Statistical Modeling of Pronunciation Variation by Hierarchical Grouping Rule Inference

Mónica Caballero, Asunción Moreno

Talp Research Center
Department of Signal Theory and Communications
Universitat Politecnica de Catalunya, Spain
{monica,asuncion}@gps.tsc.upc.edu

Abstract

In this paper, a data-driven approach to statistical modeling pronunciation variation is proposed. It consists of learning stochastic pronunciation rules. The proposed method jointly models different rules that define the same transformation. Hierarchic Grouping Rule Inference (HIEGRI) algorithm is proposed to generate this model based on graphs. HIEGRI algorithm detects the common patterns of an initial set of rules and infers more general rules for each given transformation. A rule selection strategy is used to find as general as possible rules without losing modeling accuracy. Learned rules are applied to generate pronunciation variants in a context-dependent acoustic model based recognizer. Pronunciation variation modeling method is evaluated on a Spanish recognizer framework.

1. Introduction

Modeling pronunciation variation is an important task when improving the recognition accuracy of an ASR system [1]. A common approach is to use phonological rules that allow to model pronunciation variation independently from the vocabulary. Rules define a particular change in the pronunciation of a focus phoneme(s) depending on a variable length context. Rules can be found in the phonology literature [2], or they can be learned automatically from data [3] [4], providing application probabilities to the extracted rules.

Most of data-driven methods proposed in the literature derive rules by observing the deviations when aligning canonical transcription with *correct* or surface form, obtained automatically by means of phoneme recognizer [5] or by forced alignment [3] [4]. After this procedure, a large set of rules is obtained and a selection criteria and/or pruning step becomes necessary. Moreover, the extracted rules are dependent on the training vocabulary.

In [6] a method to obtain a set of general rules is proposed. A hierarchy of more and more general rules belonging to the same transformation is induced. Afterwards, the created hierarchical network is pruned using an entropy measure. This method is very efficient to obtain a reduced set of rules as general as possible but it does not consider information given by rules belonging to the same transformation at the same level (same context length): Are the rules similar or do they have totally different context phones? How many rules share the same internal pattern? Answering these questions surely would help to find the best candidates to be general rules in a reduced rule set.

In this paper, a data-driven method for statistical modeling of the pronunciation variation is proposed. The method learns pronunciation rules automatically. A new strategy to infer a set of general rules based on Hierarchical Grouping Rule Inference (HIEGRI) algorithm is proposed. As a result we obtain a compact set of rules, flexible enough to derive alternative pronunciations for a variety of domains and vocabularies.

Learned rules are applied to derive word pronunciation models for each vocabulary word. The word pronunciation model contains all possible pronunciation variants for a word. Such an approach was also used in [3] in a context-independent recognizer framework. In this work, we expand pronunciation models to be applied to a context-dependent acoustic model based recognizer.

The rest of the paper is organized as follows. Section 2 describes the learning rule process and the HIEGRI algorithm proposed. Section 3 explains variant generation and creation of word pronunciation models. In Section 4 the details concerning the database used in this study are included. In section 5 experiments carried out in this study are shown. Finally, section 6 contains the conclusions of this work.

2. Rule learning methodology

Stochastic pronunciation rules (referenced in [1] as rewrite rules) define a transformation of a focus phoneme(s) F into F' depending on the context with a given probability. Rules can be expressed by the formalism [3] [4]:

$$LFR \rightarrow F' \; with \, a \, probability \; p_{LFR} \tag{1}$$

L and R are the left and the right contexts. Combination LFR is the condition of the rule. The tuple F, F' is the transformation the rule models, where F and F' are the focus and the output of that transformation, respectively.

The aim of the proposed rule learning method is to achieve a model for each possible transformation. The model is defined as a *Rule graph*: a tree shaped graph containing rules associated to a particular transformation. A Rule graph general example is shown in Figure 1. This Rule graph models transformation $F \rightarrow F'$. In each level of the graph, different rules with the same length condition can be found. Maximum length condition rules (most specific rules) are in the highest level. Focus of the transformation (most general rule) is set on the lowest level. Intermediate levels contain common patterns conditions for rules in upper levels. Each node of the graph is assigned the estimated probability for the rule it contains.

