In L.S. Smith & P.J.B. Hancock (Eds), 1995, Neural Computation and Psychology, 34-48. London: Springer.

Modelling Reaction Times

John A. Bullinaria

Department of Psychology, University of Edinburgh
7 George Square, Edinburgh EH8 9JZ, UK

Abstract

We discuss the simulation of reaction times in connectionist systems. The obvious way to do this involves thinking in terms of neural activations building up towards some threshold in cascaded systems, but it has also been suggested that the output activation error scores in standard back-propagation networks should also be correlated with response times. The idea is that in the more realistic cascaded processing systems, the clearer the outputs (i.e. the lower the errors), the lower the time taken to reach the threshold. If this is correct and we can consider our simple feedforward networks to be reasonable approximations to these cascaded systems, then we have an easy way to simulate reaction times. However, the validity of this has been questioned. I will discuss these issues in some detail, suggest a more principled way of extracting simulated reaction times from back-propagation networks and show how this relates to the corresponding cascaded networks.

1 Introduction

Often when connectionist modelling we wish to simulate reaction times, i.e. the time taken to produce an output after being given an input. These may take many forms: naming latencies in models of reading, times taken to recognise particular sounds in models of hearing, and so on. The obvious way to model these involves thinking in terms of neural activations building up towards a threshold in some form of cascaded system [1-6]. The basic assumption is that the response depends on a series of sub-processes and that activation cascades through these levels eventually activating the appropriate output units. Any activation at one level will feed through to the next at a rate depending on the connection weights (i.e. synaptic strengths) between the levels: there is no need for processing at one level to be completed before the next begins [1].

The natural system of equations to describe such a build-up of activation is:

$$Out_i(t) = Sigmoid(Sum_i(t)) \tag{1}$$

$$Sum_i(t) = Sum_i(t-1) + \sum_j w_{ij} Prev_j(t) - \lambda Sum_i(t-1) \tag{2}$$

so that at each timeslice *t* the output $Out_i(t)$ of each unit is the sigmoid of the sum of the inputs into that unit at that time. The sum of inputs $Sum_i(t)$ is the existing sum at time *t-1* plus the additional weight w_{ij} dependent contribution fed through from the activation $Prev_j(t)$ of the previous layer and a natural exponential decay of activation depending on some decay constant λ. This can be rewritten as:

$$Sum_i(t) = \lambda \sum_j \frac{w_{ij}}{\lambda} Prev_j(t) + (1 - \lambda) Sum_i(t-1) \tag{3}$$

which with $\lambda \to \tau$ and $w/\lambda \to w$ corresponds exactly with the equivalent formulation of Cohen et al. [5]. In the asymptotic state $Sum_i(t) = Sum_i(t-1)$ so:

$$Sum_i(t) = \sum_j \frac{w_{ij}}{\lambda} Prev_j(t). \tag{4}$$

It follows that the asymptotic state of our cascaded network is the same as a standard feedforward network with weights w/λ. Assuming the right way to train the cascading network is to adjust the weights so that it produces the right asymptotic output for each input, we can obtain exactly the same results by training the standard feedforward network in the conventional manner, e.g. by back-propagation. It would seem that any back-propagation network can be trivially re-interpreted as a cascaded network and we can easily extract response times from it in this manner.

If our neural network consists only of input and output units with no hidden layers, the process is easy to analyse (for an application see [6]). $Prev_j(t)$ becomes the constant input and the differential equation (2) is easily solved to give:

$$Sum_i(t) = \sum_j \frac{w_{ij}}{\lambda} Input_j + \left(Sum_i(0) - \sum_j \frac{w_{ij}}{\lambda} Input_j \right) e^{-\lambda t} . \tag{5}$$

The initial condition $Sum_i(0)$ may be some resting or reset value (e.g. zero) or simply that value corresponding to the previous input configuration. We shall discuss this further in the section on priming effects. Figure 1 shows typical plots of $Sum_i(t)$ for four output units with different asymptotic activations. The unit which reaches some threshold activation first may be deemed to indicate the correct network output and the time taken to reach that threshold is the reaction time. It is therefore easy to read off response times from graphs such as these and, since the curves never cross, we can tell almost immediately which output unit is going to reach the threshold first.

Once we start introducing hidden layers to give proper cascaded networks with non-linear activation functions, everything becomes much more complicated (cf. [1]). The previous layer activation $Prev_j(t)$ is no longer a constant input but a varying hidden layer activation. As we can see from Figure 1, the activations of hidden units can change radically over time and hence the rate at which the various output activations build up can have complicated time and unit dependence. It

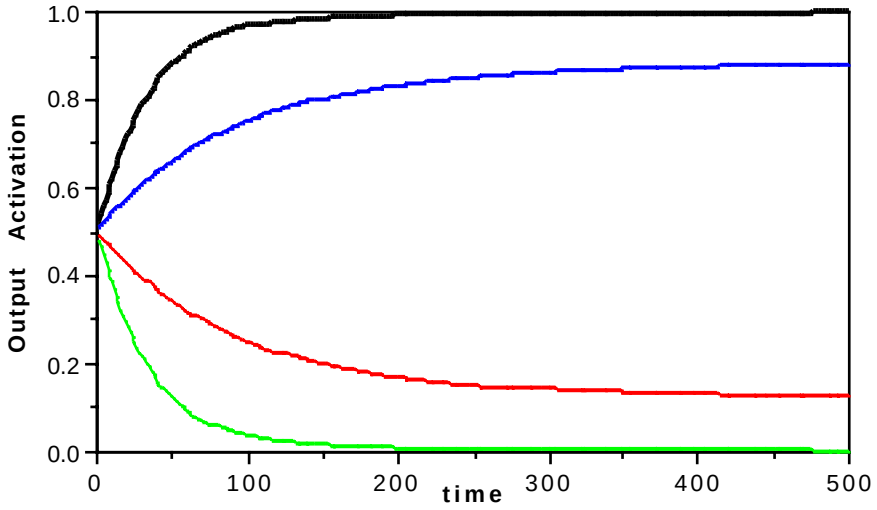

Figure 1: The build up of the Sum in the simplest cascade network with constant inputs and no hidden layer.

becomes possible for output activation curves to cross each other and we have to be very careful about choosing thresholds if the outputs of our cascaded system are to agree with the feedforward network on which it is based. Rather than attempting to derive general solutions to the differential equations here, we shall illustrate the kind of things that can happen by looking at a specific reasonably realistic model.

