Optimal Data Selection for Unit Selection Synthesis

Alan W Black and Kevin A. Lenzo

Carnegie Mellon University and Cepstral, LLC
awb@cs.cmu.edu, lenzo@cs.cmu.edu

Abstract

In this work, we address the issue of creating a set of utterances with optimal coverage for reliable, high quality concatenative synthesis, whether for general synthesis or domain synthesis. We present an automatic method that takes into account the acoustic distinctions made by a particular speaker and selects prompts from large databases of typical utterances. A general unit selection text-to-speech system created by this process can synthesize any input text, but the output is best for content intended to be similar to that in the database in terms of style, delivery, and coverage.

1. Background

Unit selection synthesis, where appropriate sub-word units are selected from databases of natural speech, seems to hold the promise of high quality natural sounding speech synthesis. However, the quality of such systems is inherently related to the quality and appropriateness of the database from which the units are selected.

In the extreme case, it has been shown [2] that if the database is deliberately and closely tailored to the intended application, high quality synthesis can be produced robustly and reliably. Many applications are of the sort where the domain can be adequately specified, such as time, weather, and even apparently open dialog systems like the CMU DARPA Communicator flight information system [12]. But not all applications are so, and a more general solution is desired. The work presented here addresses exactly that problem, how can we find the best set of utterances to record in order to cover the domain we intend our synthesizer to perform well in. The techniques presented here are suitable for both relatively limited domains to completely open domains like reading arbitrary stories or names and addresses.

When a unit selection synthesizer produces high quality output, it is bringing forward the implicit style within the unit database, as well as other factors in the system. In this paper we do not address the harder problems of appropriately modifying the units we select to allow for a richer output; unlike our earlier work on limited domain synthesis [2], however, we are building synthesizers that can say anything, though their style will still be more appropriate for the domain for which they were designed. That is, a synthesizer based on a database that cover names and addresses well will be able to read arbitrary new stories, but will sound odd, slow, and hyperarticulated. Likewise, a unit selection synthesizer built using newswire text will make every turn in a dialog system sound like a CNN report.

2. Initial attempts

For limited domain voices, the process of designing what to record is easier, even if there is not an automated process. At first approximation, the object is to include in the databases at least one occurrence of each word in the domain in each desired prosodic context. Hand design can often be adequate for such domains.

Our initial attempts to do this automatically for new domains was to take a large example of the intended output and greedily select sentences that maximized the word coverage. To some extent, this works, but obviously does not take into account the phonetic coverage of the database, and so word joins may be poor. We can take each sentence and generate the diphones that are required to synthesize it and then greedily select sentences which have the best diphone coverage. We also investigated selecting for demisyllables rather than diphones to see what sort of coverage was necessary.

As a test we took the text of Lewis Caroll's "Alice's Adventures in Wonderland" [5] to see how many sentences are sufficient given these various criteria. The Gutenberg version of Alice (`alice29.txt`) contains 26,457 words from which Festival finds 1,918 utterances. The following table shows the number of utterances needed to cover the criteria as selected by a simple greedy algorithm.

Optimize for	utts	% total
Words	979	51%
Diphones	196	10.2%
Demisyllables	312	16.2%

There are, of course, many ways to describe and define demisyllables; here we use onset as initial consonantal gestures (if any) as well as an initial portion of the vocalic nucleus, and the coda as the remainder of the vocalic portion into the the final consonantal gestures (if present). Syllable affixes were not treated distinctly from the coda. Thus, the units can be written as onset cluster - vowel and vowel - coda cluster, respectively. The demisyllable inventory and feature set is based upon [8], with some simplifications.

We can easily add other factors to the units we are trying to optimize for, be that lexical stress, pitch and/or metrical accents, position in phrase etc. As pointed out in [13] getting all possible features and all contexts is prohibitive. The addition of each multiplies the amount of data requires to systematically cover the space.

Another direction to find the right data to record is to define the space and then explicitly create a comprehensive database by design. Simple diphone databases are a prototypical example of this, we define the phone set and then what diphones can appear in the language, and then carefully design a database to ensure it has one example of each of the token types defined (e.g. [11]). This direction seems feasible for smaller inventories, but as the combination of features grow we have to make more and more decisions about pruning that space, as collecting everything would be a monumental task – let alone the post-processing steps necessary.

In these two methodologies – define features and greedily select from data, and define features and expertly design the data – two distinct aspects are missing.

1. The frequency of the units is ignored. Some unit types are much more frequent that others, this fact could help define which corners could be cut without degrading the synthesis except in rarer cases.
2. Acoustic distinctions, or their lack, are ignored. When multiplying out the various units, some will not be phonetically realized as distinct, e.g. some vowels preceding /t/ may be indistinguishable from those preceding /d/.

