Progress Report

Alex Park
04/05/2004

Summary: This progress report is an overview of the work I have been carrying out on an alternate approach to isolated word recognition/detection using the transient synchrony idea presented in Hopfield et. al, 2001. I give an overview of the background behind the approach as it was originally presented as well as recent modifications I have made in preparation for word recognition experiments on PHONEBOOK.

1 Introduction

The recognition approach detailed in this work is modeled after Hopfield's digit detection paper on the fictional *Mus Silicum* organism. In that paper, the authors describe the physiology and neural computing principles of a simple organism that is able to distinguish between different spoken utterances. The detection procedure utilized by the organism takes place in two steps that can be roughly divided into front end signal processing, followed by a template matching step performed with neural circuitry. The schematics of the front end are shown in Figure 1.

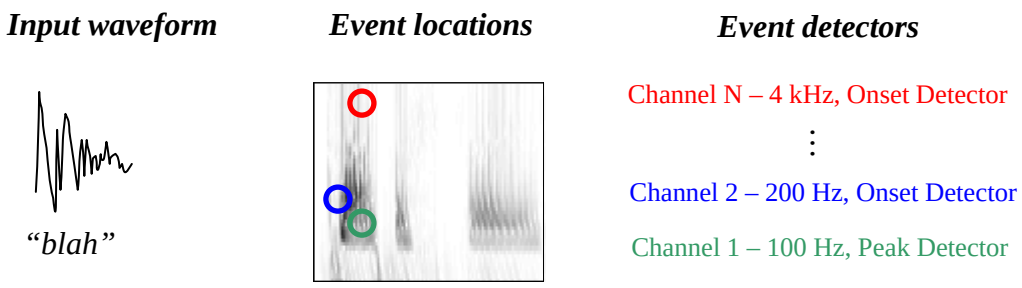

Figure 1 – Spectral event detection front end.

The incoming waveform is converted into a set of event times, each one corresponding to a particular channel illustrated by the colors. For the front end illustrated above, events are characterized by onsets, peaks, and offsets in spectral energy within different frequency ranges. Each event triggers activity in a set of *event driven* (ED) neurons whose spike rate peaks, then decays at an intrinsic rate. The decay of the spiking rates is characterized by decaying neural input currents to the ED neurons.

When the currents of the spiking neurons intersect, the spike trains synchronize for a brief period of time as shown in Figure 2. Training a word/lexical unit corresponds to making connections to the ED neurons that synchronize for that particular lexical unit.

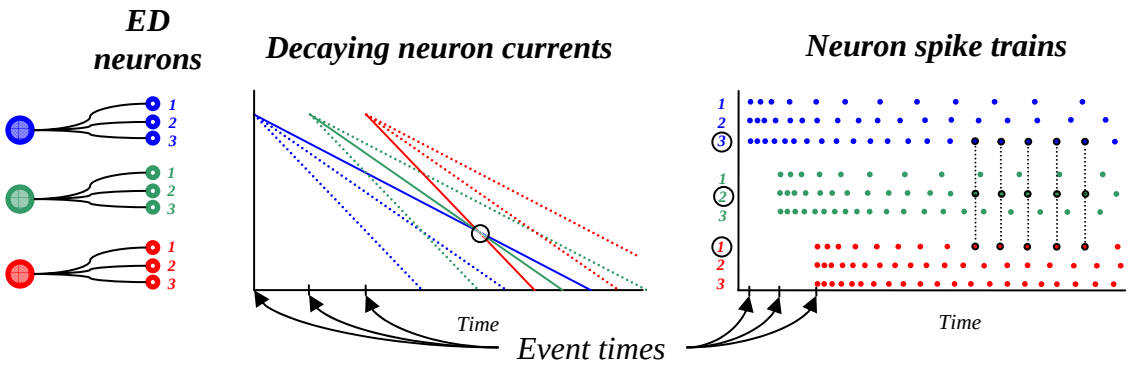

Figure 2 - Decaying input currents and spike trains from ED neurons.

The role of a lexical unit detector is to aggregate the responses of the ED neurons is to detect transient synchrony in their spiking outputs. The mechanism for detecting transient synchrony is illustrated in Figure 4, where the aggregated spike output for synchronizing and non-synchronizing spike trains are compared. In the synchronizing case, the ED neurons sum together to yield large spikes in the aggregated output, while the non-synchronizing neurons sum to an output with much more limited range.

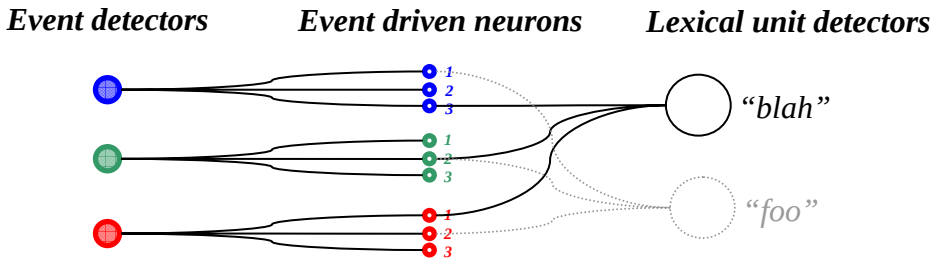

Figure 3 - Topology of neural connections.

