PHONEME-TO-GRAPHEME CONVERSION FOR OUT-OF-VOCABULARY WORDS IN LARGE VOCABULARY SPEECH RECOGNITION

Bart Decadt[1], *Jacques Duchateau*[2], *Walter Daelemans*[1] *and Patrick Wambacq*[2]

[1]CNTS Language Technology Group, University of Antwerp, Belgium
[2]ESAT - PSI, Katholieke Universiteit Leuven, Belgium
e-mail: {decadt,daelem}@uia.ua.ac.be {Jacques.Duchateau,Patrick.Wambacq}@esat.kuleuven.ac.be

ABSTRACT

In this paper, we describe a method to enhance the readability of the textual output in a large vocabulary continuous speech recognition system when out-of-vocabulary words occur.

The basic idea is to replace uncertain words in the transcriptions with a phoneme recognition result that is post-processed using a phoneme-to-grapheme converter. This converter turns phoneme strings into grapheme strings and is trained using machine learning techniques.

Experiments show that, even when the grapheme strings are not fully correct, the resulting transcriptions are more easily readable than the original ones.

1. INTRODUCTION

In large vocabulary recognition systems, the occurrence of out-of-vocabulary (OOV) words in the input speech is inevitable: the known vocabulary will never be complete due to the existence of for instance neologisms, proper names, and compounds in some languages.

The point is to react properly when an OOV word occurs, depending on the application. In automatic transcriptions of speech data – for example broadcast news data – the outcome of the recognition process is basically text-only. OOV words degrade the result as (1) the reader doesn't know when they occur and (2) each OOV word is poorly transcribed in terms of the known vocabulary.

Therefore it is useful to add some indication of the reliability of the words in an automatic transcription. One can think about a visual representation (for instance with color codes) of a word level confidence score. Also it can be decided to replace a word in the transcription by a readable representation of its phonetic contents whenever the confidence score for the word is below a threshold.

The feasibility of the latter solution is investigated in this paper. The idea is that for each word with a low confidence score, the corresponding acoustic data is sent to a phoneme recognizer. Then the resulting phoneme string is transformed into a grapheme string using an automatic phoneme-to-grapheme converter. The new grapheme string can replace the originally recognized word or – as confidence scores aren't perfect – both can be put in parallel in the transcriptions.

The paper is structured as follows. In section 2, the recognition system for large vocabulary word recognition and for phoneme recognition is described. Then in section 3 we review the phoneme-to-grapheme converter and its training. Next we introduce the database used in the experiments in section 4. The experimental results are given and discussed in section 5 and finally section 6 draws some conclusions from the proposed research and gives directions for future research.

Research funded by IWT in the STWW program, project ATraNoS (Automatic Transcription and Normalization of Speech), home page: http://atranos.esat.kuleuven.ac.be

2. RECOGNITION SYSTEM

For the experiments described in this paper, the speaker independent large vocabulary continuous speech recognition system is used that is developed at the ESAT-PSI speech group at the K.U.Leuven. An overview of the acoustic modeling can be found in [1], the search module is described in [2].

In the proposed automatic transcription system, which was developed for Dutch, the recognizer was used in two modes: for *word* recognition and for *phoneme* recognition. In both modes, the same single pass time synchronous beam search algorithm was used (not a two pass strategy with graph re-scoring as often used in current large vocabulary recognizers).

2.1. Word Recognition

The lexicon for the large vocabulary word recognition task consisted of the 40k most frequent words in newspaper texts for which a phonetic transcription was available in a pronunciation dictionary (this excludes proper names). With this lexicon, a 3.5% OOV rate was found on a test set. This is rather high due to the existence of word compounding in Dutch. In parallel research we are investigating solutions to this problem, but they are not yet incorporated in the system used in this paper.

The trigram language model we used, was trained on newspaper texts and results in a test set perplexity of 128.9.

The cross word context dependent acoustic modeling is based on a phoneme set with 38 three state phonemes and one noise state. A global phonetic decision tree defines 575 tied states, which are modeled with in total 10k tied gaussians. These numbers are rather small due to the size of the acoustic training database for Dutch, namely 6 hours of speech.