Given a phone string as input, the most specific matching rule in the graph is selected. The application probability of the selected rule is the output of the model of the transformation.

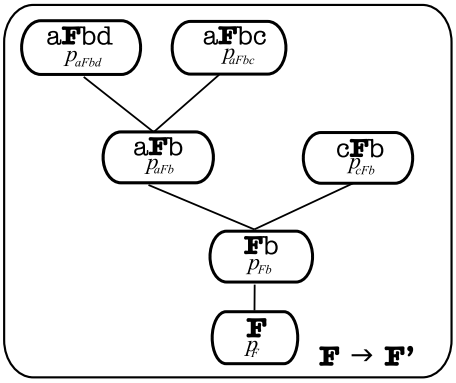

Figure 1: Rule graph model for transformation $F \rightarrow F'$

Rule learning method consists of three main steps. In the first step an initial set of rules is learned from a orthographically transcribed corpus. Second step consists on the application of HIEGRI algorithm. HIEGRI algorithm infers general rules with different length conditions and generate a preliminary graph (*HIEGRI graph*) for each transformation. General rules inferred are the commom patterns shared by rules associated to a transformation. Third step is a rule selection strategy that leads to the final Rule graph. Next sections describe each step of the process.

2.1. Obtaining an initial set of rules

Rules are extracted comparing a canonical transcription (T_{can}) with an automatic transcription that represents an hypotheses of what has been really said.

Canonical transcription is achieved concatenating word baseline transcriptions. T_{aut} is obtained by means of forced recognition. Word pronunciation model [6] is used instead of using a variety of alternative pronunciations for each word.

For each word appearing in the training data, a finite state automaton (FSA) is created representing its canonical transcription. FSA nodes are associated the acoustic model (HMM) of the corresponding phone in the word. Then, modifications are introduced to allow deletions and substitutions. For implementation issues, intermediate nodes are used between phone nodes of the word. Deletion of a phone is modeled adding an edge from one intermediate node to the following. Alternative paths are added for each possible substitute phone. Phone substitutions are only allowed between phones from the same broad phonetic group. Added edges are given a specific probability of phone deletion and phone substitution. Insertions are not considered in this study as it is not common to insert phones in Spanish language. In addition, in a preliminary experiment allowing insertions, we found that most of the insertions come from speaker's noise confused with unvoiced or plosive phones as /s/ or /p/.

An example of such an automaton for a three phone word is drawn in Figure 2. 'Ini' and 'End' nodes represent initial and final node of the FSA, respectively.

The automatic transcription (T_{aut}) and the canonical transcription (T_{can}) are aligned by means of a Dynamic Programming algorithm. Transformations (deletions and substitutions) and their associated conditions are extracted from this alignment, following these considerations:

- Focus of a transformation can be composed by one or two phonemes.
- L and R is composed by up to two phones. Context can contain word boundary symbol (represented with symbol '$') but not phones of preceding or following words. Maximum length condition is always selected.

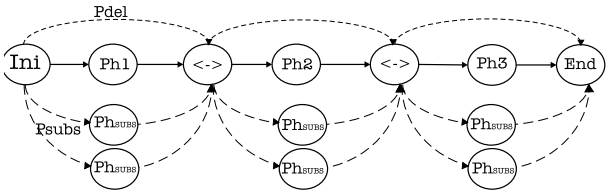

Figure 2: Finite state automaton representing the pronunciation of a word allowing deletions and substitutions

Once all training data has been parsed, transformations appearing less than N_t times are removed. This is done in order not to consider transformations due to errors in the recognizer or in the alignment phase.

Initial set of rules is composed by all the conditions associated with each remaining transformation.

