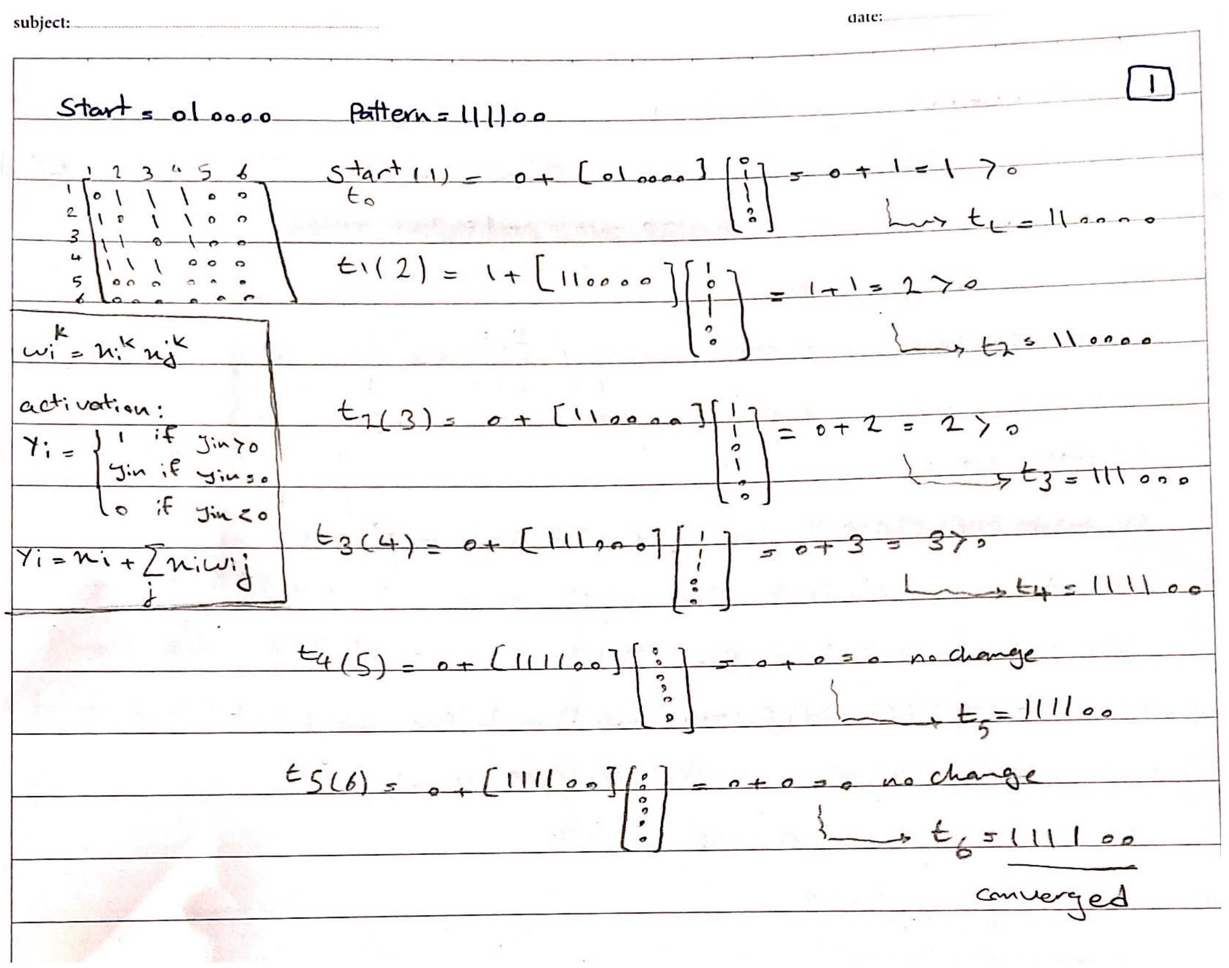
HW3 Report

Fundamentals of Computational Intelligence

Niki Nezakati
98522094
Spring 2022

(Q1



subject:

date:

$\boxed{1}$

Start = 010000 Pattern = 111100

$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 1 & 0 & 0 \\ 3 & 1 & 1 & 0 & 1 & 0 & 0 \\ 4 & 1 & 1 & 1 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

