Applying Backoff to Concatenative Speech Synthesis

Lily Liu
Stanford University
lliu23@stanford.edu

Luladay Price
Stanford University
luladayp@stanford.edu

Andrew Zhang
Stanford University
azhang97@stanford.edu

Abstract

In concatenative speech synthesis systems, the lengths of the speech units can vary from phones to syllables to entire words, providing varying levels of contextual information. Our project expands on the primary traditional methods of concatenative speech synthesis—diphone synthesis and unit selection synthesis.

We investigate the effect of varying contextual information by applying a novel system of backoff to phone selection in a concatenative speech synthesis system. We assume that units with greater numbers of matching surrounding phones provide more contextual information, so units with less matching contextual information should only be used when more matching contextual information is absent.

Thus, we use triphones, biphones, and monophones for contextual information on a phone of interest; when the unit with the most contextual information is absent, we use a tiered selection system to back off to a unit with lower phone context.

Our current system shows that the more relevant context surrounding the phone of interest, the more natural the synthesized speech sounds. It also shows that backing off to different levels of context is a fruitful way to concatenate speech units if the unit with the best context is not in the database.

1 Introduction

The aim of a concatenative speech synthesis system is to select and stitch together units of sound that best reproduce some target sentence. Common concatenative speech systems include diphone synthesis systems and unit selection synthesis systems.

Diphone synthesis generally involves recording one speaker pronouncing diphones, speech units that begin in the middle of one phone and end in the middle of the following phone. One example of each diphone is then stored in a database. To synthesize a specific utterance, the right sequence of diphones is selected from the database and concatenated together to produce one speech utterance. Signal processing is usually required to smooth transitions between diphone recordings, as well as enhance the natural sound of the utterance.

Unit selection synthesis involves a similar process of recording and storing speech units in a database, selecting the most appropriate units from the database using a cost function, and concatenating them together to produce the target. However, unit selection systems generally use larger units, ranging from diphones to sentences.

Both of these concatenative systems have strengths and weaknesses.

Diphone synthesis systems require a comparatively small database (Bunnell and Yarrington, 1998). Furthermore, the diphones stored in the database generally sound fairly natural, as the diphones are easy for the speaker to read. However, they may not be pronounced correctly, especially less common diphones.

Once the target utterance has been generated, signal processing is still required to make the result sound more natural; even after signal processing, the resulting speech can still sound artificial. Bunnell et al. raised additional concerns with diphone systems in the paper. They point out that coarticulation poses a challenge for diphone concatenation. Diphones are not context independent; surrounding phones do affect the pronunciation of the diphone of interest.

As a result, it is difficult to remove a diphone
from one particular context and introduce it into a different context without perceptible discontinuities in the speech (Bunnell and Yarrington, 1998).

Lastly, using such small units can often fail to capture word-specific effects, syllable structure, and stress patterns (Maas, 2017).

Unit selection synthesis systems require a lot more speech than diphone systems do. Access to more speech in the database generally results in storage of multiple copies of sounds, which provides greater opportunity to choose the right unit for the given context. Thus, unlike diphone synthesis, little to no signal processing is required to produce the target utterance, since the units are generally larger than diphones.

Stöber et al. discussed using words as primary synthesis units with natural-sounding results for a first version system in the paper (Stöber, 1999). Hunt and Black experimented with unit sizes, comparing syllables, diphones, phones, and half-phones (Hunt and Black, 1996). Perceptual tests found that syllables as synthesis units created more natural-sounding speech than phones, diphones, and half-phones, while half-phones performed better than diphones and phones. Thus, variable synthesis unit size is a positive aspect of unit selection synthesis.

However, calculating unit costs to determine how well a unit matches the utterance can be computationally expensive. Additionally, overall quality of the produced speech can fluctuate; some sections may sound very natural, such as if a larger unit of speech from the database was used, while other sections may sound unnatural. Lastly, Stöber et al. found that the storage complexity was much higher for words as units than diphones (Stöber, 1999). Depending on the unit size, unit selection databases may need to store an order of magnitude or more units than diphone systems (Maas, 2017).

Our approach draws inspiration from, as well as expands on, these two system types. We used a forced alignment tool to align an audiobook with its transcript, then segmented the phones and stored them into a database, noting whether each phone appeared at the front, in the middle, or at the end of a word, or just as a single phone.

