DIRECTORY NAME RETRIEVAL USING HMM MODELING AND ROBUST LEXICAL ACCESS

G. Gravier F. Yvon G. Etorre G. Chollet

Ecole Nationale Supérieure des Télécommunications / CNRS-URA 820
46, rue Barrault
75634 Paris Cedex 13, FRANCE

Abstract - A system to retrieve names in a directory from the pronunciation and the spelling is presented, for telephone quality speech in French. The recognizer is based on Hidden Markov modeling of speech units using word models for letters and phone models for name pronunciation. The directory is build automatically from a list of names using a grapheme to phoneme converter for the names and rules for spellings, each entry in the directory consisting in several pronunciation and spelling variants. From the acoustic recognition, the corresponding entry in the directory is found using dynamic alignment of symbol sequences, with insertion, deletion and substitution costs fixed or determined from the training data to take into account acoustic confusability. Experimental results show that a mixed recognition strategy combining both spelling and pronunciation improves over the spelling based strategy. The effectiveness of integrating learnt phonetic confusability information during the lexical search is also demonstrated.

1 Introduction

Automatic retrieval of names from their pronunciation and spelling is a popular topic in speech recognition, one of the most common application being directory name retrieval for telephone services. Many studies address the problem of alphabet recognition, which is known to be a difficult task because of acoustic similarities between some letters (e.g. the well-known E-set in English). When working with telephone quality speech, the confusability between letters increases drastically. Cole *et al.* presented experiments with telephone speech for the recognition of English and French alphabets [1, 2]. Letters, separated by pauses, are segmented and then classified using a MLP with phonetic features. More recently, HMM based methods have been proposed, as in [3] and [4] where letter models are used. The first study compares DTW and HMM based lexical search strategies to retrieve names from natural spelling. HMM based lexical search can also be seen as a DTW whose insertion, deletion and substitution costs are learnt from the training

data. The second study is based on a multi-level classification with a N-best approach, using DTW and a restricted grammar for lexical search.
These studies however fail to address a number of problems that are critical in the perspective of real applications. First, they greatly overestimate the variability observed in real-life spellings. Second, phonetic knowledge is mainly used to improve letter discrimination but not for DTW-based lexical access. Finally, the name itself (i.e. the pronunciation of the name) is a source of information for name retrieval applications that has hardly been used (see however [5]). In this paper, experiments on name retrieval from the pronunciations and the spellings are reported, using Hidden Markov models of speech units and robust lexical access based on phoneme or letter confusability. The name directory is build automatically and integrates pronunciation and spelling variants. The paper is organized as follows. In the next section, the database is described and the construction of the directory is explained. Acoustic modeling is then presented, along with some results. In section 4, different retrieval strategies are evaluated. The various possible improvements of this baseline system are finally discussed.

2 Database and directory

The Swiss-French Polyphone database[1] was recorded over telephone lines with 5,000 speakers calling once, a call including the pronunciation and spelling of 3 names, where a name can be a person name, a city name or any common noun. Spellings were prompted but no specific spelling guidelines were given to the callers. As a result, in addition to "standard" spellings, the corpus contains occurrences of comparisons, such as "*a comme alain*" (a not so uncommon way to minimize confusions between letters), occurrences of aeronautic-like spellings, eg. "*alpha bravo ...*", etc.
A subset of the database is used, containing 11,920 speech segments from 3,998 speakers. This subset was divided in three corpora. The first one is the train corpus, containing 5,390 segments from 3,223 speakers. The remaining items belongs to the test corpus, from which a special subset, the "*clean*" test corpus, is extracted. This "*clean*" test corpus contains the items for which the spelling conforms to the "standard" spelling conventions. The "*clean*" test corpus contains 5,015 segments from 3,097 speakers, corresponding to 3,478 different names and is used to evaluate the quality of acoustic modeling. The directory in which names are to be retrieved contains the 8,261 names presented to the callers. Given the typical size of name directories, we cannot rely upon a whole word based recognition. Therefore, each directory entry contains one or several phonetic transcription(s) of the name, plus one or several possible spellings. These phonetic transcriptions were automatically produced by a grapheme to phoneme converter developed during the

[1] The Swiss-French Polyphone database, recorded by Swiss Telecom PTT, is distributed by the European Language Resource Association.
http://www.icp.inpg.fr/ELRA

course of the Onomastica project [6, 7]. This transcriber has been explicitly devised to cope with proper names idiosyncrasies. It also contains a module for recognition of proper name origins and is likely to output pronunciation variants. The spelling variants are generated by a rule-based system. For each letter (or cluster of letters), all possible pronunciations are considered. For instance, the letter "é" can be spelled "*e accent aigu*", "*e aigu*" or "*é*"; the cluster "nn" can be spelled "*deux n*" or "*n n*"... In the remainder of the paper, a directory entry e_i will be refered to as $\{n_{i,j=1,\ldots,N_i}, s_{i,j=1,\ldots,S_i}\}$, where $n_{i,j}$ (resp. $s_{i,j}$) is the j-th pronunciation (resp. spelling) variant.