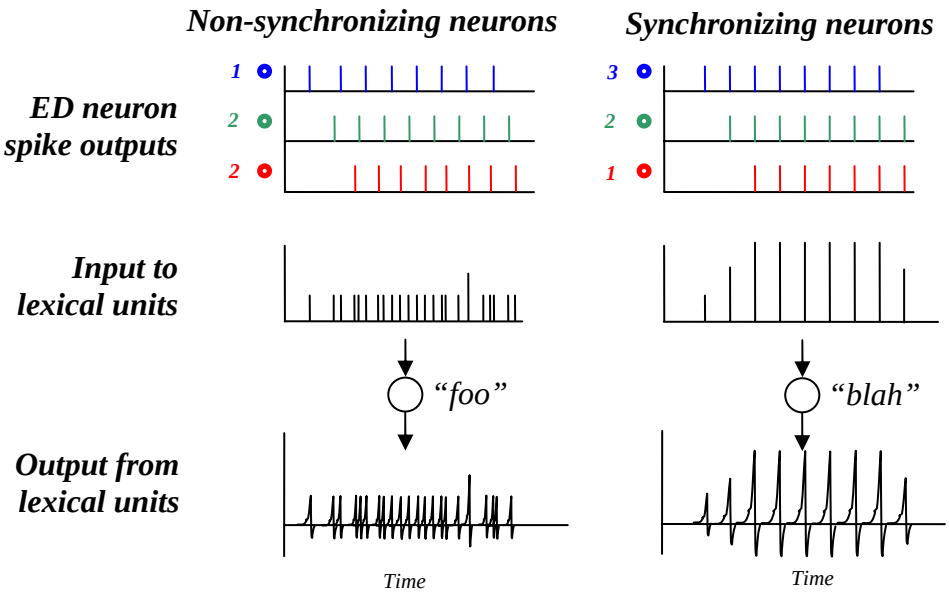

Figure 4 - Input/Output characteristics for lexical detection units in presence and absence of synchrony. Top row shows the action potentials for a set of non-synchronizing neurons (left) and synchronizing neurons (right). Middle row shows the aggregate input to the lexical detection unit. Bottom row shows the output of the lexical detection unit.

1.1 Recognition Framework

We can now consider how to perform isolated word recognition with these models. The recognition framework can be deduced from Figure 3. We "train" each lexical unit in the lexicon by making connections from the appropriate ED neurons. Recognition is performed by running the input waveform through the network and picking the lexical unit whose output response is maximum. The overall procedure is illustrated in Figure 5.

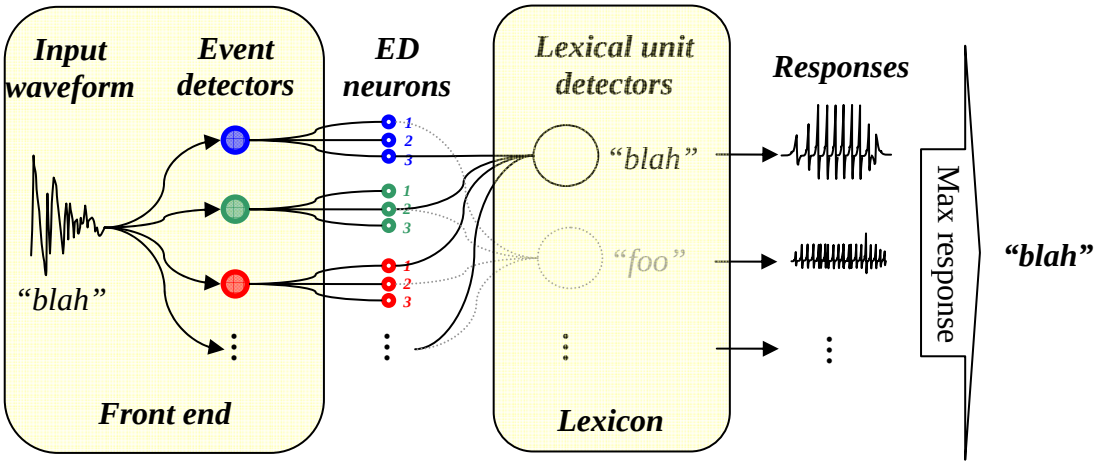

Figure 5 - Recognition pathway. Hypothesized word is given by the unit whose output response is maximal over the lexicon

As we noted earlier in this section, the overall system can be described in two main components. The front end is responsible for producing event times for each channel which drive the ED neurons. The lexical unit detectors combine the outputs of the ED neurons to perform classification over the lexicon by giving an output response, or "score", for each detector in the lexicon.

1.2 Remarks

There are several features of Figure 5 that should be noted. Note that both the frontend and the lexicon are not constrained to be of the form that we specified. That is, events do not need to be onsets and maxima in the spectrogram. Events can be lower-level spectral features as described or they can also be derived from onsets of phonetic/distinctive features. At a higher level, we can think of events in terms of occurances of phones, syllables, or even words. Likewise, the lexical units in the lexicon do not need to be words. Any unit which is temporally identifiable via event types driving the ED neurons can be used as the output of the network. For example, at a higher level, we could have the event detectors be word detectors, and the units in the lexicon be phrases, or word bundles.

Another feature that should be mentioned is that computation can be greatly reduced by not computing responses for all lexical units in the lexicon. Instead we can reduce the number of *active* lexical units by considering only a cohort set produced by first-pass recognition, or by using information from previous words to prime or "wake up" a set of related or likely next words.

In the next section we describe extensions to the framework that we have just described.

2 Extensions

The previous section outlined and summarized implementation details concerning Hopfield's work on *Mus Silicum*. There are several additional modifications that we have made in the course of our work with this approach. In particular, we examined the possibility of using non-discrete current decay rates for the ED neurons, and a method for combining multiple input examples for training a single lexical unit detector.

2.1 Non-discrete ED neuron current decay rates

The first change that was made concerned the nature of the ED neurons. Instead of using a discrete set of ED neurons for each event detector, we simulated an infinite library of ED neurons so that we could access a continuum of decay rates. Instead of selecting ED neurons to connect to during training, we construct and connect the ED neurons in question. This training process is illustrated in Figure 6.

