Continuous Phone Recognition in the Sigma Cognitive Architecture

Himanshu Joshi^{1,2}, Paul S. Rosenbloom^{1,2} and Volkan Ustun¹

Abstract

Spoken language processing is an important capability of human intelligence that has hitherto been unexplored by cognitive architectures. This reflects on both the symbolic and sub-symbolic nature of the speech problem, and the capabilities provided by cognitive architectures to model the latter and its rich interplay with the former. Sigma has been designed to leverage the state-of-the-art hybrid (discrete + continuous) mixed (symbolic + probabilistic) capability of graphical models to provide in a uniform non-modular fashion effective forms of, and integration across, both cognitive and sub-cognitive behavior. In this article, previous work on speaker dependent isolated word recognition has been extended to demonstrate Sigma's feasibility to process a stream of fluent audio and recognize phones, in an online and incremental manner with speaker independence. Phone recognition is an important step in integrating spoken language processing into Sigma. This work also extends the acoustic front-end used in the previous work in service of speaker independence. All of the knowledge used in phone recognition was added supraarchitecturally – i.e. on top of the architecture – without requiring the addition of new mechanisms to the architecture.

Keywords: Cognitive architecture, graphical models, Sigma, speech recognition, factor graphs, Dynamic Bayesian Networks, HMM

1 Introduction

Cognitive architectures model the fixed structures underlying human intelligence. They are an important approach towards realizing the original goal of AI, a working implementation of a complete cognitive system in aid of creating synthetic agents with human capabilities (Langley, Laird, & Rogers, 2009). The Sigma cognitive architecture is under development at the University of Southern California's Institute for Creative Technologies (ICT) to support the real time needs of intelligent agents – Virtual Humans – that are integrated simulations of human bodies, minds and behaviors in virtual environments, focusing on human-like interactions with real humans. Such virtual humans require a real-time combination of a broad set of human-level capabilities, from central cognitive (thought) processes to peripheral sub-cognitive (perceptual and motor) processes (Campbell, et al., 2011). Sigma predicates its ability to support Virtual Humans on the fact that its development is guided by a quartet of desiderata (Rosenbloom, Demski, & Ustun, 2016). Two of the four desiderata derive directly from this requirement of real-time combination of broad capabilities: (1) grand unification, combining both traditional cognitive capabilities and sub-cognitive capabilities such as

1
2
3
4
5
6
7 motor control, vision, and speech; and (2) sufficient efficiency, executing quickly enough for
8 anticipated applications. The third desideratum – (3) functional elegance, generating the broad
9 (cognitive and sub-cognitive) functionality necessary for human-level intelligence from a simple and
10 theoretically elegant base – brings a principle of parsimony into the design of Sigma that encourages
11 uniformity in how disparate capabilities are implemented and thus also potentially enables tight
12 coupling among them. Finally, the remaining desideratum – (4) generic cognition, aims to integrate
13 ideas from both cognitive science and artificial intelligence. This work provides an important step
14 towards grand unification, functional elegance and generic cognition. It does however at the same
15 time raise serious issues concerning sufficient efficiency that will be discussed at the end.

16 This article builds on previous work on speaker dependent isolated word recognition in Sigma
17 (Joshi, Rosenbloom, & Ustun, 2014). While the focus there was demonstrating a simple form of
18 speech processing, and understanding the extent to which the architecture can contribute towards
19 achieving it, the focus here is on taking the first step towards integrating a speech capability that can
20 process a more natural form of continuous, speaker independent speech signal by segmenting it into a
21 stream of recognized phones. The manner in which this is achieved is both constrained by key aspects
22 of what is known about human speech processing – in particular, the online, incremental nature of it –
23 and uniform with how cognitive processes are implemented within the architecture.

24 Speech processing involves a combination of challenges – the high dimensional nature of sub-
25 symbolic input, a rich high-bandwidth interplay of sub-symbolic and symbolic aspects of incremental
26 language processing, the task and the goal oriented nature of human conversations etc. – that have
27 made it a difficult capability to integrate into cognitive architectures. Traditional symbolic cognitive
28 architectures, such as Soar (Laird, 2012), interface to sub-cognitive modules outside of the core
29 architecture for perceptual processing (Laird, Kinkade, Mohan, & Xu, 2012; Laird, 2012). In contrast,
30 Sigma’s mixed (symbolic + probabilistic) and hybrid (discrete + continuous) core – based on graphical
31 models (Koller & Friedman, 2009) – has already been shown capable of a uniform non-modular
32 integration of symbolic decision making with probabilistic perception in the form of conditional
33 random fields (CRFs) and simultaneous localization and mapping (SLAM) (Chen, et al., 2011). More
34 broadly, a variety of forms of memory, learning and decision-making have been demonstrated in
35 Sigma, along with forms of perception and mental imagery (Rosenbloom, Demski, & Ustun, 2016).
36 The hypothesis underlying this work – and more broadly, the speech processing work in Sigma – is
37 that Sigma is sufficiently capable of implementing speech and providing a tight coupling of it and
38 language processing with cognition in a non-modular fashion on top of the architecture without adding
39 any language or speech specific capabilities to the architecture. As a step towards realizing this vision,
40 this article presents results from a continuous phone recognition task that demonstrates Sigma’s ability
41 to segment a stream of continuous audio into constituent phones, and to recognize those phones, in a
42 non-modular supraarchitectural fashion.

43 Sigma achieves phone segmentation and recognition by leveraging the graphical architecture
44 hypothesis – that the key to making progress on Sigma’s desiderata entails a blending of lessons learnt
45 from the history of work on both cognitive architectures and graphical models (Rosenbloom, Demski,
46 & Ustun, 2016). Although not directly biologically inspired, graphical models share many of the
47 attributes of neural networks and a number of neural network algorithms map directly onto them
48 (Jordan & Sejnowski, 2001). Graphical models have the added advantage of providing a simple path
49 for implementing symbolic processing and integrating it with sub-symbolic processing (Rosenbloom,
50 2012b; Rosenbloom, Demski, & Ustun, 2016). Sigma uses factor graphs as its graphical model of
51 choice, with a variant of the summary product algorithm as its solution method (Kschischang, Frey, &
52 Loeliger, 2001). Factor graphs are bipartite, undirected, potentially cyclic, graphical models that are
53 more expressive than Bayesian and Markov networks. They enable grand unification via a mixed
54 hybrid cognitive language, functional elegance through a small set of general mechanisms with broad
55 applicability, and sufficient efficiency and generic cognition through the range of state-of-the-art and
56 cognitively-inspired algorithms that map onto them. The summary product algorithm can be used to
57
58
59
60
61
62
63
64
65

compute marginals over all of the variables in the graph, when summarization occurs via sum/integral, or the maximum a posteriori (MAP) distribution, when summarization occurs via max.