Given a target utterance to produce, we first attempt to use a phone from the database that is an exact match—the correct target phone with the proper surrounding context. The uniqueness of our approach is addressing which phone to use if an exact matching phone is not present in the database. We can choose from triphones, bi-phones, and monophones for contextual information.

By using triphones, we attempt to synthesize speech using context both before and after our choice of phone, partially addressing some of the concerns from Bunnell et al. Our system implements a unique system that organizes potential phones into different tiers of contextual appropriateness. If desired phone is absent in the highest tier, we backoff until one is found.

This tiered system allows us to choose phones with the most appropriate level of context. As with diphone synthesis, our approach uses small speech units, since we ultimately concatenate monophones, albeit with ample contextual information. For our approach, we apply no external signal processing alongside our concatenation, to maintain a low computational cost associated with synthesis. Thus, our approach acts as an extension of both diphone and unit selection synthesis. Although this may result in a less consistent, lower quality speech sample, this allows us to focus on testing the viability of our backoff system.

2 Methods & Approach

Our pipeline (Figure 1.) is composed of two essential yet asynchronous pieces. The first part constructs the database of phones. Once the database is created, we move on to the second part (concatenative speech synthesis). The speech synthesizer uses our specialized backoff system to select phones from the database to synthesize the desired speech. We opted to use the ARPAbet to represent our phones, which provides benefits to both components of our pipeline.

2.1 Database Construction

The first necessary task in constructing the database was segmenting the audio training data and labelling each phone. Our training data was an audiobook of George Orwell’s Nineteen Eighty-Four, which provided over 9.5 hours of spoken English. To label and segment the data, we used a forced alignment tool built on Kaldi called Gentle (Ochshorn and Hawkins, 2017). We feed the audio files and transcript of the book into Gentle, which provides us with the segmentations for each word into its component phones.

Our choice to use Gentle offers two significant
Figure 1: An overview of the backoff concatenative speech synthesis model.

benefits to our pipeline. Gentle happens to also internally use CMUdict to help split the words that appear in the transcript into their component phones. However, there are potential problems with blindly using CMUdict for all aspects of our pipeline.

The speaker of the provided audio might not pronounce words exactly in line with CMUdict. In fact, most likely, a significant portion of the words may have imperceptibly mispronounced words or alternate but equally correct pronunciations. Indeed, in common usage, many people typically do not pay attention to differences in more similar phones (especially vowels).

Fortunately, while Gentle uses CMUdict to obtain preliminary pronunciations of each word, it returns an accurate classification of each individual phone, and only uses the provided pronunciation to guide word-level alignment of the transcription. This drastically stabilizes our database, which would otherwise be much less predictable, with many of the stored vowel phones containing a significant mix of phones.

Gentle labels the segmented phones by returning their ARPAbet representation. From there, we extract the timestamps that map each ARPAbet phone in the transcript to its corresponding audio data. In any cases where the Gentle was unable to recognize a word in the audio, if a word in the transcript was out of the vocabulary, or if there is an extraneous word in the transcript, we do not include any relevant phone information in our database.

Alongside each phone and its corresponding audio data, we extract another label: “B”, “I”, “E”, or “S”. phones labeled “B” appeared as the initial phone of a word, those labeled “I” were in the middle of a word (not at either end), those labeled “E” were the final phone in a word, and S phones were standalone words.

These four labels define our four main sub-databases, and all phones are stored in their corresponding-label database. This provides a convenient but useful alternative to storing all phones as triphones, biphones, or monophones. While “I” technically represents triphones, and “S” represents true monophones, “B” and “E” are two completely different kinds of diphones. “E” phones have pre-context (a phone before it as context) while “B” phones have post-context (a phone after it as context). As such, these must be stored separately if we are to fully utilize their context.

These labels provided a convenient way to locate triphone, biphone, and monophone contexts.
without the need for an unsuitable database; we can extract triphones from “I”, both ‘flavors’ of di-
phones from “B”, “I”, and “E”, and monophones from all four databases.

2.2 Concatenative Speech Synthesis

The second main part of the system synthesizes the desired speech given a target piece of text.

The target text is first translated into ARPA-
bet phones. We stored a file containing over
134,000 words and their corresponding ARPA-
bet transcriptions taken from the Carnegie Mel-
lon University Pronouncing Dictionary (Dictio-
nary, 2014). For each word in the target text,
we use the pronunciation file to return the words
ARPAbet transcription.