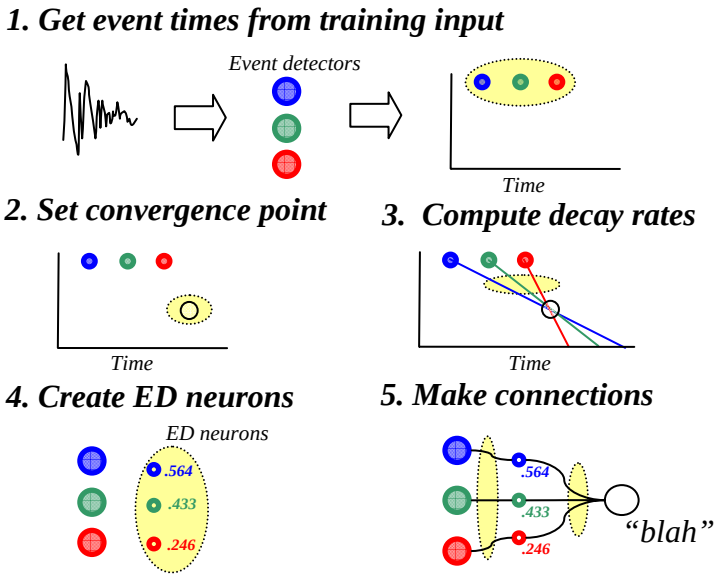

Figure 6 - Designing input specific ED neurons based on selected convergence point. In contrast to Hopfield approach, ED neurons are not selected from a pre-existing set, but are created to accommodate event times and convergence point.

2.2 Multiple Input Training

A second modification that needed to be made concerned the issue of multiple input training. In the original description of the digit detection approach, each lexical unit was trained on a single training example. For robustness and generalization, use of multiple training examples is beneficial, but the training procedure made no allowances for this possibility. Our approach to this problem, which is shown in Figure 7, was to generate an *average* set of ED neurons from the custom ED neurons created for each training

example. Mathematically, this corresponds to taking the arithmetic mean of the decay rates for channel events over all inputs in which they are present. By averaging these slopes while maintaining a common convergence point, the ED neurons decay rates generalize, becoming less convergent on any particular input, but more convergent across all similar inputs.

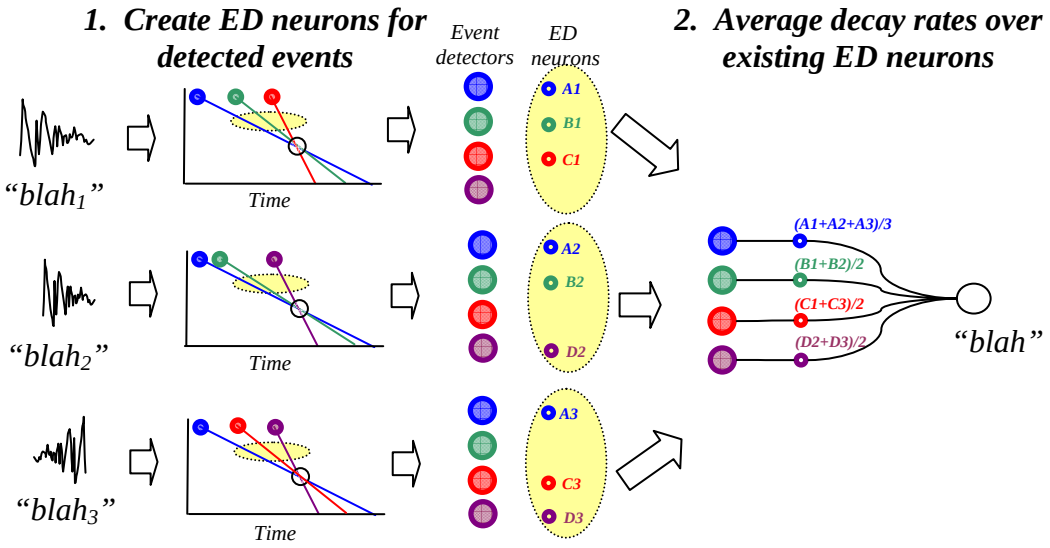

Figure 7 - Framework for using multiple examples for training a single lexical unit.

3 Model-based Event Detection

In beginning experiments on PHONEBOOK, we realized that the spectral front end was somewhat unreliable in the face of speaker variation and was limited by the band-limited nature of the channel detectors. This section describes our attempt to make use of statistical phone models for event detection.

3.1 Event detection from landmark model scores

As we mentioned before, the spectral front end described in Hopfield's paper is only one example of an event detector that can be used. An alternate event detector that we have constructed makes use of the output probabilities of landmark phone models for event detection. For the original front end, channels corresponded to actual frequency channels, and the measurements used to detect events in those channels were the spectral power. In contrast, the alternative front end we describe here has channels which correspond to class conditonal probabilities for context-dependent phone models. There are two main advantages to using the non-spectral front end. First, using channels that correspond to phones allows us to predict the connections that will be made between lexical units and ED neurons via phone transcriptions and baseforms. Second, each event channel incorporates multiband spectral information and can utilize all statistical model training techniques that have already been developed in the ASR community.

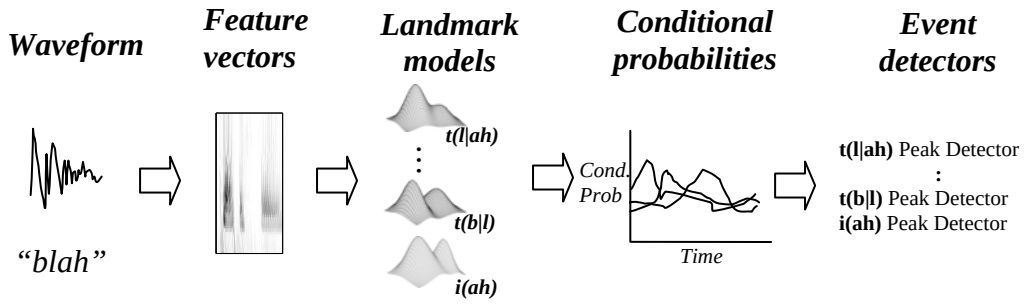

Figure 8 - Use of landmark models to create phonetic event detectors.