Sigma’s basis in graphical models enables the core of the phone recognition task – as modeled by hidden Markov models (HMMs) that process low-level spectral speech features derived from an acoustic front-end – to be implemented via the same kinds of knowledge structures it uses for cognitive processing. Furthermore, a portion of the structure of these HMMs can itself be generated automatically by diachronic processing mechanisms previously introduced into Sigma in support of reinforcement learning (RL) (Rosenbloom, 2012a). These results provide an important step towards a uniform, non-modular, speech capability in Sigma that can be coupled tightly with both language and cognition. In the process, these results also provide significant contributions towards both grand unification and functional elegance by yielding these subsymbolic capabilities without additional architectural modules or extensions.

The next section covers background material on the phone recognition task and its role in speech processing within a cognitive architecture. Sections 3 and 4 introduce the most relevant aspects of the Sigma architecture along with how the task at hand maps onto Sigma, together with a discussion of how Sigma automatically generates a portion of the graph used in this task. Section 4 then discusses the dataset that was used, the experiments conducted, and the accompanying results. Finally, the article is wrapped up in Section 5 with a summary and possible future steps.

2 Continuous Phone Recognition and Speech Processing

Speech is a non-stationary signal produced by the human articulatory apparatus that is assumed to consist of a short-term and a long-term component (Rabiner, 1989). The short-term component is of the order of a few tens of milliseconds (ms) and characterizes the development of individual sounds. The long-term component is of the order of a few hundred milliseconds and characterizes the development of sequences of sounds (Rabiner, 1989). Following the noisy channel model, the audible speech signal is assumed to stem from speech codes that cannot be directly observed. These speech codes can be individual sounds – i.e. *phones*¹ – or contextualized sounds, or even words. Small-vocabulary systems typically work at the level of individual words, whereas medium- or large-vocabulary systems model phones or contextualized phones. Phones model individual sounds and their study – phonology – is based on the assumption that spoken words are composed of smaller units of speech (Jurafsky & Martin, 2009). Choosing to model phones as the acoustic unit of choice allows speech recognizers to use available data effectively and recognize words not present in the training data. From a cognitive linguistics perspective, the parallel architecture hypothesis (Jackendoff, 2007) considers phonology to be an independent generative component of language, alongside structure (syntax) and meaning (semantics). This work focuses on recognizing a stream of spoken audio as a sequence of phones, using a generative model of phones. It serves as a first step towards integrating a full-fledged generative phonology component.

¹ Phonology distinguishes between a phoneme – an idealized sound – and a phone, the physical realization of a sound (Jurafsky & Martin, 2008). In this work, phones are of most direct relevance.

The dominant paradigm in speech recognition involves the use of the generative framework of hidden Markov models (HMMs), with one HMM per speech code, a phone in this case. An HMM consists of a Markov chain that models the evolution of the speech code over time while producing acoustic observations. Each phone is considered to be a finite state automaton (FSA) (Figure 1), as modeled by the states of the HMM. As each phone automaton progresses, it passes through the beginning, middle and finally the end of the phone while producing the acoustic observations. These observations are what are perceived – i.e. heard – by the human ear, whereas the part of the phone (beginning, middle or end) producing these observations is not observed directly but can be inferred from the acoustic observations. The acoustic observations are obtained via digital signal processing (DSP) that is designed to model the function of the human ear. Typical examples include Linear Predictive Coding (Gold, Morgan, & Ellis, 2011), Mel Frequency Cepstral Coefficients (MFCC) (Young, et al., 1997) and observations obtained from neural networks (Hinton, et al., 2012). The relationship between the states of the HMM and the acoustic observations is governed by the *perception* function. The progression of the state sequence, i.e. which state can follow the current state, is governed by the *transition* function. Together the perception and transition functions combine to yield the joint

Figure 1: A finite state automaton model of a phone. The states correspond to the beginning, middle and end of a phone. Each state can stochastically transition to itself or the next state, to account for the variable duration of phones.

Figure 2: A sequence of two phones. The continuous phone recognition task consists of identifying an arbitrarily long phone sequence given a spoken utterance. Typically, the conditional probability of the next phone given the previous phone – i.e. the phone bigram – is used to approximate the joint probability of the sequence of phones. The transition leaving state S_2 and entering state S_0 is an inter-phone transition while the rest are intra-phone transitions.

probability of the sequence of states and acoustic observations. The prior function of an HMM specifies the initial distribution over the states of the HMM. Typically, each phone starts in the beginning, i.e. state S_0 , but the HMM construct allows for more complex phone models. When a phone ends, the phone-to-phone transition – or the *phone bigram* – function (Figure 2) governs the distribution over the phones that can follow it. Each spoken utterance, modeled as a sequence of HMMs, can itself be considered a higher order HMM, where the intra-HMM transitions are ignored and only the inter-HMM transitions – i.e. those that happen from state S_2 of one phone to state S_0 of the next phone (Figure 2) – are recorded to reconstruct the utterance. Whereas previous work on isolated word recognition in Sigma used one HMM per utterance, with only one word per utterance, this work focuses on a more natural form of continuous utterance, where the number and order of the phones is not pre-determined. The only restriction here is that each utterance must end in the final state of some HMM, i.e. S_2 for the HMM. Section 4 describes how this higher order HMM, obtained via

composition of individual phone HMMs, is specified in the form of a finite state acceptor (FSA) and then mapped onto a factor graph.

