USING CONTEXT TO CORRECT PHONE RECOGNITION ERRORS

Stephen Cox

School of Computing Sciences, University of East Anglia, Norwich NR4 7TJ, U.K.

sjc@cmp.uea.ac.uk

Abstract

There are many circumstances in which it is useful or necessary to recognise phones rather than words, but phone recognition is inherently less accurate than word recognition. We describe here a post-recognition method for “translating” an errorful phone string output by a speech recogniser into a string that more closely matches the transcription. The technique owes something to Kohonen’s idea of “dynamically expanding context” in that it learns from the errors made by the recogniser in a particular context, but it uses many contexts rather than a single context to estimate the “translation” of a recognised phone. The weights given to the different contexts in estimating the translation are determined discriminatively. On the WSJCAM0 database, the technique gives a 19.2% relative improvement in phone errors (including insertions) over the baseline, compared with a 6.2% improvement obtained using dynamically expanding context.

1. Introduction

Phone recognition is preferred to word recognition in some circumstances, the most usual being when the vocabulary is too large to decode using a network of words or when some (or all) of the spoken words are unknown, or have no transcription. Phone recognition is also useful in developing the acoustic-matching component of a recognition system, for identifying pronunciation errors in the lexicon, and determining when alternative pronunciations need to be added. The accuracy obtained when recognising phones is lower than that obtained recognising words. This is because the constraints on the sequence of decoded units are weaker when these units are phones, since there is no knowledge of how the phones combine to form the words, and word sequences, in the vocabulary. This problem can be mitigated by the use of a phonotactic model. A phonotactic model is an n-gram model that models the probability of sequences of phonemes, and which is incorporated into the decoding stage of the recogniser. A high order phonotactic model (e.g. n = 6) can produce a very significant increase in phone accuracy [1] but at the expense of increased processing at recognition time. Another problem with such models is that their influence in the recogniser needs to be carefully balanced with the influence of the acoustic model, a process that can only be done experimentally.

In this paper, we describe a simple technique for “correcting” the output of a phone recogniser. Like a phonotactic model, it uses contextual information, but unlike the latter, it effectively incorporates a model of the errors made by the recogniser and uses these in the correction process. The structure of the paper is as follows: in section 2, we introduce Kohonen’s idea of “dynamically expanding context” (DEC), which was partly the inspiration for the technique. Section 3 describes the model and section 4 the speech data and recogniser used. Details of experiments and results are presented in section 5 and we end with a discussion in section 6.

2. Dynamically Expanding Context

The idea of dynamically expanding context (DEC) was proposed by Kohonen in 1986 [4, 5]. In the context of phone recognition, it works as follows: we assume that we have the output strings from a phone recogniser in response to a set of input utterances, together with the true phonetic transcriptions of the utterances. Furthermore, we assume that each output string has been aligned to its transcription string, most likely using some form of dynamic programming. Suppose \( x_1 \) is the first example in the recognised data of the phone \( x \). We refer to the corresponding aligned phone sequence in the transcription string, \( Y \), as the “translation” of \( x \). Hence we can make a “translation rule” \( x \rightarrow Y \) (note that because of deletions by the recogniser, \( Y \) may be a sequence of one or more phones rather than a single phone, and so is denoted in capitals). However, when \( x_2 \) is observed in the data, it may map to a different phone sequence \( Y' \), introducing ambiguity into the original translation rule. Suppose that each \( x_i \) has an associated left and right context \( L_i \) and \( R_i \), where \( L_i \) and \( R_i \) are phone sequences of (as yet) undefined length. The original context-free translation rule \( x \rightarrow Y \) becomes a set of \( N \) context dependent rules, \( L(c_k)xR(c_k) \rightarrow Y_k \) \((k = 1, 2, \ldots , N)\), where \( L(c_k) \) and \( R(c_k) \) are phone sequences that are sufficiently long to resolve any ambiguity in the rule set. In practice, the algorithm is implemented by beginning with a zero context translation rule for each phoneme based on the first occurrence of the phoneme. When a phone is found that gives rise to a conflict with an existing rule for a particular phoneme, the context is increased for both the new phone and for the phone (or phones) that gave rise to the existing rule, until no conflict exists (hence the name “dynamically expanding context”). The original rule is thus revised and a new rule is created based around the new phone. The final set of rules is stored.