2.2. HIEGRI algorithm

At this stage, for each transformation a large set of rules have been collected. Some of the rule conditions may supply significant knowledge while others, due to maximum length condition extraction, may be specific cases of a '*unknown-at the moment*' more general rule. HIEGRI algorithm is proposed to process the initial rule set in order to detect possible common patterns across conditions associated to a particular transformation and to develop the preliminary graph (HIEGRI graph) for each transformation, inferring a set of candidate general rules with different condition lengths. Note that HIEGRI graph is not a Rule Graph. HIEGRI graph nodes contain rule conditions but not rule associated probabilities.

The growing process of the graph consists of establishing a double hierarchy across rules nodes. Vertical hierarchy is established generating rules with more general conditions, stripping one element of the right or the left context of rule condition. Horizontal hierarchy is established between rules at the same level depending on the number of the upper level rules that have had generate a particular rule. Horizontal hierarchy defines the following classes of rule nodes (in hierarchical order):

- Grouping nodes. Initial rules nodes or rule nodes created by more than one rule in the upper level.
- Heir nodes. Rule nodes created by a grouping node.
- Plain nodes. Rest of the rule nodes.

For each transformation, initial rules are set on the highest level of the structure and are associated an identification number (*id*). The following steps are performed for each level, until the context-free rule level is achieved:

- Identify horizontal hierarchical class for each node in the level.
- Develop a lower level. This is done depending on horizontal hierarchy. Grouping nodes are the first to create more general rule nodes and plain nodes the latest ones. Inside each class of rule nodes, alphabetical order is used as the order criterion. For each rule r, two more general

condition rules, r_L and r_R can be generated, one removing one phoneme of the left context, and one removing one phoneme of the right context. r_L and r_R are placed on a lower level, are linked to r, and inherit rule r *id*s. It is possible that rules r_L or r_R are already in the lower level, just because they are rules of the initial set or because they have been created by another rule. In this case, linkage is not performed if any of r ids is already present in the lower level rule node r_L or r_R. This constraint is set in order not to let an initial rule create the same general rule twice and produces rule nodes without links to lower levels.

The situation at this stage of the algorithm is shown in Figure 3. Double hierarchical graph corresponds to the transformation $D \rightarrow *$, meaning /D/ deletion. In this example, four different rule conditions form the initial set of rules for this transformation. Dark grey is used to mark grouping nodes, heir nodes are drawn in medium grey, and plain nodes are not shadowed.

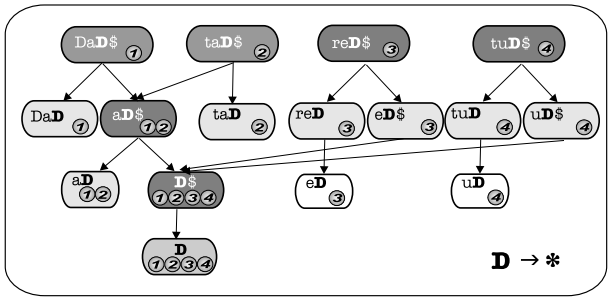

Figure 3: HIEGRI graph growing process for deletion of /D/. Different grey shadows are used to mark hierarchy at horizontal level.

The tree shaped graph is achieved parsing the hierarchical graph in a bottom-up direction erasing rule nodes not linked to its lower level, as well as their links to upper level. If a survivor rule node keeps its two bottom links, only the link with more *id*s is preserved. In Figure 4, HIEGRI graph obtained for the /D/ deletion example is shown.

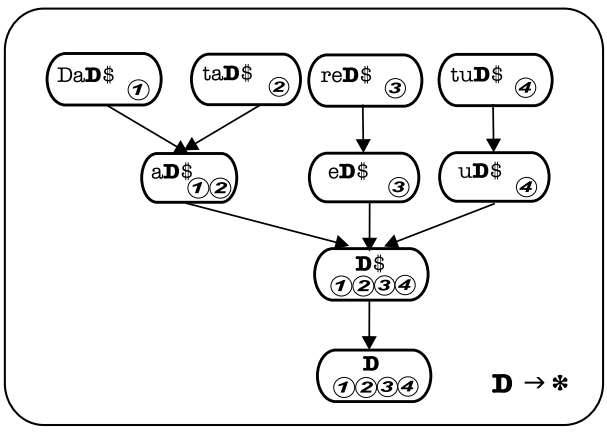

Figure 4: HIEGRI graph obtained for /D/ deletion.