2 Naming Latencies in a Reading Model

The system we shall look at in detail is a simple model of reading aloud, i.e. text to phoneme conversion, in which the response times are naming latencies. Most of what we say, though, will be applicable quite generally. A complete reading system might look something like shown in Figure 2. Here we will only attempt to model the section from the "letter buffer" to the "phoneme buffer". This sub-system has been described in some detail by Bullinaria [7]. It is basically a NETtalk style model [8] with a modified learning algorithm that obviates the need to pre-process the training data. It uses simple localist representations for its inputs (i.e. one unit for each of 26 letters) and outputs (i.e. one unit for each of 38 phonemes including a phonemic null). The complete input layer consists of a moving window of 13 blocks of 26 units and the output layer consists of two blocks of 38 units. The most highly activated unit of each output block gives the phonemes corresponding to the letter in the central input block in the context of the letters in the other input blocks. We shall concentrate on one such network with a single hidden layer of 300 units trained by back-propagation on the extended Seidenberg & McClelland [9] corpus of 2998 mono-syllabic words. This network achieved 100% performance on its training data and 98.8% generalization performance on a standard set of non-words.

Following the discussion above, it is easy to treat this feedforward network as a

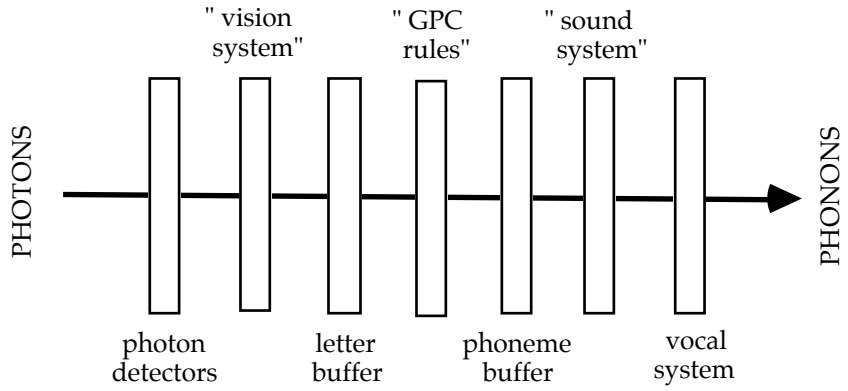

Figure 2: Possible levels in a cascaded system of reading aloud.

cascaded system and plot the build up of output activations for each input letter. To illustrate the problems involved in choosing the appropriate thresholds we present a few well chosen examples. The first graph of Figure 3 shows the build-up of output activations for the 'd' in 'dog'. This is what typically happens for regular letter to phoneme correspondences. Initially all the outputs fall until the appropriate hidden unit activations build up and then the correct phoneme output activation rises to its high value and all the others decay to their low values. The second graph of Figure 3 shows what can happen for a less regular case. The 'oo' in 'spook' is pronounced as an exception to a sub-rule and also occurs infrequently in the training data. Three things complicate this case: First, the asymptotic winner /U/ doesn't take the lead until after about 150 time slices; second, it has a relatively close rival /u/; and third, the final activation is relatively low. Indeed, it is quite possible for the maximum asymptotic activation to be less than the starting activation.

Given the complications introduced by the hidden layer, how do we now read off the reaction times? We cannot simply impose a reasonably high threshold and wait for the first output unit to reach that threshold because there is no guarantee that any unit will reach that threshold. If we impose a too low threshold it is possible that the wrong output unit will reach it first. Similarly, we cannot look for the first unit to start increasing again after the initial decrease because this does not necessarily happen for the right unit. Finally, since all the output activations will generally get multiplied by different weights in the next layer, it is not clear that imposing equal thresholds for all units makes sense anyway.

In a complete cascaded system it is the integrated activations of each layer that power the next, so it seems reasonable that the threshold we should be imposing is not on the activations themselves but on the time integrals of these activations. Figure 4 shows the time integrated activations corresponding to the graphs of Figure 3. We note several important features: first, since the activations are all greater than zero, any output will eventually reach any threshold however low the activation is; second, the complicated effects that occur at small times will contribute to the time taken to reach the threshold but do not interfere with a straightforward implementation of the threshold; and third, we can see how regularity effects could