What we want to do is find exactly which units are acoustically distinct, and take into account the distribution of units in order to design a corpus for collection that will exactly cover all the variations – with minimal redundancy.

3. Cluster unit selection

Before we address the two specific shortcomings for selecting data mentioned above, let us include a description of the unit selection algorithm we have been using, because its method helps offer an answer to those questions.

This general unit selection method is first described in [3] which also includes some techniques of both [6] and [10] in it. The general idea is to take all units of a particular type and define an acoustic distance between them. In the experiments presented here the types are phones, though they could be diphones or demisyllables. Using features such as phonetic, metrical and prosodic context find which features can best split the cluster such that the mean acoustic distance is smaller. Then recursively apply this splitting until some threshold size is achieved. Thus using CART techniques [4] we end up with a decision tree indexed by features available at synthesis time identifying clusters of acoustically similar units.

More formally, we define the acoustic distance $D(U, V)$ between two units U, and V where $|V| > |U|$ as

$$P\frac{|U|}{|V|}\sum_{i=1}^{|U|}\sum_{j=1}^{n}\frac{W_j}{n\sigma_j|U|}abs(F_{ij}(U) - F_{(i\frac{|V|}{|U|})j}(V))$$

where P is a duration penalty, $|U|$ is the number of frames in U, W_j is the weight for parameter j. $F_{xy}(U)$ is the parameter y of frame x of unit U, σ_j is the standard deviation of parameter j, and there are n parameters. The term $F_{(i\frac{|V|}{|U|})j}$ is F_{xy}, where the x index is computed as $i \times \frac{|V|}{|U|}$, and $y = j$.

We can then define the impurity of a cluster as

$$Impurity(C) = \frac{1}{|C|^2}\sum_{i=1}^{|C|}\sum_{j=1}^{|C|}D(C_i, C_j)$$

Then, using standard CART techniques, we greedily find the question that gives the best information gain, and split clusters to minimize the summed impurity of the sub-clusters.

Run-time synthesis consists of selecting the appropriate cluster using the CART trees and then finding an optimal route through these candidate units using a Viterbi decoding algorithm.

Of course there are many degrees of freedom in such a system, including the definition of an acoustic distance – both the parameterization and their relative weights – which must correlate with human perception of speech. We have found that pitch synchronous Mel-frequency cepstral coefficents to be a useful representation, and use mean weighted Mahalanobis distances over the frames in two units with a duration penalty, [3] also included delta cepstral coefficients but at least in our more recent databases we have not found them useful.

An important by-product of this particular unit selection algorithm is a classification of the acoustically distinct units in a database. Each cluster in the tree represents an acoustically distinct typical unit. Thus, given a large database with broad enough coverage, we can automatically find out what acoustically distinct units are and what are contexts they are likely to appear in.

Of course with speech, you can never be completely sure that your distinctions are fixed, but they are reasonable approximations. The distinctions found in this process are database-specific, as well as speaker and style specific, but if the seed databases is of a reasonable coverage, they may be useful in defining selection criteria for a corpus with better coverage.

4. Selecting the best database

Thus, armed with a cluster tree from a database, we can now use it to find the optimal set of utterances to cover the acoustic space. This algorithm works in three stages:

1. From a general speech databases, build a cluster tree as one would when creating a unit selection voice.
2. Using typical utterances for the domain, count the number of uses of each cluster, were the corpus to be synthesized.
3. For the same data (or otherwise), greedily select utterances which have the highest score and coverage.

As this is a bootstrap process, we have not yet been able to investigate different types of databases from which to build the initial cluster tree. What is important, though. is to have a fairly general databases with good phonetic coverage and a diversity. In our experiments, described below, we choose the largest databases we had from one of our key speakers, but we are in the process of investigating this further.

Once the initial tree is created, we take a much larger example of text from the domain. We have experimented with place names (for all zip codes in the U.S.), and people's names from movies (as in, for example, the Internet Movie Database at imdb.org) – though no results are presented from those databases here. Our main experiment has concentrated on simple stories, as general text reading is still a common application for speech synthesis. The details of that experiment are presented below.

We take the database and synthesize it with respect to the basic cluster tree and count how often each cluster is used in the synthesis of the large amount of text. As unit selection databases – even our current ones – tend to have unnatural distributions, using some real data allows the frequencies of the clusters to be better estimated.

The third stage is to greedily select the utterances which have optimal coverage with respect to the frequency-weighted cluster tree.

