Knowledge Representation and Speech Production/Perception Modelling in an Artificial Intelligence Environment

Mark Tatham
Katherine Morton

Published as: Tatham, M. and Morton, K. (1988) Knowledge representation and speech production/perception modelling in an artificial intelligence environment. In W. A. Ainsworth and J.N. Holmes (eds.) *Proceedings of Speech '88* (7th FASE Symposium), 1053-1060. Edinburgh: Institute of Acoustics

---

INTRODUCTION

The need to understand the relationship between cognitive and physical processes is most apparent at the phonetic and phonological levels of language production and perception. There is agreement that semantics and syntax call for cognitively oriented modelling, whereas the myodynamics, aerodynamics and acoustics of speech call for physically oriented modelling. Although it seems that the cognitive and physical perspectives are irreconcilable we do need to relate the two, especially when constructing models designed to simulate language as a whole.

For both cognitive and physical modelling it is appropriate to assume that we are dealing essentially with a knowledge-based system based on an adequate theory of language. We must distinguish the different types of knowledge within the system, and choose the right representational format for that knowledge. Furthermore, is the system to be given, once and for all, the entire knowledge base, or is it to somehow acquire the knowledge for itself - and if so how?

DIFFERENT TYPES OF KNOWLEDGE

Within the phonetic part of the system we can distinguish several types of knowledge. Firstly there is non-cognitive knowledge: the physical system knows to behave in a certain way: articulator response to motor control is predictable. Neural communication at a sub-cortical level ensures coordination between contracting muscles or between articulator movement, the detail of which is opaque to higher levels of the system: not only do individual components of the physical system behave predictably but much of their related behaviour is also predictable. To this extent low level sub-systems respond to control messages which do not themselves specify response procedures. The response is necessarily appropriate because its specification is itself a constraint on what higher level systems can expect. The more that behaviour is an intrinsic property of the low level systems, the more constrained the inventory available to high level systems.

There is thus a trade off: structures intrinsically organised at a low level are simple to control, but restricted in their possibilities - unorganized low level structures could do much more but require more complicated control organisation above. It seems to be the case that low level systems are mostly intrinsically organised (that is, hold their own procedural programs), requiring only simple control messages to activate them. However, limited setting of the parameters passed as part of the control message is possible. A simple, abstract analogous example:

Suppose an object comprising two numbers with the inbuilt function that they be multiplied together, and suppose this function returns its solution (the response). Further suppose that the solution is constrained to be within the range 5-7, and that one of the numbers is preset at 2. It follows that the other number is a variable adjustable within the range 2.5-3.5 - with a default of, say, 3. A simple instruction - GO - issued to this object will

return a response of 6 but a more complex Instruction - GO(2.5) - will return a response of 5. The object is said to have been tuned by the passing a value to set the variable parameter.

GO contains no procedural information for the receiving objects. To use the command the higher issuing level must have had some means of predicting its probable outcome. The high level system knows only that a certain range of behaviour is possible and what messages will invoke it: when it wants this behaviour it sends a suitable message. It has learned (constructed a model) what response to expect, and that, within limits, responses can be influenced: it does not know and need not know how all this happens.

The setting of parameters other than at their default values originates from cognitive processing. There is a decision to adjust the default response to messages. The decision needs two major lines of input: a message whose origin is phonological, and knowledge of the physical resources and how to use them. The message originates externally to this cognitive area of phonetics, whereas knowledge of the response range of the physical system is internal to cognitive phonetics. Extrinsic messages and intrinsic knowledge (or structured information) are brought together in the decision procedure. The output is messages to the physical structures.

I stress again that in the model described here the cognitive level does not know how the physical structures work: only that those structures respond in certain ways to particular messages, and that the responses can be adjusted by a tuning process which sets some parameters away from their default values without changing the basic response behaviour.

KNOWLEDGE ACQUISITION AND REPRESENTATION

As yet we are unable to specify precisely what the systems need to know in order to work properly. There are two reasons for this:

- we have insufficient data, and
- traditional theories of speech production are essentially descriptive in nature, and descriptive theories are not necessarily helpful to simulation model building.

Traditionally linguists and phoneticians use rules or productions for the representation of knowledge. There are arguments for and against the idea that the human system itself contains rules of this type - they may be just an external convenient device of the linguist. What the linguist does is set up descriptions of what is observed: the account itself is not observed. This descriptive modelling is quite different from building a simulation. A simulation is a working model - abstract or real - which must be explicit shout the internal structure of what is being emulated, rather than just account for its effects on its environment. Descriptive modelling does come into its own by being able to compare accounts of the behaviour of the real system and the simulation: if they converge then the simulation is evaluated as good.

A currently popular method of resolving the problem caused by insufficient data and an unsuitable linguistic theory is to create a device that learns the necessary knowledge for itself. Learning involves the acquisition and organisation of information by an active processing device.

