

Improved Storage Capacity in Correlation Matrix Memories Storing Fixed Weight Codes

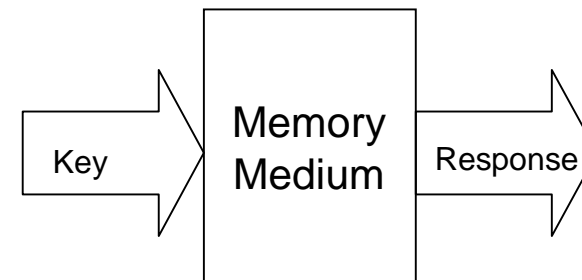
Stephen Hobson and Jim Austin
Department of Computer Science,
University of York, UK

Outline

- Associative Memory
- Binary Correlation Matrix Memories
- Fixed weight coding
- Baum's algorithm for fixed weight codes
- Improved thresholding
- Results

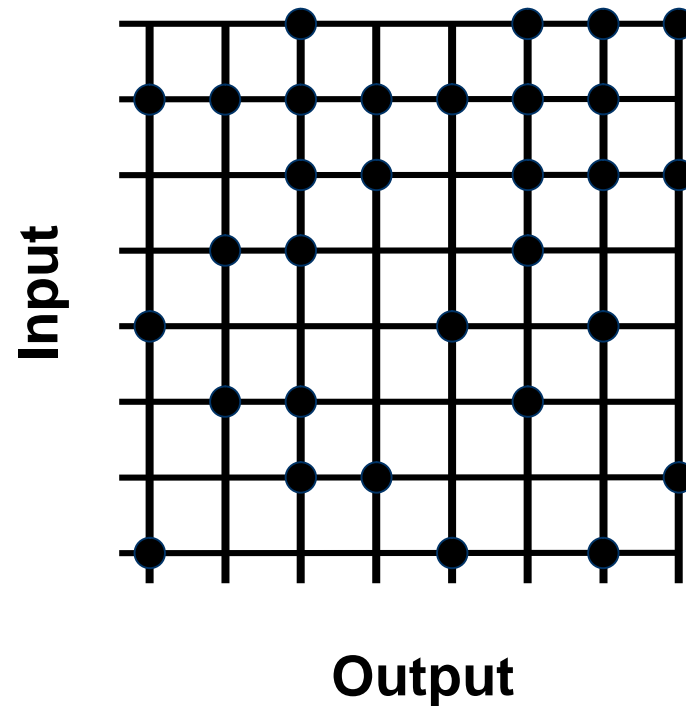
Associative Memory

- Stores data as associations between pairs of data items
- Think of a dictionary
- Not necessarily neural- but today we only consider a neural model



Binary Correlation Matrix Memories

- A single layer fully connected neural network
- Commonly represented as a matrix of weights
- Here we consider the binary version
- Known henceforth as the CMM



Memory

Correlation Matrix Memory

- Distributed
- Robust to noise
- Generalisation
- Storage capacity depends on data