The sequence of steps prior to event detection are shown in Figure BLAH. However, there are several critical differences in performing event detection with a spectral front end versus the probabilistic front end. With the spectral front end, we assume no prior knowledge of the channels which can yield events for a particular lexical unit. With a phone-based probabilistic front end, we can constrain events to occur only for phone channels that we expect to be active for that lexical unit. Thus, for the lexical unit "blah", we can look only in the channels for phone units that occur in all possible transcriptions of "blah", i.e. t(-|b), t(bcl|l), i(l), t(l|ah), etc. We can then perform event detection on each channel in relation to the other active channels.

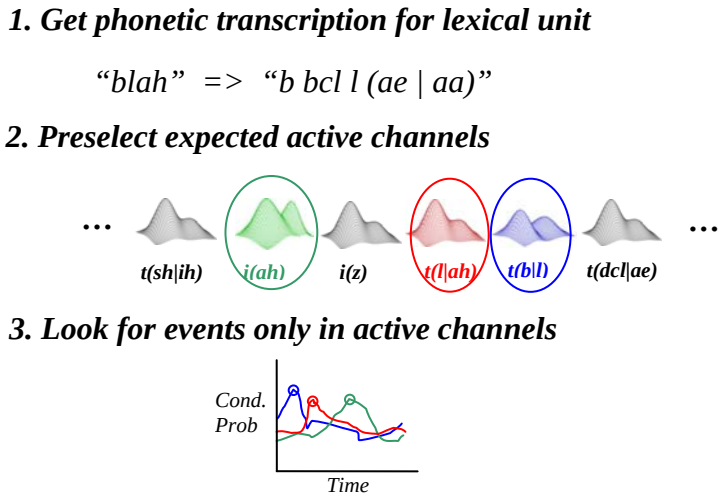

Figure 9 - Selection of active units prior to performing event detection with landmark based event detectors.

We perform event detection on the constrained set of conditional probability time tracks with a simple algorithm. First we take the global maximum probabilities across all event channels, then pick events for the represented channels from those maxima. This process is illustrated in Figure 10.

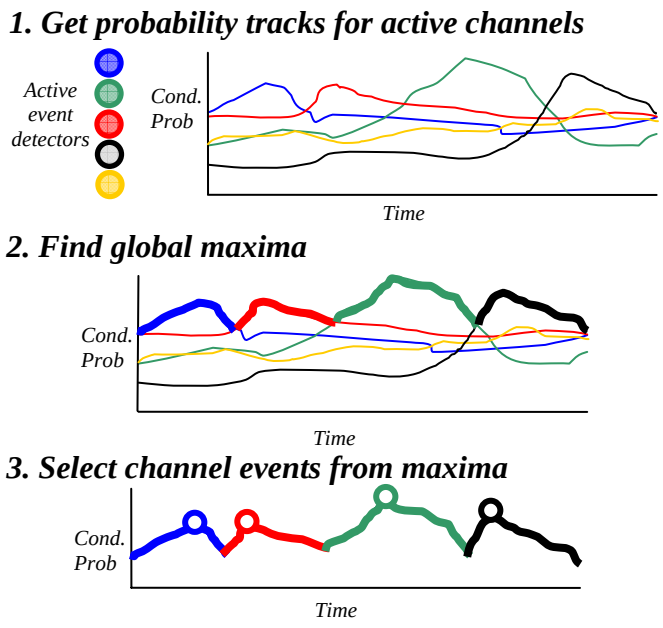

Figure 10 - Event detection algorithm. Channel events are selected only from time spans in which that channel is maximum among the active channels.

An example set of events generated by this event detection approach for the word "windshield" (utterance 'aa74_f01' from phonebook) is illustrated in Figure 11. If we order the events by their time of occurence, we can see that the event channel identities occur in the expected order, given the phonetic transcription of the word. This indicates that our event detection algorithm is at least able to preserve the order of events that we expect.

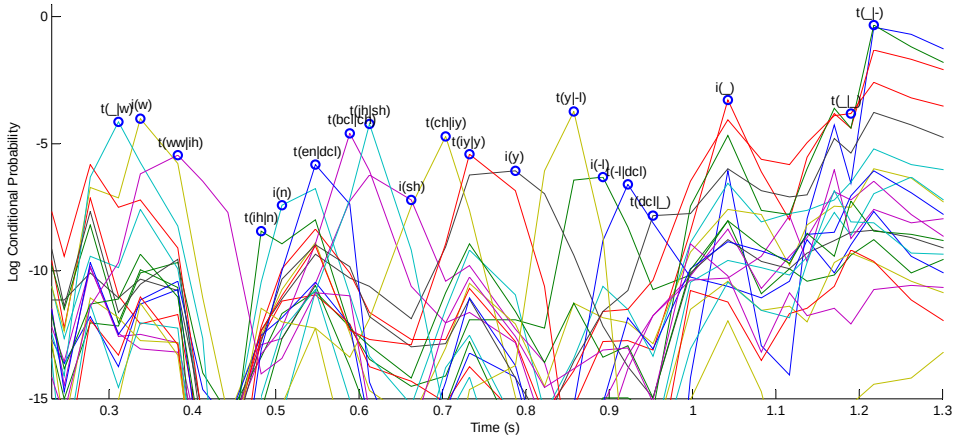

Figure 11 - Log conditional probabilities in active channels for utterance of "windshield" (tag aa74_f01), along with events detected in each channel.

Original phonetic transcription

_ w ih n dcl sh iy y l dcl _

Event channel identities ordered by event times

_|w	w	ww|ih	ih|n	n	en|dcl	bcl|ch	jh|sh	sh	ch|iy	iy|y	y	y|-l	-l	-l|dcl	dcl|_

Figure 12 - Comparison of original phonetic transcription for tag aa74_f01 with channel events from Figure 11, ordered by detection time.

