ISCA Archive
http://www.isca-speech.org/archive

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

EFFICIENT WEIGHT TRAINING FOR SELECTION BASED SYNTHESIS

Yoram Meron *Keikichi Hirose*
Department of Information and Communication Engineering, Faculty of Engineering
The University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113-8656, Japan
(meron,hirose)@gavo.t.u-tokyo.ac.jp

1 ABSTRACT

Synthesis systems, based on unit selection from databases with a large number of unit examples from different phonetic and prosodic contexts, have been shown to allow a high quality speech synthesis [1].

One of the difficult problems associated with this kind of synthesis is determining the strategy for unit selection, and tuning its parameters. In this work we suggest a way to extend the usefulness of two existing training methods, by using phoneme pairs as the basic comparison unit.

Using unit pairs is shown to significantly increase the efficiency of the exhaustive weight search training method on one hand, and refining the regression weight training method on the other.

2 BASE FRAMEWORK

The base framework we start from, is a CHATR style synthesis system [1]. For the selection of units, a target cost and a concatenation cost are defined. The target cost C^T is a weighted sum of the difference between target and candidate feature vectors:

$$C^T(t_i, u_i) = \sum_{j=1}^{p} w_j^T C_j^t(t_i, u_i) \quad (1)$$

The concatenation cost C^C is defined similarly:

$$C^C(u_{i-1}, u_i) = \sum_{j=1}^{q} w_j^C C_j^c(u_{i-1}, u_i) \quad (2)$$

Where t_i is the target unit, u_i is the candidate unit for the i-th phone, and p, q are the number of features used. These features are typically phonetic label, pitch, duration and energy, and possibly an acoustic distance measure.

Given a sequence of target phonemes, a number of candidates (taken from similar phonetic contexts) are selected from the database for each target phoneme, and a Viterbi-like algorithm is run, to find the best matching candidate sequence. This sequence is then used for synthesis.

3 EXHAUSTIVE SEARCH

In order to train the weights w_j^T and w_j^C, an automatic scheme is suggested [1] [2], which uses an objective acoustic distance between cepstral vectors of an original utterance and its time aligned, re-synthesized version. By performing an exhaustive search on many weight combinations, the weight combination which produces the synthetic version closest to the natural speech is selected. Due to the large number of parameters, this is very computation intensive (hundreds of CPU hours).

In order to accelerate the weight training process, two components of the process were examined and improved. Instead of iteratively running the *selection-synthesis-comparison* processes for each weight combination separately, it was observed that it would be more efficient to first run the selection for all the weight combinations, and then run the synthesis-comparison for the selected candidate sequences.

3.1 Unit Selection

Simple algorithmic modifications were used in order to accelerate the selection process. In the unit selection process, for each weight combination, the target and concatenation costs have to be calculated for all the candidates. As shown in Eq. (1)(2), each of C_i^t and C_i^c are a sum of context window weighted features. By pre-calculating the *unweighted* costs for each candidate in advance, the selection process is much more efficient.

Further, if the weight combinations are organized efficiently (so that each time only one weight changes), incremental calculation can further improve calculation efficiency. Using these simple measures, selection can be accelerated by about two orders of magnitude.

3.2 Synthesis and Scoring

After selecting speech units from the database, the output waveform is synthesized using PSOLA synthesis [3] (with prosody modification). The produced waveform is then coded as a sequence of cepstral vectors, time aligned (using a DTW algorithm) with the target (natural) utterance of the sentence, and an acoustic distance is calculated between the two cepstral sequences.

Looking at the results of the unit selection process (see table 1), it is seen that for a given (large) number of weight combinations tested, the number of *different* selected unit sequences is much smaller. This can be used for acceleration of the process. Since we break up the training process, and first do the selection for all weight combinations, it is possible to perform the synthesis only once for each different selected sequence.

In this example, for 120,000 different weight combinations tested, there were only 65790 different full length sequences selected (i.e. full length synthesis needs to be done only for about half the number of weight combinations). Moreover, the number of single units (8), pairs of units (24) , and unit triplets (53) actually selected is much smaller.

Units	1	2	3	5	10	full
Diff. Seq.	8	24	53	167	1096	65790

Table 1: Average number of different selected sequences of different lengths for 120,000 weight combinations. Average on 10 files (average number of phonemes per file was 60, 40 candidates per unit)

In order to take advantage of this, we synthesize and score only short parts of the full utterance. Then, we add the results of the appropriate short parts, in order to get the score of the full synthetic utterance.

Although using single phonemes for the synthesis-score process would be the most efficient, this would not account for the effects of concatenation on synthesized speech quality. The length of the synthesized short parts, N, must be 2 or more, and the actual acoustic distance calculation must be done only over the N-1 diphones (excluding the short part's edges, where no concatenation is performed).

