Modeling Liaison in French by Using Decision Trees

Josafá de Jesus Aguiar Pontes, Sadaoki Furui

Department of Computer Science, Tokyo Institute of Technology, Japan

`{josafa, furui}@furui.cs.titech.ac.jp`

Abstract

French is known to be a language with major pronunciation irregularities at word endings with consonants. Particularly, the well-known phonetic phenomenon called *Liaison* is one of the major issues for French phonetizers. Rule-based methods have been used to solve these issues. Yet, the current models still produce a great number of pronunciation errors to be used in 2nd language learning applications. In addition, the number of rules tends to be large and their interaction complex, making maintenance a problem. In order to try to alleviate such problems, we propose here an approach that, starting from a database (compiled from cases documented in the literature), allows us to build C4.5 decision trees and subsequently, automate the generation of the required rules. A prototype based on our approach has been tested against six other state-of-the-art phonetizers. The comparison shows the prototype system is better than most of them, being equivalent to the second-rank system.

Index Terms: liaison in French, post-lexical rules, speech synthesis, grapheme-to-phoneme conversions

1. Introduction

Liaison is the pronunciation of an otherwise silent final consonant of a word (w_1) in certain contexts. It is realized when the following word (w_2) starts with a vowel, a (graphic) mute "h" or some glides [1, 2]. For example, the adjective (w_1) "mes" [me(z)] ("my") is pronounced with a [z] sound when the next word (w_2) is a noun starting with a vowel, such as in "mes amis" [mezami]. However, this sound should not be pronounced when w_2 starts with a consonant, such as in the sequence "mes doigts" [medwa].

Liaisons are classified as Obligatory (Ob), Optional (Op) or Forbidden (F) [3]. *Obligatory liaisons* stand for contexts in which a consonant of liaison is to be pronounced, such as in the above sequence "mes amis".

Optional liaisons are cases where the liaison can be realized depending on the circumstances [4]. For instance, consider the noun w_1 "amis" (friend) followed by the adjective w_2 "étrangers" ("foreign"). The word sequence "amis étrangers" can be pronounced either with or without the [z] liaison consonant. When this sequence is found in isolation, liaison is likely to be realized in order to indicate the plurality of the terms. But this is not the case when another element of the context already signals the plurality, such when "les" ("the") precedes this sequence: "les amis étrangers" [lezamietʀɑ̃ʒe].

Finally, *Forbidden liaisons* are those whose realizations are avoided, even if the graphic word ends with a consonant. For example, if the noun w_1 "amis" is followed by the verb w_2 "étudient" [etydi] ("study"). In this case, the sequence "amis étudient" must be pronounced as [amietydi], as liaisons are not triggered between nouns followed by verbs.

A study conducted by Yvon et al. [5] evaluated the quality of automatic grapheme-to-phoneme (G2P) converters (phonetizers) for French produced by 8 different systems. The study revealed that the phonetizers are still problematic in French, since even the best systems are prone to make at least one error in every 10 sentences. The authors of this study consider the prediction of liaison as a major cause of the problems.

Unfortunately, very little literature is available on how these kinds of systems predict liaisons. It seems that the Bell Laboratories TTS system [6] uses a rule-based method. Within text analysis processing, traces (labels) are appended at the end of each word to indicate a possible liaison candidate. Then, these labels are tested, verifying whether there is a liaison rule applying to a context or not. Whenever there is, the phonetic sequence is changed accordingly. However, specific details about their method (such as codification technique, number of liaison rules, etc) are not specified.

Due to the intricacy of the problem, the number of rules to model this phenomenon tends to be large, their interaction becomes complex, and the code becomes difficult to maintain. That may be one major reason why the current models still produce a great number of errors, i.e. mispredictions, and this holds even if the scope is only Obligatory and Forbidden liaisons. Hence our motivation is to improve the quality of the French G2P conversions.

The focus of this study is Obligatory and Forbidden liaison contexts as indicated in the French phonetic literature. We investigate how efficiently liaison can be modeled via C4.5 decision trees learning algorithm [7]. For this purpose, we compile a database by using lexical, phonetic, morpho-syntactic and other linguistic properties which are characteristic of each context. Then we use this data for training the related models. Decision trees are used here because they allow to convert structured data sets into respective sets of hard-coded rules necessary for the implementation of a prototype system.