Inference in HMMs is traditionally carried out via the Viterbi algorithm (Rabiner, 1989) to yield the most likely sequence of states given a sequence of observations. For example, the joint likelihood of the observation sequence and the state sequence in an HMM, for three time slices, as shown in Figure 3, is given by $P(S, O|M) = a_{12}b_2(o_1) \cdot a_{22}b_2(o_2) \cdot a_{23}b_3(o_3)$ where a_{ij} is the conditional transition probability of the next state being j given the previous state was i and $b_j(o_t)$ is the probability of observing the symbol o_t in state j for time slice t . The state variable S is summarized out using the max operation to obtain the most likely sequence of states that generated the observation sequence:

$$P(O_{1..T}|M) = \max_S \left\{ a_{S_0 S_1} \prod_{t=1}^T a_{S_t S_{t+1}} b_{S_t}(o_t) \right\} \quad (1)$$

The HMM in Figure 3 is shown as the equivalent factor graph in Figure 4. Unlike the HMM, the factor graph is undirected and the transition function (T), acoustic perception function (A) and prior state distribution (Pr) are moved into explicit factor nodes. While Figure 4 shows the factor graph equivalent for one HMM, section 4 discusses how this structure is extended to both represent all the HMMs corresponding to different phones and segment the observation sequence into a corresponding sequence of phones in an incremental and online fashion.

Figure 3: Hidden Markov model (HMM) as a dynamic Bayesian network for 3 time slices. S: states, O: Observations.

3 Sigma Cognitive System and Architecture

The Sigma cognitive system and architecture is best understood in layers, in analogy with a computer system. Figure 5 shows an operationalization of the graphical architecture hypothesis in the form of layers,

with the cognitive architecture being implemented by the graphical architecture. The work here then demonstrates how phone recognition can be achieved via addition of knowledge

Figure 4: An HMM shown as a factor graph for three time slices. Pr: prior distribution over states, T: state transition function, A: acoustic perception function, S_i : distribution over states S in time slice i and O_t : observation symbol for time slice t .

structures supraarchitecturally, i.e. at the level of knowledge and skills above the cognitive architecture. This ability to integrate new such capabilities on top of the architecture is a demonstration of the desideratum of functional elegance in action.

The specification of knowledge required for this capability – and indeed all supraarchitectural capabilities – uses Sigma’s cognitive language, as supported by the cognitive architecture layer. The cognitive architecture defines a *statistical relational language* of *predicates*, *functions* and *conditionals* (Rosenbloom, Demski, & Ustun, 2016). Predicates provide relational data structures for cognitive processing, with each inducing a region in Sigma’s working memory (WM). Perceptual predicates, such as those used here for speech processing, also induce a region in its perceptual buffer. Predicates can be open world or closed world. Open world predicates default to a value of 1 and require calculation of their WM every cognitive cycle whereas closed world predicates default to a value of 0 and latch their contents across cognitive cycles. Section 4 shows how the continuous phone recognition task is implemented with a mix of closed and open world predicates.

Figure 5: The Sigma cognitive system in analogy to a computer system. Each layer in the system supports the one above. The work described here is at the level of knowledge and skills, specified using the language provided at the cognitive architecture level. Adapted from (Rosenbloom, Demski, & Ustun,

Predicates join together *typed arguments* representing objects, entities and concepts. Types can be discrete – whether symbolic or probabilistic – or continuous to enable specifying richly structured representations that are both mixed (symbolic + probabilistic) and hybrid (discrete + continuous). For example, the acoustic observation – a spectral label in this work – can be declared as a discrete numeric type – `observation(value:[0:256])` – for 256 possible spectral labels. The states of the HMM, corresponding to the states of the FSA, can be represented symbolically as `fsa_state(value:[s0 s1 s2])`. Similarly, the vocabulary of the phone recognizer can be represented as `phone(value:[aa ah ax ... sil zh])`. Predicates can optionally specify functions that govern relationships among their arguments.

Relationships between predicates are specified via conditionals. Conditionals represent generalized knowledge fragments in Sigma’s long-term memory (LTM) by blending lessons from rule based systems and probabilistic systems (Rosenbloom, Demski, & Ustun, 2016). They are built from predicate patterns – *conditions*, *actions* and *conducts* – plus optional functions. Conditions and actions are analogous in their behavior to their namesakes in rules and provide the forward momentum characteristic of procedural memory. Conditions match their arguments against evidence in working memory, whereas actions, based on their arguments, propose changes to it. An example usage of a condition-action pair to specify a rule is shown in Figure 6. This conditional creates a knowledge

fragment in Sigma's LTM. The rule specifies that whenever the current phone is in state s_0 and the previous phone is in state s_2 , the previous phone is extracted. Conducts allow bidirectional flow of information, both matching to working memory and proposing changes to it. Section 4 (Figure 15) shows an example of how conducts are important for the phone recognition task by allowing

```

CONDITIONAL Extract_Phone
  Conditions: phone_select_prev(phone: $p_1$  fsa_state: $s_2$ )
             phone_select(phone: $p_2$  fsa_state: $s_0$ )
  Actions: phone_extracted(phone: $p_1$ )

```

Figure 6: An LTM fragment that matches a pair of conditions in WM and extracts the phone in the `phone_extracted` predicate when the conditions match. The conditions specify that the state s_2 of a phone is selected in `phone_select_prev` and that state s_0 is selected in `phone_select` predicate. Note: variables – such as p_1 and p_2 – are italicized whereas constants – s_0 and s_2 – are shown in regular font.

segmentation of the continuous audio stream without needing complex logic.

Functions specify relationships among variables and can be included as part of predicate definitions. They can represent joint or conditional distributions or may not be probabilistic at all, instead specifying arbitrary multivariate relationships. In the case of probability distributions, they are restricted to the range [0,1], and for purely symbolic functions they are limited to 0 (*false*) and 1 (*true*). An example conditional with a perception function is shown in Figure 7, where a predicate is used to specify the perception function. This function specifies the conditional probability distribution $P(\text{Observation}|\text{Phone})$.

```

CONDITIONAL Acoustic_Perception
  Conducts: observation(spectral_label: $\underline{o}_t$ )
            phone_state(fsa_state: $s_t$ )
            acoustic_perception(spectral_label: $\underline{o}_t$  fsa_state: $s_t$ )

```

Figure 7: An LTM fragment to combine perception from the `observation` predicate with state information processed via the perception function from the `acoustic_perception` predicate. The perception function is the conditional distribution of the spectral label given the state. The observation $\underline{o}_t$ is underlined to indicate it is the conditioned (child) variable.

