AUTOMATIC RULE-BASED GENERATION OF WORD PRONUNCIATION NETWORKS

Nick Cremelie and Jean-Pierre Martens
ELIS, University of Gent, St.-Pietersnieuwstraat 41, B-9000 Gent (Belgium)
E-mail: cremelie@elis.rug.ac.be

ABSTRACT

In this paper a method for generating word pronunciation networks for speech recognition is proposed. The networks incorporate different acceptable pronunciation variants for each word. These variants are determined by applying pronunciation rules to the standard pronunciation of the words. Instead of a manual search, an automatic learning procedure is used to compose a sensible set of rules. The learning algorithm compairs the standard pronunciation of each utterance in a training corpus with its auditory transcription (i.e. 'how should it be pronounced' versus 'how was it actually pronounced'). It is shown that the latter transcription can be constructed with the assistance of a speech recognizer. Experimental results on a Dutch database and on TIMIT demonstrate that the pronunciation networks reduce the word error rate significantly.

1. INTRODUCTION

In most continuous speech recognizers, the acoustic models are based on sub-word units (phones, syllables,...). Hence, in order to recognize words, one needs word models describing the pronunciation of the words in terms of these units. In their most simple form, the models allow one pronunciation for each word. However, the actual pronunciation of a word may differ significantly from the prescribed one. For some words several widely accepted pronunciation variants may exist; those can be considered as *intra-word* variants. Coarticulation between words can introduce even more pronunciation variations; these can be viewed as *inter-word* pronunciation variations.

Modeling all the acceptable pronunciations of the words [1, 2, 3] is likely to result in a higher recognition accuracy, since there is more chance that the pronunciation matching the uttered speech is accommodated. The more discriminative the recognizer is, the more important this closer match may be.

We developed a method for generating word pronunciation networks representing different possible pronunciations of each word. The networks are generated automatically, starting from a single 'standard' pronunciation of each word. The pronunciation alternatives are produced by applying pronunciation rules to the standard pronunciation. These rules actually *rewrite* the standard pronunciation. Although such rules may be available from phonology, we prefer to learn them from a training set. Such an automatic training procedure facilitates the adaptation of the recognizer to new tasks and/or languages. Moreover, the pronunciation variants must fit in a probabilistic framework. Therefore one needs to know the likelihood that a rule has to be applied. This information can easily be gathered during the training procedure. In [4], we proposed a preliminary version of the pronunciation rule learning algorithm. Since then, we have further improved this algorithm, as will be described in the following sections. We will assume the sub-word units to be phones. With some appropriate modifications, the same method can be used with other types of sub-word units.

2. LEARNING PRONUNCIATION RULES

We adopt the following form for the pronunciation rules:

$$L\underline{F}R \rightarrow F'$$

This rule reads as follows: the *focus* F, when preceded by a *left context* L and succeeded by a *right context* R, can be modified to F'. F contains one or more phones and possibly a word boundary (further on denoted as %). L and R contain 0 or more phones. The part $L\underline{F}R$ is called the rule *condition*, F' is called the rule *output*. In the rule condition, we underline the part that is modified by the rule. This way it can be distinguished from the context parts, which are not modified.

In order to postulate acceptable rules, two possible phonetic transcriptions of each training utterance, a *standard* transcription T_{st} and a more realistic *auditory* transcription T_{au}, are compared to one another. T_{st} is composed of the standard pronunciations (retrieved from a dictionary and supplemented by the word boundary symbol %) of the words in the utterance. In the next section we will explain how T_{au} is obtained.

The transcriptions T_{st} and T_{au} are aligned with each other. The alignment procedure is a dynamic programming search, returning the path with the minimal total cost. This cost is the sum of local contributions given by 0 if the two compared phones are identical and 1 otherwise. The alignment path shows which phones in T_{st} correspond to which phones in T_{au}. Any conflict emerging from this alignment — i.e. a substitution, insertion of

deletion of a phone in T_{au} — leads to one or more pronunciation rules. Let F be a phone or group of phones in T_{st} differing from the corresponding phone(s) in T_{au}. We consider four different rule conditions C_i ($i = 1, ..., 4$):

- $C_1 = \underline{F}$: no context;
- $C_2 = p\underline{F}$: left context (p is the phone preceding F in T_{st});
- $C_3 = \underline{F}q$: right context (q is the phone succeeding F in T_{st});
- $C_4 = p\underline{F}q$: left and right context.

Each C_i introduces a rewrite rule $C_i \rightarrow F'$ with F' being the phone(s) in T_{au} corresponding to F.

