Neural Network Models of Language

1.0 Serial Order

1.1 Language is a complex, structured sequence of sounds that is generated by a complex, structured sequence of movements.

One way to describe the structure of such a complex sequence is to use a tree, as grammars do:

Inferred Structure:

```
S
   /\  
  NP  VP
    /  /\  
   DET N  V  A
```

Observed Sequence: the dog is ...

One problem with such descriptions is that when the sequences themselves are analyzed, actions do not appear to be performed as separate units. The form that the actions take on depends strongly on temporal context.

One temporal context effect observed in speech is called coarticulation. The articulatory configuration associated with the production of a given phoneme changes substantially depending on the identity of the phonemes nearby in the sequence.


Thus we need a model that can account both for structural complexity and for serial context effects.

1.2 Rumelhart & Norman’s Typing Model

One approach to context sensitivity is to allow parallelism. Rumelhart & Norman proposed a model of typing that does just that.

The model is a network with a processing unit for each action in the sequence to be produced. There are inhibitory links from units early in the sequence to units later in the sequence. When the activation of the unit reaches a threshold, it is turned off, disinhibiting the remaining units.
Inhibition enforces a graded pattern of activation across the units, with units early in the sequence having the most activation. Each unit pulls the appropriate finger toward a key on the keyboard in proportion to its activation, so that the overall movement of the hand in the vector sum of all parallel influences from future actions.

This model can account for many of the phenomena observed in typing, including the context-sensitive nature of movements toward the keys.

The problem is that the sequential disinhibition mechanism does not allow for certain sequence complexities such as repeated actions. As a general mechanism for sequence generation, there is a problem about which links to follow in the network. As a general mechanism for language generation or recognition, it is similar to a finite state machine.

1.3 Context Sensitive Allophones

Another solution (Wickelgren, 1969) is to introduce units that explicitly code for context. A context-sensitive allophone is a phoneme sized unit that takes into account left and right context. For example the context-sensitive allophones for the string pin are \( \{\#, p, i, n, i, n, \#\} \). More complex sequences can be represented with context-sensitive units (because they have a little memory!), but they are still not sufficient, in general, for language.
2.0 Language by Association

2.1 Associative learning systems have been proposed for language learning. These include generating past tense verbs, and categorizing strings as grammatical/ungrammatical.

The simplest case is that of a linear perceptron: $u = Wv$. Let’s define a learning rule for such a network.

Define the error for pattern $p$ as: $E_p = \frac{1}{2}(t - u)^2$.

Now, let’s make the change to the weights negatively proportional to the derivative of the error surface. This is called gradient descent search. (To simplify this derivation, first assume scalars, $t$, $u$, $w$, $v$. We’ll generalize to vectors and matrices shortly.)

$$\frac{\partial E_p}{\partial w} = \frac{\partial E_p}{\partial u} \frac{\partial u}{\partial w}$$

$$= (t - u) \frac{\partial u}{\partial w}$$

$$= (t - u)v$$

Thus,

$$\Delta w_{ij} = \eta (t_i - u_i)v_j$$

and the amount of learning is proportional to the difference between the actual output activation and the target activation.

Then, the change to the weight matrix is
Neural Network Models of Language

\[ \Delta W = \eta (t - y) \psi^T \]

and the learning rule looks like this:

\[ W_{n+1} = W_n + \eta (t - y) \psi^T \]

For multiple input patterns:

\[ W_{n+1} = W_n + \eta (T - U) V^T \]

where \( V \) is a matrix whose column vectors are input patterns, \( U \) is a matrix whose column vectors are output patterns, and \( T \) is a matrix whose column vectors are desired output patterns. This has the effect of summing across multiple input patterns.

The linear delta rule (Widrow & Hoff, 1960) assumes linear independence of input patterns.

It finds a solution to the learning problem if one exists. If not, it finds a “least squares” approximation.