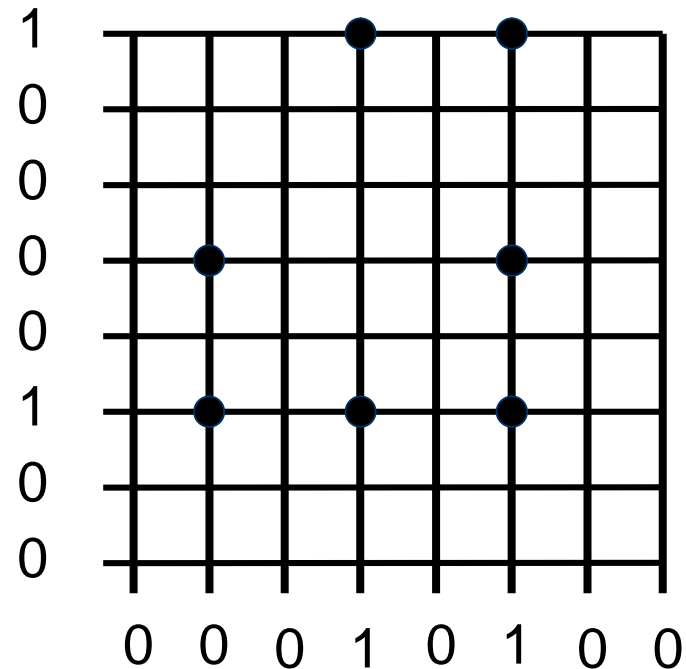
Conventional Memory

- Localised
- Noise-intolerant
- No Generalisation
- Known storage capacity

CMM Storage

- Present data pair to memory
- Set weight to 1 where there is mutual activity
- Repeat for all pairs to store

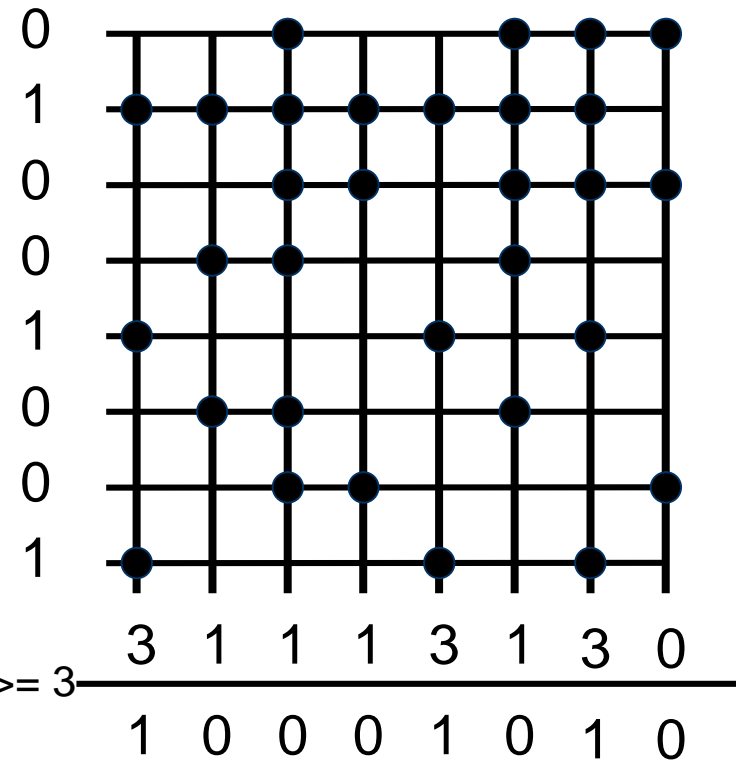
$$W = \bigvee_{i=1}^N x_i y_i^T$$



CMM Recall

- Present input pattern to memory
- Apply threshold function to resulting activity to get output code

$$y = f [Wx]$$



Applications

- Graph matching
 - Molecular databases
 - Face recognition
- Text processing
 - Address databases
- Signal processing
 - Spotting problems in Rolls Royce aero engines

The screenshot displays a software window titled "match_win32.exe" with a menu bar (File, Process, Database, Query, Help). The main area shows search statistics: "Database taught nciopen3d in 5.218 s user time 0.0 s system time" and "Query 17 solutions in 4 iterations in 0.32 s user time and 0.0 s system time". A list of results is shown, with "100 with 42 points in query matching molecule" highlighted. Below the list is a "View" button. At the bottom, three 3D molecular models are displayed in separate windows labeled "netropsin", "78", and "100", each with a "Dismiss" button underneath.

