá J. Aguiar Pontes, Sadaoki Furui
(Tokyo Institute of Technology)

1 Introduction

The phenomenon technically known as *liaison*, in which a final silent consonant of a graphic word may be pronounced under certain specific circumstances is still one of the biggest issues in automatic phonetic transcriptions of French language. Liaisons are classified into three types or major sets: Required (M), Forbidden (F) and Optional (O) liaisons. At the present work, we focus our study on the M and F cases which result in making and not making liaison respectively. Automatic phonetic transcriber systems for French language developed so far use manually generated rule-based approach for modeling liaison. Unlike these systems, our approach models liaison in a semi-automatic way using decision trees [1], bringing a number of advantages.

2 Our Method

Our approach consists of an algorithm having 5 major steps, where the first three refer to specification of a database required for training decision trees. The last two steps, which are automatic, targets the training of the trees and the translation of tree models into appropriate language representation respectively.

2.1 Classification of Liaison Rules

First, we arrange the liaison rules (R) into classes R_1, R_2, ..., R_m, with $\{i \mid 1 \leq i \leq m\}$, where each R_i has the same characteristics or attributes that determine R_i as a general or specific rule, taking the order F, M and O classification into account. The order of precedence F>M>O matters for deciding which rule to apply first. For example, a F liaison formed by verb-pronoun inversion must be defined before any other rule that treats verbs or pronouns; likewise, a F liason such as aspirated /h/ is likely to be treated first inside the context that it could appear as liaison (M or O). Although several subclassifications for each group (F, M, O) of liaison rules are presented in the literature [2], here we propose to split R taking in consideration characteristics that are applicable to all *liaison words* inside a class R_i. For example, the same contexts that determine how to pronounce the number six "six" also applies for the number ten "dix", in a way that we can gather them into a single class R_i. The position i specifies the order in which R_i is instantiated. This order is important when another rule R_t, with $\{t \mid 1 \leq t \leq m, i \neq t\}$ also treats the same liaison word in a different context, for example, when it belongs to an expression or exception. In this case, the exception needs to be treated first, that is $t < i$. In the case of words with complex prediction, such as "plus", we split it into several classes R_i's, ordered from the most restrictive contexts to the least restrictive ones.

2.2 Pre-classification of the Attributes

Second step is to name each attribute of a given class R_i and pre-classify it into a limited number of outcomes. Let Y_i be the set of attributes Y_{i1}, Y_{i2}, ..., Y_{in} that determines R_i, with $\{j \mid 1 \leq j \leq n\}$. Then, we pre-classify each Y_{ij} into possible categories for that attribute, assigning a general label L_{ijk} to each possible category, with $\{k \mid 1 \leq k \leq p\}$ where p is the cardinality of Y_{ij}. It is recommended to use labels that represent the respective meaning of that value, for example, one attribute labeled *next_word_vch* that pre-classifies the next word in terms of vowel, consonant and "h" type would have as possible values the strings "vowel", "consonant", "aspirated_h", "mute_h" and "none". Sometimes the liaison word itself can be used as a category, once they are limited and determinative of liaison, such as "les", "des", "nous", "vous", etc. Every Y_{ij} must have one label similar to "none" that identifies a word/token which does not apply as a category for that attribute. This will force node splits in the tree, granting that the maximum number of Y_i's are used in order to precisely distinguish contexts. The labels contain useful information required to identify the context in which a F, M or O liaison occurs. Here we assume that every token of a text has already been treated by both lexical phonetic transcriber and POS tagger subsystems, making possible this pre-classification process. The more accurate and complete this process is designed and populated, the more precise liaison models we have.

2.3 Database Specification

The third step is to define a database for each R_i containing the attributes Y_{i1}, Y_{i2}, ..., Y_{in} pre-classified with corresponding target output attribute λ_i's. Let P_i be a vector containing the cardinality of each Y_{ij} defined as $P_i = [p_{i1}, p_{i2}, ..., p_{in}]$. Then we define our database DB_i for each R_i as the *Cartesian Product* of the non-categorical attributes Y_i's concatenated with the categorical output attribute λ_i's, such as DB_i=concatenation($Y_{i1} \times Y_{i2} \times ... \times Y_{in}$,$\lambda_i$'s) containing $p_{i1} \times p_{i2} \times ... \times p_{in}$ tuples. λ_i is the liaison information, defined here as a mnemonic string containing useful information of how to treat that context, such as the type of liaison (F, M, O); the *consonant of liaison*; a liaison justification code; a double acceptable pronunciation code; or simply, a label such as "skip" meaning that this R_i is not responsible to treat a certain combination of Y_i's, letting this responsibility to another rule R_t with $t > i$; or a label such as "incorrect" meaning that this combination of Y_i's results in incorrect grammar.

2.4 Training the Tree Models

In the fourth step, we use the above-defined database DB_i to train a decision tree DT_i. There is no need to include in the training set the tuples which λ_i has "incorrect" value because they are not informative. However, λ_i containing "skip" value is informative because it forces the node split, limiting the cases of liaison that this DT_i is able to solve as originally planned. Two essential settings, independently of the decision tree algorithm used, are 1) the minimum number of tuples in a node that the algorithm uses for deciding the split of a node to target different λ_i should be one tuple and 2) the trees should not be pruned. This tree over-fitting

* フランス語の自動音素表記の改良，ジョザファ・アギアル・ポンテス，古井貞熙 (東工大)

guarantees that every leaf achieves 0% of misclassification errors, building trees that map exactly as designed in the database.