With the above lexicon and modeling, a 14.7% WER was found on the test set. This is higher than the typical error rate for speaker independent large vocabulary recognition due to the small acoustic models and the high OOV rate. For comparison, with a similar type of acoustic modeling we achieved a 7.3% WER on the well known Wall Street Journal (WSJ) recognition task for the November 92 evaluation test set (trigram, 20k word vocabulary, 1.9% OOV rate, 69 hours of acoustic training data).

2.2. Phoneme Recognition

For the phoneme recognizer, the *lexicon* consists of the 38 phonemes. The acoustic modeling is the same as for the word recognizer.

As statistical phoneme sequence model, a 5-gram was used. This way, phoneme sequences that are typical for Dutch are preferred in the phoneme recognizer. A training database for this 5-gram was developed through a forced alignment of the 6-hour acoustic database, allowing for multiple pronunciations per word (as given by the pronunciation dictionary) and for intra word and cross word assimilation rules.

The phoneme recognizer achieves a 25.6% phoneme error rate (sum of insertions, deletions and substitutions) on a separate test set.

3. PHONEME-TO-GRAPHEME CONVERTER

To carry out the task of phoneme-to-grapheme (P2G) conversion, we used TIMBL, a memory-based machine learner (for details on TIMBL, see [3]). Memory-based learning (MBL) is based on the hypothesis that in domains like language processing, where relatively few regularities compete with many sub-regularities and exceptions, a *lazy* form of learning (keeping in memory all examples and using similarity-based reasoning on all examples at classification time) is superior to an *eager* learning approach (extracting rules or other abstractions from the examples and using these to handle new cases, see [4] for some evidence). Furthermore, the results of research on a similar task (grapheme-to-phoneme conversion, see [5]), suggest that MBL may be very well suited for our task, P2G conversion.

TIMBL is a software package for MBL implementing a wide range of algorithms, weighting metrics, and other parameters. It can take as input patterns (or instances) of feature values with a corresponding class symbol (supervised, example-based learning). The feature values of the instances in P2G conversion consist of a phoneme in focus with a certain amount of context, i.e. the preceding or following phonemes of the phoneme in focus. The class symbol of these instances is the graphemic representation of the phoneme in focus position.

During the learning phase, TIMBL stores all instances from a training set in memory and collects statistical data about these instances. The class of new instances is extrapolated from the class of the most similar instance(s) (the so called *nearest neighbor(s)*) from the training set, given an operationalization of similarity. We will describe here only the algorithms which we used in our experiments, for a full description of the implementation of all available algorithms and metrics, we refer to [3].

The basic similarity between two instances is computed using an *overlap metric*. In the case of our symbolic, nominal data (phonemes as features), this means that similarity between two patterns is the number of features for which the two patterns have the same value. Obviously, this would in general give bad results as not all features are equally relevant for solving a particular task. We use an information-theoretic approach (*information gain* in its form normalized for number of values per feature; i.e. *gain ratio*, see [6]) to weigh the relevance of the different features. We will call this algorithm IB1-IG, introduced in [7]. Another factor of importance in MBL is the number of nearest neighbors that is taken into account to extrapolate from (the parameter k). Finally, we have used in our experiments the IGTREE algorithm (see [8]), a decision tree-based heuristic approximation of MBL which is more efficient than IB1-IG.

4. DATABASE

The 15 hour acoustic database on which the training of the P2G converter is based, is a part of the currently developed 10 million word Spoken Dutch Corpus [1]. It consists of data from about 60 speakers, reading aloud books. Note that this is a different database, with other speakers, than the 6 hour database used to train the acoustic models and the 5-gram statistical phoneme sequence model.

The training database for the P2G converter was constructed as follows. First phoneme recognition was executed on the acoustic data, producing a phoneme string for each sentence in the database.

[1] Home page: http://lands.let.kun.nl/cgn/ehome.htm

Parameter settings of TIMBL	Accuracy (% ± SD) at:	
	word level	grapheme level
IGTREE	46.3 (± 2.0)	76.4 (± 1.2)
IB1-IG, k=1	46.4 (± 2.0)	76.2 (± 1.3)
IB1-IG, k=3	46.5 (± 2.0)	77.3 (± 1.3)
IB1-IG, k=5	46.5 (± 2.0)	77.4 (± 1.4)

Table 1. Results with different settings in TIMBL.

Next, so called *compound graphemes* are used to shorten the grapheme strings for some typical Dutch grapheme sequences that represent only one phoneme. For example, in the Dutch word *slaap* (sleep), with pronunciation /slap/, we replaced the grapheme sequence *aa* with the *compound grapheme A*.