3.1.1 Example – Different test inputs presented to single detector

The potential of this approach can be seen by considering the testing scenario in Figure 13, where a lexical unit is trained on an utterance of the word “windshield”, then tested on three different utterances: “windshield”, “westfield”, and “accompanied. The outputs of the training and testing phases are shown in Figure 14 and Figure 15, respectively. First, the events detected for the previous example (tag ***aa74_f01*** for the utterance “windshield”) are used to train a lexical unit detector for the word “windshield”. Because we are setting the ED neuron decay rates to converge exactly, we see synchronization occuring in the spike trains, as well as a strong peak in the output current for the trained lexical unit.

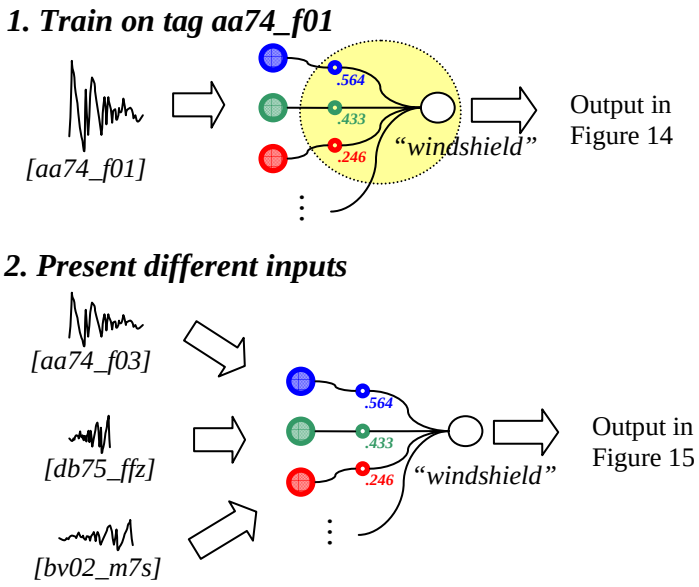

Figure 13 - Testing scheme for Example 1. Lexical unit is trained on utterance aa74_f01 and outputs are compared for utterances aa74_f03, db75_ffz, and bv02_m7s.

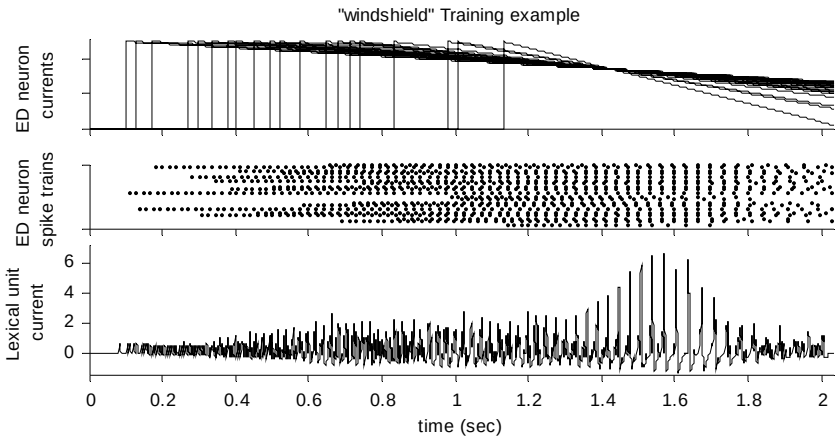

Figure 14 - Output of lexical unit for training input "windshield" (tag aa74_f01).

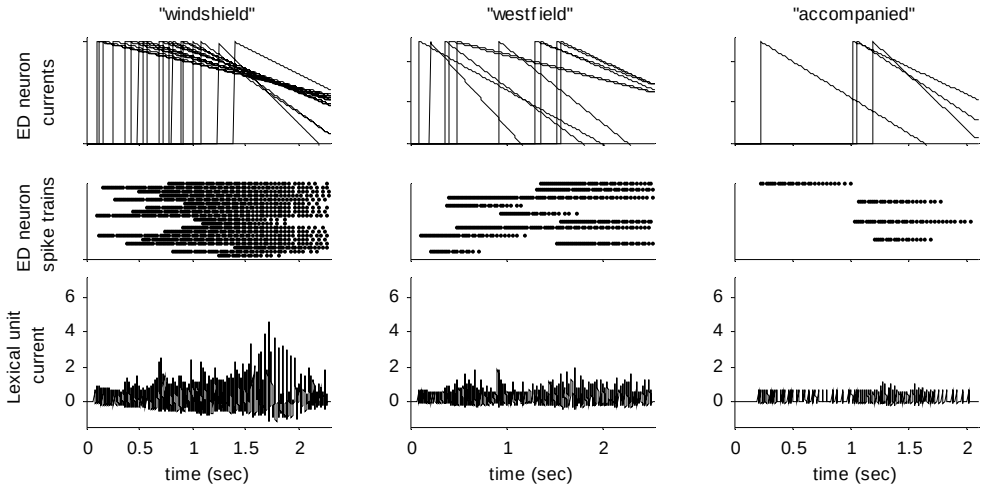

Figure 15 - Outputs for test utterances "windshield" (tag aa74_f03), "westfield" (tag db75_ffz), and "accompanied" (tag bv02_m7s) tested on "windshield" lexical unit trained on tag aa74_f01.

We now present the trained lexical unit with three unseen spoken utterances: “windshield” (tag ***aa74_f03***), “westfield” (tag ***db75_ffz***), and “accompanied” (tag ***bv02_m7s***). The response of the windshield lexical unit detector to each of these inputs is shown in Figure 15.

Of the three inputs, the “windshield” utterance triggered the most event detections among the active channels. By observing the converging nature of the ED neuron current decay rates, we can also see that the temporal order of the events was similar to that of the training example. The decay rate convergence leads to temporal synchrony of the ED neuron spike trains, which in turn leads to a peak in the lexical unit currents. In contrast, we can see that the events detected for the other two utterances occurred out of their expected order, leading the current decay rates for their ED neurons to diverge. This in turn leads to poor synchrony among the neuron spike trains and no peak in the aggregated lexical unit output. This example illustrates how a lexical unit responds strongly to examples similar to the ones it was trained on, and weakly to examples that are different.

