ECE 657 Assignment 2

Akshay Pala

Uche McDaniel Elekwa

Oghenerukevwe Ahwin

Question 1

$$J(n) = \frac{1}{2}|e(n)|^2 = \frac{1}{2}[y_d(n) - y(n)]^2 \quad -(1)$$

$$y(n) = \sum_{K=1}^{N} w_K(n)\,\phi\{x(n), c_K, \sigma_K\} \quad -(2)$$

$$J(n) = \frac{1}{2}\left[y_d(n) - \sum_{K=1}^{N} w_K(n)\,\phi\{x(n), c_K, \sigma_K\}\right]^2 \quad -(3)$$

$$J(n) = \frac{1}{2}\left[y_d(n) - \sum_{K=1}^{N} w_K(n)\, e^{-\left[\frac{(\|x(n)-c_K(n)\|)^2}{2\sigma_K^2(n)}\right]^2}\right]^2 \quad -(4)$$

$$w(n+1) = w(n) - \mu_w \frac{\partial}{\partial w} J(n)\Big|_{w=w(n)} \quad -(5)$$

$$\psi(n) = [\phi\{x(n), c_1, \sigma_1\}, \ldots \phi\{x(n), c_N, \sigma_N\}]^T \quad -(6)$$

a. Show that: $w(n+1) = w(n) + \mu_w e(n)\psi(n)$ -(7)

Comparing (5) and (7)

$$\mu_w e(n)\psi(n) = -\mu_w \frac{\partial}{\partial w} J(n)\Big|_{w=w(n)}$$

Comparing (5) and (7)

$$\mu_w e(n)\psi(n) = -\mu_w \frac{\partial J(n)}{\partial w}\Big|_{w=w(n)}$$

$$e(n)\psi(n) = -\frac{\partial J(n)}{\partial w}\Big|_{w=w(n)} \quad -(8)$$

$$\frac{\partial J(n)}{\partial w} = \frac{\partial J(n)}{\partial e(n)} * \frac{\partial e(n)}{\partial w(n)} \quad \text{– From chain rule}$$

$$J(n) = \frac{1}{2}|e(n)|^2 \quad \therefore \frac{\partial J(n)}{\partial e(n)} = e(n)$$

$$e(n) = y_d(n) - \sum_{k=1}^{N} w_k(n)\,\phi\{x(n), c_k, \sigma_k\}$$

$$\frac{\partial e(n)}{\partial w(n)} = -\sum_{k=1}^{N} \phi\{x(n), c_k, \sigma_k\}$$

$$= -\left[\phi\{x(n), c_1, \sigma_1\}, \cdots \phi\{x(n), c_N, \sigma_N\}\right]^T$$

$$\therefore \frac{\partial J(n)}{\partial w(n)} = e(n) \cdot -\left[\phi\{x(n), c_1, \sigma_1\}, \cdots \phi\{x(n), c_N, \sigma_N\}\right]^T$$

Remember (6)

$$\therefore \frac{\partial J(n)}{\partial w(n)} = -e(n)\psi(n)$$

From (8)

$$e(n)\psi(n) = -\frac{\partial J(n)}{\partial w(n)}$$

$$\therefore w(n+1) = w(n) + \mu_w e(n)\psi(n)$$

b. $$c_k(n+1) = c_k(n) - \mu_c \frac{\partial}{\partial c_k} J(n)\Big|_{c_k = c_k(n)} \quad -(9)$$

Show that: $$c_k(n+1) = c_k(n) + \mu_c \frac{e(n) w_k(n)}{\sigma_k^2(n)} \phi\{x(n), c_k(n), \sigma_k\} [x(n) - c_k(n)] \quad -(10)$$

comparing (9) and (10)

$$\frac{e(n) w_k(n)}{\sigma_k^2(n)} \phi\{x(n), c_k(n), \sigma_k\} [x(n) - c_k(n)] = -\frac{\partial J(n)}{\partial c_k}\Big|_{c_k = c_k(n)} \quad -(12)$$

$$\frac{\partial J(n)}{\partial c_k} = \frac{\partial J(n)}{\partial e(n)} * \frac{\partial e(n)}{\partial c_k}$$

From (a), $\frac{\partial J(n)}{\partial e(n)} = e(n)$

From (i), $e(n) = y_j(n) - \sum_{k=1}^{N} w_k(n)\, e^{-\left(\frac{\|x(n) - c_k(n)\|^2}{2\sigma_k^2(n)}\right)}$

let $z = \frac{-\|x(n) - c_k(n)\|^2}{2\sigma_k^2(n)}$ — (ii)

$$\frac{\partial e(n)}{\partial c_k} = \frac{\partial e(n)}{\partial z} * \frac{\partial z}{\partial c_k}$$