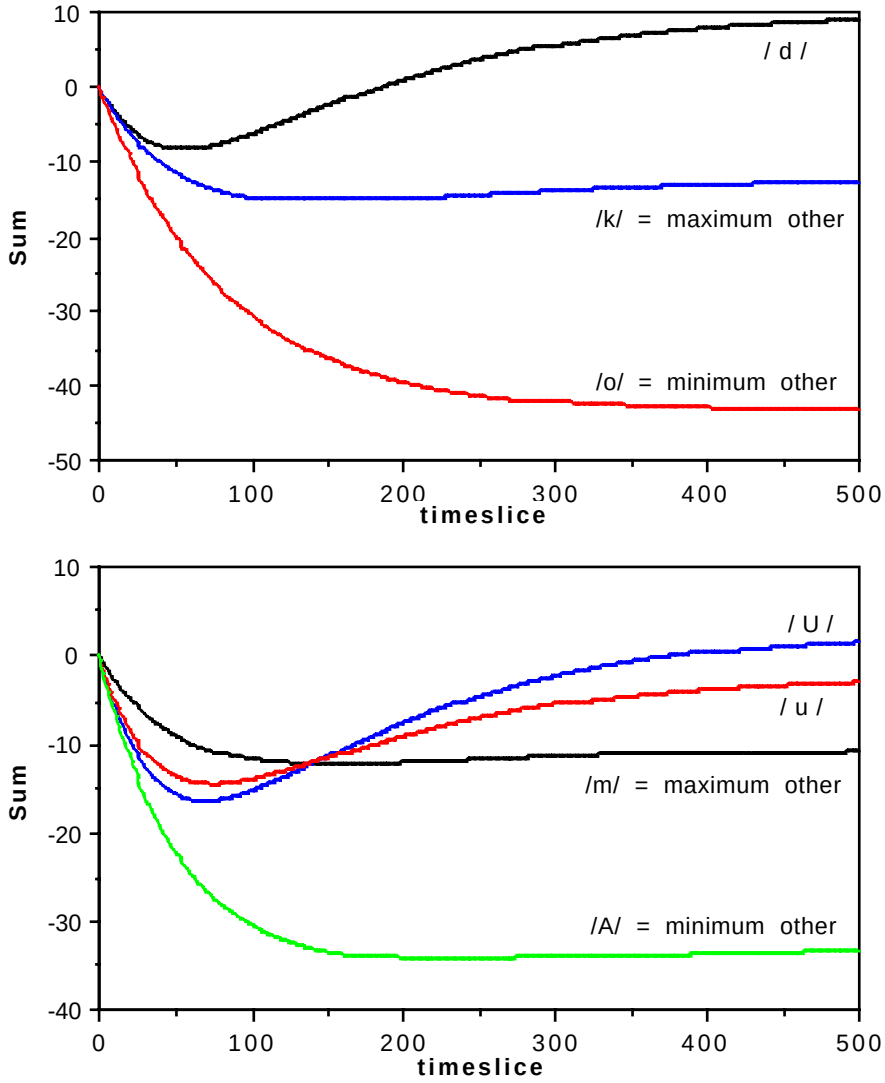

Figure 3: The build up of Sum in the reading model for the regular 'd' in 'dog' and the irregular 'oo' in 'spook'.

arise in the response times. Note that, even if this is not *technically* the right thing to do, we see from the graphs that it does give a much fairer and unambiguous indication of the time taken to process each letter.

To simulate response times for whole words we have to decide if they should be the sum of the response times for the individual letters or if we should assume that parallel processing takes place and the overall response time is that taken for the slowest letter. Experimental evidence suggests that (for mono-syllabic words at least) we should take the naming latency to be given by the longest response time of

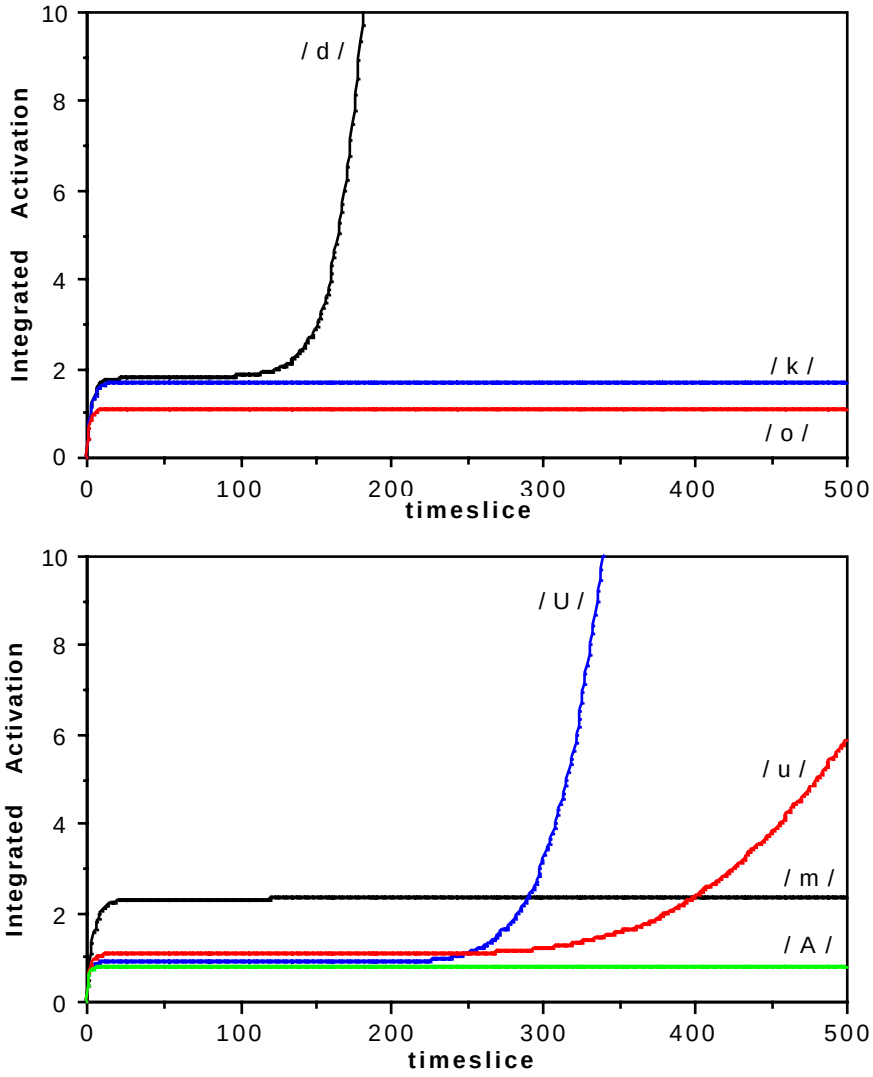

Figure 4: The build up of integrated output activation in the reading model for the regular 'd' in 'dog' and the irregular 'oo' in 'spook'.

the constituent letters (i.e. otherwise our simulated response times bear little relation to human naming latencies). In a complete system we will clearly have to worry about the details of this parallelization process and it will be quite likely that different phonemes will require different integrated activations to drive the process to completion. However, the best we can do here is assign each phoneme unit the same simple threshold.