After processing a representative training set in the above manner, a possibly large number of rules is obtained. Generally this set will be too large to be practical, and moreover it will contain rules that are too specific. In order to prune rather unlikely pronunciation rules, we gather two different frequency counts during the training:

1. The number of times the different rule conditions occur in the T_{st} transcriptions of the training utterances. These numbers indicate the *coverage* of each rule.
2. The number of times the rule had to be applied on the training utterances, given that the rule condition was observed in the standard transcriptions. From these numbers we compute the *rule application likelihood* by dividing by the frequency count of the rule condition.

On the basis of the coverages and the application likelihoods, a considerable part of the rules can be rejected. Indeed, it is feasible to remove rules with a very low coverage, as the loss of information will be minimal. Similarly, rules with a small application likelihood describe a very rare transformation and can therefore be excluded as well.

In a further pruning step we try to remove specific rules in favor of more general ones. For that purpose we define the term *parent rule* of a rule. Rule R_a is a parent of rule R_b, if R_a and R_b have the same focus F and the same output F', *and* if moreover R_b is obtained by extending the context of R_a with the appropriate phones. Now suppose that the application likelihood of the parent rule R_a is similar (within some limits) to the one of R_b. In such a case it seems acceptable to remove R_b, since its transformation is already accurately described by R_a: applying R_a yields the same result, with the same likelihood.

The above pruning steps reduce the original set of rules to a more compact and more general set. Note that, for each rule condition appearing in this set, an identity rule is defined implicitly ($L\underline{F}R \rightarrow F$). Obviously, some of the rules can still be parent rules of others (but with a different application likelihood). Evidently, if a particular rule R_b is applicable to a given example, all of its parent rules are also applicable. E.g. if `ft%b` $\rightarrow$ `d` is applicable (like in `heft%be`), the parent rule `t%b` $\rightarrow$ `d` is applicable as well. In this case we would only apply the most specific rule, as this one provides the maximum context overlap between the rule condition and the example to be transformed. Generally, we can maintain the following principle when applying the rules: of all applicable rules, the ones being parents of other applicable rules are *not* applied. Note that still more than one rule can be applied to a given example, but each applied rule will lead to a different output and will be as specific as possible. This strategy has an important consequence. Since the rule `t%b` $\rightarrow$ `d` would not be applied in the above example, it can actually be considered as a rule `*t%b` $\rightarrow$ `d`, with `*` being any phone different from `f`. Indeed, knowing that a more specific rule exists (`ft%b` $\rightarrow$ `d`), the rule `t%b` $\rightarrow$ `d` will only be applied when the `t` is not preceded by an `f`. If we do take this into account, it becomes necessary to recompute the frequency counts on the training data. Consider a particular training example (i.e. a part of a transcription T_{st} and the corresponding part of T_{au}). Let S be the set of rules that are applicable to the example *and* return the correct result (i.e. the rule condition *and* the rule output match the example), then the example must only be counted as an example for the most specific rule in S. This is the only rule in S that has no parents in S.

3. DETERMINATION OF T_{au}

In the first version of our pronunciation rule learning algorithm [4], we suggested to use an auditory transcription T_{au} determined by a human expert. However, when manually obtained transcriptions are not already available, creating them may be too costly a job. Hence we developed an algorithm that relies on the speech recognizer outputs in order to determine a reasonable auditory transcription T_{au} for each training utterance.

For each utterance we start with T_{st}. From this standard transcription, a simple pronunciation network is built. The network has an entry state and one additional state per phone in T_{st}. Between two consecutive states, we put a transition emitting the corresponding phone in T_{st}. A deletion transition is added in parallel with each emitting transition, thus allowing the deletion of the phone. Additionally, an insertion transition is provided on each state, to allow the insertion of garbage units. Evidently, all transitions in the model receive adequate transition probabilities. The above explanation is depicted on figure 1.

Once the pronunciation network is constructed, a constrained alignment of the acoustic data with this network is performed by the recognizer. The deletion and in-

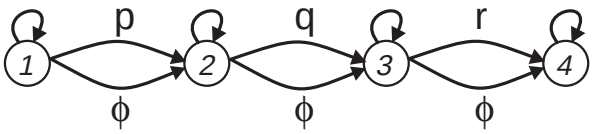

Figure 1: *Pronunciation model derived from* T_{st} = `pqr`*;* ϕ *denotes a deletion transition.*

sertion transitions in the model facilitate or improve the alignment whenever the actually uttered phone sequence differs from the expected sequence T_{st}. By examining the alignment path, some hints for improving the T_{st} model are collected:

1. A phone p modeled in T_{st} may appear deleted in the alignment. It is possible that p was indeed not pronounced. However, the deletion may also indicate that a phone different from p was pronounced. For that reason we supplement the deletion arc with one transition for every phone different from p.