Match ID	Points in Query Matching Molecule
17	66
78	58
100	42
108	64
116	67
117	68
118	61
119	40
124	43

CMM Storage Capacity

- Capacity depends on data stored
- Can store $K > N$ associations
- Representation and threshold function are important



Fixed Weight Codes

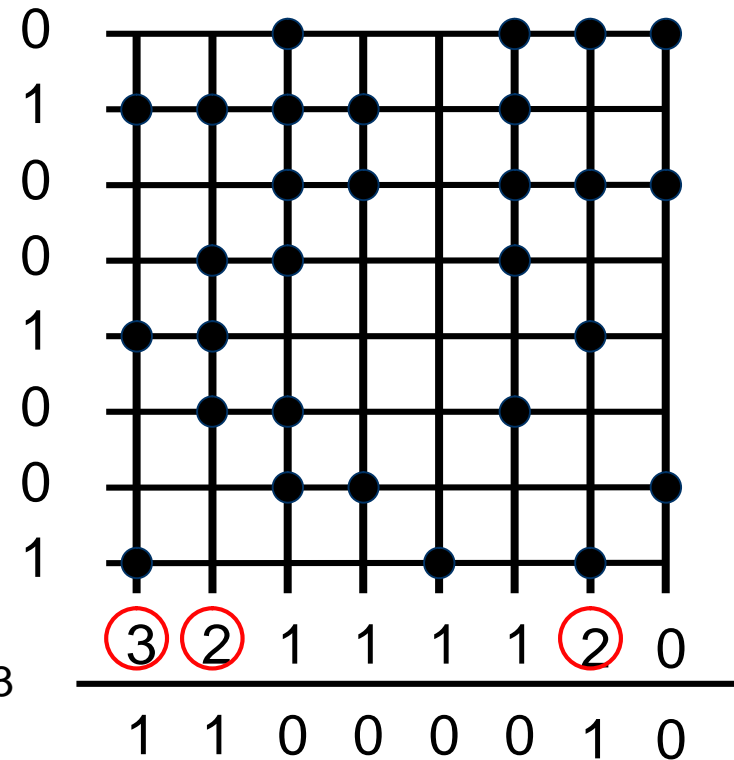
- Fixed number of bits set to 1 in a vector
- Therefore output codes have a known weight
- Sparse coding is beneficial
- However, unary representation loses distribution

```
1 1 0 0 0 0 0 1 0 0 0 0
0 0 0 1 0 0 0 1 0 0 1 0
0 0 1 0 0 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 0 1 1 1
0 1 0 0 1 0 0 1 0 0 0 0
0 0 1 0 1 1 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0 0 1
0 0 0 1 0 1 0 0 0 1 0 0
```

L-max Thresholding

- Set the L largest activities to 1
- Otherwise, 0
- Enables robust recall of fixed weight codes

L-max threshold: L=3



Baum Codes

- Algorithm given by Baum et al. to generate fixed weight codes
- Divide code into L coprime sections
- Creates codes with low “overlap”
- Deterministic method