3 Acoustic modeling

3.1 Models and grammar

The first processing stage consists in acoustic decoding. Speech is encoded using 12 Mel frequency cepstral coefficients and log energy, with first and second order derivatives, computed every 10 ms on 25.6 ms frames. To estimate phone models for recognition of names and letter models for recognition of spellings, the training corpus is first automatically segmented from its orthographic transcription, using speaker-independent phoneme models. The latter are also used as bootstrap models. Phoneme model parameters are then re-estimated using standard Baum-Welch algorithm. Letter models are created by concatenating the bootstrap models that correspond to the letter pronunciation, and then performing re-estimation. Because of the possible liaisons after some words such as "*deux*", some letters can have two models, one with the liaison and one without.
A simple grammar is also used during the decoding procedure. This grammar describes an utterance as a name followed by a spelling where a name can be any sequence of phonemes and a spelling any sequence of letters. It is important to realize that our grammatical model allows one or the other part of the utterance to be skipped by the speaker, a very likely situation in real applications. A short silence is forced after each letter in the spelling. In fact, a preliminary study, inspired from [8], showed that forcing a short silence gives better accuracy for letter recognition. Using the grammar defined above, the recognizer finds out the most likely sequence in the speech signal.

3.2 Results and discussion

In this section, results concerning speech unit recognition are presented and discussed, allowing for an independent evaluation of the acoustic models. Table 1 reports the phone and letter recognition rate and accuracy. The accuracy corresponds to the number of correctly recognized units minus the number of insertions and is a much more reliable measure of the recognition quality than the correct recognition rate. As can be seen, though the recognition rate is quite good, there are many phoneme insertions.

	phonemes		letters	
	correct	accuracy	correct	accuracy
train	58.71	14.42	83.16	74.56
clean test	57.99	-6.72	81.53	69.72

Table 1: Recognition rate (in %) and accuracy for phonemes and letters

Reducing the number of insertion can be done by introducing a fixed probability for the transition between two models. The smaller the probability, the smaller the number of insertions. This, however, tends to increase the number of deletions. Figure 1 plots the correct recognition rates and accuracy on the train corpus as a function of the fixed probability. The name retrieval rate is also plotted. However, it can be noted that the name recognition rate on the training corpus does not really improve as the fixed transition probability increases. Those results also point out the fact that letter recognition is much more reliable than phone recognition.

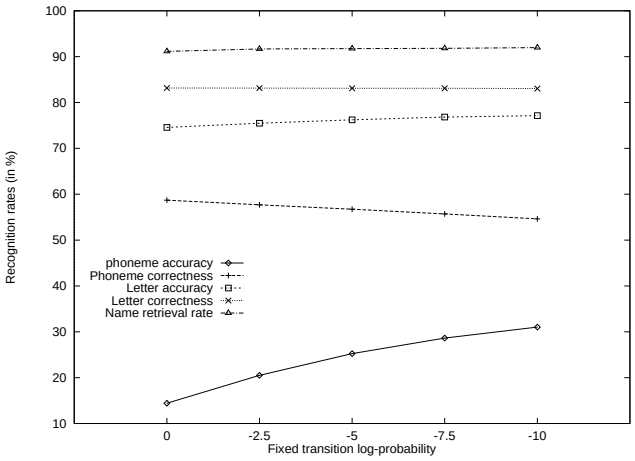

Figure 1: Recognition rates on the train corpus as a function of a fixed transition penalty

A more detailed study of the letter recognition errors outlines the standard confusions between acoustically similar letters such as "b" and "d", "f" and "s" or "p" and "t". As noted in many studies on alphabet recognition (eg. [3, 9]), letter HMMs are not really able to distinguish two letters whose spellings only differ by a short transitional acoustic event. For example, letters "m" and "n" shares the same vowel and are separable by the last consonant. But phonemes /*m*/ and /*n*/ are quite similar, and their confusion is also one of the most common substitution at the phoneme level.