From (ii), $e(n) = y_j(n) - \sum_{k=1}^{N} w_k(n)\, e^{z}$

$$\frac{\partial e(n)}{\partial z} = -\sum_{k=1}^{N} w_k(n)\, e^{z}$$

$$z = -\left(\frac{x(n)^2 - 2x(n)c_k(n) + c_k(n)^2}{2\sigma_k^2(n)}\right)$$

$$= \frac{2x(n)c_k(n) - c_k(n)^2 - x(n)^2}{2\sigma_k^2(n)}$$

$$\frac{\partial z}{\partial c_k} = \frac{2x(n) - 2c_k(n)}{2\sigma_k^2(n)} = \frac{x(n) - c_k(n)}{\sigma_k^2(n)}$$

$$\frac{\partial e(n)}{\partial c_k} = \frac{x(n) - c_k(n)}{\sigma_k^2(n)} * -\sum_{k=1}^{N} w_k(n)\, e^{z}$$

$$\therefore \frac{\partial J(n)}{\partial c_k(n)} = e(n) * \frac{x(n) - c_k(n)}{\sigma_k^2(n)} * -\sum_{k=1}^{N} w_k(n)\, e^{z}$$

comparing (3) and (4)

$$\phi\{x(n), c_k(n), \sigma_k(n)\} = e^{-\frac{\|x(n) - c_k(n)\|^2}{2\sigma_k^2(n)}}$$

$$\frac{\partial J(n)}{\partial c_k(n)} = -\frac{e(n)\, w_k(n)}{\sigma_k^2(n)} \phi\{x(n), c_k(n), \sigma_k(n)\} (x(n) - c_k(n))$$

the above is the equivalent to (12) from earlier.

$$\therefore c_k(n+1) = c_k(n) + \mu_c \frac{e(n) w_k(n)}{\sigma_k^2(n)} \phi\{x(n), c_k(n), \sigma_k\} [x(n) - c_k(n)]$$

c. $$\sigma_k(n+1) = \sigma_k(n) - \mu_\sigma \frac{\partial}{\partial \sigma_k} J(n) \Big|_{\sigma_k = \sigma_k(n)} \quad -(13)$$

show that:

$$\sigma_k(n+1) = \sigma_k(n) + \mu_\sigma \frac{e(n) w_k(n)}{\sigma_k^3(n)} \frac{\phi\{x(n), c_k(n), \sigma_k\}}{(\|x(n) - c_k(n)\|)^2} \quad -(14)$$

comparing (13) and (14)

$$\frac{e(n) w_k(n)}{\sigma_k^3(n)} \frac{\phi\{x(n), c_k(n), \sigma_k\}}{(\|x(n) - c_k(n)\|)^2} = -\frac{\partial J(n)}{\partial \sigma_k} \quad -(15)$$

$$\frac{\partial J(n)}{\partial \sigma_k(n)} = \frac{\partial J(n)}{\partial e(n)} * \frac{\partial e(n)}{\partial \sigma_k(n)}$$

From (a), $$\frac{\partial J(n)}{\partial e(n)} = e(n)$$

From (4), $$e(n) = y_j(n) - \sum_{k=1}^{N} w_k(n)\, e^{-\left(\frac{\|x(n) - c_k(n)\|^2}{2\sigma_k^2(n)}\right)}$$

let $$z(n) = -\|x(n) - c_k(n)\|^2 \quad -(16)$$

Using chain rule,

$$\frac{\partial e(n)}{\partial \sigma_k(n)} = \frac{\partial e(n)}{\partial z(n)} * \frac{\partial z(n)}{\partial \sigma_k(n)}$$

From (5) $\frac{\partial e(n)}{\partial z} = -\sum_{k=1}^{N} w_k(n) e^z$

$$z = -\left(\frac{[x(n) - c_k(n)]^2}{2} \right) * \sigma_k^{-2}(n)$$

$$\frac{\partial z}{\partial \sigma_k} = -2 * (x(n) - c_k(n))^2 * \sigma_k^{-3}(n)$$

$$= (x(n) - c_k(n))^2 \, \sigma_k^{-3}(n)$$

$$\therefore \frac{\partial e(n)}{\partial \sigma_k(n)} = \frac{|x(n) - c_k(n)|^2}{\sigma_k^3(n)} * -\sum_{k=1}^{N} w_k(n) e^z$$

$$\therefore \frac{\partial J(n)}{\partial \sigma_k(n)} = e(n) * \frac{|x(n) - c_k(n)|^2}{\sigma_k^3(n)} * -\sum_{k=1}^{N} w_k(n) e^z$$