Once we obtained a complete phonetic transla-
tion of the target text, we proceeded with building
the output speech word by word, phone by phone. The goal is to find the best phones in the database
that will join together to produce the most natu-
ral speech. The best phone in the database is, of
course, the exact phone we are looking for, with
the exact surrounding context we are looking for.

Take for instance, the word “win.” If the phone
of interest is “ih” preceded by “w” and followed
by “n,” we want to search the database for a phone
“ih” that has “w” and “n” phones preceding and
following it; as this is the exact context we desire,
using this phone should result in more natural out-
put speech.

With enough training data, the phone with the
exact context should be present in the database.
However, this is not always the case. Our train-
ing data could have been an insufficient amount of
speech or biased towards certain phone sequences.

Whatever the reason may be, we cannot possi-
bly expect that the phone with the exact context
we desire is already present in the database. Our
goal then becomes producing a phone from the
database that shares as much context with the ex-
act phone as possible—finding a phone that is good
enough.

Thus, we used a tiered backoff system to search
for phones in the database if the exact phone with
the exact context we desire is not present. The tier
structure determined the way we searched through
the database, and depended on the location of the
target phone.

2.2.1 Beginning phones

Phones at the beginning of a word have a follow-
ing phone and no preceding phone; thus local con-
text is provided through the phone on its right. As a result, we decided that the next best phones
would be intermediate phones with the exact fol-
lowing phone, as these provide biphone context.

If these phones were not present in the
database, we then searched for any other inter-
mediate phones, beginning phones, and single
phones. These provide the least amount of context
with the target phone, so these are searched for
last.

This produced the following overall order for be-
beginning phones:

1. Matching beginning phones with the exact
following phone
2. Matching intermediate phones with the exact
following phone
3. Matching intermediate phones, end phones,
or singular phones

2.2.2 Intermediate phones

If the target phone was both preceded and fol-
lowed by phones, it was considered to be an inter-
mediate phone. Local context for these phones is
provided by phones on both sides. As a result, we
decided that the next best phones should share at
least one side of context. We first searched for end
phones that had the same previous phone as the
target or intermediate phones with the same previ-
ous phone as the target phone.

If these phones were not present, we then
searched for beginning and intermediate phones
that were followed by the same phone as the target
phone. This search prioritizes matching end and
intermediate phones over matching beginning and
intermediate phones. We chose this order because
we assumed that preceding phones have more con-
textual influence on pronunciation than following
phones. This order also reflects the fact that be-
ginning phones tend to be pronounced more dis-
tinctly; since the target phone is preceded by a
phone, the output speech might sound a little un-
natural if a phone in the middle of a word was un-
necessarily emphasized.

If none of the above phones are present in
the database, the biphone contexts have been
exhausted. We then searched for intermediate
and single phones that matched the target phone.
We searched for these last because, although they matched the target phone itself, they shared no local context with the target phone.

This produced the following overall order for intermediate phones:

1. Matching intermediate phones with the exact preceding and following phone
2. Matching end or intermediate phones with the exact preceding phone
3. Matching beginning or intermediate phones with the exact following phone
4. Matching intermediate or single phones

2.2.3 End phones
Phones at the end of a word have a preceding phone and no following phone; thus local context is provided through the phone on its left. As a result, we decided that the next best phones would be intermediate phones with the same preceding phone as the target phone. Since end and beginning phones have no preceding phones, this was the only biphone context available. If these phones were not present in the database, we searched for matching beginning phones, single phones, and intermediate phones with any preceding and following phones. These are the last options as they share the least amount of context with the target phone.

This produced the following overall order for intermediate phones:

1. Matching end phones with the exact preceding phone
2. Matching intermediate phones with the exact preceding phone
3. Matching beginning, intermediate, and single phones

2.2.4 Single phones
With single phones, there is no within-word context to consider. Since single phones are also standalone words, we figured that these phones are usually emphasized and pronounced clearly. To take this into consideration, we chose to select matching beginning phones first as the next best phones. If these were not present in the database, we searched for matching end phones. We searched for matching intermediate phones last. The selection order for single phones follows the clarity of pronunciation; the pronunciation of beginning phones most closely resembles the pronunciation of single phones. Since the pronunciation of end and intermediate phones can be overshadowed by surrounding phones, we searched for these second to last and last.