If we score an utterance only by the number of uses of each of the units needed to synthesis it, we would continually select only utterances containing very high frequency units, and never provide examples of less frequent units. Thus, we want an iterative selection process that notes instances that have already been selected and appropriate gives new examples of existing items a lower score.

The basic selection algorithm is

```
todo\_list = all sentences
while todo\_list
  score all utterances in todo\_list
  remove all utterances with
    score 0 from todo\_list
  select best utterances
    mark its items in the cluster tree
  remove best utterance from todo\_list
```

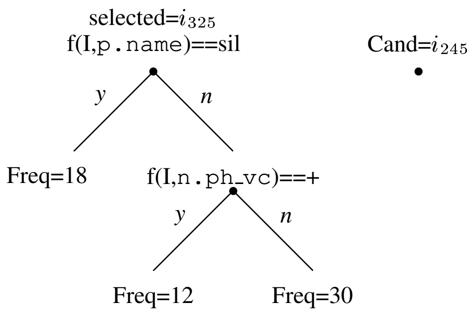
The score mechanism deserves further explanation. For each phone item in a candidate utterance we recursively step down the cluster tree, asking each question until we reach a node already marked with an item from a previous selected sentence. We then ask the question at that node to see if the existing item and the candidate item can be distinguished (i.e., one answers "yes" to the question and the other "no"). If they can be distinguished, the candidate item is given a score of sum of the frequencies on all leafs subordinate to its answer node.

If they cannot be distinguished by the current question, the candidate item is given a score of zero, which is slightly pessimistic, but justified given the objective function. If the node is a leaf node the candidate item is also given the score of zero (as we already have an example). We score all sentences in the corpus and find the one whose sum its item's score the highest.

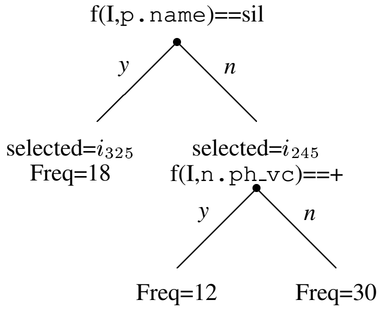
Selecting that utterance, we then add its items to the cluster trees. We again recursively follow the cluster tree questions for each item until the distinguishing node is found. The currently selected item is then moved down one level to mark the appropriate yes/no node below and the newly selected item is used to mark the other node.

Thus, as sentences are selected, we fill out the coverage of the trees, and as we fill in the high frequency parts of the cluster tree, the less frequent ones become more important.

An example may help illustrate this. If we had the following condition where item i_{325} was selected on a node with question "f(I,`p.name`) == sil" (the previous item to I is a silence). We have a candidate item i_{245}. If the answer to "f(i_{325},`p.name`) == sil" is "yes," and "f(i_{245},`p.name`) == sil" is "no" (i.e. these can be distinguished) then the score for i_{245} is the sum of the frequencies below the node. In the case 42.



If the sentence contains i_{245} is eventually selected, the tree is modified as follows



A special condition is needed at the start, as no units have been selected to check if potential new ones can be distinguished. In this case, the score is defined to be the number of occurrences of that phone; that is, the sum of the frequencies of every cluster in the tree.

There are various strategies possible for this scoring. We are looking for the incrementally best addition each time. Some sentences may contain some good units in general but their items are not distinguished immediately from the currently selected items and hence will be scored with 0 even though that sentence my ultimately have been a good example.

The exhaustive search for the best set of items is of course computational very expensive so some compromise is necessary, though we have not investigated if this current strategy is the most reasonable given that constraint. But this method does seem at least partially adequate. Also, this greedy selection technique is not dependent on the order of the sentences in the corpus.

It is important to note that, despite using a greedy algorithm to select utterances, the resulting coverage is still complete, and covers all necessary distinctions – and thus is optimal, given the objective function.

5. Example

As we wish the result to be unencumbered by copyright restrictions, we started with a set of 19 out-of-copyright novels from Project Gutenberg [9]. This is a total of 1.2 million words. If synthesized, the audio would take just under 4 days to play.

As a starting cluster tree, we used the CMU Communicator domain voice. KAL, the voice used for that (created from an author of this paper), speaks with a standard US English accent. We used a version of the limited domain corpus consisting of around 900 utterances and built a standard unit selection synthesis (clunits) voice from it. That voice performs well on Communicator like utterances, but isn't particularly good on general synthesis.

We then synthesized the complete 1.2 million word corpus, counting the usages of each cluster in the clunits cluster tree. As we did not need to actually generate the audio, we skipped the final part of selection and synthesis once the required unit types were selected, but this pass still takes several hours.

