COSC421 2015 Assignment 1 (15%)
Phonological working memory

There are two options for this assignment. The first involves programming; the second is an essay. Both options are based around Burgess and Hitch's (1999) well-known model of phonological working memory—i.e. memory for short sequences of letters, digits or words.

Option 1: A neural network for remembering phone numbers

In this option, your task is to implement a neural network which receives as input a sequence of numbers (imagine they arrive as a spoken sequence), retains these in memory for a short time, and then reproduces this sequence as output.

The architecture of the network should look as shown in Figure 1.

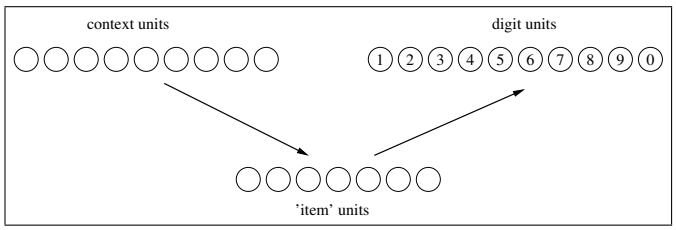

Figure 1: Architecture of the network

The network comprises:

- A bank of 10 **digit units**, representing the digits 0–9.
- A bank of 9 **context units**, which represent a sequence of temporal contexts.
- A bank of (around!) 7 **item** units, representing temporary associations between digits and contexts.

All units have real-valued **activations** ranging from 0–1.

The digit units use a **localist** encoding scheme: to represent a digit, we set the activation of one unit to 1, and of all other units to 0.

The context units should be hardwired (preprogrammed) to encode a sequence of seven temporal contexts. Each context is represented by three active units, as shown in Figure 2. At each temporal context, three units are on. Note that this layer uses a **population coding** scheme (as discussed in Lecture 2): representations of successive temporal contexts partly overlap.

The arrows in Figure 1 denote **fully connected** layers: thus every context unit is connected to every item unit, and every item unit is connected to every digit unit. (These connections are unidirectional, and initially have random **strengths**, also in the range 0-1.)

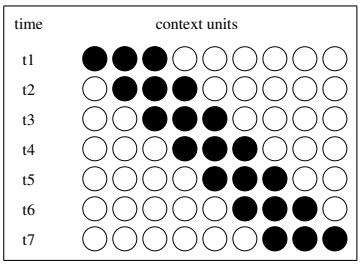

Figure 2: Sequence of activations in the context unit layer. (Black units are 'on'; the others are 'off'.

1. During **encoding**, the network receives a sequence of digits, one at each time step, and at the same time it steps through its preset sequence of temporal context representations. At each time step, item units compete, so that the most active receives extra activity, and the others having their activity reduced. After activity in the item units has been updated to reflect competition, **Hebbian learning** takes place in all the connections in the network, according to the following rule: the change $\Delta_H w$ in the strength of a connection from u_1 to u_2 is given by

   $$\Delta_H w = \alpha u_1 u_2$$

   (where α is a small positive learning constant). This rule has the effect of strengthening connections between units which are both on, while leaving other connection strengths unchanged. At the end of each time step, the strength of each connection is reduced by a constant amount $\Delta_D w$, to model the decay of memory traces. Also, *the winning item unit should be inhibited*, so that it can't compete to store subsequent digits. (You may need this inhibition to last for more than one time step.)

2. During **rehearsal**, the sequence of context representations is replayed, evoking a sequence of activation patterns in the item units (which are again sharpened through competition) and a sequence of activation patterns in the digit units (which are likewise sharpened through competition). At each time step, after activations are determined, another round of Hebbian learning occurs to 'relearn' the original pattern. (Without rehearsal, the pattern would eventually be forgotten, because of the general decay of connection weights over time.) Rehearsal of the whole sequence can happen an arbitrary number of times.

3. During **retrieval**, we perform the same operations as during rehearsal, but at each time step we produce the winning digit unit as an output. Thus retrieval results in the production of a sequence of digits.

Code

You are free to implement the network in any language. You will have to experiment to find suitable values for α, $\Delta_H w$ and $\Delta_D w$, and to implement the sharpening effect of

competition, and the inhibition of the winning item unit.

Testing the network

The **main behaviour** your network should show is an ability to retain a sequence of seven digits (a phone number) over at least one iteration of rehearsal.

In addition to this, you should implement at least one **extension behaviour**, which reproduces a property of real phonological working memory. Here are some options. (You can think of alternatives if you like.)

- **Transposition errors.** If you make the activation of units a bit noisy, you should be able to get errors where two successive digits have their positions swapped when they are recalled—so, for instance, instance *0123* is recalled as *0213*. (This effect is made possible by the coarse coding of temporal contexts, plus self-inhibition of item units.)
- **Interference.** If you present the system with 'distractor' digit sequences while it is rehearsing, you should cause it to forget its original sequence pretty effectively.
- **Primacy and recency effects.** You may find that your network is better at remembering the first and last digits in a sequence than the middle digits. If so, you can try and explain why these effects arise.

Report

You should also submit a report about your code. The report should contain four sections:

- How the code implements the network model (including the code for your extension behaviour);
- How to run the code (so I can test it);
- Some examples of its results (including its performance on the main and extension behaviours);
- A brief evaluation of how well it does, and what might be done to improve it.

Submission and marking

You should submit the code and the assignment by email to me (`alik@cs.otago.ac.nz`) by 5pm on **Monday of Week 7** (i.e. on **Monday August 22nd**). 10% of available marks will be deducted for each day late.

I'll give equal weight to the code and the report. (But the 'report mark' is likely to reflect the quality of the code, so in that sense the implementation is the main focus.)

Option 2: Essay

The second option for the assignment is an essay, with the following title:

> Describe in detail Burgess and Hitch's (1999) neural network model of the phonological loop. Your description should include: (i) the architecture of the network, (ii) the rules governing how connection strengths are changed, (iii) the activation patterns in the network during encoding, rehearsal and recall, and (iv) an explanation of the behaviour of the network, including an explanation of how it reproduces human phonological working memory performance.

To answer this question, your main reference should obviously be Burgess and Hitch (1999). But you can also briefly consult other papers—for instance, papers which Burgess and Hitch cite, or which cite them.[1]

Note: the model described in Option 1 is a simplification of Burgess and Hitch's actual model. You need to describe the actual model, not the simplification.

Submission and marking

You should submit the essay by email to me (`alik@cs.otago.ac.nz`) by 5pm on **Monday of Week 7** (i.e. on **Monday August 22nd**). 10% of available marks will be deducted for each day late.

You will be marked on the clarity and organisation of the essay, and on evidence that you have read the assigned paper in detail and understood the model described.

References

Burgess, N. and Hitch, G. (1999). Memory for serial order: A network model of the phonological loop and its timing. *Psychological Review*, **106**, 551–581.

[1] You can get these papers online from Web of Science. Go to the Otago Library webpage, then follow links to 'Article Databases' → 'W' → 'Web of Science via Web of Knowledge'. This is a fantastic resource—if you don't already know about it, you should try it!