2.3. Selection of final set of rules

The objective of this last step is to select as general as possible rules modeling each transformation without losing modeling accuracy. This step obtains the final Rule graph containing the probabilities for each particular rule in it. The selection strategy consist of iteratively generating subgraphs based on the HIEGRI graph.

Before entering into selection method details, it is necessary to explain how probabilities are assigned into a given Rule graph.

2.3.1. Assigning rule probabilities

Rule probabilities are approximated by rule relative frequencies. Frequency counts are collected for each node rule r in the graph. Data files are parsed in order to get counts of the times the rule condition is seen in the database (ns_r), and the times the transformation occurs in that context (no_r). Counts are assigned to the most specific rule found in the graph. Rule r probability, p_r, is obtained as no_r/ns_r.

2.3.2. Selection strategy

Selection process starts considering only the most general rule node and evaluates if it is worth adding nodes corresponding to more specific rules by means of a cost function.

Cost function is the entropy of a graph, defined as:

$$H_G = \sum_{r=0}^{R} H_r \tag{2}$$

where R is the number of rule nodes in a graph and H_r is the entropy of a rule node r. H_r is calculated with the expression:

$$H_r = p_r log_2 p_r + (1 - p_r) log_2 (1 - p_r) \tag{3}$$

Selection process is an iterative algorithm. It begins considering a subgraph containing only the most general rule node. We called it subgraph as it is a part of the HIEGRI graph.

For each iteration, nodes candidate to be added to the current subgraph are identified. A node is considered a candidate if it is linked to any of the existing nodes in the current subgraph, and if node no count is greater than a given threshold no_{th}. Different subgraphs containing each candidate node are created. H_G is evaluated for each new subgraph. Note that rule probabilities for each different subgraph can be different, since they depend on the existing nodes in each subgraph, as was explained in section 2.3.1[1]. Subgraph providing the maximum entropy reduction, if any, is selected. Selected subgraph is considered the new initial subgraph to continue the process if the entropy reduction ($\Delta \mathrm{H}_G$) is greater than a given threshold $\Delta \mathrm{H}_{Gth}$.

The process iterates until there are no more candidates in the graph or until adding existing candidates do not provide enough entropy reduction.

Figure 5 illustrates one iteration of the selection process following the example of /D/ deletion. Subgraph containing the two lowest rule nodes (D and D$) has identified candidate rule nodes to be added (marked with dotted lines). Subgraphs created for each candidate are shown in the right part of the figure.

[1]Note that is not necessary parsing training data each time entropy of a new subgraph has to be evaluated. Actually, data is parsed once and different counts are collected in order to be able to get counts for each new subgraph.

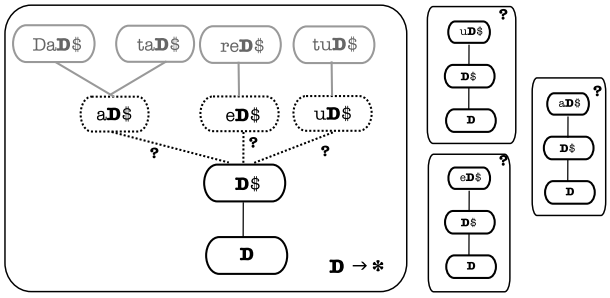

Figure 5: Selection of final rule set procedure. At this stage, current subgraph nodes are marked in black and candidate nodes are marked with dotted lines. Right part of the Figure shows subgraphs created for each candidate.

After applying the selection process, final Rule graphs for each transformation are achieved.
It is important to note:

- Rule nodes in intermediate levels can be left without counts, having probability zero. Those rules stay in the graph indicating that it is not possible to perform a transformation with that condition unless another phone is also present (condition of an upper level rule).
- Inferred rules in lower level could have been assigned a probability greater than zero. These rules kept the counts of rule nodes not selected to appear in the final Rule graph. If ΔH_G is zero, counts come from rule nodes not seen more than no_{th} times.