Applying this cascading procedure to the reading model we can obtain naming latencies for all the words in the training data. The timescale λ and the threshold have to be chosen carefully: The timescale must be small enough that the stepwise

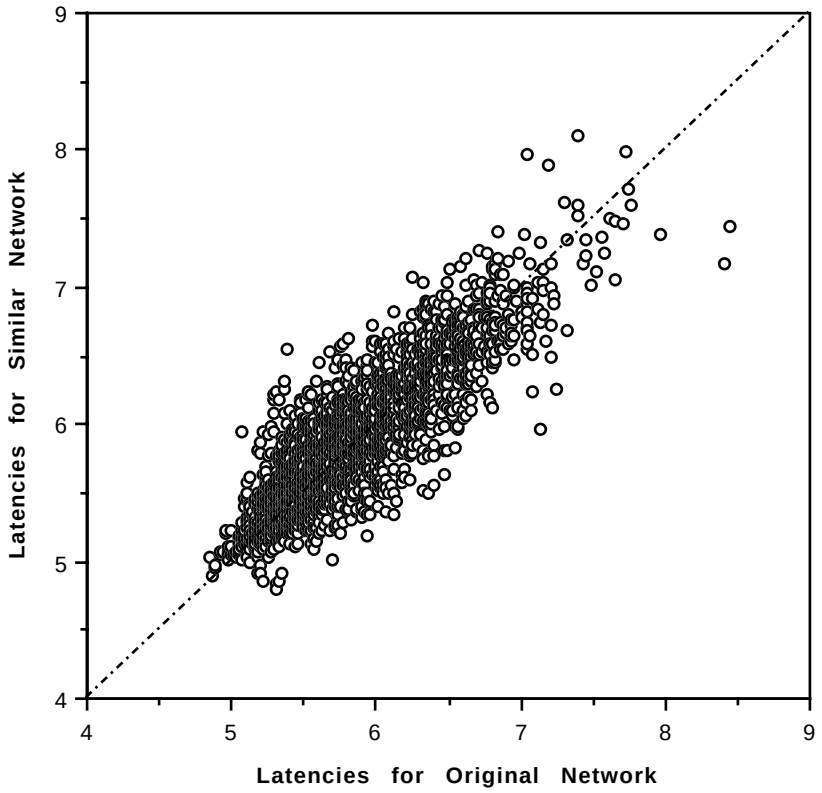

Figure 5: Variation in naming latencies between different networks.

build-up of activation is a reasonable approximation to a realistic continuous build-up, yet large enough that the simulations do not take too long to run. The threshold must be small enough that low activation units do not take an unreasonably long time to reach it, yet not too low that the wrong unit reaches it first (e.g. the /m/ in Figure 4). We found that a timescale of $\lambda = 0.2$ and an integrated activation threshold of 20 were appropriate for our model. The output phonemes produced for each input word corresponded exactly to the training data and we obtained a reasonably realistic distribution of naming latencies that could be compared with the experimental results for humans. These latencies correlated extremely well (Pearson $r = 0.9998$, Spearman $\rho = 0.9996$) with those obtained using $\lambda = 0.1$ instead, indicating that $\lambda = 0.2$ was an acceptable value to use.

For both the Taraban & McClelland [10] and Waters & Seidenberg [11] exception words and controls we found a significant regularity effect as observed in humans ($F = 175.5$, $p < 0.0001$ and $F = 52.9$, $p < 0.0001$) but no frequency nor interaction effects ($p > 0.1$). A separate t-test on the Waters & Seidenberg exception words did show a frequency effect ($t = 3.6$, $p < 0.0019$). The lack of significant frequency effects, apart from this, is in conflict with experiment but easily understood. To get the network to train in a reasonable time we had to logarithmically compress the word frequency distribution. There is good reason to

suppose that training with a more realistic frequency distribution will produce a more realistic frequency effect in the naming latencies.

We might ask if the simulated reaction times we obtain are consistent across networks. Figure 5 shows how the latencies compare for the training words for two identical networks starting with different random initial weights and having the words presented in different random orders. We see that there is a reasonable correlation ($r = 0.87$, $\rho = 0.85$) but find a spread similar to the inter-subject differences found in humans.

The situation is slightly more complicated for non-word data. The network actually generalizes quite well, but still the outputs are rather marginal with very low activations for some non-words (i.e. those based on exceptional words). This results in some unrealistically large simulated naming latencies. We can hope that these problems will go away in more realistic networks that incorporate basins of attraction in the output layer or simple lateral inhibition between output units. In the meantime, we simply imposed a time cut-off (of 10) on the troublesome non-words. For the Glushko [12] non-words and controls we then found a significant non-word effect ($F = 65.5$, $p < 0.0001$), regularity effect ($F = 15.7$, $p = 0.0011$) and interaction effect ($F = 9.6$, $p = 0.0023$). However, unlike in humans, we found no significant difference between the regular words and regular non-words ($t = 0.80$, $p = 0.43$) and no pseudohomophone effect ($t = 0.73$, $p = 0.46$).