Finally, for each sentence, the (shortened) grapheme string is aligned with the corresponding phoneme string. To do this, the *Dynamic Programming* algorithm [9] was used, allowing for null symbols in both the grapheme and the phoneme string. This results in total in 129k sample conversions from phoneme string to grapheme string at the word level, and in about 600k phoneme-grapheme pairs.

The training database for the P2G converter then consists of these phoneme-grapheme pairs. The input for the converter is the phoneme and its three-phoneme context. The output is the corresponding grapheme.

We want to stress that graphemes aligned with a null symbol (representing phoneme deletions by the phoneme recognizer), were removed from the training database, because these pairs are not useful as training material: we will not know for previously unseen data where these null symbols (deletions) may occur.

5. EXPERIMENTS

5.1. Evaluation of TIMBL on P2G conversion

With the training database, we conducted some experiments using TIMBL at different parameter settings (IB1-IG with k set at 1, 3 or 5, or IGTREE). The performance of TIMBL at each parameter setting was obtained by doing *ten-fold cross-validation* (10 CV): the database was partitioned into 10 pieces, and every part was used as a test set, while the remaining 9 parts served as training set, after which averages over the ten test sets were computed. The results at word level (percentage of words for which all graphemes were correctly predicted) and at grapheme level (percentage of correctly predicted (compound) graphemes) are presented in Table 1. The best scoring algorithm is IB1-IG, with k set at 3 or 5.

A major source of errors are ambiguous phonemes: in Dutch, some phonemes can be spelled in different ways, and TIMBL lacks context phonemes or other (syntactic, semantic, ...) information to resolve the ambiguity. A typical example is the Dutch verb *worden* (to become): the first and third person singular of the present tense are both pronounced /wOrt/, but written differently - *word* in the first person, and *wordt* in the third person. Without morphological or syntactic cues, the correct spelling cannot be identified.

When looking at the accuracy of each grapheme in particular, we noticed that TIMBL has more difficulties converting a phoneme to a *compound grapheme* than to a normal grapheme. It is, however, not possible to abandon the concept of *compound graphemes*: this would lead to more graphemes being aligned with a null symbol, and thus less training material as these graphemes are removed from the training database.

5.2. Evaluation within a speech recognizer system

On the 3.6k word test set used in section 2 to evaluate the recognition system, we find a 55.2% accuracy at the word level for the P2G converter. Unfortunately this number is only the average accuracy over all test set words.

On the 3.5% OOV words, a word level accuracy of only 7.9% is found: the OOV words are often long words, or a-typical for Dutch. The word level accuracy on the recognition errors (including the OOV words) is 19.2%, one of the reasons for this low accuracy is that difficulties in the acoustic data (for instance a bad pronunciation for a word) will result in errors in both the word recognizer and the phoneme recognizer.

From [10], we know that (for a recognition task with a WER as in our experiments) the threshold in the confidence measure can be adjusted so that about 75% of the recognition errors are tagged as *uncertain word* (thus missing 25% of the errors), while tagging (wrongly) only 10% of the correctly recognized words.

If we suppose that the 75% tagged recognition errors will be converted with the 19.2% accuracy average for recognition errors, and the 10% tagged correct words with the accuracy average for correct words (which is 59.9%), then transcriptions in which all tagged words are re-written by the P2G converter will result in a slightly higher WER: about 16.0% instead of the 14.7% mentioned in section 2.

But this does not mean that the resulting transcriptions are less readable than the original ones. Albeit only 19.2% of the wrongly recognized words is transcribed correctly by the P2G converter, 41.0% is transcribed with at most 1 error (counting each substituted, inserted or deleted letter as an error), and 62.6% is transcribed with at most 2 errors.

A lot of the words with only few errors can be understood by a reader. Moreover the transcribed words often do not exist in Dutch, giving the reader a clear lead that that word is uncertain (the original transcription is a concatenation of existing words, known by the recognizer).