All the knowledge, as specified by the predicates, functions and conditionals is compiled into a single graph. Processing on the graph is performed by the Sigma cognitive cycle that is designed to correspond to the 50ms cognitive cycle in humans (Rosenbloom, 2009). Figure 8 shows this cycle and the operations it performs, as broken down into phases. Cognitive cycles begin and end with input and output phases that provide the basic transduction necessary to interface to the outside world. In the context of this work, the input phase performs the pre-processing necessary to obtain the spectral labels that correspond to the acoustic signal, while nothing occurs in the output phase. The two major phases of the cognitive cycle are the *elaboration* phase – which consists of memory access, perceptual processing and reasoning – and the *adaptation* phase, which among other things modifies working memory and does learning of functions. The elaboration phase, via message passing, combines the information available in the perceptual buffer – as provided from the input phase to the perceptual

Figure 8: The four phases of the Sigma cognitive cycle. *Input* and *Output* are minor phases, for the basic transduction necessary to interface with the outside world. The *Elaboration* phase consists of message passing until quiescence and the *adaptation* phase proposes changes based on the just concluded elaboration phase. Adapted from (Rosenbloom, Demski, & Ustun, 2016).

predicates, once per cognitive cycle – with state information in WM from the previous cognitive cycle and knowledge encoded as conditionals in LTM to reason about whether a phone was recognized this cognitive cycle. Once no new messages are available to send, decisions can be made about changes proposed to WM. For example, the most likely state across all phones – after the current perception is combined with state information – can be selected as part of the adaptation phase. A conditional that performs such selection is shown in Figure 9, with the information thus obtained from this selection then available for further inference during the elaboration phase of the next cognitive cycle. Thus, the rule shown in Figure 6, which tests the `phone_select` predicate, is matched based on the results of the previous cognitive cycle. The result of this rule, whether a new phone is recognized, is immediately available in the `phone_extracted` predicate because it involves simple condition matching with no selection.

4 Continuous Phone Recognition in Sigma

```

CONDITIONAL Select_Best_Phone_State
Conditions: phone_state(phone:p fsa_state:s)
Actions:   phone_select(phone:p fsa_state:s)

```

Figure 9: A rule that selects the most likely phone-state pair from the `phone_state` predicate into the `phone_select` predicate.

towards building a more natural form of spoken language understanding capability. This task has aspects of hierarchical sequential streaming classification that are representative of language processing in general, but on a scale that should allow for a gradual march towards realizing a full language capability in Sigma. This section discusses the mathematical formulation of the phone recognition task followed by a discussion summarizing various architectures with different tradeoffs that are used to perform this task. Additionally, the constraints imposed by what is understood from human language processing are discussed, along with how they affect both the architecture and the performance of the agent.

As noted previously, each phone is modeled as a three-state HMM, with the states of the HMM corresponding to the beginning, middle and end of the phone. However, a choice must still be made regarding the representation of the acoustic signal. The source audio is treated as a sequence of acoustic signals, with each chunk of acoustic signal converted to a corresponding block of parameter vectors that the recognizer can use as spectral features. This process is meant to model cochlear processing of the speech signal along with cortical feature extraction.

The source acoustic signal is first processed using a traditional DSP front-end

The phone recognition task was chosen both to extend the previous work in speech recognition in Sigma and to take the next step

Figure 10: A naïve Bayes factor graph for a multidimensional observation vector. One acoustic perception function is needed per dimension. The information from all dimensions is combined multiplicatively. The prior on the state is not shown but it is obtained via the combination of the state information from the previous time slice and the state transition function.

and converted into the frequency domain after splitting the signal into successive overlapping windows. The amount of source signal used is 25ms and there is a 15ms overlap between successive windows, thereby generating a new data frame for every 10ms of the source signal. This frequency-domain signal is then filtered using a bank of twelve mel-filters to obtain a vector of twelve filtered signal values. The scale and center frequencies of the filters are designed to correspond to the processing in the human cochlea (Denes & Pinson, 1993). The output – the *mel-spectrum* – is then brought back into the time domain to obtain better phonetic discrimination. This vector of processed audio is the *mel-frequency cepstral coefficient* (MFCC) feature. Each such vector of MFCC values is one feature and is converted to a spectral label (Gray, 1984) via a clustering algorithm from the Hidden Markov Model Toolkit’s (HTK) HQuant tool (Young, et al., 1997).

Such a spectral label suffices to yield good performance in speaker dependent isolated word recognition, but may face severe performance degradation in the face of speaker independent, phonetically varied data, as is the case for the current task. To mitigate this performance degradation, the original twelve-dimension MFCC vector is augmented with delta and double-delta feature vectors, where the delta vector represents the change in the MFCC from the previous frame to the current frame and the double-delta vector represents the change in MFCC from two frames ago. This yields a 36-dimension vector for each 10ms acoustic frame. The energy from each vector is also appended, yielding a 39-dimension feature vector that consists of four feature types: the MFCC, the delta MFCC, the double-delta MFCC, and the corresponding energy in each of these vectors. Each feature type can be independently clustered to obtain a spectral label using HQuant. This yields a four-dimensional vector of spectral labels for each 10ms acoustic frame for use in the continuous phone recognition task. The four dimensions of this feature vector are combined multiplicatively, in a naïve-Bayes fashion as shown in Figure 10, to yield a perceptual distribution over the states of the HMM. Combining information across multiple dimensions in this fashion makes the assumption that each dimension is independent of the other and corresponds to using a diagonal covariance matrix in a traditional acoustic front-end based on Gaussian mixture models (Young, et al., 1997).

The phone recognition problem can now be specified mathematically: given a sequence of observations $\{o_1, o_2 \dots o_T\}$, obtain a sequence of M phone labels $\{ph_1, ph_2 \dots ph_M\}$. Each observation o_t is a vector of four-dimensional features. Note, the length of the observation sequence and the length of the phone sequence can be different and the length of each utterance varies arbitrarily. Given a set of HMMs, one for each phone from the alphabet P , and the set of all possible strings of phones, P^* , the phone recognizer can be conceptualized using Bayes’ theorem as the sequence of phones $\overline{Ph}$, that maximizes the product of two factors:

$$\overline{Ph} = \underset{\overline{Ph} \in P^*}{\operatorname{argmax}} P(\overline{O}|\overline{Ph}) \cdot P(\overline{Ph}) \quad (2)$$

The factor $P(\overline{O}|\overline{Ph})$ is the probability of the observation string $\overline{O}$ – a sequence of four-dimensional spectral labels – being generated by the sequence $\overline{Ph}$ of phones. This is obtained from the perception function of the HMMs. As discussed above, the perception function used here is a naïve-Bayes classifier that uses the four dimensions of the feature vector for each 10ms frame to yield a distribution over the states of the HMM. The Sigma conditional in Figure 7 showed how the front-end is programmed when the observation is one-dimensional. Extending the front-end to handle four-dimensional input involves specifying four such conditionals, one per dimension of the observation. In this fashion, the acoustic front-end can be set up for one time slice, where the acoustic signal is processed using HMM models obtained via HTK’s training to generate a distribution over the states of phones.