The various discrepancies between the network's naming latencies and those found in humans were discussed in some detail by Bullinaria [7]. These problems, together with problems modelling certain types of dyslexia, led us to conclude that the model still required the incorporation of an additional semantic route between the text and phonemes. Activation flowing into the output units via this route will clearly have an effect on the simulated naming latencies. We also expect reaction time effects to arise from the other levels in Figure 2, e.g. common strings of letters may be easier to recognise than others, common strings of phonemes may be easier to pronounce. It is therefore premature to expect our simulated naming latencies to be totally realistic at this stage, but our results so far give us good reason to expect this approach will work well given more realistic back-propagation networks.

3 Priming Effects

Thus far we have taken all the $Sum_i(0)$ to be zero. In this section we consider what happens if they are non-zero, i.e. the system is primed.

The most natural values to use for the $Sum_i(0)$ are whatever values the $Sum_i(t)$ happen to be due to the previous letter processed. This is likely to result in the well known priming effects found in humans (e.g. [13, 14]). An obvious potential problem with this is that, if the previous output is strongly activated (e.g. as is the /d/ in 'dog'), then we need that activation to be reduced sufficiently quickly once the phoneme is spoken lest it triggers the output mechanism again and produces a form of stutter. This can easily be achieved for our model by choosing the threshold sufficiently high. Another problem is that, if we are assuming a certain amount of parallel processing, it is not immediately obvious which should be taken as the previous letter to be processed. Table 1 shows the effect of various priming letters

Prime	time	Sum(0)	time
sp<u>oo</u>f	3.87	+2.0	8.10
sp<u>oo</u>k	4.12	+1.0	7.79
f<u>oo</u>d	4.46	0.0	7.39
b<u>e</u>d	5.54	–1.0	6.88
sp<u>o</u>t	5.80	–2.0	6.23
b<u>oo</u>k	6.24	–3.0	5.58
d<u>o</u>g	6.47	–4.0	5.18
<u>t</u>ip	6.79	–5.0	5.02
ti<u>p</u>	7.15	–6.0	4.97
t<u>i</u>p	7.23	–7.0	5.01
-	7.39	–8.0	5.09
b<u>a</u>t	7.42	–9.0	5.17
<u>b</u>at	7.74	–10.0	5.25

Table 1: The effect on the speed of phoneme production for the 'oo' in 'spook' of priming with particular previous inputs and various values of Sum(0).

on the speed of output for the 'oo' in 'spook'. We see that priming with the word 'spook' itself and the closely related word 'spoof' both produce a significant reduction in the naming latency. Priming with certain unrelated words can produce an increased latency. Interestingly, many apparently unrelated words can also reduce the latency.

Choosing all the $Sum_i(0)$ to be zero would seem to be a fair way to average over these priming effects. However, Table 1 also shows that setting all the $Sum_i(0)$'s to be uniformly less than zero can also improve the naming latencies. (This corresponds to resetting all the $Out_i(0)$ to some low value after each output is produced.) The precise value that minimises the naming latency varies from word to word, but setting $Sum_i(0) = -3.0$ gives a good overall reduction. Figure 6 illustrates how this occurs. It is feasible that such a reset mechanism may also be an efficient way for real brains to operate as well. This introduces yet another free parameter into our simulations. If changing this parameter simply re-scales the reaction times, it would not pose too much of a problem. Figure 7 shows that, although the speed up is far from uniform, there is a some degree of correlation ($r = 0.79$, $\rho = 0.74$), the type effect found previously still remains ($F = 59.9$, $p < 0.0001$ and $F = 13.2$, $p < 0.001$) and it introduces no additional frequency nor interaction effects ($p > 0.1$). If we allow the network to be primed by the previous letter in each word, the correlation is poorer ($r = 0.73$, $\rho = 0.67$) which is not surprising since that letter is essentially random.

Throughout our discussion so far we have treated the network processing unit thresholds as just another network weight ($\theta_i = -w_{iO}/\lambda$) and when we set $Sum_i(0)$ we are effectively balancing the sum of weighted inputs against this threshold. One could argue that it would be more reasonable to take the threshold out of the *Sum()* and into the *Sigmoid()*. This is equivalent to using $Sum_i(0) = -\theta_i$ instead of 0 in our existing formulation. In fact, for our model, the thresholds are all quite small (mean