3.1.2 Example – Single test input presented to different detectors

The second example concerns the outputs of different lexical units for a single test input and simulates the process that would occur in word recognition/detection. The testing scheme is illustrated in Figure 16. Separate lexical unit detectors are trained, then presented with a single test input. The outputs, which are shown in Figure 17, show that the windshield lexical unit has the most events in common, the best convergence, and the strongest output response of all three trained units. During recognition, we would pick out the strongest response as our hypothesized unit.

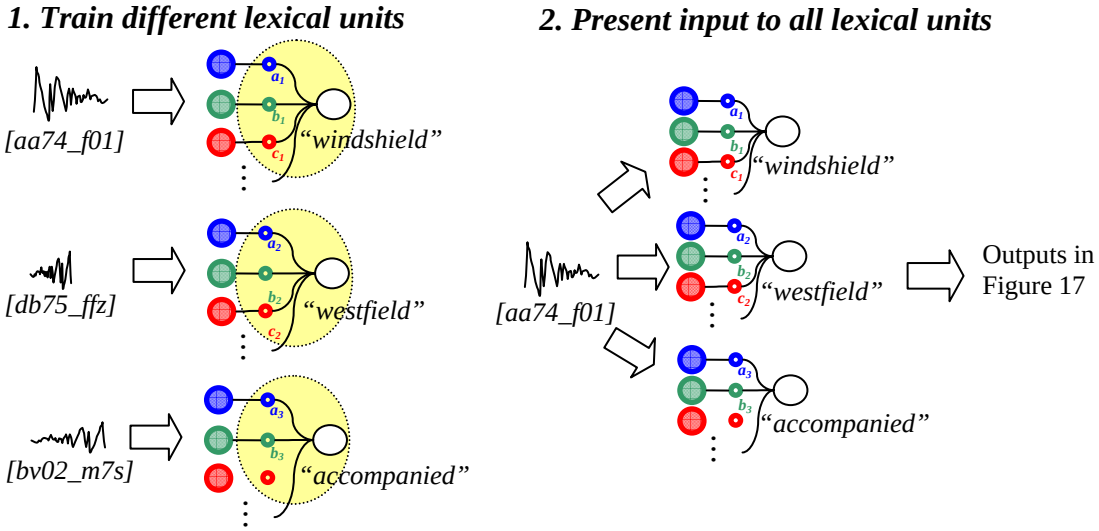

Figure 16 - Testing scheme for recognition example. Lexical units are trained on different words, then tested with the same unseen utterance of the word "windshield" (tag aa74_f03).

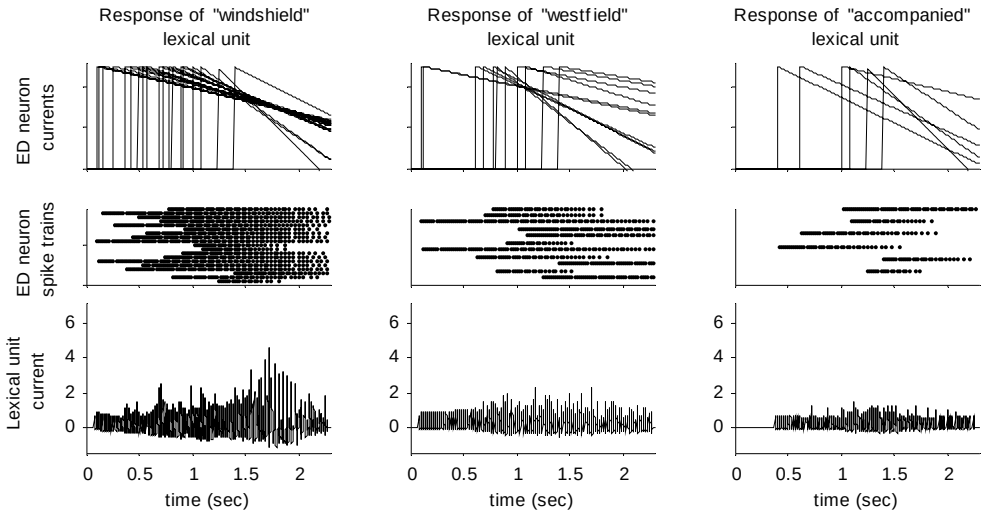

Figure 17 - Responses of 3 different trained lexical units to common test input, an utterance of the word "windshield" (tag aa74_f03).