All the different unit sequences (selected by the unit selection process) of the decided length are synthesized, and an acoustic distance is calculated between them and the (time aligned) relevant part of the natural utterance. After this is done for all short sequences, a second pass goes over all the selected whole length sequences, and for each sequence, sums the scores of all the short parts which appeared in it.

3.3 Results

Table 2 shows the results of different training conditions. 120,000 weight combinations, and 10 utterances were used for training. The weights selected by each method were used, and the average acoustic distance was calculated over 25 test utterances. A listening test (playing 12 sentences to 10 listeners), was used to find the listeners' preference, and rank the training methods accordingly.

In the last two lines of table 2, a time limit was used, forcing the full iteration method to use only a smaller number of weight combinations. This number of weight combinations was then randomly picked (from the already calculated 120,000 combinations), and the best weight combination was selected and used for synthesis (the table shows the average score of repeating this random selection 10 times).

Method	Run time [sec]	Acoust. dist.	Ranking
Full iterations	99600	1.532	1
2 phonemes	290	1.538	3
10 phonemes	2040	1.539	2
(350 comb.)	(290)	1.686	5
(2450 comb.)	(2040)	1.579	4

Table 2: Run time (average on 10 train files with average sequence length 60 phonemes) and acoustic distance results (shown for 25 test files)

The training set is obviously small, but this is exactly because the full search method was too computationally intensive to allow more utterances to be used (requiring about 1.5 weeks to run on a Pentium 450).

Naturally, no training method can achieve a better acoustic distance score than the full iteration method (at least on the training set, as it directly minimizes the tested quantity). However, from a practical point of view, when a limit on computation time is enforced, the results produced by the method suggested above, outperform the full iteration method.

4 REGRESSION TRAINING

In the method described above, given a set of weights, a sequence of pairs, and their scores are found. Going the opposite way, given the scores of the pairs, we would like to find the set of weights which would have selected the pairs with the best scores. This is what regression training tries to achieve.

In [4], a training method is suggested in which the target costs are trained using a regression method, and the concatenation costs are trained in the iterative search method. This method offers the following advantages:
1) Computation is reduced, by separately training target and concatenation costs,
2) Different costs can be trained for different phonetic contexts (reflecting the variance in feature importance for different phonetic contexts), and
3) it makes use of many observations, and thus are more robust than the exhaustive search method, (which chooses only one data point).

On the other hand, training target and concatenation costs separately might not be optimal (as they are not independent). Further, this method does not take into account prosodic modifications done at synthesis time (for systems which do perform this process).

4.1 Suggested Method

A new training method was tested, which performs regression training of target and concatenation weights simultaneously, and is computationally feasible. The basic idea of [4] was modified as follows:

1. *Pairs* of concatenated units (instead on single units) are compared to “natural” pairs, thus allowing training of target and concatenation costs together.
2. Units are prosodically modified before concatenation, allowing closed loop training.
3. Data driven phonetic context clustering is added to improve the phonetic context modeling.

4.2 Instance score calculation

The score for each pair instance is calculated in a way, similar to that described above in section 3.2. Pairs of units are synthesized and compared to the natural utterance, however, the comparison is run on *all* possible candidate pair combinations - there is no pre-selection stage.

As we are trying to model both the target and concatenation weights, the comparison should include the target phoneme *and* the effect of the concatenation, but *not* the previous or next phonemes. Therefore, the comparison is carried out on the frames shown in figure 1: including the second pair member, and a short overlap with the first pair member, but excluding frames influenced by the next phoneme. In this way, the comparison result is decided mostly by the target cost and much less by the concatenation cost (as an average phoneme duration is between about 50 and 100[ms], while the concatenation influences a region of about 20[ms]). To increase the effect of concatenation (which is argued to contribute strongly to the perception of synthetic speech quality), the part of the cost relating to frames in the concatenation region is stored separately, and an independent multiplicative parameter (α) can control it’s weight.

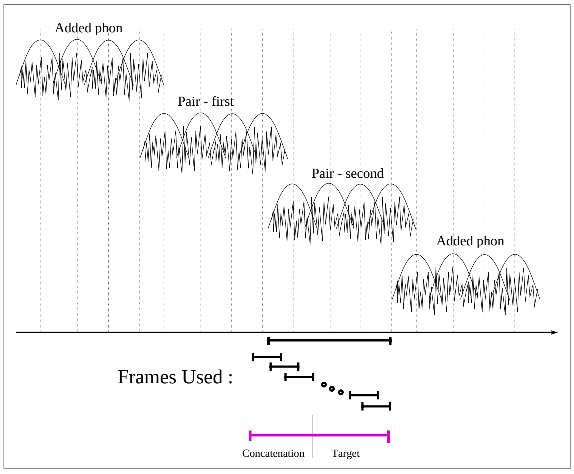

Figure 1: Frames used for calculating distance measure.