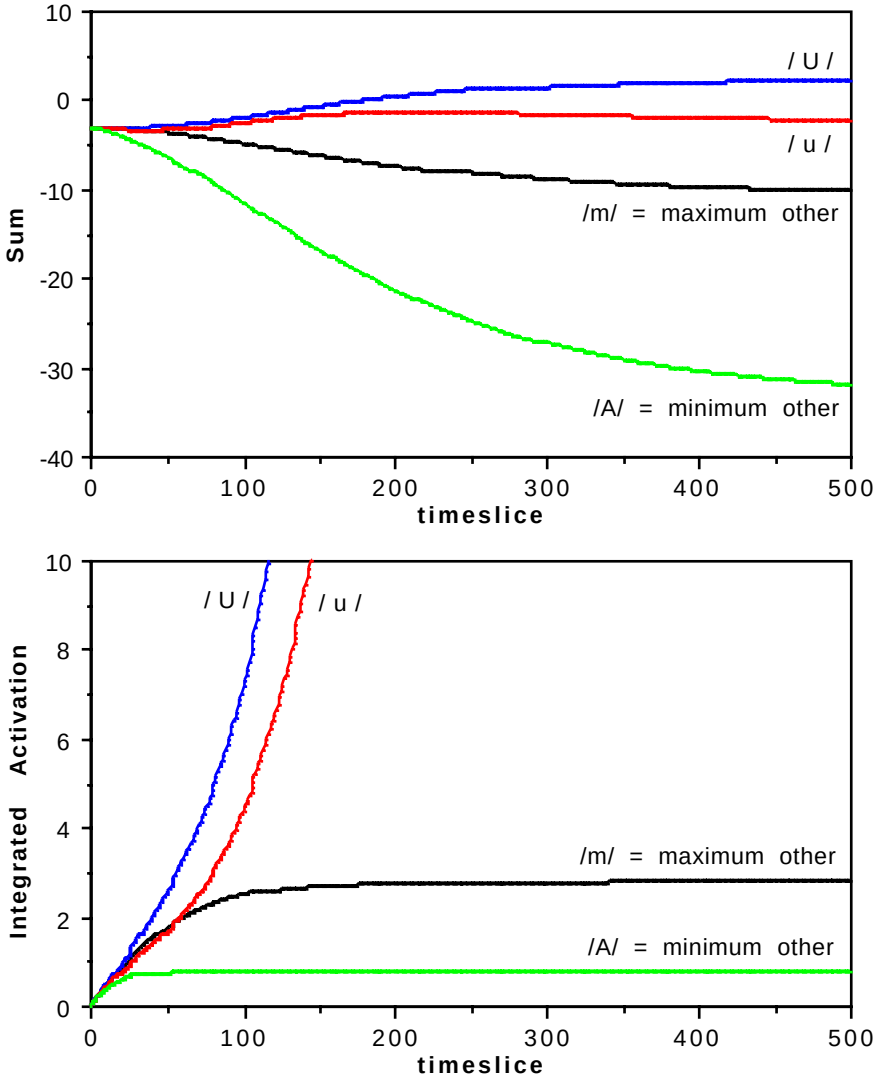

Figure 6: The changes in performance on the 'oo' in 'spook' brought about by having Sum(0) = –3.0 rather than 0.0 .

output threshold 0.9, s.d. 1.0; mean hidden unit threshold 1.3, s.d. 0.6), so the new naming latencies correlate quite well with the old ($r = 0.93$, $\rho = 0.91$). A related variation, that also needs to be checked, takes into account the fact that in the natural resting state the hidden unit activations are non-zero and hence we really have non-zero inputs into the output units. Incorporating this into our model we get yet another set of naming latencies that correlate well with both our new set ($r = 0.98$, $\rho = 0.98$) and the original set ($r = 0.96$, $\rho = 0.95$).

A final point that may have a bearing here, is that real brains operate with limited precision and so it is quite possible that very high and low $Sum_i(t)$'s will

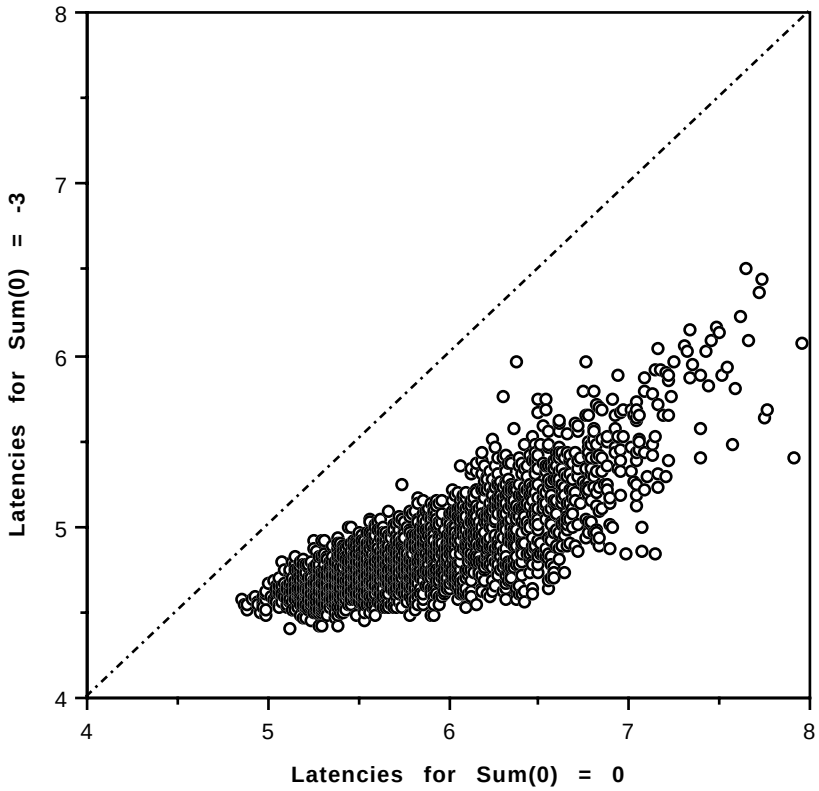

Figure 7: Comparison of simulated naming latencies for two values of Sum(0).

never get learnt. This will clearly be yet another effect on the speed of response when the input is changed.

In summary then, we have seen how various priming effects can be modelled easily in this kind of cascaded system and that resetting all units to the same negative *Sum(0)* can result in a general overall improvement in speed. The results from numerous plausible variations of the basic approach are all quite highly correlated and all result in the significant type effects found in humans.