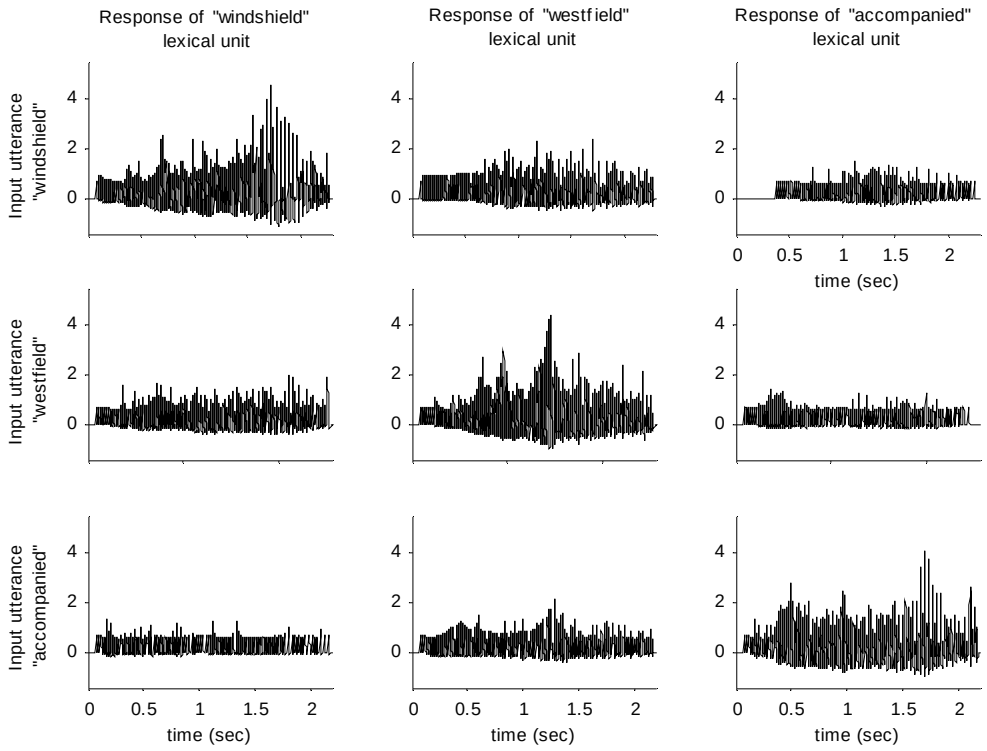

Figure 18 - Responses of 3 different lexical units to 3 different test inputs. Each column represents the response of a particular lexical unit to different inputs, while each row represents the response of different lexical units to a particular input. Recognition is simulated by taking the most active response in each row.

We can extend this example to include other detectors. In Figure 18, outputs are shown for 3 different lexical units, responding to 3 different test inputs. In each mini recognition experiment, corresponding to the rows of the figure, we can see that the appropriate lexical unit gives the strongest response.

4 Training from baseforms

In PHONEBOOK, the train and test sets have disjoint vocabularies. The immediate consequence of this fact is we do not have any examples from which to train the lexical units. In this type of situation, we can not use the landmark model event detection approach described in the previous section to train the lexical units. Instead, we need a way to generate training events from a word baseform or phonetic transcription. The comparison of phonetic transcription with event channel ordered by occurrence in Figure 12 gives us a clue as to how this might be accomplished. Although the relative timing between events gives a great deal of information about the word identity, much of this information is encoded in the ordering of these events. Thus, we can get a useful *approximation* to the event times using just the order of events taken from the space of phonetic transcriptions produced by the baseform of the lexical unit.

The procedure for deriving landmark event times to train a lexical unit from its baseform is shown in Figure 19. During training, we obtain all possible phonetic transcriptions for a lexical unit by transforming its baseform. The phonetic transcriptions can then be converted into a set of landmark model identities, corresponding to event channels. The ordering of these events can be used to derive uniform times. During testing, we get event times from the probability tracks, then strip the explicit timing info from the events to leave them with order-derived times.

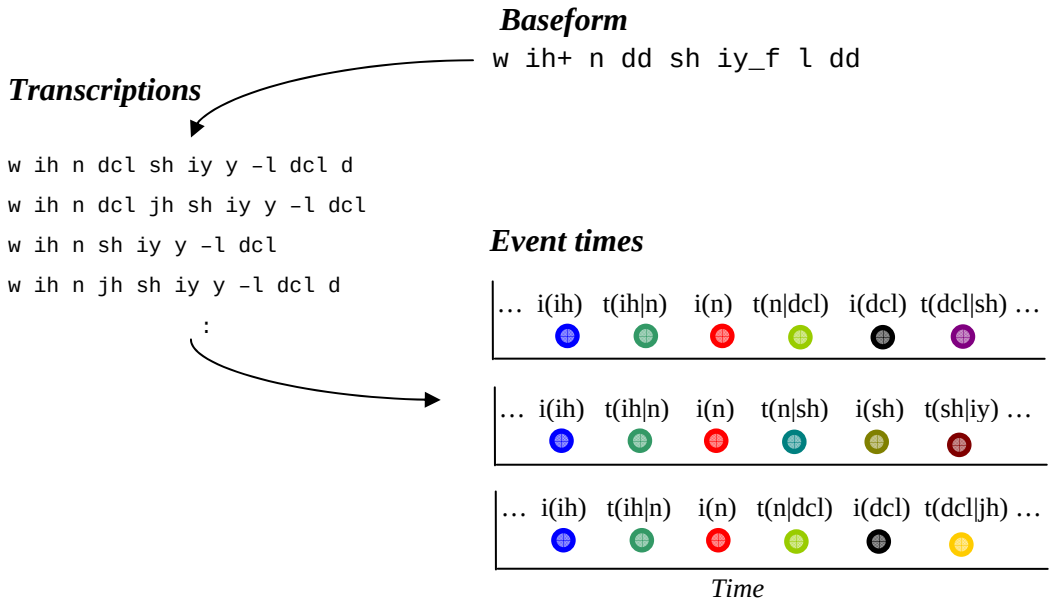

Figure 19 - Getting events from baseforms. Each unique phonetic transcription yields a unique set of ordered events. All event sets are used to train a single lexical unit detector.

4.1.1 Example

We can replicate the example described in Section 3.1.2 but using the train-from-baseforms approach to train the lexical unit detector. Figure 20 and Figure 21 duplicate the outputs shown in Figure 17 and Figure 18 of Section 3.1.2, respectively. Comparing

the event times between Figure 17 and Figure 20, we can see that the latter figure has roughly the same number of event occurances in each column, but has uniform timing between the events. This is a consequence of removing relative timing information during event detection in order to match events derived from baseforms. Although the outputs of the lexical unit are smaller in magnitude than in Figure 17, the relative differences between the outputs are preserved. The same result can also be seen in Figure 21.