This is done for each of the utterance unit pairs, and for each of the training utterances. The acoustic distance score of the pair’s comparison are saved together with the value of all it’s target and concatenation sub costs. This data is used for regression training, which tries to predict the acoustic distance score from a given set of target and concatenation costs.

4.3 Phonetic Clustering

When only target weights are trained, the regression training can be performed separately for different phonemes or phoneme sets, which allows finer context modeling. Adding concatenation costs to the training, makes use of phoneme pairs, thus making it possible to increase the context sensitivity of the trained weights, by training different weights to different phoneme *pairs* or sets of pairs.

In this experiment, a data driven clustering method was used to cluster phonetic clusters. The clustering is performed by building a decision tree, in which each non terminal node contains a phonetic context question of the type : *“Does the phoneme in the left/right belong to* $PhonSet_i$ *?”* where $PhonSet_i$ is one of several subsets of phonemes (with some common phonologic features). The construction of the tree starts with one node (the root), which corresponds to all phoneme pairs. We then recursively split the remaining pairs into two sets by greedily choosing the best question. In this case, the best question is the question which splits the pairs into two sets, such that the error of regression modeling of the two sets is minimal.

Stop criteria define when the splitting process should be stopped - either a lower limit on the number of training examples (below which estimation of the regression parameters may not be reliable), or a lower limit on the regression error improvement due to a split.

4.4 Experiment

An experiment was conducted to test the performance of the suggested regression training method, with different clustering conditions.

Experiments were performed using different regression training conditions. First, one set of parameters were trained for the whole database (all the phonemes together). Next, different sets of weights were trained for each one of 32 phoneme separately. Finally, phonetic clustering was used to separate the data into clusters, and weights were trained for each cluster separately.

It was confirmed that the clustering reduced the regression error prediction of the acoustic score, and improved the acoustic distance score on test utterances.

The results of a listening experiment (playing 9 sentences to 10 listeners) showed that listeners ranked first the synthetic utterances produced by training regression parameters on 50 clusters, ranked second the utterances produced by the regression parameters trained separately for each phoneme, and third the training done on all data together (1 cluster). However, listeners noted the differences between synthetic utterance quality was minor.

Another test was run, increasing α (the multiplicative parameter for the concatenation costs) for the case of 50 automatically trained clusters. In a listening experiment, 10 listeners scored 9 sentences. The listeners' preference was quite consistently $\alpha = 2 > \alpha = 1 > \alpha = 4$.

5 CONCLUSION

Improvements to two methods of weight training for selection based synthesis were presented.

For the the exhaustive search approach, breaking the select-and-score process into separate selection and scoring was shown to enable more efficient calculation. Faster training can increase the size of the search space, allowing better weight combinations to be found. Alternatively, more sentences can be used for training, making the found weights more robust.

Simple algorithmic changes were shown to accelerate the selection process by two orders of magnitude. For the scoring process, acceleration is a little more complex, and is achieved by breaking the whole utterance into short unit sequences, thus avoiding repetitions in calculations. This was shown to accelerate calculations by two orders of magnitude. A listening experiment showed that given the same amount of computing resources the suggested method produces better results.

An improvement to the regression training method was introduced, which allows simultaneous training of target and concatenation weights, taking into account prosodic modifications at synthesis time. Different regression coefficients were trained for different phonetic contexts (selected by data driven clustering), to improve regression prediction. A listening experiment showed that the clustering improved the synthesis results. Increasing the weight of concatenation costs was also shown to improve synthesis quality.

References

[1] N. Campbell and A. Black, "Prosody and the selection of source units for concatenative synthesis", In J. Van Santen, R. Sproat, J. Olive and J. Hirschberg, editors, *Progress in Speech Synthesis*, Springer Verlag, 1995

[2] A. Black and N. Campbell, "Optimizing selection of units from speech databases for concatenative synthesis", Eurospeech95, pp 581-584

[3] F. Charpentier, E. Moulines, "Pitch Synchronous Waveform Processing Techniques for Text to Speech Synthesis Using Diphones", Speech communication Vol. 9, pp 453-467, 1990

[4] A. Hunt and A. Black, "Unit selection in a concatenative speech synthesis system using large speech database", ICASSP96, V1, pp 373-376

[5] A. Black and P. Taylor, "Automatically Clustering Similar Units for Unit Selection in Speech Synthesis", Eurospeech97, pp 601-604