2.5 Translating Trees into Source Code

The final step extracts a set of rules from each DT_i into a convenient programming language representation. Here we use scripts able to perform the translation directly from the trees generated by a decision tree tool to the control flow statements of the target programming language. The basic algorithm for this translation phase comprehends a lexical analyzer and a code generator. The lexical analyzer reads a tree file, retrieves and identifies tokens, recognizes among which branch patterns they belong to, tag them with meta-labels that identify the patterns, and puts them into appropriate data structure. Then, the code generator reads this data structure, verify similarities among the values of the attributes in order to optimize expressions, pushes and pops attribute labels and values in a stack, discard expressions that result in "skip" value for λ_i, decides which sort of code to generate based on the meta-labels and finally writes the optimized source code for the target language. This translation process is then repeated for all the m DT_i's. Then we just insert this automatically generated code in the application making necessary adjustments.

3 Experiment Setup

The experiments are done using French text extracted from NHK [3] news and their corresponding audio files recorded from 4 native speakers. 16,289 words distributed in 618 sentences compose our evaluation text corpus extracted from 103 short articles. At first, we listened to the audio files and manually labeled the liaisons occurrences in that text according to native speakers taken as reference. Our labeling consisted of identifying, classifying and annotating F, M and O liason cases on the text corpus. Next, we submitted our text corpus to Loquendo [4] TTS to generate synthesized speech from which we could generate our baseline transcriptions and annotate them repeating the same procedure as described above. Loquendo uses manually generated rule-based approach for modeling liaisons. Finally, we generated transcriptions of our text corpus using our prototype transcriber system.

4 Results and Discussion

First, we confirmed that native speech was totally accurate, serving as a good reference to our experiments. Next, we compiled our results grouping them in the three categories of liaison as follows.

4.1 Required Liaisons

In a total of 691 cases of required liaison, the baseline transcriber could correctly predict 89%, while our transcriber was able to detect 95%, with an improvement of 6%.

We also observed that in the case of our prototype transcriber 79% of the errors were due to expressions not included in our database. It means that if we had an extensive list of expressions where required liaisons happen, we could have even better results. The other 21% of errors were due to either inaccurate classification of attributes during the modeling phase or exceptions not included in our models.

In our text corpus, we also observed that 67% of the required liaisons were due to the articulation of 10 most frequent liaison words ("aux","ces","des","en","les","leurs","sans","ses", "son","un") with their context. Removing these most frequent cases, we verified that our transcriber is in average 17% more efficient than the baseline in detecting non-trivial cases of required liaisons.

4.2 Prohibited Liaisons

It is meaningless to count the total number of prohibited liaisons inside a text because all boundaries between words where either required or optional liaisons cannot happen are candidates for that account. Thus we report here just the absolute number of errors detected in prohibited liaisons, which were 15 in the baseline and 17 in our transcriber.

Both baseline transcriber and our prototype generated approximately the same number of prohibited liaisons errors. 47% of our errors were due to inaccurate classification of attributes during the modeling phase and 53% refer to exceptions not included in our models.

4.3 Optional Liaisons

As for the optional liaisons, we are interested to know how many cases our transcriptions diverge from the native speaker's option. In this case, best transcriptions are those which the number of divergence is lower, modeling optional liaisons to sound more naturally.

In the whole text corpus, there were 308 cases of optional liaisons. While the baseline transcriber produced 123 diverging optional liaisons, our transcriber produced 102, showing that our optional liaison models were able to generate 17% better predictions than the baseline approach. Although some optional liaisons are more frequent than others, predicting them to match native speakers' preference is not trivial, since subjective factors influence in their choice.

5 Conclusions and Future Work

In this paper we have proposed a new approach for modeling post lexical grapheme to phoneme rules for French language liaison phenomena based in decision trees. We investigated our method comparing its output transcriptions with the ones supplied by the previous approach. In general, we observed that our transcriber was able to provide 6% more accurate required liaison prediction than the previous method. We also verified that better results depend on how extensive and complete we specify the attributes and their classification in the second phase of our method. The advantages over the previous method are: 1) standardization and simplification on the process of rule generation, 2) once data is correctly prepared, we have complete automation of the process, 3) ease maintenance and correction of the rules not appropriately modeled. Our approach simplifies the liaison modeling in a way that no other knowledge than phonological realization is required, discarding the complexity involved in manual rule codification in a programming language. Future work in this area include providing maintenance in our database; studying in more detail about the optional liaisons and generating multiple transcription models.

References

[1] Quinlan, J.Ross. "C4.5: Programs for Machine Learning". Morgan Kaufmann Publishers Inc. San Mateo, CA, USA 1993.

[2] Côté, Marie-Hélène. "Phonologie Française". pp. 91-105, Version avril 2005

[3] http://www.nhk.or.jp/daily/french

[4] http://www.loquendo.com