4 Lexical search

4.1 Search strategy

The second step consists in retrieving the name in the directory from the recognized strings. This is done by calculating the dissimilarity measure between the recognized form r and each lexical entry e_i according to:

$$D(r, e_i) = \beta(\min_j d(r_s, s_{i,j})) + (1 - \beta)(\min_j d(r_n, n_{i,j}))$$

where r_s (resp. r_n) is the recognized spelling (resp. pronunciation). The retrieved name is the one for which the dissimilarity is the smallest. Parameter β allows to balance the respective contributions of the spelling and pronunciation. Dissimilarity $d(.,.)$ is computed by dynamic alignment using specific costs for each possible substitution, insertion and deletion. Those costs are usually arbitrarily fixed. However, as some pairs of symbols (letters or phonemes) are more confusable than others, it seems natural to assign a smaller cost for the substitution of confusable symbols. Therefore, the weighted cost for the substitution of x by y is $-\log(p(y|x))$, where $p(y|x)$ is estimated on the training corpus. The same procedure is used to determine insertion and deletion costs. This approach is somewhat similar to HMM based alignment procedure in [3]. Instead of using phonetic knowledge to achieve a better discrimination as in [9], this knowledge is learnt from the training data and used for name retrieval rather than for symbol recognition. Setting $\beta = 0.5$ and without fixed transition probability between models, name recognition rate achieves 52.11% with binary costs, and 82.11% with weighted costs.
For the experiments on lexical access, a fixed transition log-probability of -7.5 is used. This choice is rather arbitrary, since the insertion costs are taken into account directly in the computation of the dissimilarity measures.

Results on the entire and *clean* test corpus are reported for various values of parameter β in figure 2.
Unsurprisingly, results are slightly better on the *clean* test corpus. Using only the pronunciation ($\beta = 0$), the recognition rate is around 23% on both corpus, while using only the spelling it is around 76% (resp. 81%) on the entire (resp. *clean*) test corpus. The best recognition rate is achieved for $\beta = 0.625$ with 79.09% of the names recognized correctly, which reflects that even though the pronunciation of the name is a valuable source of information for the name retrieval procedure, spellings remain the more reliable piece of information.

4.2 Pre-selection

The name retrieval procedure described in the previous section is extremely time-consuming, as it requires the computation of dynamic alignments between, the recognized spelling and pronunciation, and each variant of each lexical entry. Obviously, this brute force approach is only applicable in the context of small or medium sized name directories.

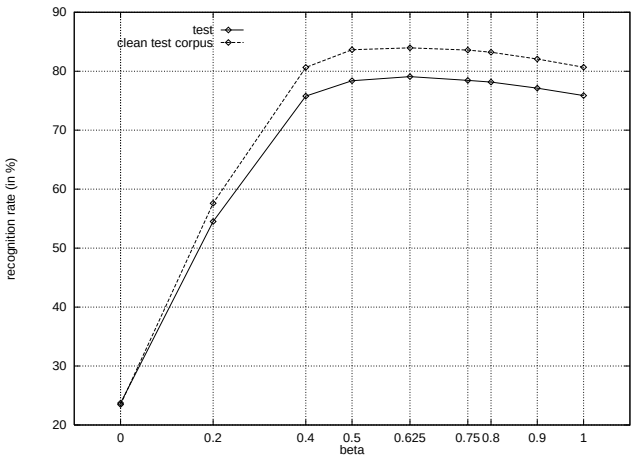

Figure 2: Name recognition rate as a function of β

As a first step towards very-large directory name retrieval, an alternative search strategy is evaluated in this section. This new strategy consists in the two following stages: first, on the basis of the spelling alone, a set of N possible entries are selected using a fast approximative lexical search; this subset is then exhaustively explored, according to the procedure defined in the previous section.

The approximative lexical search implements a variant of the algorithm originally proposed in [10] for error-tolerant lexical recognition. Given a possibly erroneous form, this algorithm efficiently retrieves every lexical entry whose edit distance (computed as the number of deletions, insertions and substitutions between two forms) to the form fells below a pre-defined threshold d. This search procedure crucially relies upon a finite-state representation (tree-based, in this implementation) of the lexicon, which makes it possible to optimize the computation of edit distances. Furthermore, using uniform unitary costs for weighting insertions, deletions and substitutions allows to compute, given the edit distance between x and a lexical prefix s, a lower-bound of the minimum edit distance between the searched form and every lexical entry whose prefix is s. As a consequence, the traversal of the tree can be dramatically speeded-up, since the exploration of entire branches can be stopped after the computation of the distance between the form and lexical prefixes.