A possible final Rule graph for the /D/ deletion example can be seen in Figure 6, where only four rule nodes have been selected.

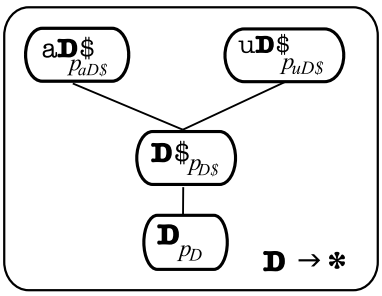

Figure 6: Rule graph model for transformation $D \rightarrow *$

3. Generating word pronunciation models

Learned rules are used to derived word pronunciation models for each word of the recognizer vocabulary. A word pronunciation model is represented with a Finite State Automaton. This FSA integrates all possible variants for a given word.

In order to achieve a word pronunciation model that represents pronunciation of a word in context-dependent acoustic models (CD-HMM), a FSA representing transcription in phones is developed in a first step. This phone-FSA also contains the '*' symbol to represent deletion of a phone. The FSA with CD-HMM will be derived from this phone-FSA.

For each word of the vocabulary a phone-FSA is initialized representing word canonical transcription. This FSA will be referenced as the canonical branch. Each node of the FSA represents a phone of the transcription (See Figure 7). Beginning from the canonical branch, in a left-to-right direction, rules are applied to generate variants. Each time a rule is applicable, variant is only generated if rule probability is greater than P_{min}. P_{min} allows to control the number of generated variants.

For each new variant a new branch (variant branch) is added to the FSA. A variant branch begins with the output of the transformation and continues with the remaining phones of the canonical transcription. First edge of the new branch is the edge to the output node, and it is given the probability of the rule generating such variant. Probability of the edge of the canonical branch is readjusted.

Once the canonical branch is entirely explored, the process continues exploring the created variant branches until there is no more branch to explore.

Figure 7 represents the generated phone-FSA for the word 'vid'. Canonical transcription for this word is /v i D/. /D/ deletion model, shown in the examples along the paper, is applied to generate variant /v i/. Selected rule in the Rule graph model is '$D\$ \rightarrow *$', with $p_{D\$}$.

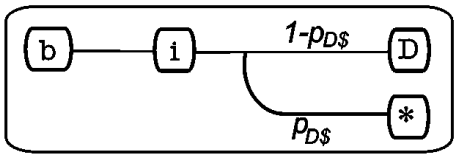

Figure 7: Phone-FSA created for the vocabulary word "vid" applying /D/ deletion model.

Such an automaton can be expanded in a straightforward manner, branch by branch, to another FSA whose nodes represent context-dependent acoustic models.

In this work, CD-HMM are demiphones [7], a contextual unit that models the half of a phoneme taking into account its immediate context. Therefore, a phone is modeled by two demiphones: '$l - ph$' '$ph + r$', where l and r stay for the left and the right phone context, respectively, and ph is the phone.

Figure 8 illustrates the obtained word pronunciation model with demiphones for word 'vid'. 'F' stays for the boundary symbol.

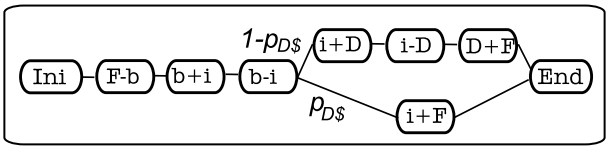

Figure 8: Word pronunciation model FSA created for the vocabulary word 'vid'. Nodes are associated CD-HMM models

4. Database

All the experiments performed were carried out on the Spanish SpeechDat II database. The database of Spanish as spoken in Spain was created in the framework of the SpeechDat II project. The database consists of fixed network telephone recordings from 4,000 different speakers. Signals were sampled and recorded from an ISDN line at 8KHz, 8 bits and coded with A-law. SpeechDat database contains 3,500 speakers for training and 500 speakers for test purposes. Database is accompanied by

a pronunciation lexicon representing word transcriptions in 30 SAMPA symbols.