4 Rushing and Other Detrimental Effects

Another well known effect found in human readers is that they make more mistakes when they are required to operate under time constraints, i.e. when they are rushed. In fact, such speed-accuracy trade-offs occur quite generally [15, 1]. In our model, this corresponds to setting the threshold to a lower value so that the outputs are produced more quickly than normal. Figure 8 shows that for sufficiently large thresholds we have a simple linear relation between the thresholds and mean reaction times. Using successively smaller thresholds eventually introduces errors into the network outputs. Fortunately, Figure 8 also shows that our speed-accuracy

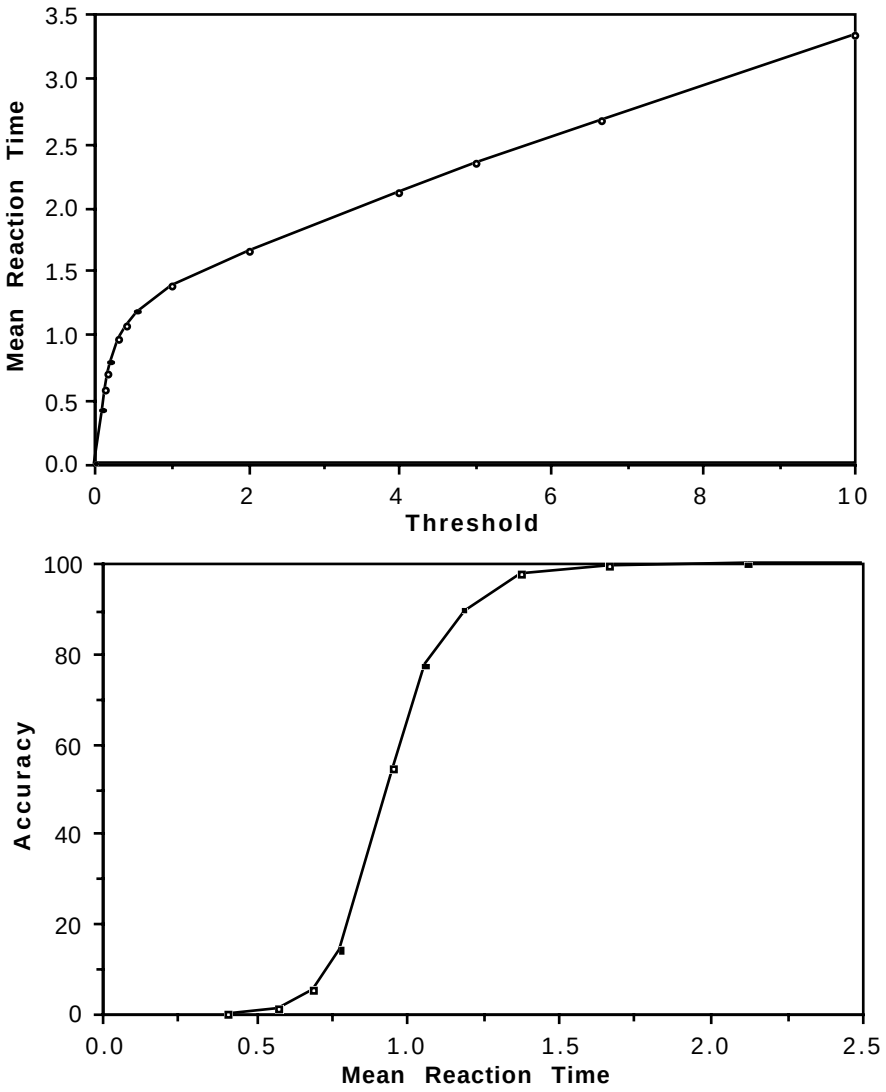

Figure 8: The speed versus accuracy tradeoff for the Sum(0) = –3 case.

trade-off curve compares well with those of humans [1, 15]. The first errors are generally regularized vowels followed by wrong vowels and other regularizations. It is easily checked that, as long as our thresholds are set sufficiently high, the precise values have little effect on the simulated latencies (for thresholds 20 and 10 we have r = 0.997, ρ = 0.999; for 20 and 4 we have r = 0.986, ρ = 0.994). Real brains will presumably normally operate in this region where the error rate is reliably low.

Finally, one might argue that for total consistency we should also have our

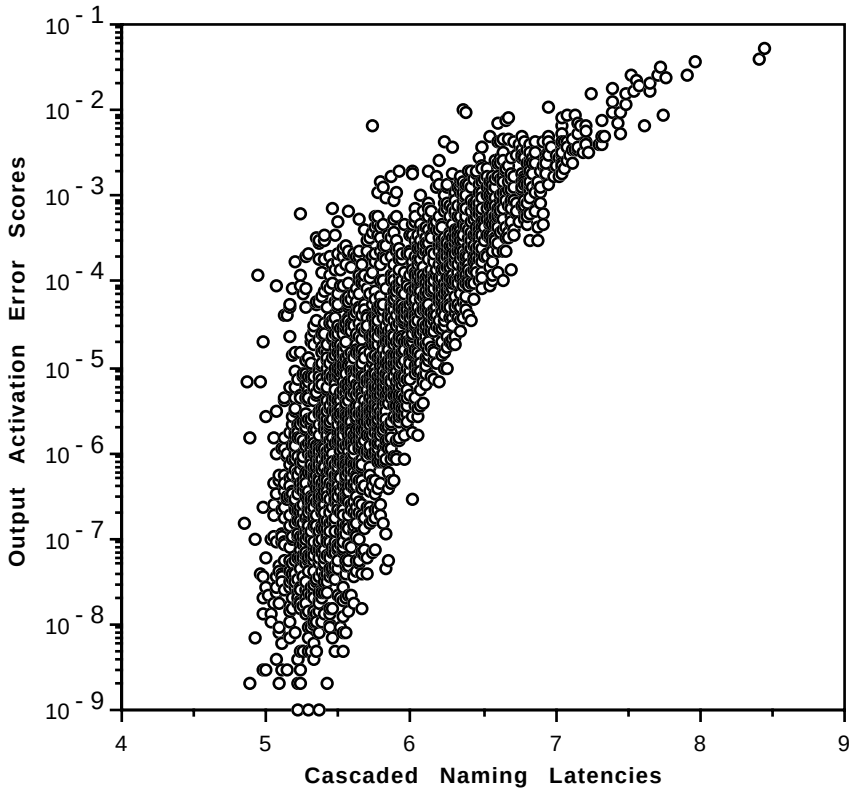

Figure 9: The relationship between our cascaded naming latencies and standard output activation error scores.

input activations building up over time. Doing this uniformly, scatters the naming latencies yet again ($r = 0.86$, $\rho = 0.82$). Given the likelihood that some letters will be detected more rapidly than others and at rates depending on their contexts, we can expect even more scattering than this in a completely realistic system. Taking this one step further, it is easy to imagine how other manipulations of the input activations might be used to simulate the effects of reading degraded type faces, "bad light" and so on.

