ASR IN A HUMAN WORD RECOGNITION MODEL: GENERATING PHONEMIC INPUT FOR SHORTLIST

Odette Scharenborg, Lou Boves and Johan de Veth

A^{2}RT, Department of Language and Speech
University of Nijmegen, The Netherlands
{O.Scharenborg,L.Boves,J.deVeth}@let.kun.nl

ABSTRACT

The current version of the psycholinguistic model of human word recognition Shortlist suffers from two unrealistic constraints. First, the input of Shortlist must consist of a single string of phoneme symbols. Second, the current version of the search in Shortlist makes it difficult to deal with insertions and deletions in the input phoneme string. This research attempts to fully automatically derive a phoneme string from the acoustic signal that is as close as possible to the number of phonemes in the lexical representation of the word.

We optimised an Automatic Phone Recogniser (APR) using two approaches, viz. varying the value of the mismatch parameter and optimising the APR output strings on the output of Shortlist. The approaches show that it will be very difficult to satisfy the input requirements of the present version of Shortlist with a phoneme string generated by an APR.

1. INTRODUCTION

'Speech recognition' is investigated by automatic speech recognition (ASR), which studies the recognition of speech by computers, and psycholinguistics, which studies human speech recognition (HSR). Although the two fields are related, their aims are different. The central issue in ASR is minimising the number of recognition errors, whereas HSR aims at building models that simulate and explain human speech recognition. Although both ASR and HSR intend to investigate the entire process from the acoustic signal to the recognised units, models of HSR tend to cover only parts of the human speech recognition process, unlike the end-to-end-systems in ASR. An integral model covering all stages of the human speech recognition process does not yet exist. For instance, virtually all psycholinguistic models start from a discrete segmental representation instead of the acoustic signal. Furthermore, in building partial models little attention is paid to the feasibility of interfacing modules with models of other stages [1: pp.145-150].

Despite the gap that separates ASR from HSR, there is a growing interest in possible cross-fertilisation [1: pp.145-150, pp.49-54]. From the viewpoint of ASR, there is some hope of improving performance by incorporating essential knowledge about HSR into current ASR systems [1]. Researchers of human word recognition hope to deploy ASR approaches to integrate partial modules into a convincing end-to-end model [1]. The research programme introduced in this paper attempts to look both ways: by trying to build an end-to-end HSR model, we hope to identify pieces of HSR knowledge that can be used to improve ASR performance.

The psycholinguistic model used in this study is Shortlist [2]. As a (partial) model of HSR, Shortlist accounts for a wide range of results from psycholinguistic experiments related to word recognition. Shortlist is a two-stage process. In the first stage, an exhaustive lexical search yields a shortlist of (typically) maximally 30 word candidates that are roughly consistent with the phonemic input. This search is repeated from scratch for each phoneme that is detected in the input. The activation of words in the shortlist is determined by their degree of fit with the phonemic input. If a phoneme in a word matches the input, the word activation is increased by 1; for each mismatching phoneme the word activation is reduced. The amount of reduction is controlled by the mismatch parameter. In the second stage, the activated words in the shortlists compete with each other by means of their initial activation and the inhibition of other words in the list. When all phonemes in the input are processed, the word with the highest activation is selected.

Despite its explanatory power the current version of Shortlist requires substantial modifications to turn it into a comprehensive account of HSR. First, the input of the current implementation must consist of a single string of discrete phoneme symbols. Although subjects can perform phoneme detection tasks, few -if any- scientists believe that this implies that speech signals can be segmented into a sequence of discrete sounds. Moreover, the current implementation of the search in Shortlist makes it difficult to deal with insertions and deletions in the input phoneme string. As a result, the present implementation gives a large premium to inputs with the ' correct' number of symbols. This is not realistic, since in spontaneous speech many insertions, substitutions, and deletions occur. In the remainder of this paper, this problem will be referred to as the 'insertion and deletion problem'. Although the limitation to a single phoneme string comprising the correct number of symbols can be overcome by changing the implementation of the search, we think that it is useful -also for ASR- to investigate the extent to which an operational front-end that converts acoustic signals into a discrete segmental representation can fulfil the requirements of a single string with the ' correct' number of elements.