The pre-selection implementation follows the same lines, with one slight difference: we are interested in retrieving a fixed number N of lexical neighbors, rather than all the neighbors within a given distance. Accordingly, the threshold d is set to be equal to the Nth closest distance, and let this threshold decrease as better lexical candidates are retrieved. If this patch appears to somehow slowen the search, the preselection stage remains however extremely fast. Computation of an entry is about 9 times faster with a pre-selection of

100 lexical neighbors with a very small decrease in terms of recognition rate.

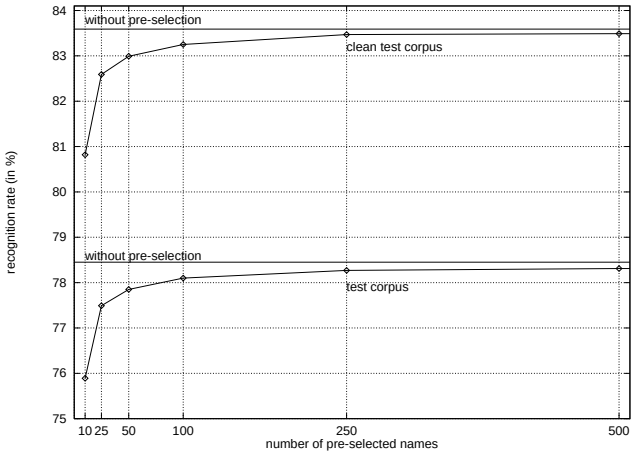

Figure 3: Recognition rates on the train corpus as a function of a fixed transition penalty

Figure 3 plots the recognition rates as a function of the number of neighbors selected. As N increases, the recognition rate tends toward the rate obtained with no pre-selection. With only 250 neighbors, the recognition rate is very close to the optimal one and the computation times are severely reduced.

5 Conclusion

Experiments on letter recognition using Hidden Markov models outlined that this technique does not avoid standard confusions between phonetically similar letters. However, the knowledge of letter confusability can be used to improve the lexical search by adapting the dynamic alignment costs to the possible confusions. In addition, an algorithm for the selection of a list of possible entries speeds up the computation time required for a lexical access with a very small loss in the recognition rate. The evaluation presented in the study were made with real-life (i.e. unconstrained) natural pronunciations and spellings.

This survey presents a baseline system on which many improvements can be done. A simple bigram language model, estimated on the lexicon itself, could be used to help the acoustic decoding step. Also, when looking at the 10 closest entries on the lexicon instead of the first one, the recognition rate is 88.66% (against 79.09% for the closest entry) on the entire test corpus. Rescoring strategies of the 10 best hypothesis is also a mean to improve this baseline system. Finally, nothing is done at current time to deal with "non standard" spellings. With the grammar defined, "non standard" spellings maybe recognized as a sequence of phones or letters, thus introducing errors

during the lexical search. It seems obvious that a system able to detect and ignore (or use) "non standard" spelling should perform better on the entire test corpus.

Acknowledgment

This work is partially supported by SwissCom.

References

[1] Ronald Cole, Krist Roginski, and Mark Fanty. English alphabet recognition with telephone speech. In *EuroSpeech*, pages 479–482, 1991.

[2] P. Schmid et al. Real-time, neural network-based, French alphabet recognition with telephone speech. In *EuroSpeech*, pages 1723–1726, 1993.

[3] D. Jouvet et al. Speaker-independent spelling recognition over the telephone. In *Int. Conf. on ASSP*, volume 2, pages 235–238, 1993.

[4] Jean-Claude Junqua et al. An n-best strategy, dynamic grammars and selectively trained neural networks for real-time recognition of continuously spelled names over the telephone. In *Int. Conf. on ASSP*, pages 852–855, 1995.

[5] Michael Meyer and Hermann Hild. Recognition of spoken and spelled proper names. In *EuroSpeech*, pages 1579–1582, 1997.

[6] M. Schmidt, S. Fitt, C. Scott, and M. Jack. Phonetic transcription standards for European names (ONOMASTICA). In *EuroSpeech*, volume 1, pages 279–282, 1993.

[7] François Yvon. *Prononcer par analogie: motivation, formalisation et évaluation*. PhD thesis, ENST, 1996.

[8] David Pye. Automatic recognition of continuous spelled Swiss-German letters. Technical report, IDIAP, 1994.

[9] Philipos C. Loizou and Andreas S. Spanias. High-performance alphabet recognition. *IEEE Trans. on Speech and Audio Processing*, 4(6):430–445, November 1996.

[10] Kemal Oflazer. Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, 22(1):73–89, 1996.