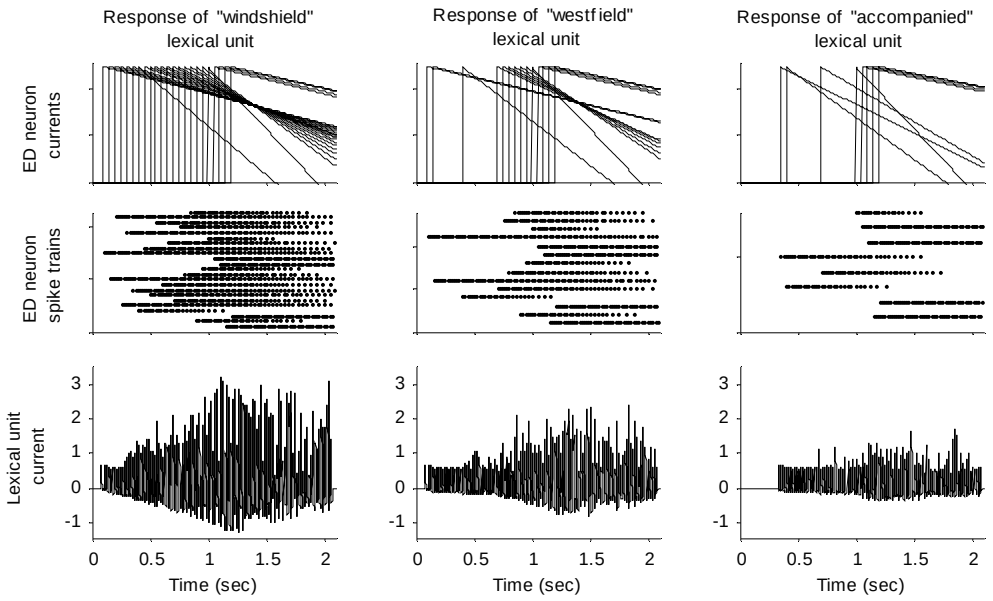

Figure 20 - Responses of 3 different trained lexical units to common test input, an utterance of the word "windshield" (tag aa74_f03) . Lexical units in each of the columns were trained from baseforms.

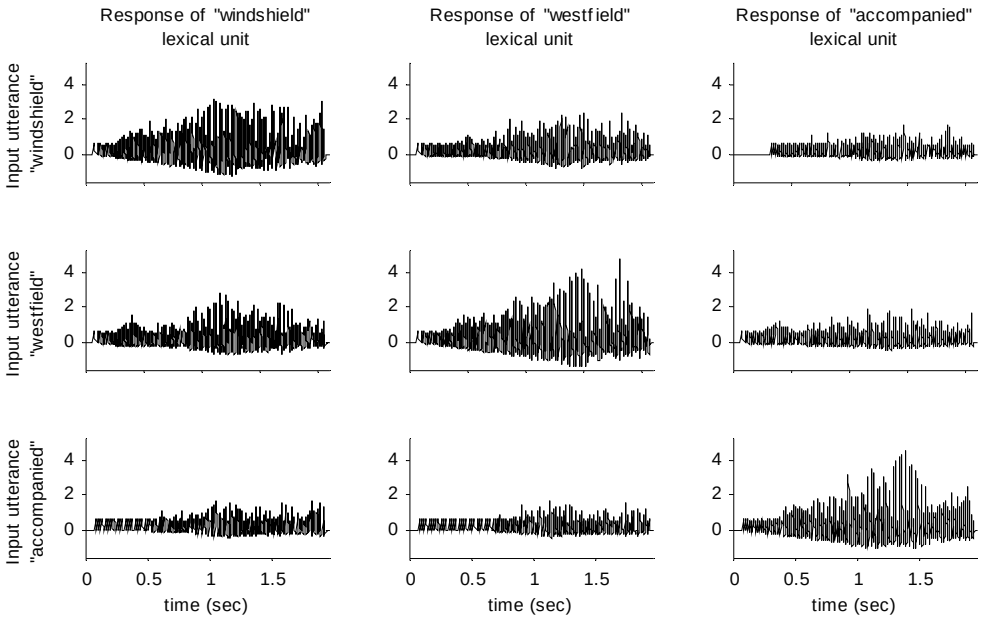

Figure 21 - Responses of 3 different lexical units to 3 different test inputs. Each column represents the response of a particular lexical unit to different inputs, while each row represents the response of different lexical units to a particular input.

4.2 Discussion

There are clear advantages and disadvantages to using the train-from-baseforms approach. One benefit is that we do not need to know any information besides the phonetic transcription in order to train a lexical unit. The glaring negative aspect of this approach is that all information about relative timing between phonetic events is lost, since the ordering is the only factor in determining event times. This is especially unfortunate since the ability to incorporate relative temporal information for template matching is one of the main benefits of the Hopfield approach.

One way to address this issue is to use the baseform training approach to "bootstrap" the unit detectors by providing an initial set of parameters, then to gather timing information during testing. Unlike a monolithic word graph which has parameters that have global implications, we expect that the lexical unit parameters can be easily updated online given confidence from higher level cues about the correctness of the hypothesized output. That is, once a word has been heard and confirmed, then the relative timing of the events can be used to update the ED neuron characteristics for that lexical unit detector. This proposed framework is both physiologically plausible and may be a compelling way to incorporate feedback into recognition.

5 Future

We have several immediate directions for future work.

1) First, in order to evaluate the full potential of this approach, we plan to use the techniques described in previous sections to perform a recognition experiment on the Phonebook corpus. To reduce computation, we plan to generate a cohort set for each test utterance using a traditional recognizer to limit the number of active lexical units.

2) Using negative events to inhibit lexical unit output. So far we have relied on using positive examples of events to affirm the identity of a lexical unit. However, it is also useful to use events that provide strong *negative* evidence against a lexical unit. For example, if we look at the output for the lexical unit "westfield", a prominent peak in the 'i(n)' channel should significantly inhibit that lexical unit from firing.

3) Use phonetic features to perform event detection.

4) As we mentioned at the end of the previous section, it is easier to update the parameters for individual lexical unit detector than for a full word graph. This offers the plausible way to perform online, unsupervised learning.