Words that are highly similar to the input phoneme string will be more likely to make it to the shortlist than words that only match at the cost of several substitutions, deletions, or insertions. Moreover, it has been easier to derive substitution costs from acoustic/phonetic distances between phonemes than to define insertion and deletion costs. Therefore, it is important to try and derive an input phoneme string from the acoustic signal that is as close as possible to the number of phonemes in the lexical representation of the spoken words.

The research described in this paper is an attempt to investigate how far ASR can go in terms of decoding acoustic signals into a string of discrete phoneme symbols that meet the input requirements of Shortlist, and at the same time see to what extent the present implementation of Shortlist can adapt to an input afflicted by insertions and deletions. Obviously, this requires that we develop an Automatic Phone Recogniser (APR). In optimising this APR for Shortlist, we have to pay special attention to the insertion and deletion problem. We have considered two approaches:

- The value of the mismatch parameter of the Shortlist model is varied: 'Mismatch parameter'. This approach investigates the room offered by the present implementation of Shortlist to handle 'errors' in its input.
- The APR is optimised on the output of Shortlist. The conventional way to optimise an APR is to make its output as similar as possible to some reference transcription of the test corpus, but that is not necessarily the best possible input for Shortlist. The optimisation criterion that we used in this approach is the percentage of utterances for which the correct word was present in the shortlists: 'Optimisation on the output of Shortlist'.

The goal of the exercise is to obtain a better understanding of the current mismatch between what ASR can provide and what Shortlist needs in terms of interface to acoustic speech signals. As a result, we expect to be in a better position to determine the most promising ways towards a remedy of that mismatch.

In the next section, the material used for the experiments and the design of the experiments will be described. Section 3 will present and discuss the results obtained with the experiments. Finally, in Section 4, we draw some conclusions.

2. METHOD

2.1. Material

2.1.1. Corpora

For training and testing the APR, data from the Dutch Directory Assistance Corpus (DDAC2000) [3] are used. The corpus is recorded over the telephone and consists of Dutch city names pronounced by unknown speakers. The material to train the acoustic models of the APR comprises 24,559 utterances. Each utterance consists of a Dutch city name or 'ik weet het niet' ('I don't know') pronounced in isolation. The reason for this strict selection is that we wanted to obtain the cleanest possible phone models with a training corpus for which only unique canonical transcriptions of the words are available. Including longer utterances in the training corpus would probably yield more contaminated models. Moreover, the models obtained with this corpus are optimally matched to the requirements of the test.

The independent test set consists of 10,510 utterances with a total number of 11,523 words. These utterances may also contain disfluencies and connected speech responses like 'haarlem noordholland' (i.e., a city name plus the name of a province).

2.1.2. Phone models, lexicon, and language model

For the APR, 36 context independent phone models, 1 silence model, 1 model for filled pauses, and 1 noise model were trained. Each phone model consists of 3 pairs of 2 identical states, one of which can be skipped. The maximum number of Gaussians trained per state is 128. The models were trained on the 'optimised transcriptions' of the training material (cf. Section 2.1.3) using the training procedure described in [4].

The lexicon of the APR only contains the 36 trained phones and three additional models for garbage, filled pauses, and silence. Uni- and bigram language models (LMs) for the APR were trained on the optimised phonemic transcriptions of the training material. The 'words' in these LMs are in fact phones. The LM was only used in the experiments with the APR.

2.1.3. The Shortlist lexicon and test corpus transcriptions

When people speak they often do not adhere to the standard dictionary pronunciation. In spontaneous speech many substitutions, deletions and insertions occur [4]. Consequently, the number of actually produced phonemes may differ from the number of phonemes in the standard transcription. The psycholinguistic theory underlying Shortlist does not make claims about the manner in which humans cope with pronunciation variation. Specifically, there is nothing in the theory that promotes the exclusive use of citation forms in the mental lexicon. To avoid unnecessary problems due to pronunciation variation we decided to add frequent variants to the Shortlist lexicon.