The remainder of this paper is organized as follows: Section 2 explains our method for data collection; Section 3 describes the training and prediction algorithms; Section 4 presents the results and discussions related to the evaluation of the proposed method, while conclusions and future perspectives are given in Section 5.

2. The Creation of the Database

The creation of the database is a manual process, which consists of three major steps. The first is the data selection and clustering, the second is the specification of features (attributes) and values, and the third is the creation of data sets. Each of them is explained in the next subsections.

2.1. Step 1: Data Selection and Clustering

Rules of pronunciation (R) for words where the liaison phenomenon occur can be obtained from the French phonetic literature [1, 2, 4]. We collected about 1500 contexts of liaison in order to build the set of training data. Here we show how to organize this knowledge in a way that it can be computationally handled, aiming at reducing the complexity involved in coding the rules, and at the same time, allow easy maintenance of the model.

Most of the liaisons can be identified by contextual features. Contextual features are described in terms of attributes and values extractable from the contexts where the variations occur. They refer to information like "current lexical word" w_1, "part-of-speech (PoS) of the next word", "initial phoneme of next word", and so on. Based on phonological/linguistic knowledge, we manually cluster the rules of pronunciation R into sets of rules R_i with $\{i \mid 1 \leq i \leq m\}$. Each R_i is compiled from a number of similar liaison cases, i.e. cases sharing common contextual features. For instance, the words "six" and "dix" ("six" and "ten") are grouped together into a single set R_i, given the fact that the contextual attributes and values determining the pronunciation of one also hold for the other.

The index i not only lists the set of rules, but also allows to disambiguate between different sets sharing the same word. In this case, the index is used for ranking. Particular/exceptional cases in which a liaison word is used are likely to have precedence over general cases with the same word being used. In such a situation, the particular cases should be handled first, by one set of rules, let us say R_i, and the general cases by another one, R_t, with $\{t \mid i < t \leq m\}$. This is because the amount of data required to distinguish exceptions from general cases is very high if all their features were used to make a single decision tree.

To predict pronunciation of the final consonants in French, care needs to be taken with respect to the order in which the rules are tested/applied because of ambiguities [6]. This is done in the Bell Laboratories French TTS system by applying overruling strategies. However, this might make the final code difficult for humans to understand, find bugs, and correct them. Thus, instead of overruling, we establish *priorities* with respect to the type of liaison context. In order to set up priorities, we recommend to cluster the sets of rules in the following order: Forbidden > Obligatory > Optional. The preliminary evaluation of known Forbidden liaisons reduces the search space for the cases left, given that less tests are needed to handle them.

Another weak point that deserves special attention when predicting liaisons is the fixed expressions or locutions. Numerous expressions are composed of a fixed sequence of tokens, such as "tout à coup" of length 3 and "Jeux Olympiques" of length 2. Consider the length of a fixed expression as the number of tokens of which it is composed. Since the length of such expressions as well as the relative position of token (causing a variable pronunciation of its final consonant) may vary, the number of attributes required to identify them varies accordingly. Hence, the identification of fixed expressions of different lengths require a huge number of attributes, values and tests, if they were handled all together in a single set of rules.

In order to reduce this complexity, we suggest splitting fixed expressions into several sets of rules, such as R_2, R_3, R_4, etc. Every set is compiled from expressions having the same length. In addition, the greater the length of a fixed expression, the smaller the index i in R_i is needed, in order to resolve possible ambiguities likely to occur between overlapping fixed expressions of different lengths. The creation of rule-sets dedicated to handle fixed expressions of equal length yields a simplification of other rule-sets dealing with the same words/tokens, but in other contexts. This is because a word/token with variable pronunciation of its final consonant is supposed to be pronounced differently depending on whether it is (part of) a fixed expression or not. Hence, processing together fixed expressions of equal length has the advantage of reducing the complexity of rule sets.

In our case, we compiled about fifty sets of rules to account for liaisons. This number may vary for other data or implementations. Next, given a set of rules, Subsection 2.2 shows how to specify the features necessary for identification of liaison contexts.