Simulation therefore involves setting up a knowledge based system which should incorporate a shell capable of inputting information from the environment and organising it as part of a learning procedure. A simple form of the device might organise the data into patterns for spotting redundancy and repetition by inductive reasoning, and evolve sets of rules to express the patterning and generalisable features of the data. This part of the device would constitute the learning mechanism and the subsequent knowledge base. Organisation of the knowledge might be by means of an intrinsic inferencing process.

The commonest rule representation used in linguistic modelling of this kind is the production, if/then or condition/action rule. Inferencing is by means of forward chaining (whereby solutions are reached by accumulating considerations of new data) and/or backward chaining (in which hypotheses are set in advance and the data examined in terms of those hypotheses).

One form of knowledge representation uses a classifier system, often seen in the phonological and phonetic components of a linguistic description. A classifier represents the properties of an object in terms of arrays. Each cell in the array contains a value (usually binary) expressing the contribution of that feature to the identification of the target object as described by the complete array. The distinctive feature matrix in phonology, for example, is just such a set of classifiers. In phonetics we may well want the values in the cells to be more complex than simply binary. The binary alphabet though can be as useful to us as a multi-valued alphabet, by setting up dependent arrays associated with the feature specification of a single target object. Thus, one array may be used to filter various primary important features, another to drop to a more detailed representation of features pointed to by the first array, and so on, where the various levels of representation are formally defined.

In a simulation model of speech perception where some form of variability reduction is essential, one could imagine classifiers set up as masks to filter features which are always present in the data, often present, and sometimes present. A perception system would proceed through layers of masks to reduce variability and finally attempt a rule based template match based on a points scoring system and directed by means of some stored knowledge such as elementary phonotactic rules. A full classifier system has the means of evaluating the effectiveness of rules in the matching process and of weighting them accordingly.

An important feature of satisfactory knowledge representation is that the knowledge itself should be separate from its manipulation - procedural program control or inferencing processes should not be mixed with knowledge. In linguistics this was an important insight by Chomsky, though even today the production format for knowledge representation is sometimes erroneously taken as some procedural algorithm.

EXAMPLES OF PHONETIC KNOWLEDGE REPRESENTATION AND LEARNING

A. Data-driven learning

Four pieces of data are presented to the device and a generalisation is formulated about VOT (voice onset time). The data is expressed in Prolog-like notation, and is available for four stops: S1, S2, S3, S4.

```
stop(S1), excitation(S1, voiced), position(S1, initial), vot(S1,  0ms).
stop(S2), excitation(S2, v-less), position(S2, initial), vot(S2, 35ms).
stop(S3), excitation(S3, v-less), position(S3, initial), vot(S3, 45ms).
stop(S4), excitation(S4, v-less), position(S4, initial), vot(S4, 40ms).
```

The device learns by inbuilt rule the generalisations:

```
vot(s, X) :- stop(s), excitation(s, voiced), (X = 0).
vot(s, X) :- stop(s), excitation(s, v-less), (X >= 35).
```

B. A classifier system for learning generalisations

In this example the data given before can be taken as the output of a set of feature detectors which correspond to the parameters represented in a series of arrays used as three masking classifiers: coarse-, medium- and fine-grained.

1. coarse-grained mask applied to output of feature detectors:

feature names	S1	S2	S3	S4
excitation	1	0	0	0
position	1	1	1	1
vot	0	1	1	1

where,

for excitation, 0 = voiceless, 1 = voiced;

for position, 0 = not initial, 1 = initial;

for vot, 0 = none, 1 = some.

Generalisation: there are different sounds which may or may not have voiced excitation, are initial position and may or may not have vot: initial position is redundant information for this data.

2. medium-grained mask:

feature names	S1	S2	S3	S4
excitation	1	0	0	0
position	-	-	-	-
vot	0	1	1	1

where,

' - ' = not relevant.

Generalisation: there are different sounds which a simple binary representation of excitation or VOT cannot discriminate.

3. fine-grained mask:

feature names	S1	S2	S3	S4
excitation	1	0	0	0
position	-	-	-	
vot	0	35	40	45

Generalisation: there are 0-excitation sounds which can be discriminated by a more detailed representation of VOT.

C. Learning based on hypothesis formation

In this example a hypothesis is generated based on the first datum and then continually updated as new data is presented to the device. Sx is an actual example of a signal, s is a variable generalised from cumulative Sx's.

datum 1	stop(S1), excitation(Sl, voiced), position(S1, initial), vot(S1, 0ms)
hypothesis 1	stop(s), excitation(s, voiced), position(s, initial), vot(s, 0ms)
datum 2	stop(S2), excitation(S2, v-lass), position(S2, initial), vot(S2, 3Sms)
hypothesis 2	stop(s), position(s, does_not_discriminate)
	stop(s1), excitation(s1, voiced), vot(s1, 0ms)
	stop(s2), excitation(s2, v-less), vot(s2, 35ms)

datum 3	stop(s), excitation(S3, v-lass), position(S3, initial), vot (S3, 45ms)
hypothesis 3	stop(s), excitation(s, v-less), vot(s, X),(X = 35) or (X = 45)
datum 4	stop(s), excitation(S4, v-less), position(S4, initial), vot (S4, 40ms)
hypothesis 4	stop(s), excitation(s, v-less), vot(s, X), range(s, vot, 35-45)

And now the device can be tested by inputting some new data, in this case deliberately defective in that the only two features supplied from the detector are that a stop is detected and that it is voiceless:

defective datum	stop(s), excitation(s, v-less)
response	stop(s), position(s, initial), vot(s, 35-45)

The device has used its latest hypothesis to repair the defective data supplied, and provided an output response.