2. It may occur that the acoustic score on a segment s is small for the phone p it is aligned with, while on the other hand it is large for a phone q different from p. In such case we add an extra transition emitting q in parallel with the one emitting p.

3. Some segments in the utterance may not be aligned with any phone in T_{st}. Hence at the appropriate position in the model we insert an extra transition emitting the acoustically most likely inserted phone.

In a second iteration the acoustics are aligned with the improved model. The phone sequence along the alignment path is proposed as T_{au}.

4. PRONUNCIATION NETWORKS

The set of pronunciation rules contains two types of rules: word internal rules and inter-word coarticulation rules. The latter ones describe pronunciation changes at the word boundaries. While the word internal rules are applied to individual words, the coarticulation rules are applied to word pairs. This results in a number of pronunciation variants. All variants of a word are compiled into a pronunciation network, using an algorithm related to the one presented in [3]. Obviously, we must take into account the likelihood of the pronunciation variations, determined by the application likelihoods of the applied rules. Likelihoods originating from coarticulation rules are not considered at this point since we will combine them with the language transition probability between words (see further). First, we build a left-to-right model for each variant (as in figure 1, but without deletions and insertions). At the entry and the exit of each model, we include the identity of the applied coarticulation rule. Transitions with an emission originating from a word internal rule receive a probability equal to the application likelihood of that rule; all other transitions have a probability equal to 1. A transition is now identified by its emission *and* its probability. Second, we combine these models into a tree, going from left to right. There is a new root node for each rule by which the word can be entered. Finally, nodes that have an identical tree attached to them (seen to the right, including the rule identities on the exits) are merged. An example is shown in figure 2.

The resulting models have different *conditional* entries and exits: a particular exit of a model can only connect to a

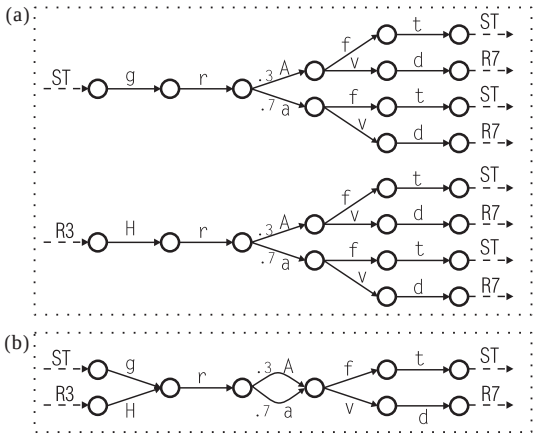

Figure 2: *Generation of Pronunciation Network for the variants* `graft`, `grAft`, `Hraft`, `HrAft`, `gravd`, `grAvd`, `Hravd` *and* `HrAvd`. *(a) Situation after tree encoding. (b) Situation after merging states. The dashed arrows are conditional entries and exits; indicated are the indices of the rules from which they originate (ST = standard pronunciation).*

'compatible' entry of an other model; this means that they originate from the same coarticulation rule. The transition probability P_t between an exit of a word and the matching entry of another word is now given by (bigram grammar):

$$P_t = P(R_c, w_2|w_1) = P(w_2|w_1) \cdot P(R_c|w_1, w_2)$$

The first factor originates from the grammar, while the second one can be estimated by $P(F'|L\underline{F}R)$, i.e. the application likelihood of the applied coarticulation rule R_c.

5. EXPERIMENTAL RESULTS

The recognizer used in the experiments is the Neural Net / Dynamic Programming Hybrid Continuous Speech Recognizer, described in [5, 6, 7]. This recognizer adopts a connectionist approach to model stochastic segments, implying a fairly good discriminative behaviour. The system is trained and tested on a Dutch (Flemish) database and on the American English TIMIT database.

Three different recognizer setups are compared. In the baseline system, the word models implement the standard pronunciation of each word (although deletions and insertions are allowed). In both the second and the third setup we use pronunciation networks derived from pronunciation rules, but the rules are obtained in two different ways: on the one hand from automatically generated auditory transcriptions (PRONNET-AUTO), on the other hand from manually determined auditory transcriptions (PRONNET-MAN) that are available for both databases. Note that, on average, the number of states in the pronunciation networks is just about 15% higher than in the corresponding standard pronunciation networks, so there is little influence on the CPU cycles required for the recognition.