2.2. Step 2: Specification and Classification of Attributes

Taking phonological/linguistic knowledge into account, Step 2 stands for the manual retrieval and labeling of attributes and values needed for the identification of liaison contexts. Examples of possible attributes are "PoS (of the current word)", "final letter (of the current word)", "next word starting phoneme", etc.

Let Y_{ij} be an attribute among the n_i attributes (Y_{i1}, Y_{i2}, ..., Y_{in_i}) of a set of rules R_i, with $\{j \mid 1 \leq j \leq n_i\}$. The possible values for the attribute Y_{ij} are classified into p_{ij} categorical values by assigning a label L_k to each category, with $\{k \mid 1 \leq k \leq p_{ij}\}$. For example, by applying this labeling procedure to the attribute "next word starting phoneme" (*nw_start_phon*), one may obtain the following categorical values: "vowel", "consonant", "aspirated_h", "mute_h" and "other".

Most of the values for these labels can be grouped into very few categories. A category is normally defined by a group of elements with the same behavior regarding liaison, such as words, phonemes, letters, PoS, punctuation marks, etc. We call them here as *categorical labels*. They will allow to predict liaisons based on the categories that the elements fall in.

One important point at this step is that every attribute Y_{ij} must have one extra categorical label such as "other", indicating that a given element does not apply to any category of that attribute. This additional label will force the node to split in a decision tree, allowing scrutinized use of the available attributes in order to distinguish each target value on the basis of the context. The more precisely and completely the categories are specified, the better the trees will be populated and trained.

2.3. Step 3: Database Specification

Step 3 consists in the creation of the training database, once the above procedures have been completed. Each set of rules R_i corresponds to a table T_i, composed of Y_{i1}, Y_{i2}, ..., Y_{in_i} attributes and one target output attribute called λ_i. A table T_i is filled by computing, over all attributes, the Cartesian Product of all values of an attribute against all values of the next attribute. Once this is done, we append manually the target attribute λ_i to the table. These operations are summarized by the following Equation:

$$T_i = append(Y_{i1} \times Y_{i2} \times ... \times Y_{in_i}, \lambda_i's), \quad (1)$$

where λ_i is the information required for handling the pronunciation of a word-final consonant. It can assume four types of values. The first type specifies a set of features required for handling a liaison. In practical terms, it is a string composed of five fields with the following data:

1. an index i, representing the set of rules R_i in charge of handling the case;

2. a pointer to a function that performs one specific kind of change to the phonetic sequence (insertion, deletion or replacement of phonemes, considering possible coarticulation effects);

3. a liaison phoneme;

4. an extra phoneme to be used in case of vowel co-articulation;

5. an additional phoneme to be used in case of available alternative pronunciation;

With this information properly specified, it is possible to perform the phonetic changes caused by liaison in the word boundaries.

The second type of information addresses the cases where no phonological change should be applied to a particular context, implying that the ordinary lexical pronunciation of the current word should remain unchanged. The label "ordinary" is used for this purpose. The third type is meant to convey that the current set of rules R_i does not contribute at all to handle this context, letting another set R_t, with $t > i$, take care of it. A label such as "skip" is used for this purpose. This label is required because the solution for a given context may not be found in a tree. In that case, the execution pointer needs to "escape" from a leaf node and continue the search in the adjacent trees, until the solution is found. Finally, the fourth type refers to a combination of values for attributes yielding incorrect grammar, phonetic combination or illogical sequence. Since these combinations do not comply with linguistic constraints, the label "invalid" is used to refer to such cases.

This process of table creation is repeated for all m sets of rules. As a result, m tables are generated, with each one corresponding to a set of rule R_i. The Equation 2 captures the total number of tuples ($\#Tuples$) contained in this database.

$$\#Tuples = \sum_{i=1}^{m} \prod_{j=1}^{ni} pij \qquad (2)$$

In practice, the implementation of these procedures led to the creation of a database having approximately 50 tables (for liaisons) containing a total of around 30 000 tuples. However, since tuples which are linguistically invalid (λ_i="invalid") are not supposed to match any text input sequence, we removed them, leaving only about 7 000 entries.

3. Training and Prediction Algorithms

This section is divided into two parts. First, we explain how to use the data to create decision trees. Second, we discuss how to connect these trees in order to process a given input text. Each of these algorithms is described in the next subsections.