Although this database does not contain spontaneous speech, speakers are not professional and do not always pronounce accurately. SpeechDat database comprises speakers covering all regional variants from Spain, so pronunciation variation due to different accents is also present.

5. Experiments

This work was developed in an in-house ASR system. The system uses Semicontinuous Hidden Markov Models (SCHMM). Speech signals are parameterized with Mel-Cepstrum and each frame is represented by their Cepstrum C, their derivatives ΔC, $\Delta\Delta$C, and the derivative of the Energy. C, ΔC, and $\Delta\Delta$C are represented by 512 Gaussians, respectively, and the Energy derivative is represented by 128 Gaussians. Each demiphone is modeled by a 2 states left to right model.

5.1. Rule generation

Rules are trained with a set of 9,500 utterances extracted from the Spanish SpeechDat II training set. Rule training set is composed by 6,470 phonetically rich sentences and 3,029 words. This set contains 67,239 running words and a vocabulary of 12,418 different words.

In order to obtain automatic transcriptions, probabilities of deletion and substitution in the word pronunciation models are adjusted empirically to 0.01. To determine the initial set of rules, minimum number of times a transformation has to be seen to be considered, N_t is fixed to 20.

With these values, 53 transformations are detected belonging to 31 different focus. Rules giving higher probabilities belong to transformations corresponding to deletion processes. This was not surprising, since it is known most substitution phenomena can be handled by HMMs.

In the selection process no_{th} is set to 10. Different rule set sizes are achieved varying ΔH_{Gth}. Setting a small value for ΔH_{Gth} provides a large set of rules. Those rules are very dependent on the training vocabulary and so are the application probabilities. As ΔH_{th} grows, specific rules dissappear in front of general inferred rules. Rule set decrease its size and become more independent of the vocabulary, but, in contrast, probabilities are smoothed and become lower. Table 1 shows sizes for different rule sets obtained varying ΔH_{Gth}. Rule set size decreases more than 50% when ΔH_{Gth} is set to 10^{-2}.

$\Delta \mathbf{H}_{Gth}$	**0**	**10^{-3}**	**10^{-2}**
Rule set size	364	306	141

Table 1: Rule set sizes varying ΔH_{Gth}

In order to compare our proposed rule learning methodology, a baseline rule set was created. The baseline rule set is composed by rules of the initial rule set. This rule set is obtained without applying HIEGRI algorithm and consequently without applying final rule selection strategy. no_{th} is set as a selection criterion. Rules that happens more than no_{th} times are selected. Due to this selection some transformations are left without rules, decreasing the number of transformations to 29 corresponding to 22 focus. Total number of obtained rules is 117. The number of the obtained rules in this case is lower than the size of the rule set obtained with HIEGRI. It has to be considered that in HIEGRI selection process, general rules are kept in the set, with a probability estimated with counts of rules not seen more then no times and/or not providing enough information. Specific rules that provides information are kept, as well. In the baseline rule set selection, rules which no is below to no_{th} are directly not considered.

Figure 9 shows the envelope of rule probabilities histograms for different rule sets: the baseline rule set, and three sets obtained with the method proposed in this paper, varying ΔH_{th}. It can be observed that baseline rule set and rule sets obtained with HIEGRI selecting a small ΔH_{Gth} are similar for probabilities higher to 0.1. Below 0.1, HIEGRI rule sets introduce general rules. When ΔH_{Gth} is increased the Figure shows the smoothing effect.