The second factor $P(\overline{Ph})$ represents the prior joint probability of the string of phones and is commonly referred to as the language model. For a phone string $P(\overline{Ph})$ of length M , the prior is given by $P(\overline{Ph}) = P(ph_{1 \dots M}) = P(ph_1 \wedge ph_2 \wedge \dots \wedge ph_{M-1} \wedge ph_M)$. This is an M -dimensional function for a string of length of M . However, this function is too unwieldy to compute or specify for arbitrarily

long phone sequences. To alleviate this problem, a phone bigram function – a conditional probability function that assigns a probability to the next phone given the previous phone – is used. When multiplied together, these conditional probabilities and the prior on the first phone approximate $P(Ph_{1...M})$:

$$P(Ph_{1...M}) \approx P(Ph_1) \cdot \prod_{i=2}^M P(Ph_i|Ph_{i-1}) \quad (3)$$

The $P(Ph_i|Ph_{i-1})$ term specifies the conditional probability of observing a particular phone, after having observed the previous phone. This reduces the size of the factor to $|P|^2$ where $|P|$ is the number of phones in the phone alphabet. Because unseen bigrams will have probability of 0, smoothing must be applied to generalize this function. This ensures there is a potential path between each phone pair, albeit at a lower probability in the presence of unseen phone bigrams. An example of a language of phones is shown in Figure 11, in the form of a finite state acceptor (FSA) (Mohri & Riley, 2008; Jurafsky & Martin, 2009). The nodes represent the phones in the alphabet of P and the arcs represent the probability of a transition from one phone to the next. Each of the states of the language defined in the FSA shown in Figure 11 is modeled individually as an HMM. This language model and subsequent smoothing are both generated by HTK’s HBuild tool (Young, et al., 1997), from the utterance transcripts provided with the dataset used in this work.

Now that the nature of the acoustic front-end along with the nature of the phone language that must

Figure 11: A Moore style finite state acceptor (FSA) with special start and end states. The $s/$ and $/s$ states are both mapped to the sil phone in the work here. The probabilities on the arcs are the conditional probabilities of the possible next phones occurring given the current phone. For the sake of clarity only two phones (aa and zh) are shown.

be recognized is specified together with HMM models for individual phones, the task of phone recognition can be thought of as generating a transcript by following the most likely path through the state space obtained by composing the language FSA in terms of the states of individual HMMs and expanding it for the length of the utterance. This is the task of inference and is typically performed by the Viterbi algorithm. The standard formulation of the Viterbi algorithm requires both the availability of the entire utterance prior to inference and the existence of a data structure that records the most likely state that resulted in the current state, for each time slice. At the end of the utterance, once the most likely final state is determined, the data structure is traversed to obtain the most likely state sequence corresponding to the

observation string. This takes the form of a speech decoder augmented by control structures – referred to as ‘tokens’ in the speech literature (Young, Russell, & Thornton, 1989; Ran & Wang, 2008) – that record whenever the final state S_2 of a phone transitions to the state S_0 of another phone. Token passing is used in several speech recognition platforms, for example Kaldi (Povey, et al., 2011) and HTK (Young, et al., 1997). Each token accumulates a summary of the states it has visited in sequence. At the end of the utterance, the best token – corresponding to the most likely path – is selected and its history is parsed to obtain the desired transcript. However, it has been long known that human speech processing is online: humans incrementally segment an utterance into words and assign these words an

interpretation as they hear them (Marslen-Wilson & Welsh, 1978; Marslen-Wilson, 1987), although with delays of up to 250ms.

```

CONDITIONAL Combine_Prior_State
  Contacts: phone_state_current(phone:pc fsa_state:sc)
           phone_state_previous(phone:pp fsa_state:sp)
           state_trans_func(phone:pp fsa_state:sp phone:pc
fsa_state:sc)

```

Figure 12: An LTM fragment illustrating how the state information from the previous time slice, available in the `phone_state_previous` predicate, is processed via the transition function in the `state_trans_func` predicate and made available for the current time slice in the `phone_state_current` predicate. The current phone p_c and current state s_c are underlined to indicate they are the conditioned variables.

To achieve online, incremental segmentation and extraction, Sigma uses a summary of the past, in the form of a predicate that stores the previous state, plus a window of limited constraint from the future (as explained below) and a simple rule that detects transitions between phones. Because only a limited constraint from the future is used, the results are not as accurate as full Viterbi, where the whole utterance is processed before any transcript is obtained, but it is possible to come close, as will be seen in Section 5.

Figure 4 showed the factor graph for a single HMM, unrolled for three time slices, with a one-dimensional observation for the sake of simplicity. This graph can be modified, by parameterizing the transition and acoustic factors to handle multiple HMMs. The perception function now represents a concatenated perception function, obtained by merging the perception functions of all of the HMMs. The transition function can be obtained by first merging the intra-state transition functions for each HMM and then adding in the inter-HMM transitions by introducing a transition from the final state S_2 of each phone's HMM to the beginning state S_0 of each other phone, as shown in Figure 2. The transition into state S_0 from state S_2 of the previous phone is the inter-phone transition. The other transitions that do not involve leaving state S_2 or entering state S_0 are intra-phone state transitions. The likelihood of the inter-phone transition is obtained from the phone bigram function specified by equation (3). This requires the transition function be specified in terms of transitions from the previous phone and state to the current phone and state. The conditional that combines state information from the previous time slice after processing with the transition function is shown in Figure 12.

Figure 13: The factor graph for the HMM from Figure 4 as extended to account for all phones. This is accomplished by parameterizing the factor functions and variables with the phone (p). The messages needed for calculating the posterior at $S_{2,p}$ are also shown.

The modified factor graph, including the parameterized perception and transition functions, is shown in Figure 13, fully unrolled for three time slices. Additionally, all messages required to calculate the posterior at node $S_{2,p}$ are explicitly shown. The posterior distribution at each node j is based on the product of the forward message $\mu_{i \rightarrow j}$, the backward message $\mu_{k \rightarrow j}$ and the information from the acoustic perception factor $\mu_{o_j \rightarrow j}$. Once all forward and backward messages are exchanged after accounting for observations, the full posterior for each time slice is available in the nodes $S_{1,p}$ through $S_{3,p}$. In this setting, the most likely sequence, as obtained by the Viterbi algorithm, can then be obtained by selecting the most probable phone from the posteriors available at each state node after bidirectional message passing over the entire utterance. Next it is discussed how to process the utterance in an online fashion, without requiring the specify the full graph.