D. Connectionist paradigm knowledge representation

In this example two cell arrays, A and B, are set up each with three entries labelled as:

cell array A	cell array B
initial	vot 0ms
voiced	vot > 35ms
v-less	vot n/a

Each cell in array A is connected to each cell in array B. Initially each connection is neutrally weighted with a value of 0. Data is presented to the device in the form of associations of data features:

1. vot(S1, 0ms) :- excitation(S1, voiced), position(S1, initial)

2. vot(S2, 35ms) :- excitation(S2, v-less), position(S2, initial)

3. vot(S3, 45ms) :- excitation(S3, v-less), position(S3, initial)

4. vot(S4, 40ms) :- excitation(S4, v-less), position(S4, initial)

A rule is incorporated which increments connection weights by 1 if a connection between the associated arrays is activated. Thus,

a	*b*	*c*	*d*	*e*

initial	- vot 0ms	0	1	1	1	1
initial	- vot >35ms	0	0	1	2	3
initial	- vot n/a	0	0	0	0	0
voiced	- vot 0ms	0	1	1	1	1
voiced	- vot >35ms	0	0	0	0	0
voiced	- vot n/a	0	0	0	0	0
v-less	- vot 0ms	0	0	0	0	0
v-less	- vot >3Sms	0	0	1	2	3
v-less	- vot n/a	0	0	0	0	0

where,

a = initial indexing of connection strengths

b = after 1st training data

c = after 2nd training data

d = after 3rd training data

e = after 4th training data

- and so on ...

The device acquires and updates knowledge which is represented by weighting patterns across connections. A rough equivalence with a production representation can be seen by simply describing this pattern in terms of productions. Although so simple to be almost unworthy of the connectionist label, in the sense that the device is adjusting to its environment in an ordered fashion it could be said to be learning during training sessions, hut whether it is acquiring rules is an arguable point. *Rules written by the observer of the connection patterns are a product of the mind of the observer and have not been directly observed themselves.*

Furthermore the pattern of connections, it can be argued, does not of itself have any meaningful content - whereas the variables and their relationships in a production-based system *do* have meaningful content. To the extent that such content *is* present in a connectionist device it could only reside in the pattern of firing cells within a specified array. Any attack on connectionism would focus on this concept of meaningful content -the counter attack being, of course, the belief that meaning is imposed from outside the system and has no business being incorporated within the system.

E. The object-oriented approach

Simple messages are received by objects which cause them (if the message is one of the set of recognised messages) to behave in the way internally specified. In the discussion earlier it was suggested that such general messages could be accompanied by tuning messages designed to set particular parameters of the behavioural process.

The Pascal-like code which follows defines the behaviour of an abstract object - VOT, and allows internal provision of some standard length of VOT for stops in general, but

provides also for the external control of that length under certain (cognitively dominated) circumstances:

```
object VOT (class: aerodynamic_constraint);
variables
closure[place~of~articulation] : boolean;
airflow[pulmonic] : boolean;
vocal_cord_vibration : boolean;
vocal_cord_tension : boolean;
{***} air_pressure[supraglottal] : tunable;
air_pressure[subglottal] : integer;
voice_onset_time : integer;
IN_messages
execute(time: integer) : boolean;
tune_variable(degree: integer) : boolean;
OUT_messages
fail, done : boolean;
begin {definition object VOT)
done :- false;
fail :- false;
receive(IN_messages);

while execute do
begin
  if airflow(pulmonic] and
  if closure[place_of_erticulation] and
  if air_pressure[supraglottal] < air_preesure[subglottal] and
  if vocal_cord_tension
          then vocal_cord_vibration := true
          else vocal_cord_vibration := false;
release_stop;
  while air_pressure[supraglottal] >= air_pressure[subglottal] do
  begin
   {***} if not tune_variable then
         begin
            vocal_cord_vibration := false;
            voice_onset_time := standard;
         end
         else
         begin
            vocal_cord_vibration := false;
            voice_onset_time := standard * degree;
         end;
         reduce(air~pressure[supraglottal], voice~onset~time);
   end;

   decrement(time);
end; {while execute}

done := true;
cancel (IN~messages);
send (OUT_messages);
```

```
end; {definition object VOT}
```

CONCLUSION

This paper has been about the representation of phonological and phonetic knowledge in simulation modelling of the human speech production and perception processes. I have emphasised the COGNITIVE PHONETICS approach, drawing attention to the concept of the cognitive control of low level physical structures and shown how we model this. Finally, the paper discusses learning strategies which overcome the lack of suitably oriented phonetic and phonological knowledge within the discipline of linguistics.