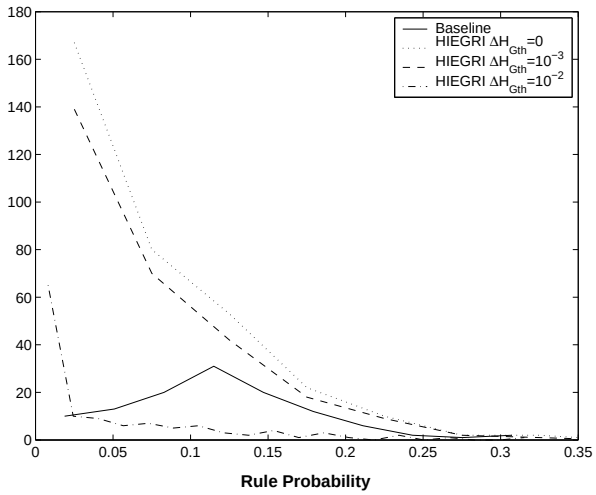

Figure 9: Envelopes of histograms of rule probabilities for different rule sets

5.2. Recognition results

Demiphones are trained with a set of 40,900 utterances, containing phonetically rich sentences and words. Training set has a total of 357,948 running words and a vocabulary of 20,062 different words.

Recognition task consists on phonetically rich sentences. Test set is composed by 1,570 sentences containing 4,744 different words. A trigram language model is create modeling all SpeechDat sentences. There is a total of 11,878 different sentences with a vocabulary of 14,300 words. Perplexity of the created language model is 68.

3,874 words appearing on the test set were seen in the rule training process. This figure means a vocabulary matching of 81.66 % between training and testing data. Having that matching percentage, selecting a small value of ΔH_{Gth} seems the most convenient option.

Three rule sets are applied to the recognition vocabulary: Baseline rule set, and HIEGRI rule sets with ΔH_{Gth}=10^{-3} and ΔH_{Gth}=10^{-2}. Varying P_{min} different number of variants per word is obtained.

Majority of the generated variants for this vocabulary results to be homophones with other words in the lexicon. Therefore, rule probabilities play an important role in order not to increase the word confusability.

Results of the recognition experiments are summarized in Table 2. Table contains WER% as well as the average number

of variants per word (V/W) generated for each rule set. Reference result, obtained without variants in the lexicon, or one entry per word, is situated in each column. In Spanish, good performance can be achieved with only one entry per word.

Baseline rule set produces a small number of word variants even when P_{min} is fixed at a small value. Rule sets obtained with HIEGRI generates up to 2.26 variants per word. Selecting intermediate P_{min} values, rule set with ΔH_{th}=10^{-2} obtains the highest number of variants per word. This rule set has less rules than the other HIEGRI sets, but rules are more general and in consequence more applicable.

All the results obtained are below the WER obtained without variants. Best relative improvement is 2.64%, obtained with a HIEGRI rule set. Recognizers behaviour when adding variants is remarkable since the large quantity of added homophones in the lexicon, and it shows that phone-learned rules can be applied with good results to context-dependent acoustic models based recognizers.

Table 2: *Recognition performance for different rule sets: baseline rule set, and rule set obtained with HIEGRI with ΔH_{th} and P_{min}.*

	Base Rule		$\Delta \mathbf{H}_{th}$ = 10^{-3}		$\Delta \mathbf{H}_{th}$ = 10^{-2}	
p_{min}	**WER**	**V/w**	**WER**	**V/w**	**WER**	**V/w**
0.02	9.82	1.53	9.72	2.26	9.77	2.26
0.05	9.75	1.44	9.77	1.86	9.68	2.05
0.07	9.72	1.41	9.81	1.64	9.59	1.78
0.09	9.62	1.29	9.62	1.39	9.60	1.36
0.10	9.71	1.26	9.57	1.30	9.65	1.33
0.12	9.64	1.14	9.69	1.23	9.75	1.03
1.00	9.83	1.00	9.83	1.00	9.83	1.00

Figure 10 shows the graphical representation of the evolution of WER adding variants to the lexicon for the different created rule sets. Depending on the selected ΔH_{th}, V/W interval where maximum improvement is achieved, varies. It can be seen that baseline rule set and rule set obtained with ΔH_{th}=10^{-3} obtaine maximum performance in a small interval of variants per word. Rule set obtained with ΔH_{th}=10^{-2} mantains its maximum WER reduction for a larger margin of variants per word.