From Figure 13, it can be seen that as the utterance proceeds from left to right, the forward messages do not change, only the backward messages do, based on the new observations. This forward computation therefore can be rolled into a single time-slice structure that records a summary of the utterance heard thus far. The graph structure that achieves this is shown in Figure 14. Node F_p maintains the state of the utterance up to the previous time slice, acting as a prior for the current time slice. A new observation enters the graph via O_t to be combined with the existing state information; that is, the transitioned prior state information. This processing all happens as part of the elaboration phase. During the subsequent adaptation phase, this result is slid back into F_p via a diachronic architectural mechanism so as to act as the prior for the next time slice. This forward going ‘sliding window’ mechanism, including the copying at the end of the cognitive cycle, can be implemented by: declaring a closed world predicate – a signal to Sigma that unknown values in the predicate are assumed to be false (0) and that the contents of its segment of WM should be latched across cognitive cycles – turning on diachronic processing, and specifying the corresponding LTM fragments. The decision node E – although not needed to implement the sliding window mechanism – computes the argmax from equation (2), in a local fashion, using information from the previous and current time

Figure 14: Forward-processing template structure using a sliding window of one observation. The unrolled graph from Figure 13 is collapsed into one processing stage, with the prior (F_p) combined with the current observation (O_t) to obtain the posterior for the utterance up to time slice t , in node $F*N_p$. This posterior is slid back into F_p as part of the adaptation phase – shown by the dotted line – to act as the prior for the next time slice. Also shown is the decision node E that performs the selection of the best phone using state information, based on the rules in Figures 6 and 9.

slices. The rules that detect this phone transition and extract the completed phone into the Ph predicate were shown in Figures 6 and 9. This rule based mechanism performs selection in an online fashion, committing to a transcript based on the utterance processed thus far, without requiring the full utterance or the implied backwards pass of the Viterbi approach.

Keeping in mind that there is

a lag of about 250ms in segmentation during human parsing (Marslen-Wilson, 1987), a limited amount of constraint from the future can be used, in the form of backwards messages, for human-like processing. Figure 15 shows how the trellis from Figure 14 can be extended to account for information from a limited future. Using the template from Figure 14, the constraint from the past – as obtained via the F^*N_p predicate – is combined with a message from the acoustic perception factor and the backward messages from the future portion of the graph. The future portion of the graph consists of a pair of perception and state predicates for each time slice that generates the backward message from it. The number of time slices that are added determines how much future constraint is incorporated into the processing. The decision node E works as before, but uses state information from the current time slice ($S_{t,p}$) and the future ($S_{t+1,p}$). This is an extension of the sliding window described earlier, and works similarly except over a wider window. A new observation is introduced every cognitive cycle, in the final time slice of the future portion of the graph. This triggers a chain of backwards messages

Figure 15: The continuous phone recognizer graph using four time slices of constraint from the future: O_{t+1} to O_{t+4} (40ms). For the sake of clarity, only one dimension is shown for the acoustic perception factors at each time slice.

along the horizontal length of the graph, which is combined with the forward information to obtain an approximate posterior. The quality of the posterior obtained at the $S_{0,p}$ predicate is a function of the amount of constraint used from the future; that is, on how many time slices are explicitly represented in the graph. The next section evaluates the effect this has on the results, along with a benchmark against an implementation of the full Viterbi algorithm.

This overall approach to phone segmentation and recognition differs from the classical formulation of the Viterbi algorithm, where a complete backwards pass occurs once the entire utterance has been processed. The approach described here instead uses a limited constraint from the future combined with a summary of the past, and extracts the transcripts in a completely online manner.

5 Experimental Results

For these experiments, the TIMIT database (Garofolo, et al., 1990; Zue, Seneff, & Glass, 1990) was used, which contains a total of 6300 utterances amounting to ~5.4 hours of speech. Despite its age, this corpus is still widely used to test new speech recognition architectures and models, and has several desirable properties for this task: tractable size, phonetic variation and compositional aspects of hierarchical sequential classification tasks. The speech community has developed a standard procedure – a recipe – that was used to perform all the experiments. Using a standard database with a standard procedure allows ready comparisons with other approaches. To ensure the training and test sets don't contain sentences by the same speaker, the recipe uses 3696 sentences for training and 192 for testing.

In accordance with the standard way of using TIMIT (Lee & Hon, 1989), the set of 61 TIMIT phone labels are reduced to a smaller set of 48 phones for training. In addition to the 48 phones, the glottal stop ‘q’ is retained for training but ignored for testing, whereas it was removed entirely by (Lee & Hon, 1989).² Furthermore, for evaluation of decoded transcripts, Lee & Hon (1989) suggest reducing the number of phones from 48 to 39 by treating, for example, the voiced closure {vcl} the same as {sil}. While such a mapping is based on rules obtained from phonetics (Lopes & Perdigao, 2011), the purpose here was not to explore the effects of these rules but to explore the feasibility of Sigma to recognize in an online and incremental manner a continuous stream of audio into constituent phones.

All of the preprocessing of the audio and the training of the parameters for these experiments was performed by tools from HTK. First, preprocessing computed the MFCC features and generates the spectral label codebook (via HTK’s HQuant tool). The spectral codebook was then used to convert the TIMIT test and training sets into corresponding spectral labels using HTK’s HCopy tool. The parameters required for this task include the HMM transition and perception models and the bigram language model. The performance of the phone recognizer is determined by using the Levenshtein distance (Lopes & Perdigao, 2011) between the output transcript and the reference transcript. This metric accounts for the three kinds of discrepancies that can occur when comparing two sequences: deletion, where a phone is present in the reference transcript but not in the recognizer’s output; insertion, where a phone not present in the reference transcript is inserted by the recognizer in its output; and substitution, when the wrong phone is recognized by the recognizer. HTK’s HResult tool is used to compare Sigma’s output with the reference transcript to compute the performance metrics³.