3.1. Training Decision Tree Models

We use the data of each table T_i to train a corresponding DT_i decision tree, which is in charge of handling similar contexts. Due to the particularities of each group of rules, every tree requires a specific set of features for determining the variable pronunciations. This technique allows individual trees to focus just on a limited set of features, i.e. only those necessary for handling a limited number of contexts.

As explained in Section 2.3, whenever a set of rules does not contribute to handle a context, the label "skip" is used. The

tuples containing this label for λ_i are informative for the training of the trees because they force node split, differentiating contexts from each other and delimiting the scope that a DT_i is able to handle. Without this label, it would not be possible to know whether the search for the solution should stop or not, after reaching a leaf node.

In addition, the following two settings of the decision tree classifier are important: 1) a tree should be able to hold at least *one instance* (object) per node and 2) the trees should *not be pruned*. These settings allows for the creation of all possible valid paths (from the root to the leaf nodes), given that the training data has been previously generated by the computation of the Cartesian Product. Because of this, the values of the attributes are carefully examined by the trees during the test. The purpose of overfitting the trees to the data is to guarantee that every leaf achieves perfect performance as designed in the training database.

We avoid pruning the trees for two reasons. The first is because a single leaf node can contain the prediction of up to *five* interdependent parameters (cf. 2.3). This set is required by the G2P converter for performing the appropriate changes to the phonetic sequence. Since *all* these parameters are output together for *a single prediction*, the *whole* set would become ineffective as soon as *any* of them contained incorrect information concerning the handling of the current context. For example, even if the final consonant phoneme were correctly predicted, it would be useless if it were accompanied by incorrect information of *how to change* the phonetic sequence.

Second, notice that we utilized the Cartesian Product operation for the generation of all possible combinations of categorical values for the attributes of a table. Assuming that the data has been correctly specified, the training process should *cover all valid possibilities*, and consequently the predictions based on this data should be correct.

Although the trees are not pruned, this method does allow to generalize predictions of liaisons with respect to unseen words. This is possible because decisions concerning the goodness of fit of a word with regard to a set of rules, are made not only on the basis of lexical entries, but also on the basis of *clusters of words*, such as PoS labels. So, even if a word is not explicitly included in the training database, decisions regarding the realization of liaison or not are made on the basis of these categories.

We use the code generated for the decision trees to create m Modules. A Module is a piece of code in charge of preprocessing and testing the attributes/values of the input text according to its corresponding decision tree. In the next subsection, we explain how this happens.

3.2. The Prediction Algorithm

The prediction algorithm stands for the integration of the automatically generated trees into the G2P converter, as well as for the description of the various steps the algorithm goes through (stepper) during run-time.

This integration occurs in the post lexical analysis routine of the G2P converter. Initially, it is assumed that every word/token ending with one of the possible liaison consonants allows in principle for liaison, and should therefore be analyzed. To this end, given the relative position of analysis of a word/token, we extract and store in memory several general parameters concerning its context. For example, the current word, some of the neighbouring words, their parts-of-speech, the lexical phoneme sequence of the current and following word, etc.

Then, Module 1 is called, performing the following operations: 1) classification of the input parameters concerning the current position of analysis, assigning the same set of labels defined for rule-set R_1 during Step 2 and 2) use of the tree DT_1 to predict liaison for this context. If the current word/token and context are identified by this tree, then the information specified in λ_1 is used for resolving this case (change the phoneme sequence). However, if the label "ordinary" is found, no phonological change is necessary. Otherwise, Module 1 does not contribute anything to process this context. The label "skip" returned by DT_1 contains this information, and the task is transferred to Module 2.

The next Modules are built in a similar fashion. Each one performs the classification required for its attributes, given the current position of analysis of the input text, followed by the tests checking the familiarity of this context. Thus, every Module has its own mechanisms to identify whether a certain context is within its purview or not. If none of the Modules is able to recognize a context, then the phoneme sequence remains unchanged.

Once the analysis of the current word is finished, the algorithm proceeds to the next word/token ending with a consonant that may cause liaison, repeating the same process over and over, until the end of the input.

4. Comparison with State-of-the-Art G2P Converters

The experiments were done using a French text corpus extracted from the newspaper *Le Monde* of January 1987. This corpus, used only for evaluation purposes, has approximately 26 000 words/tokens distributed in about 2 000 sentences [5]. Roughly, it contains 1 500 liaison candidates, of which about 600 are obligatory. Manually prepared phonetic transcriptions are also available for this data, including pronunciation variations.