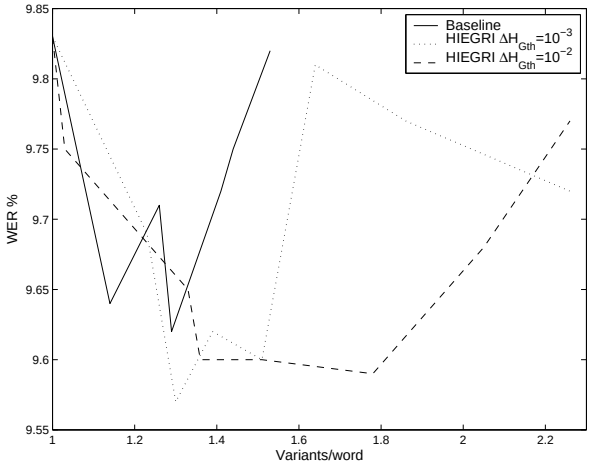

Figure 10: Evolution of WER adding variants/word for different rule sets

6. Conclusions and Future work

We have presented a pronunciation variation modeling method based on learning stochastic pronunciation rules automatically. The heart of the method is the HIEGRI algorithm that from an initial set of rules, inferres general rules and arranges them on a graph. To obtain the final Rule graphs, a selection strategy based on the HIEGRI resultant graph is proposed. Selection strategy is guided by the entropy calculated over the graph. Learned phone-based rules are applied to generate word pronunciation models that substitute pronunciation dictionary in a CD-HMM based recognizer.

Application of HIEGRI algorithm allows to generalize the rule set making it applicable to other vocabularies. As a result, the obtained rule set is able to generate more variants per word than a typical rule learning method. Applying variants to the recognizer improves the recognition accuracy. Achieved improvement with the proposed method is quite stable for a big interval of variants/word.

We are planning to apply this rule learning methodology based on HIEGRI algorithm in a open-vocabulary test set, in order to evaluate its generalization potentiality. In addition, since acoustic models are trained using canonical transcription, an improvement is presumed when applying pronunciation variation modeling to the acoustic models training process.

7. Acknowledgements

This work was granted by Spanish Government TIC 2002-04447-C02. We would like to thank Enric Monte for his help in the development of this work.

8. References

[1] Strick, H. and Cucchiarini, C., 1999. Modeling pronunciation variation for ASR: A survey of the literature. Speech Communication, Vol 29, Issues 2-4, pp. 225-246, November 1999.

[2] Ferreiros, J. and Pardo, J.M., 1999. Improving continuous speech recognition in Spanish by phone-class semicontinuous HMMs with pausing and multiple pronunciations. Speech Communication, Vol 29, Issue 1, pp. 65-76, September, 1999.

[3] Cremelie, N. and Martens, J.P., 1999. In search of better pronunciation models for speech recognition. Speech Communication, Vol 29, Issue 2-4, pp. 115-136, November, 1999.

[4] Kessens, J., Wester, M. and Strick, H., 2003. A data-driven method for modeling pronunciation variation. Speech Communication, Vol 40, Issue 4, pp. 517-534, June 2003.

[5] Korkmazskiy, F. and Juang, B.H., 1998. Statistical modeling of pronunciation and production variations for speech recognition. Proceedings of ICSLP 98,Sydney, Australia.

[6] Yang, Q., Martens, J.P., Ghesquiere, P.J. and Compernolle, D.V., 2002. Pronunciation Variation Modeling for ASR Large improvements are possible but small ones are likely to achieve. Proceedings of ISCA Tutorial and Research Workshop:Pronunciation Modeling and Lexicon Adaptation for Spoken Language. Colorado, USA, September 2002.

[7] Mariño, J.B., Pachés-Leal, P., Nogueiras A., 1998. The Demiphone versus the Triphone in a Decision-Tree State-Tying Framework. In Proceedings ICSLP, Sydney, Australia, 1998, Vol. I, pp. 477–480.