This produced the following overall order for intermediate phones:

1. Matching single phones
2. Matching beginning phones
3. Matching end phones
4. Matching intermediate phones

Phones that match the search criteria for each phone type are added to a list. In our preliminary system, we randomly choose a phone from this list of matching phones. Once we have chosen phones for each one in the target text, the phones are stitched together to form words. We also add silence between each word.

3 Experiments

3.1 Using Context to Select Phones
We wanted to confirm that using a phone with greater surrounding context was better than using a singular phone with less context. For example, we suspected a phone taken from a context that shares the right and left phones as in the target utterance should be selected over a phone that shares only the left phone with the target context.

We took this to somewhat of an extreme, comparing a system that chooses the correct phone completely randomly, without using any surrounding context, to a system that finds either a perfect context match, or backs off to a random choice within the tier. As expected, choosing the phone with more context resulted in more naturally sounding speech.

3.2 Relative Importance of Context
We investigated which contexts should be exhausted if the exact context is not in the database. For example, during the process of building our system, we heavily tested and judged which context was more valuable. For instance, a preceding context is more valuable than a following context in terms of overall smoothness of output speech. Many of these insights are reflected and explained under Concatenative Speech Synthesis.
3.3 Phone Selection by Pitch Comparison

While we do divide up potential phones into tiers, we ultimately still choose a phone from our database, albeit within the highest tier available. Instead, one experiment we ran was attempting to match the ending pitch (from the third quarter) of the previous phone with the beginning pitch (from the second quarter) of the next.

We tested methods using fast Fourier transforms as well as autocorrelation, and combinations. However, we found a slight but marginal increase. Due to noise and because we are working with very short audio samples, the frequency estimations occasionally varied greatly.

3.4 Modifying Silence between Words

To improve fluency and overall sound quality of the target speech, we adjusted the amount of silence between each word. With a 150ms pause between words, the synthesized speech sounded like it was rushing between words. However, when we tested pauses longer than 300ms, the produced speech sounded noticeably slow. Thus, we settled on a 200ms pause between words, a natural compromise that produced better-sounding speech.

4 Conclusion & Future Work

Our current model shows that in synthesized speech concatenation, the more relevant context surrounding the phone of interest, the more natural the synthesized speech sounds. It also shows that backing off to lower levels of context is a fruitful way to concatenate speech units if the unit with the best context is not in the database.

Furthermore, the model was also able to improve if relative pitches were considered for the selection of the best phone to use after finding the units with the best context. Lastly, the hand-tuning of silences between words was a crucial final step in improving the final synthesized speech.

Our system has several benefits compared to the two types of speech concatenation systems mentioned previously.

Unlike diphone synthesis systems, our system does not require the recording and labeling of speech, as this is done automatically by forced alignment and segmentation using audio and text files that are free to anyone on the web. Thus, it is more time-efficient and does not need additional hardware or software to record and process the raw recordings. Also, our system could potentially have more accurate sounds immediately after concatenation.

It was shown that synthesized speech using phone concatenation improves drastically if phones are taken from units that have the same surrounding context; thus, because our units and phones are taken from English words, the synthesized text could also capture more word-specific effects, syllable structure, and stress patterns.

Compared to unit selection synthesis systems, our system does not require as large of a database, and the storage complexity is lower. Because we only concatenate individual phones, our system is also less computationally expensive. This tends to result in a more uniform overall quality of synthesized speech.

However, there are limitations to our system. Because the units are selected using backoff, if the database does not include enough units with enough context for the concatenation of the target sentence, then the resulting output may not sound as good. This challenge can be minimized, however, if the dataset includes units from a large and diverse input set.

Another restriction is that the system needs a dataset segmented from speech that is relatively uniform so as to keep the pitch fluctuations as small as possible in general cases.

Currently, work is ongoing to investigate the effect of volume on unit and phone selection.

Further work to expand this model includes adding onto the database of phones to include more speakers and data, selecting the best phone from the larger database, then adjusting the pitch and duration of the phones by using PSOLA. A last step that could be taken to improve the system could be implementing a prosody model to make the final synthesized speech sound more natural.

Acknowledgments

Many thanks to Professor Andrew Maas and the CS224S staff for all of their advice and support.

References


## A Supplemental Material

Our codebase and detailed demos (sample outputs and comments) can be found at

https://www.dropbox.com/sh/fzgtmvakfttiez9/AACSCzpVxVVTl1p-FYUCavp8a?dl=0