For the generation of these new forms, we used the *Weka* decision tree (d-tree) tools [5] and the bottom-up d-tree approach described in [4]. In building the d-trees, only the left and right neighbour of the phones were used as input to the d-tree algorithm. The d-trees were used to generate multiple potential pronunciation variants for all words in the training and test corpus. An ASR trained on the canonical transcription of the training corpus, running in forced recognition mode, was then used to find the most likely variant for each word in the training and test corpus. The phonemic transcriptions of the two corpora were then updated by replacing the canonical transcriptions by the 'best matching' variant. Finally, new phone models in the APR were trained using the updated ('optimised') transcriptions of the training corpus.

The d-trees attach a score to all pronunciation variants they produce, which can be considered as a prior probability. This allows us to select the most likely pronunciation variants to add to the Shortlist lexicon. To this end, all variants produced by the d-trees were ranked according to their prior probability. A threshold was then determined such that adding all variants with a higher probability to the Shortlist lexicon resulted in an average of 2.5 variants per word. The number of 2.5 variants/word was chosen because independent ASR experiments (cf. [4]) have shown that a higher number of variants will cause a deterioration of the recognition performance, unless the prior probabilities of the variants can be used in the recognition process (which Shortlist does not).

2.2. Experiments

We have used two measures in our experiments, viz. phone error rate (PER) and the proportion of utterances for which the correct word was in the shortlist. PER, in its turn, is evaluated by comparing the output of the APR against two reference transcriptions 1) the original phonemic transcriptions and 2) the optimised transcriptions in the test corpus. Phone Error Rate (PER) is defined as:

$$PER = (S + I + D)\big/ N \qquad (1)$$

with *S, I, D,* and *N* the number of substitutions, insertions, deletions, and total number of phones, respectively.

The results of the 'mismatch parameter' experiments can only be evaluated through the operation of Shortlist, in terms of the percentage of utterances for which the correct answer is present in the shortlist. In addition, we record and report the proportion of utterances for which the correct word had the highest activation at the very output of Shortlist.

2.2.1. *Mismatch parameter*

Our first attempt to loosen the input constraint of Shortlist focuses on the implementation of Shortlist itself, namely on the mismatch parameter. The reasoning behind this is that when the deactivation of a word by the mismatch parameter is reduced, mismatching phonemes will be penalised less severely, resulting in a better ability of Shortlist to recognise input strings containing insertion and deletion errors.

In a series of experiments, the value of the mismatch parameter was reduced from its default value 3.0 to 0.0 in steps of 0.5. As input, the output strings of the APR were used.

2.2.2. *Optimisation of the APR on the output of Shortlist*

The second approach to reduce the insertion and deletion problem of Shortlist is to optimise the APR on the output of Shortlist. In this way, the criteria an input string of Shortlist must comply with to be recognised correctly are optimally met by the output strings of the APR.

A straightforward way to control the number of phonemes and their identity in the output of the APR is through the word insertion penalty (WP) and the language model factor (LMF), respectively. WP determines the trade-off between insertions and deletions: when WP increases, fewer phonemes will be inserted. LMF determines the weighting of the acoustic and the language model contribution to the total probability of a phoneme: by increasing LMF, the contribution of the LM probability to the total probability of the phoneme is increased.

In this experiment, WP and LMF were simultaneously optimised. The values of LMF and WP that produced the lowest error rate for the test set were regarded as optimal. The cost function used for the optimisation process was the proportion of utterances for which the correct word was not present in the shortlists created after the last input phoneme was presented.

3. RESULTS AND DISCUSSION

Table 1 shows the effect of decreasing the value of the mismatch parameter from the default value 3.0 to 0.0. The effect is measured in terms of the percentage of utterances for which the correct word was present in the shortlist and the percentage of utterances within this set for which the correct word was not recognised ('Not rec.'). Furthermore, the percentage of utterances for which the correct word was recognised is shown. A word is recognised if it has the highest activation.

As can be seen in Table 1, lowering the value of the mismatch parameter increases both the percentage of utterances for which the correct word was present in the shortlist, as well as the percentage of correctly recognised utterances. These results confirm our intuition that reducing the value of the mismatch parameter results in a better ability of Shortlist to deal with input strings containing insertion and deletion errors. Words containing insertion and deletion errors are penalised less severely and therefore recognised more often.