Remember from (3) and (4)

$$\phi\{x(n), c_k(n), \sigma_k(n)\} = e^{-\frac{||x(n) - c_k(n)||^2}{2\sigma_k^2(n)}}$$

$$\therefore \frac{\partial J(n)}{\partial \sigma_k(n)} = -\frac{e(n) w_k(n)}{\sigma_k^3(n)} \phi\{x(n), c_k(n), \sigma_k(n)\} [|x(n) - c_k(n)|]^2$$

... proving (15)

Hence,

$$\sigma_k(n+1) = \sigma_k(n) + \mu_\sigma \frac{e(n) w_k(n)}{\sigma_k^3(n)} \phi\{x(n), c_k(n), \sigma_k(n)\} [|x(n) - c_k(n)|]^2$$

Question 2: The Capacity Of Hopfield Associative Memory

The Hopfield Associative memory established prior to this publication was intended to demonstrate capabilities that allow nascent neural networks to some degree recall a pattern/memory from a partial version of it. This paper "*The Capacity of Hopfield Associative Memory*" was introduced to analyze the internal structure of Hopfield Networks.

On the high level, the state of each neuron is represented by two bi-stable elements of value 1 and -1 representing one bit of information. The information flows through each neuron transmitting using synaptic weight T such that neuron i transmitting to neuron j represents T_{ij}. T is considered fixed because the learning has taken place. The connection matrix recovers one of m memories using probe vector p where the probe vector < hamming distance $n/2$ away from the fundamental memory; where n is the number of neurons in the network and the fundamental memory is the original memory. The present state of the system can be defined by x. The neurons handle the computation where each node's new value is given by computing other nodes components and return -1 if the sum is negative and +1 for sum equalling or exceeding zero. With x_j =$\pm$1 representing the j^{th} neuron, the new state of the i^{th} neuron is determined by: $x\_i = sgn[\sum\_\{j = 1\}\^\{n\}\{T\_\{ij\}x\_\{ij\}\}] = +1 or -1$ where +1if = ≥ 0 and -1 if < 0.

The paper also presented two modes of changing the states x *Synchronously* or *Asynchronously* and as well highlighted the indication of the computational power existing in the system which is because of the Hebbian Theory and Associative recall. Synchronously updating the states involves simultaneous computation while asynchronously updating the state involves updating each state individually. The memory is defined such that they are fixed points in a system; It is defined mathematically where if the binary n-vector is a memory then for each neuron i = 1,….,n we have $x\_i = sgn[\sum_{j=1}^{n} T_{ij}x_{ij}]$.

These memories were also required to be *attractors* to influence memories around them with similar states and to map to their memory by repeated iteration. If a memory is close to the probe vector, the system will shift so it proceeds towards network stability centered around that memory then correct majority of the error in the probe vector (Fundamental Vector).

With the sequences of association and memory within the network structure, issues were raised: The nature of the memory encoding rules that will allow a desired structure of associations to be programmed into the network and the capacity of the resultant network to recall memories with some measure of error correction.

As we go further, we will cover sections of contributions this paper has made in the analysis of Hopfield Network: The sum of outer products connection matrix construction which is the basis for the results and connections of all parameters mentioned in the paper, Use of other possible connection matrices was talked about briefly but not implemented due to its overall difficulty and complexity, use of examples, the analysis of the various kinds of memory stability where the radius of attraction around fixed points was introduced along with some modes of convergence, later on there will be a summary on the inclusion of the concept of asymptotic capacity as the capacity heuristics and how capacity is measured in growth rather than an exact number, the other contributions from the research paper injected lemmas which were used to solidify the hypothesis scripted on their research paper.

Outer Product Connection: The paper talks about the encoding rule where $x = x^{(1)}, x^{(2)}, \dots, x^{(m)}$ is an m-set of n-dimensional binary ± 1 column vectors which are to be stored and are known as the fundamental memories. For each memory, $x^{(a)}$ the $n\,x\,n$ matrix is formed such that $T_a = x^{(a)}\left(x^{(a)}\right)^T . -I_n$ where I denotes the n x n identity matrix and Thus $T_{(a)}$ is the outer product of itself. In summary, the Hopfield connection matrix for the set of m memories is defined as $T = \sum_{a=1}^{m} * T_a$. Which is the sum of the outer products. Once T has also been calculated, all the information regarding $x^{(a)}$ is forgotten.
The update of the component of x happens by randomly and independently replacing the i_{th} component of vector T^x

Alternative Connection Matrix: There are other possible connection matrices regarded here required having m fundamental memories as fixed points. This means that the fundamental memories be exactly ordinary *eigenvectors* . λ With this, it was postulated that the memories will certainly have fixed points. Then; $sgn\left(\left(T_x^{(a)}\right)i\right) = sgn\left(\lambda^{(a)}(a)x_i^{(a)}\right) = x_i{}^a$.