Recognizer	Total reference labels (N)	Correctly identified (H)	Total deleted (D)	Total substituted (S)	Total inserted (I)	Accuracy (H-I/N) %
HTK Decoder using Viterbi	6785	3816	954	2015	387	50.54
Sigma incremental, no future information	6785	2850	2385	1550	99	40.55
Sigma incremental, 6 future time-slices	6785	3731	1067	1987	363	49.64
Sigma incremental, 12 future time-slices	6785	3782	1041	1962	315	51.10
Sigma incremental, 20 future time-slices	6785	3801	1039	1945	317	51.35

Table 1: Summary results from the phone recognition task. All the parameters were trained using the HTK tool suite. The top row shows results from HTK’s Viterbi decoder. The other rows show results from Sigma’s incremental decoding approach, using limited constraint from the future. N indicates the number of phone labels in the reference transcripts, whereas the other columns indicate numbers obtained in various configurations.

Experiments were run using the HTK decoder, based on the Viterbi algorithm, and in Sigma with varying amounts of future constraint, ranging from 0 time slices up to 20. The results can be seen in Tables 1 and 2. The top row of Table 1 shows baseline results using HTK’s Viterbi decoder. There are three main results. First, Sigma’s online and incremental recognition and segmentation of the acoustic signal into a corresponding sequence of phones improves monotonically with more future

² Removing a phone entirely seems to create blanks or holes in the transcript when using HTK’s HLED tool. This workaround is suggested by GMTK’s TIMIT recipe (Bilmes & Zweig, 2002).

³ All of the code, including the audio features and scoring scripts are available online at: https://bitbucket.org/hima_cogarch/sigma_phone_recognizer

constraint. This is apparent when considering the number of deletion errors goes down as the amount of future information used for decoding increases. For example, the number of deletion errors when no future information is used is high in row 3 but goes down when 60ms of future information is used as more phones are recognized in row 4. This increase in the number of phones recognized is reflected in the increased numbers of correctly identified phones as well as the substitution and insertion errors. However, both these kinds of errors go down in number, with a corresponding increase in the number of correctly identified phones, as more future information is used in rows 5 and 6. Second, with sufficient future constraint – but still bound by the amount of lag in human parsing (with 10ms per time-slice) – it is comparable in accuracy to the results from HTK’s Viterbi decoder that operates on the full utterance. Third, Sigma is far from sufficiently efficient for this task. Table 2 lists the average cognitive cycle time for each of the Sigma configurations from Table 1. As expected, the cognitive cycle time – which should remain limited to 50ms – increases with the number of time slices used, ranging from a factor of 5 too slow to nearly a factor of 400.

Configuration	Cognitive cycle time (ms)
Sigma incremental, no future information	250
Sigma incremental, 6 future time-slices	5811
Sigma incremental, 12 future time-slices	14119
Sigma incremental, 20 future time-slices	19110

Table 2: Average cognitive cycle times as a function of the amount of future constraint incorporated.

6 Conclusion and Future Steps

A speaker independent phone recognition task was implemented on top of the Sigma cognitive architecture. The top-level result is that the recognition is performed in an online and incremental manner, with a single forward pass through the utterance that uses limited constraint from the future. This is in line with what is understood about spoken language processing in humans. All of the knowledge required to perform online phone recognition is specified supraarchitecturally. The accuracy of the resulting system is comparable to that of a narrow AI system that uses the same parameters but that assumes the entire utterance is available at the start of decoding; however, the amount of time required for this processing in Sigma is significantly more.

This work builds on earlier work by extending the acoustic perception function to handle a multi-dimensional representation of the acoustic signal and extending the transition function to represent both intra-phone and inter-phone transition parameters. The phone recognizer reuses the diachronic processing mechanism already available in Sigma to generate portions of the graph automatically. Experiments show that performance improves as the amount of information used from the future increases. Because the acoustic signal is introduced in the graph via perceptual predicates, and is combined – via LTM fragments in the form of conditionals – with the state of the agent within the same cognitive cycle that processes all other forms of knowledge in Sigma, this work is an important demonstration of functional elegance in action while taking a step towards grand unification. It does however fall significantly short on sufficient efficiency.

While previous work showed how Sigma’s cognitive cycle can process an acoustic signal to recognize one word (Joshi, Rosenbloom, & Ustun, 2014), this work shows how this same cognitive cycle can recognize and segment a stream of continuous audio signals in terms of its constituent phones. This hints at the ability of the cognitive cycle to blend cognitive and sub-cognitive processing – a broader form of unity and uniformity – as is necessary for a full implementation of spoken language processing.

The most obvious next step for this work is to extend the current model to perform word-based speech recognition; i.e., decode the string of words from the sequence of acoustic observations. This should yield an important step in the close coupling of speech and language, the higher ambition guiding the spoken language processing work in Sigma. The parameters used for phone recognition in this article were obtained from HTK using embedded Viterbi training, in a non-incremental manner. Exploring incremental learning of the parameters required for phone recognition is also therefore an interesting potential next step. Such learning was demonstrated for isolated-word recognition, and so can hopefully extend to this more complex case as well.

The perception function is currently a discrete factor, with the spectral label representing a cluster in the audio spectrum. Although the performance of the local online recognizer is comparable to that of the Viterbi decoder, as implemented by the HTK toolkit, the discrete nature of the acoustic signal here is a major bottleneck in further improving recognition performance. Better performance is expected using a mixed factor, where the spectral dimension is a continuous function such as that represented by a Gaussian mixture model. More recent speech systems use recurrent neural networks to model the acoustic perception function, yielding the best performance to date on TIMIT. Exploring mixed neural-probabilistic factors or discrete-continuous Gaussian mixture model based factors is an interesting future direction for this work. Additionally, exploring solutions to mitigate the sufficient efficiency issues is an important step towards achieving the overall vision of combining speech and language on top of the architecture in a non-modular fashion. This would potentially take the form of reducing the cognitive cycle time via parallel algorithms at the graphical layer.

Acknowledgements

This work was sponsored by the U.S. Army. Statements and opinions expressed may not reflect the position or policy of the United States Government, and no official endorsement should be inferred. The authors would like to thank Ruchir Travadi, of USC's SAIL laboratory, for insightful discussions during the course of this work.