When it is desired to correct a previously unseen output string, the context around each phone in the string is increased from zero until the string \( L_xR \) matches the left hand side of one of the rules, and the phone is replaced by the phone sequence on the right hand side of the rule. Of course, it is possible that no instances of a particular phoneme in the context that it appears in the recognised string were seen in the training data, and in this case, the recognised phone is left unaltered.

In [4, 5], phone-based speaker dependent recognisers in Finnish and Japanese were used, each with a vocabulary of 1000 words, spoken in isolated word fashion. Each word was spoken and recognised eight times: seven of the recognition strings were used to train the rules and the other string was used for testing. Transcription accuracy of the test strings was raised from 68% to 90% for the Finnish speaker and from 83.1% to
94.3% for the Japanese speaker, with about 70% of the errors being corrected in each case.

### 3. A Probabilistic/Discriminative Approach to the Use of Context

The main drawback of the DEC approach to correction is that it makes no use of the translation probabilities available from the training data—in fact it does not use the concept of probability at all. DEC deletes many of the low context rules (which tend to be in conflict) regardless of how potentially useful they are in a probabilistic sense. A rule may be correct and occur many times, but a single instance of a conflict is enough to change it to rule that operates at a higher context, which inevitably has less application. In this section, we suggest a probabilistic approach to translation.

In [3], Jelinek and Mercer proposed a method of smoothing estimates of trigram probabilities of words. The smoothed estimate of a trigram $Pr(w_i|w_{i-1}, w_{i-2})$ is given as:

$$\tilde{Pr}(w_i|w_{i-1}, w_{i-2}) = \lambda_1 Pr(w_i|w_{i-1}, w_{i-2}) + \lambda_2 Pr(w_i|w_{i-1}) + \lambda_3 Pr(w_i),$$

where the RHS terms are the maximum-likelihood probabilities estimated from “held-out” training data and $\lambda_1 \geq 0$ and $\sum \lambda_i = 1$. The motivation for equation 1 is that when counts of the word sequence $(w_{i-2}, w_{i-1}, w_i)$ are low (inevitable for most sequences, even when a very large training corpus is used), they need to be “smoothed” using counts of lower contexts of word $w_i$. We can apply the same principle to the estimates of the probabilities of different translations of a phone $x$ in different contexts. For compactness, let us represent a particular context of $x_i$ by a single symbol $C_i$, which subsumes $L_i$ and $R_i$. Then the joint occurrence of a phone $x_i$ within a certain context $C_i$, together with its translation $Y_i$, can be represented by the triple $(x_i, C_i, Y_i)$, and the number of counts of such an occurrence as $C(x_i, C_i, Y_i)$. We prefer to smooth the joint counts directly and so our model may be written as

$$\tilde{C}(x_i, C_i, Y_i) = \sum_{k=1}^{M} \lambda_k C(x_i, C_i^k, Y_i)$$

where $C_i^k$ is the $k$’th context of $x_i$ and $\lambda_k \geq 0, \sum \lambda_k = 1$. Note that by dividing the RHS of equation 2 by the total number of different triples $(x_i, C_i, Y_i)$, it becomes the smoothed joint probability of a recognised phone, its context and its translation. However, it is simpler to work in terms of counts. The contexts may be enumerated as the set of all possible pairs of $L$ - $R$ phone contexts between a minimum number of phones, $C_{min}$, and a maximum number, $C_{max}$. ($C_{min}$ and $C_{max}$ are the same for $L$ and $R$ contexts). For instance, using $C_{min} = 1$ and $C_{max} = 2$ gives 4 possible contexts, $C_i = \{l_i x_i r_i\}, C_{i+1} = \{l_{i+1} x_{i+1} r_{i+1}\}, C_{i+2} = \{l_{i+2} x_{i+2} r_{i+2}\}, C_{i+3} = \{l_{i+3} x_{i+3} r_{i+3}\}$, where $l$ and $r$ represent single phones.