$w_i^k = x_i^k x_j^k$

activation:

$$Y_i = \begin{cases} 1 & \text{if } y_{in} > 0 \\ y_{in} & \text{if } y_{in} = 0 \\ 0 & \text{if } y_{in} < 0 \end{cases}$$

$$Y_i = x_i + \sum_j x_i w_{ij}$$

$$\underset{t_0}{\text{start}}(1) = 0 + [010000]\begin{bmatrix}0\\1\\1\\1\\0\\0\end{bmatrix} = 0 + 1 = 1 > 0 \quad \rightarrow t_1 = 110000$$

$$t_1(2) = 1 + [110000]\begin{bmatrix}1\\0\\1\\1\\0\\0\end{bmatrix} = 1 + 1 = 2 > 0 \quad \rightarrow t_2 = 110000$$

$$t_2(3) = 0 + [110000]\begin{bmatrix}1\\1\\0\\1\\0\\0\end{bmatrix} = 0 + 2 = 2 > 0 \quad \rightarrow t_3 = 111000$$

$$t_3(4) = 0 + [111000]\begin{bmatrix}1\\1\\1\\0\\0\\0\end{bmatrix} = 0 + 3 = 3 > 0 \quad \rightarrow t_4 = 111100$$

$$t_4(5) = 0 + [111100]\begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix} = 0 + 0 = 0 \text{ no change} \quad \rightarrow t_5 = 111100$$

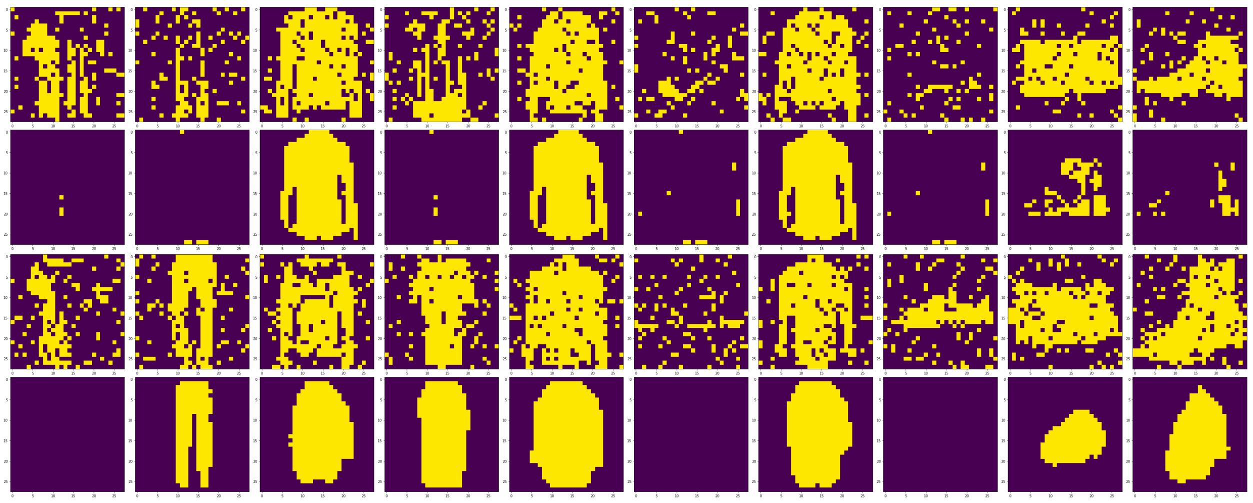
$$t_5(6) = 0 + [111100]\begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix} = 0 + 0 = 0 \text{ no change} \quad \rightarrow t_6 = 111100$$

converged

Hopfield Network for Fashion MNIST Dataset (Q2

The dataset images were turned into bipolar format (-1,1) to be used in the hopfield network. The network sizes are 6 , 600 and 60000. I used a sign activation function which, with execution and debugging, 80 had the best accuracy as the bias.

Result images:



Accuracy Table:

Noise / Network Size	6	600	60000
10%	0.17	0.19	0.18
30%	0.07	0.03	0.03
60%	0.00	0.00	0.00

(Q3

.3.1

The Traveling Salesman Problem consists of finding the shortest route possible that traverses all cities in a given map only once. This problem is NP-Complete. This implies that the difficulty to solve it increases rapidly with the number of cities, and we do not know in fact a general solution that solves the problem. For that reason, it's currently considered that any method able to find a sub-optimal solution is generally good enough.