Because we wish to use the result as prompts in recording, and hence we want the utterances to be relatively easy to say, we then pruned the complete text databases to give more reasonable sentences to from which to select. All candidate sentences were between five and twenty words in length.

After initial tests and noting that the scoring technique does not cater properly for repeated words, as items are not added to the tree during scoring but only on selection of the best utterance, repeat words are doubly scored. We added the restriction that candidate sentences do not contain any repeated words, which is not an unreasonable condition anyway.

The result of these restrictions gives us a total 34,796 utterances from which we select candidates. The candidate search is also computational expensive; although we do not do full unit selection synthesis, we still do a significant part of that process. Also, because the database is big, we cannot store all the synthesized utterance structures, and need to re-calculate on each pass. Thus, the initial pass to find the best selected takes over an hour on our current setup. The selection algorithm runs continuously over the data selecting the utterance that best contributes units to the cluster tree, and ignores utterances that add nothing. As this process proceeds, the cycles do get faster.

In our test, the first, i.e. most contributable utterance is

Allow me to interpret this interesting silence.

which is from Bronte's "Emma." This sentence contains six different vowels, and nine different consonants. We were also amused to find "Humpty Dumpty sat on a wall." was included in the set of selected utterances, from "Through the Looking Glass".

After several days of computing, we ended up with a list of 241 sentences. On inspection, some of these sentences were very unusual, and some are hard to say, so we removed some of these, even though we are aware this is affecting the distribution of units we want in the database. To some extent, unusual sentences are expected, as our selection process is trying to maximized coverage and therefore mis-spellings, and unusual text (like letters spelled out) will have a higher score, for example "Deed you A I N T" from "Huckleberry Finn." After hand checking the complete list and removing difficult to say or unnatural utterances we were left with a total of 221 utterances.

The number seems smaller than we expected, though it is perhaps not surprising, given this is effectively selecting one example of each cluster. But in order to get more examples, we ran the selection again, excluding the sentences from first selection list, and so generated a second list of 146 utterances. We could repeat this process a number of times.

6. Evaluation

We took the two sets of sentences, (221 and 146) and recorded KAL delivering them. We built basic unit selection voices, using the standard FestVox build process, and hand corrected the phonetic labels of each. Three test voices were actually built, one with just the 221, one with just the 146, and one with the combined set. We spent some time tuning the voices to find good parameters, such as the right cluster size and features used for selection. However, for the best quality, we know we would need to do more correction of the labels.

A set of 20 sentences had been previous created for testing purposes; one of these sentences (the first sentence of Alice) was contained in the 34,796 utterances training set but not in the selected set, but the rest of the test set were independent.

Judging speech synthesis quality is not easy, even – perhaps especially – if you listen to a lot of it. In general, it is fairly easy to reliably determine what is much better, but in close cases where multiple factors may affect the quality of speech (e.g. joins, prosodic smoothness and segmental quality) such judgments become not so clear, and subjects, when questioned, may differ.

We synthesized the twenty sentences, and one of the authors listened to randomly ordered examples form each comparison. Opinions were collect in terms of A being better than B, B better than A or equal quality.

	A	B	A=B	better
txt_221 vs txt_146	4	8	8	txt_146
txt_221 vs txt_367	6	10	4	txt_367
txt_146 vs txt_367	3	12	3	txt_367

It is clear that the largest set is best, but it is interesting that that it is not so clear which of the 221 and 146 sets are best, even when one is 50% larger than the other. The 146 is often smoother, but typically has less variation in prosody.

Relative quality helps in deciding directions, but this does not determine if the resulting voice is good enough for real applications. A second test was done on these voices with respect to further five different test sets.

alice 20 sentences from "Alice", which was part of the training set (none of these test sentences from "Alice" were actually in the set of recorded utterances).

timit 20 sentences from the TIMIT databases, a phonetically balanced set of 452 sentences [7].

comm 20 sentences from the CMU Communicator testing suite, used in a speech dialog system.

festvox 20 sentences from the abstract of [1].

story 20 sentences from a novel that was not part of the original database.