References

- Bilmes, J., & Zweig, G. (2002). The graphical models toolkit: An open source software system for speech and time-series processing. *Acoustics, Speech, and Signal Processing (ICASSP), IEEE International Conference on*, 4, IV-3916.
- Campbell, J. C., Hays, M. J., Core, M., Birch, M., Bosack, M., & Clark, R. E. (2011). Interpersonal and leadership skills: using virtual humans to teach new officers. *In Proc. of Interservice/Industry Training, Simulation, and Education Conference*, 11358.
- Chen, J., Demski, A., Han, T., Morency, L.-P., Pynadath, D., Rafidi, N., & Rosenbloom, P. S. (2011). Fusing symbolic and decision-theoretic problem solving + perception in a graphical cognitive architecture. *Proceedings of the 2nd International Conference on Biologically Inspired Cognitive Architectures*, (pp. 64-72).
- Denes, P. B., & Pinson, E. (1993). *The speech chain: The physics and biology of spoken language*. Macmillan.
- Garofolo, J., Lamel, L., Fisher, W., Fiscus, J., Pallett, D., & Dahlgren, N. (1990). DARPA, TIMIT Acoustic-Phonetic Continuous Speech Corpus CD-ROM. *National Institute of Standards and Technology*.
- Gold, B., Morgan, N., & Ellis, D. (2011). *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. Hoboken, NJ: John Wiley & Sons.
- Gray, R. M. (1984). Vector quantization. *ASSP Magazine*, 1(2), 4-29.

- 1
- 2
- 3
- 4
- 5
- 6
- 7 Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen,
- 8 P., Sainath, T. & B., K. (2012). Deep neural networks for acoustic modeling in speech
- 9 recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*,
- 10 29(6), 82-97.
- 11 Jackendoff, R. (2007). A parallel architecture perspective on language processing. *Brain*
- 12 *Research*(1146), 2-22.
- 13 Jordan, M. I., & Sejnowski, T. J. (2001). *Graphical Models: Foundations of Neural Computation*.
- 14 Cambridge, MA: MIT Press.
- 15 Joshi, H., Rosenbloom, P. S., & Ustun, V. (2014). Isolated word recognition in the Sigma cognitive
- 16 architecture. *Biologically Inspired Cognitive Architectures*(10), 1-9.
- 17 Jurafsky, D., & Martin, J. (2009). *Speech and Language Processing: An Introduction to Natural*
- 18 *Language Processing, Speech Recognition, and Computational Linguistics*. Upper Saddle
- 19 River, NJ: Prentice Hall.
- 20 Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*.
- 21 Cambridge, MA: MIT Press.
- 22 Kschischang, F. R., Frey, B. J., & Loeliger, H. (2001). Factor graphs and the sum-product algorithm.
- 23 *Information Theory, IEEE Transactions on*, 47(2), 498-519.
- 24 Laird. (2012). *The Soar cognitive architecture*. Cambridge, MA: MIT Press.
- 25 Laird, J., Kinkade, K., Mohan, S., & Xu, J. (2012). Cognitive robotics using the soar cognitive
- 26 architecture. *Cognitive Robotics AAAI Technical Report, WS-12-06*.
- 27 Langley, P., Laird, J. E., & Rogers, S. (2009). Langley, Pat, John E. Laird, and Seth Rogers. *Cognitive*
- 28 *Systems Research*, 10(2), 141-160.
- 29 Lee, K., & Hon, H. (1989). Speaker-independent phone recognition using hidden Markov models.
- 30 *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(11), 1642-1648.
- 31 Lopes, C., & Perdigao, F. (2011). Phone recognition on the TIMIT database. Speech Technologies. In
- 32 I. I. (Ed), *Speech Technologies* (pp. 285-302). INTECH Open Access Publisher.
- 33 Marslen-Wilson, W. D. (1987). Functional parallelism in spoken word-recognition. *Cognition*, 25(2),
- 34 71-102.
- 35 Marslen-Wilson, W. D., & Welsh, A. (1978). Processing interactions and lexical access during word
- 36 recognition in continuous speech. *Cognitive Psychology*, 10(1), 29-63.
- 37 Mohri, M. P., & Riley, M. (2008). Speech recognition with weighted finite-state transducers. *Springer*
- 38 *Handbook of Speech Processing*, 559-584.
- 39 Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek,
- 40 P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., Vesely, K. (2011). The Kaldi speech
- 41 recognition toolkit. *IEEE 2011 workshop on automatic speech recognition and*
- 42 *understanding*.
- 43 Rabiner. (1989). A tutorial on hidden Markov models and selected applications in speech recognition.
- 44 *Proceedings of the IEEE*, 77(2), 257-286.
- 45 Ran, W., & Wang, R. (2008, February). A framework and token passing model for continuous speech
- 46 recognition with dynamic Bayesian networks. *Proceedings of the Fifth IASTED International*
- 47 *Conference on Signal Processing, Pattern Recognition and Applications*, 187-194.
- 48 Rosenbloom. (2009). A graphical rethinking of the cognitive inner loop. *Proceedings of the IJCAI*
- 49 *International Workshop on Graphical Structures for Knowledge Representation and*
- 50 *Reasoning*.
- 51 Rosenbloom. (2012a). Deconstructing reinforcement learning in Sigma. *Proceedings of the 5th*
- 52 *Conference on Artificial General Intelligence*, (pp. 262-271).
- 53 Rosenbloom. (2012b). Graphical models for integrated intelligent robot architectures. *Proceedings of*
- 54 *the AAAI Spring Symposium on Designing Intelligent Robots: Reintegrating AI*.
- 55 Rosenbloom. (2015). Supraarchitectural capability integration: From Soar to Sigma. *Proceedings of*
- 56 *the 13th International Conference on Cognitive Modeling*.
- 57
- 58
- 59
- 60
- 61
- 62
- 63
- 64
- 65

- 1
2
3
4
5
6
7 Rosenbloom, P. S., Demski, A., & Ustun, V. (2016). The Sigma cognitive architecture and system:
8 Towards functionally elegant grand unification. *Journal of Artificial General Intelligence*.
9 Accepted.
- 10 Young, S. J., Russell, N. H., & Thornton, J. H. (1989). Token passing: a simple conceptual model for
11 connected speech recognition systems. Cambridge: Cambridge University Engineering
12 Department.
- 13 Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., & Woodland, P. (1997). *The*
14 *HTK book (for HTK version 3.4)*. Cambridge: Cambridge University Engineering
15 Department.
- 16 Zue, V., Seneff, S., & Glass, J. 9.-3. (1990). Speech database development at MIT: TIMIT and
17 beyond. *Speech Communication*, 9(4), 351-356.
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65