In [3], Jelinek and Mercer estimated occurrence probabilities of word trigrams using a maximum likelihood criterion to estimate the weights in their model. In our case, the target translation $Y$ for any $x$ is known, and it therefore possible to use a discriminative criterion to train the weights.

For unseen strings from the recogniser, we will choose the translation for $x$ within a context $C$ that has the highest number of smoothed counts i.e.

$$Y^* = \arg\max_{j} \tilde{C}(x, C, Y_j).$$

Suppose that for a certain recognised phone $x_i$ in context $C_i$, the correct translation is $Y_c$, and there also exist counts of other translations of $x$ in the same context $C_i, C(x_i, C_i, Y_j), j \neq i$. Define

$$d_i = (\tilde{C}(x, C_i, Y_i) - \tilde{C}(x, C_i, Y^*)),$$

where $Y^* = \arg\max_{j \neq i} \tilde{C}(x, C_i, Y_j)$ i.e. $Y^*$ is the translation with the highest number of smoothed counts, not considering the correct translation (in this context) of $Y_i$. Hence $x_i$ is correctly classified if $d_i > 0$. Choosing the $\lambda_i$’s to maximise $\sum d_i$ is not an effective strategy, as it tends to find $\lambda$’s that increase the $d_i$’s that are already positive rather than $\lambda$’s that transform negative $d_i$’s into positive ones. We use the logistic

$$z_i = \frac{2}{1+e^{-\alpha d_i}} - 1 \quad \alpha > 0$$

to transform the $d_i$’s: $z_i \rightarrow 1$ for positive $d_i$, $z_i \rightarrow -1$ for negative $d_i$, and the higher the value of $\alpha$, the sharper the transition. We use a search algorithm that maximises $\sum z_i$ subject to $\lambda_k \geq 0, \sum \lambda_k = 1$ (Sequential Quadratic Programming method). $\alpha$ is set to 0.8 and its value was found to have little influence on the effectiveness of the algorithm.

### 4. Speech Data and Recogniser

We conducted our experiments with a subset of the WSJCAM0 database [6]. The speech data used was parameterised to a 39-d vector consisting of 12 MFCCs + velocity + acceleration coefficients and log energy coefficients. The training-set consisted of 8246 utterances from the si tr portion of WSJCAM0, a total of about 90 hours of speech from 92 speakers. It was used to train a set of 45 HMMs of monophones, each model being a three state, left-right model having a 25 component Gaussian mixture model associated with each state. A phonotactic bigram language model was estimated from the transcriptions of the same data. Using monophone models with a large number of mixture components per state was found to give better phone recognition performance than using triphone models with a smaller number of components. The test-set was the non-adaptation sentences of the si dt set of WSJCAM0, consisting of 773 utterances from 20 different speakers. Phone recognition results for the training- and test-sets are given in Table 1. These results represent a reasonably high baseline to work from and the fact that the train and test set results are very close also suggests that the recogniser is well-trained.

The recognition utterances were then aligned to their transcriptions using a simple form of dynamic programming:

- the distance metric between phones $x_i$ and $x_j$ was

$$D(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j \\ 1 & \text{otherwise} \end{cases}$$

- the “cumulative distance” was computed as

$$U(i, j) = D(x_i, x_j) + \min\{U(i-1, j-1), U(i-1, j), U(i, j-1)\}$$

Table 1: Phone error rates on the training and test sets

<table>
<thead>
<tr>
<th></th>
<th>% Correct</th>
<th>% Accuracy</th>
<th># phones</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training-set</td>
<td>66.80</td>
<td>57.04</td>
<td>601594</td>
</tr>
<tr>
<td>Testing-set</td>
<td>66.43</td>
<td>56.36</td>
<td>54270</td>
</tr>
</tbody>
</table>
It was noted that there were instances where a more sophisticated matching technique might have improved the alignment. “Beginning of sentence” and “End of sentence” markers were also added to give some context to the initial and final phones in an utterance.