5 Non-cascaded Response Times

It has been suggested (by Seidenberg & McClelland [9] and many others since then) that the output activation error scores in standard back-propagation networks should also be correlated with response times. The idea seems to be that in the more realistic cascaded processing systems we have been considering, the clearer the outputs (i.e. the lower the errors), the lower the time it should take to reach the threshold. If this is correct, then we have an easy way to simulate reaction times. However, the validity of this has been questioned and it has been demonstrated

explicitly for our model of reading that changing the precise mathematical relation used to relate the error scores to the simulated response times can have quite significant effects on the results [7].

The first thing to notice is that, if the procedure we have adopted here to simulate reaction times is correct, then it is only the activation of the winning output unit of the slowest parallel process that matters. Taking into account the error score contributions due to faster parallel processes or other units having not quite zero activation seems to be the wrong thing to do. Indeed, Figure 9 shows that the relationship between the error scores and the cascaded latencies for our reading model is far from simple. In particular, we see that the error score distribution is excessively skewed. There *is* a significant correlation ($\rho = 0.72$) and the high cascaded latencies *do* correspond to high error scores, so we still get the strong type effect, but apart from that we cannot say much. The various other non-cascaded estimations of naming latencies suggested by Bullinaria [7] do not do much better: Inverse Log Errors $r = 0.75$, $\rho = 0.72$; Log Sum Luce Ratios $r = 0.68$, $\rho = 0.67$; Sum Inverse Inputs $r = 0.56$, $\rho = 0.56$.

We see from Figures 3, 4 and 6 that there are two main contributions to the time taken to reach threshold. First, the time taken for the output activations to build up to near their asymptotic values and second, the asymptotic slope of the curve which is trivially related to the critical asymptotic output activation. In fact, even the inverses of these critical output activations are surprisingly poor indicators of the cascaded results ($r = 0.63$, $\rho = 0.76$), so it is hard to see what other than our full cascaded approach could do much better than the simple output error scores.

6 Conclusions

We have seen how it is easy to re-interpret simple feedforward back-propagation networks as cascaded systems. This provides a principled procedure for simulating reaction times complete with independent priming and rushing effects that are rather difficult to model in non-cascaded systems.

Looking at a particular reading model, we find significant correlations between the simulated naming latencies of virtually any two variations of the basic approach and strong word type effects seem to be obtained whatever we do. The standard supposed relationship between reaction times and the output activation error scores seems to have some basis but must clearly be treated with caution and we will have to be much more careful when looking for weaker effects in our models.

References

1. McClelland, J.L. (1979). On the time relations of mental processes: An examination of systems of processing in cascade. *Psychological Review*, **86**, 287-330.
2. Dell, G.S. (1986). A Spreading-Activation Theory of Retrieval in Sentence Production, *Psychological Review*, **93**, 283-321.
3. McClelland, J.L. and Rumelhart, D.E. (1981). An Interactive Activation Model

of Context Effects in Letter Perception: Part 1. An Account of Basic Findings, *Psychological Review*, **88**, 375-407.
4. Rumelhart, D.E. and McClelland, J.L. (1982). An Interactive Activation Model of Context Effects in Letter Perception: Part 2. The Contextual Enhancement Effect and Some Tests and Extensions of the Model, *Psychological Review*, **89**, 60-94.
5. Cohen, J., Dunbar, K. & McClelland (1990). On the control of automatic processes: A parallel distributed processing model of the Stroop task, *Psychological Review*, **97**, 332-361.
6. Norris, D. (1993). A quantitative model of reading aloud, Technical report, MRC APU, Cambridge.
7. Bullinaria, J.A. (1994). Representation, Learning, Generalization and Damage in Neural Network Models of Reading Aloud, Submitted.
8. Sejnowski, T.J. and Rosenberg, C.R. (1987). Parallel Networks that Learn to Pronounce English Text, *Complex Systems*, **1**, 145-168.
9. Seidenberg, M.S. & McClelland, J.L. (1989). A distributed, developmental model of word recognition and naming, *Psychological Review*, **96**, 523-568.
10. Taraban, R. & McClelland, J.L. (1987). Conspiracy Effects in Word Pronunciation, *Journal of Memory and Language*, **26**, 608-631.
11. Waters, G.S. & Seidenberg, M.S. (1985). Spelling-sound effects in reading: Time-course and decision criteria, *Memory & Cognition*, **13**, 557-572.
12. Glushko, R.J. (1979). The Organization and Activation of Orthographic Knowledge in Reading Aloud, *Journal of Experimental Sciences: Human Perception and Performance*, **5**, 674-691.
13. Meyer, D.E., Schvaneveldt, R.W. & Ruddy, M.G. (1974). Functions of graphemic and phonemic codes in visual word-recognition. *Memory & Cognition*, **2**, 309-321.
14. Tanenhaus, M.K., Flanigan, H.P. & Seidenberg, M.S. (1980). Orthographic and phonological activation in auditory and visual word recognition. *Memory & Cognition*, **8**, 513-520.
15. Wickelgren, W.A. (1977). Speed-accuracy tradeoff and information processing dynamics, *Acta Psychologica*, **41**, 67-85.