Examples: In the examples section, the parameters used were values n = 5 representing the number of neurons needed to store three fundamental memories. $x^{(1)} = (+ + + + +)^T$ $x^{(2)} = (+ - - + -)^T$ $x^{(3)} = (- + - - -)^T$ and included a probe vector $x = (+ \; - \; - \; + +)^T$ at distance 1 from $x^{(2)}$. After a series of computation using their hypothetical algorithm, they had some findings where if they begin with a probe x, the fixed point: may not be one of the memories or it may not be the nearest memory. The study of the convergence behavior of Hopfield's algorithm is very complex. In their example, all three fundamental memories however do happen to be fixed.

Stability: The fundamental memories is expected to be recoverable in some sense, it was stated that there has to be at least some fixed points in the vector mapping to give $sgn(T_x)$. They also introduced some error correction by a generic assumption about a "forced choice" model in which, if some components are not known, they are guessed and are right half the time. Then going further, the paper talk about the possibilities of convergence for the asynchronous case; First, the sphere of radius pn may be directly or monotonically attracted to its fundamental memory center, Second, with a high enough probability but not probability 1, a random step is in the right direction, the third possibility while not mentioned in the paper stated that components can change back and forth during their sojourn.

Capacity Heuristics: The capacity of the memory was said to be represented as a growth function rather than a precise value. It was discussed in details on page 469 of the paper.

Lemmas are introduced in the rest of the paper section with equations to solidify the proofs and research so far made in the paper by providing rigorous results and tests.

This paper creates a clear internal mechanics on Hopfield network. The steps were well defined down from the abstraction to the execution. The Hopfield network has huge potentials in the

development of great algorithms for pattern matching and associative memory to retrieve lost or corrupted data

Question 3

Using all 441 points

Spread	MSE
1	0.0003
3	0.1157
5	0.2157
7	0.3588
9	0.4693
11	0.5821
13	0.5662
15	0.6553

The mean squared error increases as the spread of the function increases. A spread value of 1 seems to provide the most accurate results.

Random 150 points

Spread	MSE
1	0.8062
3	0.8128
5	0.8732
7	0.8743
9	0.8378
11	0.8805
13	0.8692
15	0.8782

K-Means

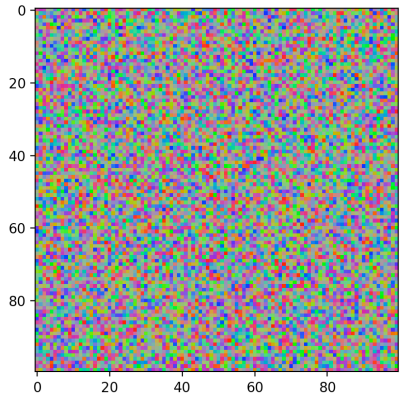
Spread	MSE
1	0.7817
3	0.7826
5	0.7848
7	0.7560
9	0.9222
11	0.8040
13	0.8744
15	0.8891

With the spread fixed at 1, the network performs better when using all 441 points as centres. However, the K-means approach seems to provide a more accurate result when compared to choosing 150 random centres.

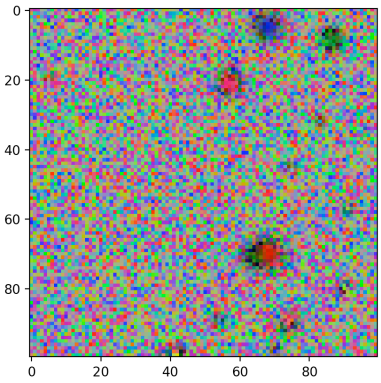
Question 4

This model was created using the MiniSom library in Python.