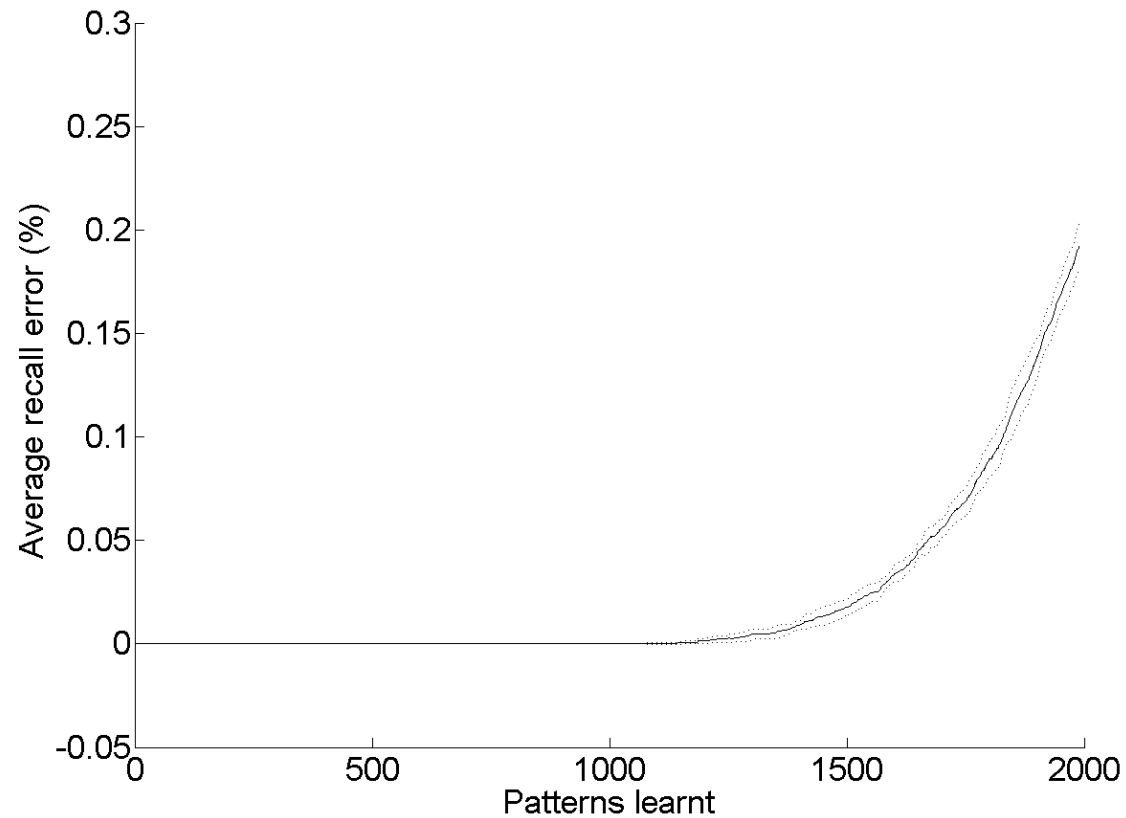
Length = 10, Weight = 3

```
1 0 0 0 0 | 1 0 0 | 1 0
0 1 0 0 0 | 0 1 0 | 0 1
0 0 1 0 0 | 0 0 1 | 1 0
0 0 0 1 0 | 1 0 0 | 0 1
0 0 0 0 1 | 0 1 0 | 1 0
1 0 0 0 0 | 0 0 1 | 0 1
0 1 0 0 0 | 1 0 0 | 1 0
0 0 1 0 0 | 0 1 0 | 0 1
```

The Simulation

1. Create empty CMM
2. Generate input code by Baum's algorithm
3. Generate random output code
4. Store pair in CMM
5. Present all previously learnt inputs to CMM and measure recall error when using L-max and L-wta
6. Return to 2