Table 1: Effect of decreasing the value of the mismatch parameter.

Mismatch	In shortlist		Recognised
(M)	%	Not rec. (%)	(%)
3.0	42.3	33.9	27.9
2.5	43.1	33.1	28.9
2.0	48.6	35.6	31.3
1.5	51.4	35.4	33.2
1.0	58.5	40.4	34.9
0.5	66.9	45. 6	36.4
0.0	76.5	46.0	41.3

The column 'Not rec' shows that when M=3.0, 33.9% of the utterances for which the correct word was present in the Shortlist were not correctly recognised. This percentage increases to 46.0% when M is reduced to 0.0. Thus, it appears that the proportion of the utterances for which the correct word is included in the shortlist grows faster than the proportion of the utterances that are correctly recognised. Apparently, the competition stage is not able to deal with too liberal an input.

An analysis of the output showed that only in rare cases does the decrease of the mismatch parameter cause a word that was recognised correctly at a higher value to fail. This is contrary to what has been observed in many ASR experiments in which the number of different words allowed into the search is increased. In those experiments, it is almost invariably observed that adding words/variants causes many changes in the final output, but a substantial proportion of these changes appear to be new errors, which trade off the number of improvements. We expect that an in-depth analysis of the way Shortlist profits from relaxing its input constraints can show directions for improving ASR.

Table 2 shows the results for the 'optimisation on the output of Shortlist' experiments at the level of the APR. To show the effect of improving the transcriptions of the test material, the performance of the APR relative to 1) the canonical phonemic transcriptions of the test corpus ('Tr_{Orig}'), and 2) the optimised transcriptions of the test material ('Tr_{Opt}') are presented. The column 'Av. #ph/utt' shows the average number of phonemes per utterance in the output of the APR. Finally, the number of insertions, deletions, and substitutions when the input strings are compared to the optimised transcriptions of the test material are shown. The 'B(aseline)' row shows the performance of the baseline system. The 'O(ptimised)' rows show the performance of the system when the output of the APR is optimised on the output of Shortlist. The value of the mismatch parameter in these experiments was 3.0 and 0.0, i.e., the two extreme values used in the experiment described above.

Table 2: Effect of the 'optimisation on the output of Shortlist' evaluated at the level of the APR.

APR Output	PER Tr_{Orig}	PER Tr_{Opt}	Av. #ph/utt	#Ins	#Dels	#Subs
B	38.28	34.42	7.35	3,657	9,030	16,097
O: M=3.0	40.83	37.34	8.26	8,744	4,657	17,817
O: M=0.0	56.10	53.43	9.69	21,922	2,434	20,317

As can be seen in Table 2, it is clear that the PER for the optimised transcriptions is lower, but this result must be interpreted with some caution. After all, the optimised transcriptions have been generated with essentially the same acoustic models as used in the APR.

With respect to the PER for the phoneme strings optimised on the output of Shortlist, Table 2 shows an increase in the error rate relative to a 'straightforward' optimisation on the acoustic signal per se. This suggests that the present implementation of Shortlist prefers a phonemic input that is not necessarily an optimal representation of the acoustic speech signal (but rather some representation that is closer to a symbolic representation of the words that happens to be present in the lexicon). The deterioration is especially large for M=0.0. The increases in PER are due to an increase in the average number of phonemes per utterance (cf. 'Av. #ph/utt'). As can be seen in Table 2, the number of substitutions remains about the same when the number of phonemes per utterance increases. As expected, the number of deletions decreases strongly; however, the increase in insertions is larger, leading to a higher PER.

Table 3 shows the results for the 'optimisation on the output of Shortlist' experiments at the level of Shortlist. The 'In shortlist' columns show the absolute and relative number of utterances for which the correct word was present in the shortlist, and the percentage of these utterances for which the correct word did not have the highest activation. The columns 'Recognised' show the absolute and relative number of correctly recognised utterances.