A five point score was used, 5 being indistinguishable or nearly indistinguishable from recorded speech, 4 being errors but understandable, 3 being understandable with difficulty, 2 bad but some part discernible and 1 nearly incomprehensible or worse. We had 4 people listen to these examples.

testset	Listeners Mean Scores				Total	
	1	2	3	4	mean	rank
alice	4.4	4.15	3.3	3.95	3.95	1
timit	3.75	3.75	2.95	2.85	3.32	4
comm	3.7	3.9	2.4	3.25	3.31	5
festvox	3.75	4.05	3.25	3.4	3.61	3
story	4.0	4.3	3.05	3.95	3.82	2

It is clear to anyone listening to the voices that it is most appropriate for reading stories, which is expected as that is the domain the voices were selected for. The TIMIT sentences are, of course, more complex, as they were deliberated chosen to have good phonetic coverage. The communicator data is a different style, and its quality is not as good, even though it is understandable; it does not have the fluency and appropriateness the limited domain voice has, which confirms our hypothesis that domain voices can always sound better than general purpose voices, because they are more appropriate to the task at hand.

7. Future Work

It is clear that, although we have had some success, a number of other related experiments must be tried, and we are in the process of carrying them out.

First is a proper investigation into how the initial cluster tree affects the selection, both from the distribution of the original data, and from the speaker. It is clear that each speaker will have a different acoustic differentiation of units, as their physiologies and speaking strategies differ. Thus, using speaker-specific acoustic model cluster trees and selecting data appropriate for them may give better results.

We have built a number of other voices from the 367 utterance set and, although they are surprisingly good, none are as good as the KAL voice. This may of course be due to other factors, such as style and experience in delivering speech to be made into a synthesizer.

We have made initial studies using a tree built for AWB (a Scottish English speaker) from a 452 (US) TIMIT database. When this is used against the same 34,796 utterances, a different set of utterances is selected. Although the first utterance is the same as the KAL selection list, the second and subsequent are different. However, we have not yet had time to compare the quality of a voice built from that selection with speaker AWB, compare with AWB using the KAL based selection.

An important use for this work is to find the best subset to record to reasonable synthesis it, given a finite known corpus. For example, which utterances should be recorded to build a voice capable of building a synthesizer of particular text in a high-quality, characteristic voice. Initial experience on selected for Alice suggest at the quality we are producing here, we would need around 257 utterances, about 13.3% of the total, and the growth is sub-linear in the size of the text. Such compression, while maintaining character and quality, can be very useful for audio books, manuals, and other materials.

8. Summary

The generated prompt list, examples and code used to select data is distributed as part of FestVox. On line examples discussed in this paper are available from `http://festvox.org/dataselect/`.

We presented a method for finding a small set of utterances from a large database that optimally cover the acoustic-phonetic space with respect to a particular speaker. The quality of voice produced from the selection is far superior that a voice build form a more general database. The resulting voice is better for data, while retaining the style of the database from which it was trained.

9. References

[1] Black, A., and Lenzo, K. Building voices in the Festival speech synthesis system. http://festvox.org, 2000.

[2] Black, A., and Lenzo, K. Limited domain synthesis. In *ICSLP2000* (Beijing, China., 2000).

[3] Black, A., and Taylor, P. Automatically clustering similar units for unit selection in speech synthesis. In *Eurospeech97* (Rhodes, Greece, 1997), vol. 2, pp. 601–604.

[4] Breiman, L., Friedman, J., Olshen, R., and Stone, C. *Classification and Regression Trees*. Wadsworth & Brooks, Pacific Grove, CA., 1984.

[5] Carroll, L. *Alice's Adventures in Wonderland*. Macmillan, 1865.

[6] Donovan, R., and Woodland, P. Improvements in an HMM-based speech synthesiser. In *Eurospeech95* (Madrid, Spain, 1995), vol. 1, pp. 573–576.

[7] Fisher, W., Doddington, G., and Goudie-Marshall, K. The DARPA speech recognition research database : specifications and status. In *Proceedings of the DARPA workshop on speech recognition* (1986), pp. 93–99.

[8] Fujimura, O. C/D model: a computational model of phonetic implementation. In *DIMACS Proceedings*, E. Ristad, Ed. Am. Math. Soc., 1993.

[9] Hart, M. Project Gutenberg. http://promo.net/pg/, 2000.

[10] Hunt, A., and Black, A. Unit selection in a concatenative speech synthesis system using a large speech database. In *ICASSP-96* (Atlanta, Georgia, 1996), vol. 1, pp. 373–376.

[11] Lenzo, K., and Black, A. Diphone collection and synthesis. In *ICSLP200* (Beijing, China., 2000).

[12] Rudnicky, A., Bennett, C. Black, A., Chotimongkol, A., Lenzo, K., Oh, A., and Singh, R. Task and domain specific modelling in the carnegie mellon communicator system. In *ICSLP200* (Beijing, China., 2000).

[13] van Santen J., and Buchsbaum, A. Methods for optimal text selection. In *Eurospeech97* (Rhodes, Greece, 1997), vol. 2, pp. 553–556.