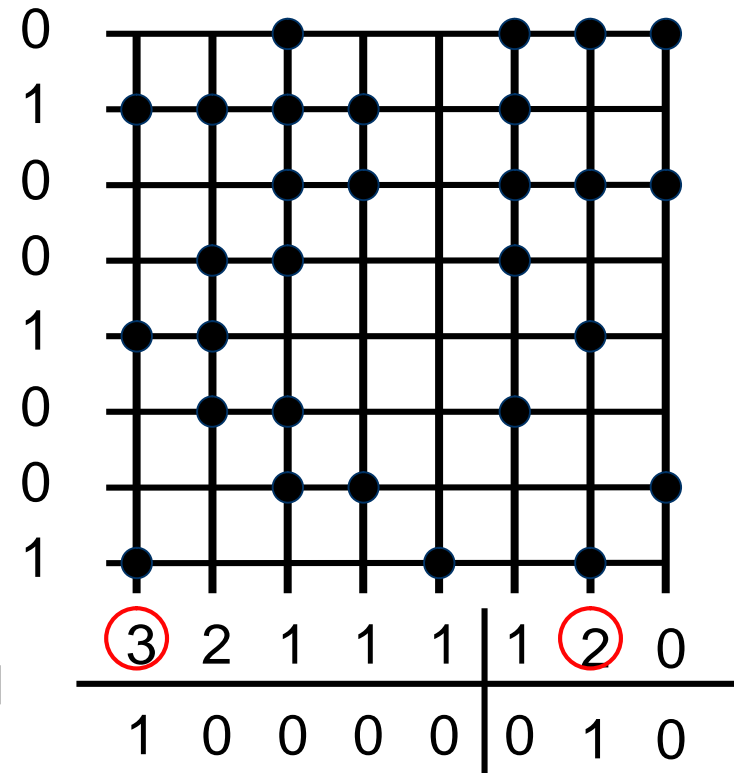
Results - Input 512/16, Output 256/4



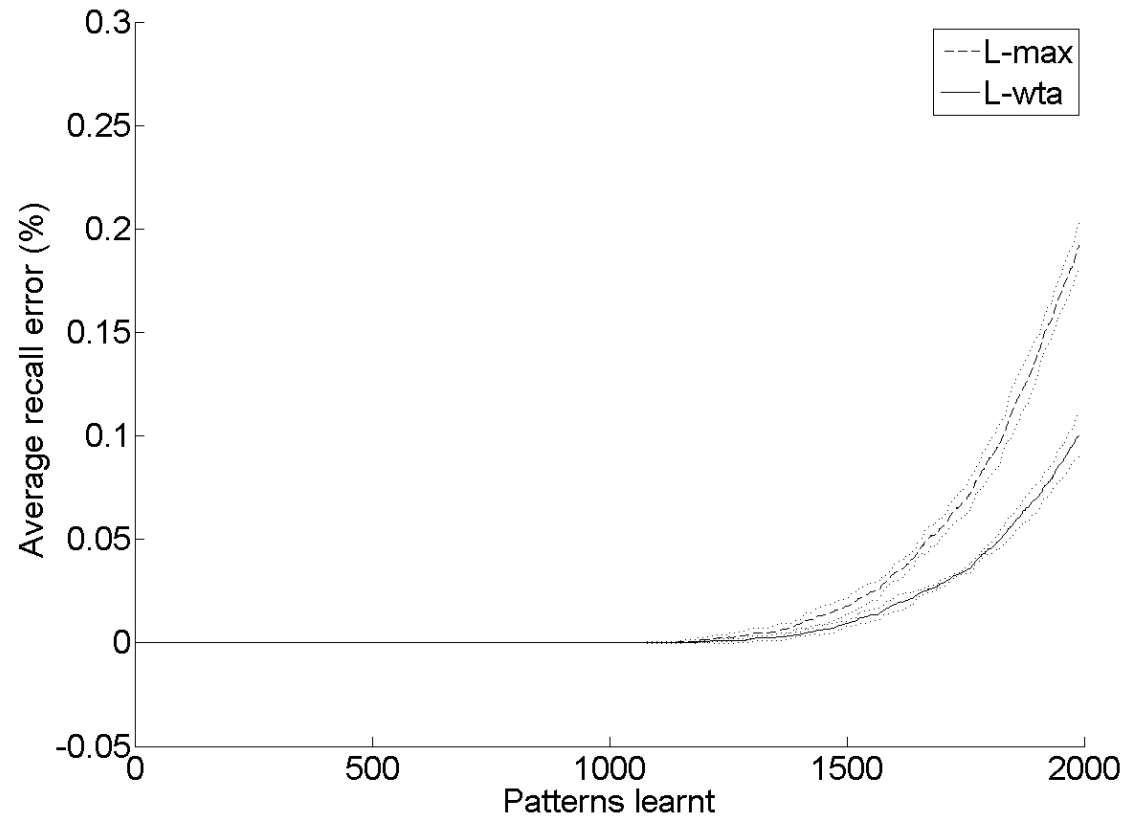
L-wta Thresholding

- Apply a winner-takes-all approach to each section of the code
- Constrains output from thresholding function
- Provides a more robust thresholding function for Baum codes