Yvon et al. used this corpus to conduct an objective evaluation of G2P conversion for French TTS synthesis in 6 systems (B, C, D, F, G and H). Their investigation included global aspects of the G2P conversions. One of the aspects investigated was the nature and diversity of sources of errors. They concluded that liaison was one of the major sources of errors among the synthesizers. In our case, we use these results to compare the amount of errors caused by *misprediction of liaisons* between these systems and our prototype.

Our experiments consisted in providing the corpus to our prototype system as input, in order to generate the G2P conversions. Next, the automatic conversions were aligned with the text corpus, as well as with the transcribed corpus. For words/tokens with a unique pronunciation, finding a mismatch was as simple as comparing two strings, given that the phonetic sequences were aligned and the phonetic alphabet identical for both transcriptions. For words/tokens with multiple pronunciations, all possible pronunciations were generated, as indicated in the transcribed corpus. Matches were identified whenever one of the optional pronunciations for a word/token corresponded to the automatically produced phonetic sequence. However, if none of the optional pronunciations corresponded to the one produced automatically, a mismatch was marked. Next, a list of all mismatching pronunciations was generated.

From this list, we selected the words/tokens ending with one of the graphic consonants considered as possible liaison candidates. This list was filtered then again by separating entries with a liaison error from entries having other types of error.

Consequently, a list of words/tokens containing liaison related errors was compiled. Table 1 shows the result of our method compared with the results previously obtained.

Table 1: Absolute number of liaison errors produced by previous 6 systems compared with our prototype system.

System	B	C	D	F	G	Prototype	H
#L. Errors	111	123	76	49	38	34	15

Unfortunately, we lack information concerning the methods used by these 6 systems to predict liaison. Hence, we cannot gain insight into the superiority of one method over another, although the results are still comparable. In addition, since the types of liaison errors produced by these 6 systems are unknown, it is not possible to compare them and determine which types of liaison are most difficult to predict. Nevertheless, the available figures allow us to get an idea concerning the performance of the proposed method from the point of view of the state-of-the-art G2P converters.

5. Conclusions and Future Work

This paper proposed a novel approach for modeling liaison in French by using C4.5 decision trees. We compared the predictions concerning Obligatory and Forbidden liaisons generated by our prototype system with those generated by six other state-of-the-art systems. The comparison shows that our method outperforms most of the current converters, being equivalent to the second-rank system.

Relying exclusively on linguistic/phonologic knowledge to deal with the problem at hand, our approach dispenses with the complexities of manual coding of rules in a programming language. We are able to automate the production of rules given a training database, although the latter requires human intervention. The proposed method also allows for standardization and simplification of the rule generation process due to the fact that similar rules are grouped together. In addition, the distribution of the rule-sets in conjunction with appropriate priority settings contribute to alleviate the impact of maintaining the model.

Future work in this field includes: 1) enlargement of the database used by the prototype system; 2) investigation of Optional liaison contexts, using annotated corpora and 3) research on other aspects of G2P conversions, such as proper nouns.

6. References

[1] Fouché, P., "Traité de prononciation française", 435–479, Klincksieck, 1959.

[2] Encrevé, P., "La liaison avec et sans enchaînement: phonologie tridimensionnelle et usages du français", Éditions du Seuil, 1988.

[3] Delattre, P., "Principes de phonétique française l'usage des étudiants anglo-américains", Middlebury College, 1951.

[4] Pierret, J-M., "Phonétique historique du français et notions de phonétique générale", Peeters Louvain-La-Neuve, 98-101, 1994.

[5] Yvon, F. et al., "Objective evaluation of grapheme to phoneme conversion of text-to-speech synthesis in French", Comput. Speech Lang., vol.12, 393–410, 1998.

[6] Tzoukermann, E., "Text analysis for the Bell Labs French text-to-speech system", Proc. ICSLP98, Sydney, Australia, vol.5, 2039–2042, 1998.

[7] Quinlan, J.R., "C4.5: programs for machine learning", Morgan Kaufmann Publishers Inc. San Mateo, CA, USA, 1993.