Table 3: Effect of the 'optimisation on the output of Shortlist' evaluated at the level of Shortlist.

APR Output	In shortlist			Recognised	
	Abs.	%	Not rec. (%)	Abs.	%
B: M=3.0	4,442	42.3	33.9	2,937	27.9
O: M=3.0	4,720	44.9	9.4	4,278	40.7
B: M=0.0	8,043	76.5	46.0	4,343	41.3
O: M=0.0	8,885	84.5	65.5	3,063	29.1

Although the performance of the optimised phoneme strings deteriorates in terms of PER, the performance at the level of Shortlist is better than the baseline output. As can be seen in the 'In shortlist' columns, more phonemes per utterance lead to more utterances for which the correct word was present in the shortlist. This corroborates the suggestion that Shortlist prefers input that matches forms in its internal lexicon, rather than a precise account of the acoustic signal.

With M=3.0 (the default value of the mismatch parameter) optimising the APR so as to maximise the proportion of utterances for which the correct word appears in the shortlist leads to a small improvement in terms of the 'In shortlist' measure. However, the improvement in terms of the proportion of utterances that are correctly recognised is much larger. This is due to the fact that only in 9.4% of the utterances the correct word in the shortlist does not end up with the highest activation. For M=0.0 the situation is completely different. With the APR output optimised on the acoustic signal the number of utterances for which the correct word enters the shortlist is almost twice as large as for M=3.0. However, the proportion of utterances in which the correct word also has the highest final activation decreases dramatically. The final result is that the recognition performance in the B: M=0.0 condition is almost the same as in the O: M=3.0 condition. Optimising the APR to maximise the proportion of utterances with the correct word in the shortlist with M=0.0 makes things even worse: the cost function does indeed improve somewhat, but the final recognition performance deteriorates enormously. This shows again that the competition process in Shortlist does not handle input strings with many insertions too well. Analysis of the errors suggests that Shortlist prefers long words, which consume a large number of input phonemes, at the cost of shorter words in the presence of insertions.

4. CONCLUSIONS

The 'mismatch parameter' experiments have shown that the search process that selects the words to enter the shortlist must trade the proportion of utterances for which the correct word enters the shortlist against the capability of the competition stage to guarantee that the correct words ends up with the highest activation. The search in the present implementation of Shortlist is probably not optimal. Therefore, we will investigate the possibility of adapting the first stage of Shortlist in such a way that insertions and deletions are handled more appropriately, without producing a shortlist that is difficult to handle for the competition stage.

Optimising the phoneme string on the output of Shortlist increases the number of utterances for which the correct word was present in the shortlist. Comparing the results of the conditions where the value of the mismatch parameter is 0.0 and 3.0, we found that the increase in number of utterances is larger in the case of 0.0 but at the cost of a dramatic increase in PER and decrease in number of utterances where the correct word has the highest activation.

In conclusion, the approaches we considered in this study show that the gap between what ASR can provide in terms of a phonemic decoding of speech signals, and the requirements that models such as Shortlist make for their discrete symbolic input is quite substantial. However, we are confident that we can narrow the gap by simultaneously adapting the search in Shortlist and the performance of the APR.

5. ACKNOWLEDGEMENT

For converting the d-trees into FSGs, software developed at the International Computer Science Institute (ICSI), Berkeley, USA, was used.

6. REFERENCES

[1] Smits, R., Kingston, J., Nearey, T., Zondervan, R. (Eds), *Proceedings Speech Recognition as Pattern Classification Workshop*, Nijmegen, 2001.

[2] Norris, D., "Shortlist: a connectionist model of continuous speech recognition", *Cognition 52*, 189-234, 1994.

[3] Sturm, J., Kamperman, H., Boves, L., den Os, E., "Impact of speaking style and speaking task on acoustic models", *Proceedings ICSLP*, pp. 361-364, 2000.

[4] Wester, M., "Pronunciation variation modelling for Dutch automatic speech recognition", Ph.D. thesis, University of Nijmegen, the Netherlands, 2002.

[5] Weka-3 Machine Learning software in Java, http://www.cs.waikato.ac.nz/ml/weka/index.html.