5. Experiments and Results

5.1. Dynamically Expanding Context

The dynamically expanding context (DEC) algorithm was run on the training-data and produced a set of 428 000 rules, so on average, each phoneme occurred in about 428 000/45 ≈ 9500 different contexts. Results for the translated test-set strings were: % Correct = 67.36, % Accuracy = 58.80. This is a small improvement on both % Correct and % Accuracy on the baseline (see Table 1). The algorithm found translations for 33631 of the 54270 decoded phones, i.e. for 62% of the phones. Table 2 gives a breakdown of the number of translations made at each context level and also shows the percentage of these translations that were correct. The table shows that the number of translations peaks when the total context is 3 phones, but the translation accuracy increases as context increases. The high context translations are much more accurate than the lower context ones, but occur very infrequently. The small improvement on baseline accuracy obtained contrasts with the much larger number of symbols describing the context. However, accuracy was significantly worse: % Correct = 61.09, % Accuracy = 44.74.

5.2. Discriminative Approach

The training-set (recognised utterances aligned to transcriptions) was used to record counts of triples \(C(x_i, C_i, Y_i)\) with a minimum context \(C^\text{min} = 1\) and a maximum context \(C^\text{max} = 3\), which gives 9 possible different contexts for a phone. This gave \(3280000\) entries with an average of 1.54 counts/entry.

A preliminary experiment was run on the test-set in which the \(\lambda_i\)'s of equation 2 were set equal to 1/9 rather than being estimated. The result was % Correct = 69.69, % Accuracy = 64.79, a significant improvement on both the baseline (66.43/56.36) and on the DEC result (67.36/59.08). The improvement in the accuracy is striking, with the number of insertions being nearly halved when compared with the baseline at the cost of about 5% more deletions.

When the \(\lambda_i\)'s were estimated using a search algorithm that maximised \(\sum_i z_i\) (equations 5 and 4) subject to \(\lambda_k \geq 0\) and \(\sum_k \lambda_k = 1\), the result was % Correct = 70.61, % Accuracy = 64.72, which is a 1% increase in % Correct over the unweighted result at the same accuracy. Table 3 shows the value of the weights estimated for each context together with the number of matches to each context in the test-set data. Table 3 shows that the only contexts that have non-zero weight, presumably because this highly-occurring low context provides little accurate classification information. It is interesting that all the higher contexts (> 4 phones) have a zero weight, probably because they occur infrequently and their probabilities are poorly estimated. Because of this result, it was not attempted to use contexts higher than 3 in any subsequent experiments.

Table 4 gives a summary of the complete results for the baseline and the three experiments described. The statistical significance of the test-set results was evaluated using McNe-
mar’s test [2]. To compare algorithms directly, each phone in
each transcription is labelled as having been decoded either cor-
rectly or incorrectly by an algorithm, which means that inser-
tions are ignored. Hence significances were computed only for
results in column three of Table 4. All results in this column are
significantly different from each other at \( p \)-values < \( 1 \times 10^{-4} \).

6. Discussion

We have proposed a technique for correction of phonetic strings
which is based on estimation of the correct transcription phone
given a decoded phone occurring within a context. The tech-
nique was inspired by the “dynamically expanding context” (DEC)
algorithm of Kohonen but uses a weighted combination of
different levels of context rather than the highest context re-
quired to differentiate strings, as in the original algorithm. Giving
equal weight to all contexts increased % Correct by 3.2%
(absolute) over baseline and % Accurate by 5.7%. By estimat-
ing the weights using a discriminative criterion, % Correct was
increased by a further 1% with no significant change in accu-
racy, which is in effect a 19.2% improvement in relative error
over the baseline. So the algorithm achieves a small but signifi-
cant increase in performance at very little computational cost
(“translation” of a phone consists of hash table lookups and
weightings and runs very quickly) and storage cost that is prob-
ably small compared with the storage required by the recogniser
(storage of the contextual string hash table can be minimised by
an appropriate tree structure).

It could be argued that it would be more effective to use
a higher order n-gram language model during decoding rather
than attempt to correct errors by post-processing (recall that
a bigram model was used for decoding in these experiments).
However, using n-grams with \( n > 2 \) requires a more complex
and more computationally expensive decoder. Also, the algo-

}