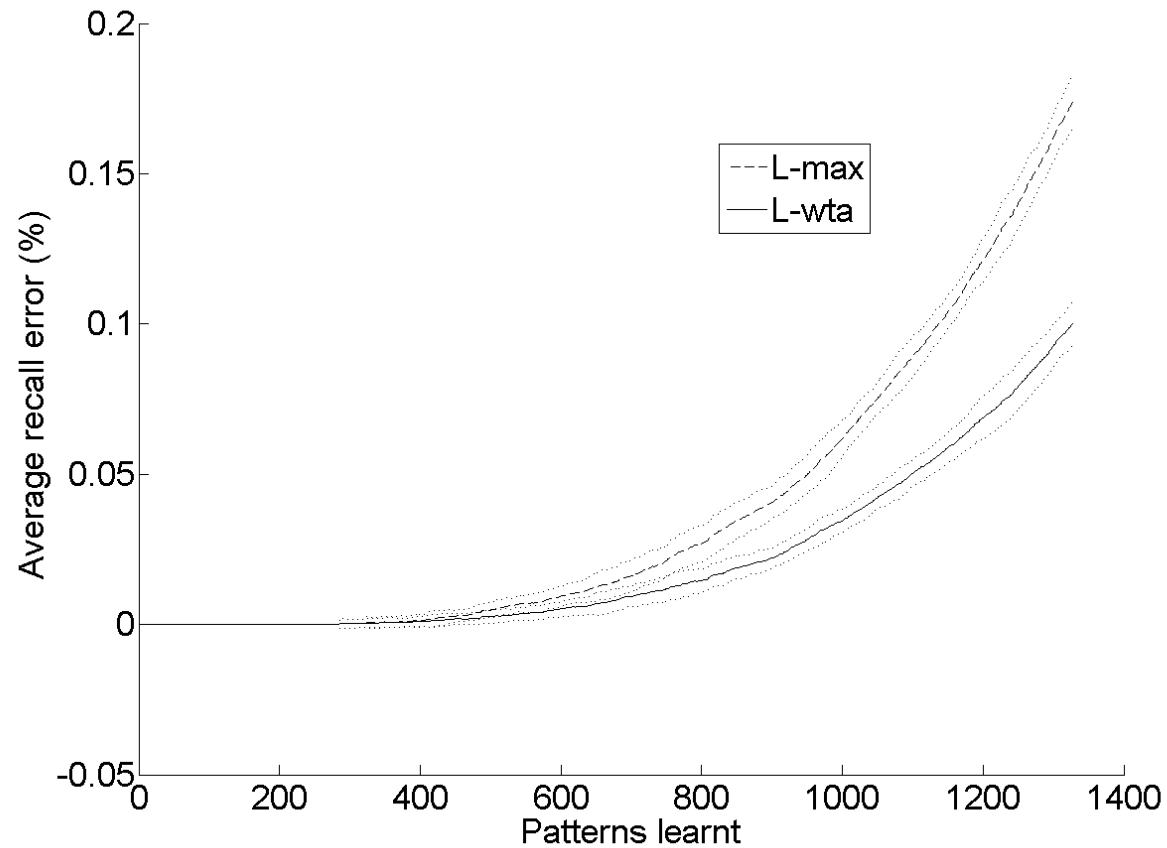
L-wta threshold, $S=[5,3]$



Results - Input 512/16, Output 256/4



Results - Input 256/4, Output 512/4



Results - Observations

- L-wta demonstrates consistently higher storage capacity when compared to L-max on CMMs storing Baum codes
- This benefit increases as the memory becomes increasingly saturated
- On average, increase is approximately 15%

Thank you

- <http://www.cs.york.ac.uk/arch>
- <http://www.cybula.com>