As examples we give the longest words that are wrongly recognized by the word recognizer. They are compounds, and OOV words for the recognizer. The transcription by the word recognizer and by the P2G converter is given.

programmaproducent	→ programma producent (word rec.)
(program producer)	→ programaprodusent (P2G)
gespreksonderwerp	→ gesprek zonder werk (word rec.)
(topic of conversation)	→ gespreksonberwerp (P2G)
speelgoedmitrailleur	→ speelgoed moet hier (word rec.)
(toy machine gun)	→ spergoetnietrijer (P2G)

As for the word recognizer, only the first result can be read easily. The second result means *conversation without work*, the third *toys must here*. Although *onderwerp* is in the vocabulary of the recognizer (*mitrailleur* is not), it wasn't found due to the linking phoneme *s*.

As for the P2G converter, no one will doubt about the meaning of the first and second result (although with 2 and 1 errors respectively).

The third result, with 9 errors, reflects the problem with loan words, which in Dutch generally preserve their original (a-typical for Dutch) spelling and pronunciation. Given the training databases used, both the phoneme recognizer and the P2G converter produce strings that are typical for Dutch.

6. CONCLUSIONS

In this paper, a combination of phoneme recognition and P2G conversion was used to transcribe uncertain words in the text output of large vocabulary recognizers.

The preliminary results of this system are quite promising. Although at word level only a rather low percentage of the words can be transcribed fully correct, this accuracy may still be useful because many of the orthographically transcribed words can be understood easily by a reader.

Several improvements to the system are still possible. For instance, the output of the P2G converter can be sent to a spelling checker with a large vocabulary as a post-processing module. As an analysis of the transcription errors indicates, this can improve the system considerably: some sounds in Dutch can be written in two ways and the context in the P2G converter is often too short to make the right decision.

Also, both the 5-gram statistical phoneme sequence model in the phoneme recognizer and the P2G converter are trained on Dutch in general, not specifically on OOV words. It may be better to train on OOV words only, as the properties of OOV words (typically loan words or proper names) may differ from the properties of Dutch in general.

Another direction for further research is the use of a more sophisticated description of the phoneme recognizer result. At this moment, the input for the P2G converter consists of only one phoneme string for a word. This means an important loss of information which may be useful for the converter. The use of a phoneme graph, possibly including probabilities for the phonemes, could be a solution to this problem.

7. ACKNOWLEDGEMENT

The authors would like to thank Erik Tjong Kim Sang and Veronique Hoste from the CNTS research group for their support during the development of the proposed system.

8. REFERENCES

[1] J. Duchateau, K. Demuynck, D. Van Compernolle, and P. Wambacq, "Fast and accurate acoustic modelling with semi-continuous HMM," *Speech Comm.*, vol. 24, no. 1, pp. 5–17, Apr. 1998.

[2] K. Demuynck, J. Duchateau, D. Van Compernolle, and P. Wambacq, "An efficient search space representation for large vocabulary continuous speech recognition," *Speech Comm.*, vol. 30, no. 1, pp. 37–53, Jan. 2000.

[3] W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch, "TiMBL: Tilburg memory based learner, version 3.0, reference guide," ILK Technical Report 00-01, Tilburg University, 2000, Available from `http://ilk.kub.nl`.

[4] W. Daelemans, A. van den Bosch, and J. Zavrel, "Forgetting exceptions is harmful in language learning," *Machine Learning, Special issue on Natural Language Learning*, vol. 34, pp. 11–41, 1999.

[5] W. Daelemans and A. Van den Bosch, "Language-independent data-oriented grapheme-to-phoneme conversion," in *Progress in Speech Processing*, J. P. H. Van Santen, R. W. Sproat, J. P. Olive, and J. Hirschberg, Eds., pp. 77–89. Springer-Verlag, Berlin, 1996.

[6] J. R. Quinlan, C4.5*: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, 1993.

[7] W. Daelemans and A. van den Bosch, "A neural network for hyphenation," in *Artificial Neural Networks 2: proceedings of the International Conference on Artificial Neural Networks*, I. Aleksander and J. Taylor, Eds., Amsterdam, 1992, pp. 1647–1650, Elsevier.

[8] W. Daelemans, A. van den Bosch, and A. Weijters, "IGTree: using trees for compression and classification in lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, pp. 407–423, 1997.

[9] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *Journal of the Association for Computing Machinery*, vol. 21, no. 1, pp. 168–173, 1974.

[10] T. Kemp and T. Schaaf, "Estimating confidence using word lattices," in *Proceedings of EuroSpeech*, Rhodes, Greece, Sept. 1997, vol. II, pp. 827–830.