The Dutch training database has a vocabulary of 413 different words and contains speech material from 80 different speakers (13 sentences per speaker). About 500

	Task 1	Task 2
WER Baseline	7.19%	24.73%
WER PRONNET-AUTO	4.60%	20.83%
Relative Improvement	36%	16%
WER PRONNET-MAN	3.97%	19.56%
Relative Improvement	45%	21%

Table 1: *Word Error Rates (WER=D+I+S) and Relative Improvement (compared to Baseline) when pronunciation networks are used for Dutch tasks.*

	SX	SI
WER Baseline	3.77%	13.45%
WER PRONNET-AUTO	3.10%	10.85%
Relative Improvement	18%	19%
WER PRONNET-MAN	3.28%	11.36%
Relative Improvement	13%	16%

Table 2: *Word Error Rates and Relative Improvement when pronunciation networks are used for* TIMIT.

pronunciation rules survive the pruning; 80% of them are coarticulation rules. Two different recognition tasks are considered. In Task 1, the vocabulary is an extension of the one of the training set (+15% new words). A bigram, trained on a set of sentences different from the ones in the training set, is used as the language model. Task 2 is completely different from the training set. There are 80% new words in the vocabulary. It is an ATIS-like task with inherently more difficulties (pauses, noises, more confusable words, ill-formed sentences). A bigram language model for this task is determined on a dedicated training set. For each task a test set containing utterances from 10 different speakers is available. The recognition results with the different recognizer setups are shown in table 1.

In the TIMIT case, the recognizer is trained on the SX+SI training sentences. The vocabulary (6227 different words) and the bigram language are both determined on this training set. The pronunciation rules are learned on the SX set. About 750 pronunciation rules are retained; again 80% are coarticulation rules. The recognition tests are run on the SX, resp. SI test set. The results are shown in table 2.

These results indicate that dealing with the pronunciation variants does yield a significant improvement of the recognition performance. Apparently, the proposed strategy is capable of learning rather general pronunciation rules. Indeed, even on Task 2 (Dutch) and on the SI task (TIMIT) — tasks which differ strongly from the original rule training set — there is still a significant reduction of the word error rate. A more detailed investigation revealed that the major part of the improvement originates from the coarticulation rules rather than from the word internal rules.

As to the different rule learning strategies, it appears that the rules derived from automatically generated auditory transcriptions (used in PRONNET-AUTO) perform about equally well as the ones derived from manual transcriptions (PRONNET-MAN). On TIMIT, the PRONNET-AUTO setup even yields slightly better results while on the Dutch task, the PRONNET-MAN setup performs best. One possible explanation is that the acoustic models for TIMIT are better trained than those for the Dutch database. Hence the automatic auditory transcriptions are more reliable for TIMIT than they are for the Dutch task. It is very likely that the automatic transcriptions catch some of the peculiarities of the recognizer. This can explain why the setup PRONNET-AUTO sometimes outperforms PRONNET-MAN.

6. CONCLUSION

In this paper we introduced a fully automatic method for generating word pronunciation networks representing the different pronunciation variants of the words. The networks are generated through pronunciation rules, which in turn are learned automatically from a speech training corpus. The learning algorithm relies on the outputs of the speech recognizer. The method was tested on a Dutch database as well as on the TIMIT database. The results show that the pronunciation networks outperform single pronunciation word models. Moreover, the pronunciation rules appear to be general enough to remain effective on new tasks with a vocabulary different from the one of the training set.

REFERENCES

[1] P. Schmid, R. Cole, M. Fanty (1993), "Automatically generated word pronunciations from phoneme classifier output," in *Proceedings of ICASSP-93*, pp. II-223 - II-226.

[2] R. Cardin, Y. Normandin, E. Millien (1993), "Inter-Word Coarticulation Modeling and MMIE Training for Improved Connected Digit Recognition," in *Proceedings of ICASSP-93*, pp. II-243 - II-246.

[3] R. Mercer, P. Cohen (1987), "A method for efficient storage and rapid application of context-sensitive phonological rules for automatic speech recognition," in *IBM J. Res. Develop.* Vol. 31, No. 1, January 1987, pp. 81-90.

[4] N. Cremelie, J.P. Martens (1995), "On the use of pronunciation rules for improved word recognition", in *Proceedings EUROSPEECH-95*, pp. 1747-1750.

[5] J.P. Martens, A. Vorstermans, N. Cremelie (1993), "A new Dynamic Programming/ Multi-Layer Perceptron Hybrid for continuous speech recognition," in *Proceedings of EUROSPEECH-93*, pp. 1937-1940.

[6] N. Cremelie, J.P. Martens (1994), "Time Synchronous Heuristic Search in a Stochastic Segment Based Recognizer," in *Proceedings ICSLP-94*, pp. 275-278.

[7] J. Verhasselt, I. Illina, J.P. Martens, Y. Gong, J.P. Haton (1997), "The importance of segmentation probability in segment based speech recognizers," in *Proceedings ICASSP-97*, pp. 1407-1410.