To solve it, we can try to apply a modification of the **Self-Organizing Map (SOM)** technique.

If instead of a grid we declare a *circular array of neurons*, each node will only be conscious of the neurons in front of and behind it. That is, the inner similarity will work just in one dimension. Making this slight modification, the self-organizing map will behave as an elastic ring, getting closer to the cities but trying to minimize the perimeter of it thanks to the neighborhood function. To ensure the convergence of it, we can include a learning rate, α , to control the exploration and exploitation of the algorithm. To obtain high exploration first, and high exploitation after that in the execution, we must include a decay in both the neighborhood function and the learning rate. Decaying the learning rate will ensure less aggressive displacement of the neurons around the model, and decaying the neighborhood will result in a more moderate exploitation of the local minima of each part of the model. Then, our regression can be expressed as:

$$nt + 1 = nt + \alpha t \cdot g(we, ht) \cdot \Delta(e, nt)$$

Where α is the learning rate at a given time, and g is the Gaussian function centered in a winner and with a neighborhood dispersion of h. The decay function consists of simply multiplying the two given discounts, γ , for the learning rate and the neighborhood distance.

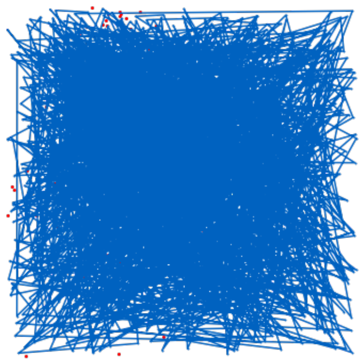
$$\alpha t + 1 = \gamma\alpha \cdot \alpha t, ht + 1 = \gamma h \cdot ht$$

Finally, to obtain the route from the SOM, we associate a city with its winner neuron, traverse the ring starting from any point and sort the cities by order of appearance of their winner neuron in the ring. If several cities map to the same neuron, it is because the order of traversing such cities have not been contemplated by the SOM (due to lack of relevance for the final distance or because of not enough precision). In that case, any possible order can be considered for such cities.

.3.2

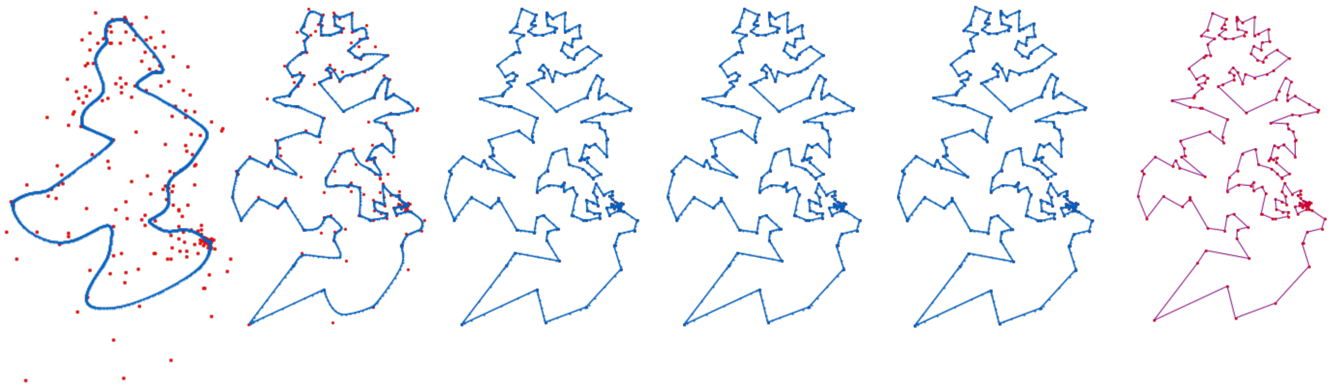
Results of the network with the learning rate of 0.8 after 100000 Iterations:

Before the training:



Training until decay or finishing iterations:

```
Radius has completely decayed, finishing execution at 24487 iterations
Route found of length 9852.426235838131
```



Resources:

https://diego.codes/post/som-tsp/