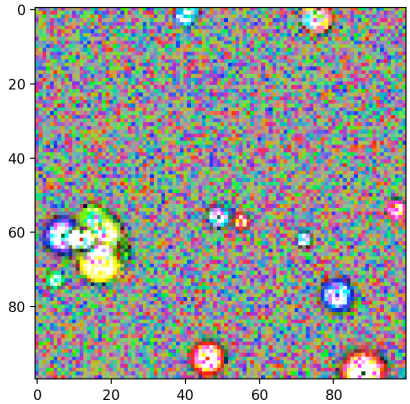
Untrained model



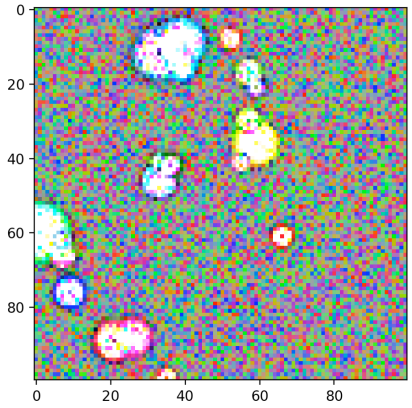
Epoch 20, Learning Rate of 1



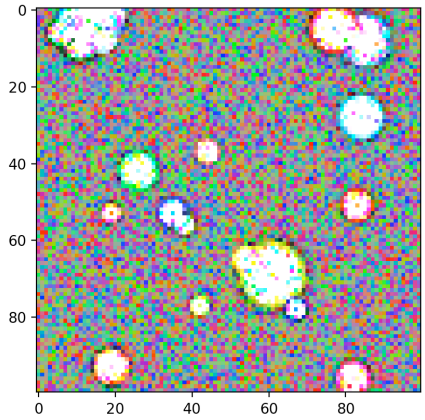
Epoch 20, Learning Rate of 10



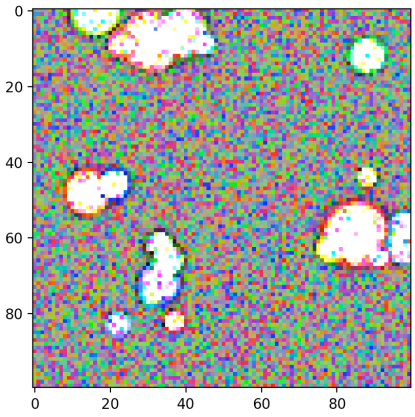
Epoch 20, Learning Rate of 30



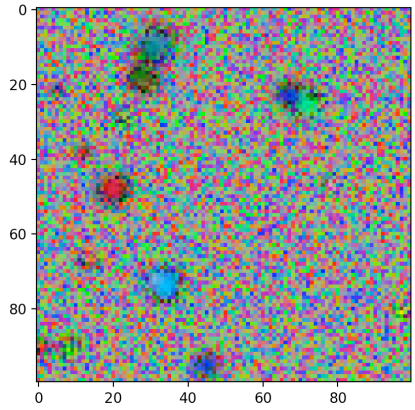
Epoch 20, Learning Rate of 50



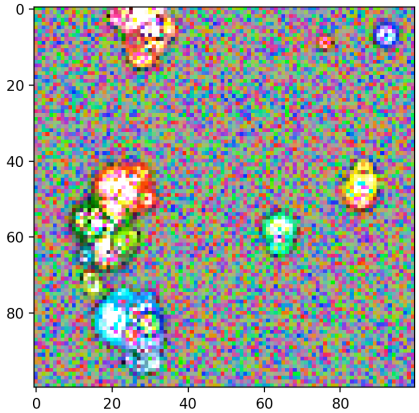
Epoch 20, Learning Rate of 70



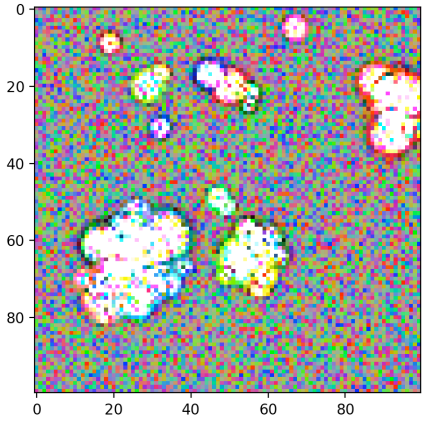
Epoch 40, Learning Rate 1



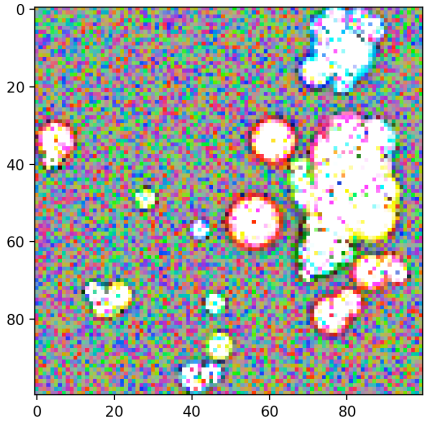
Epoch 40, Learning Rate of 10



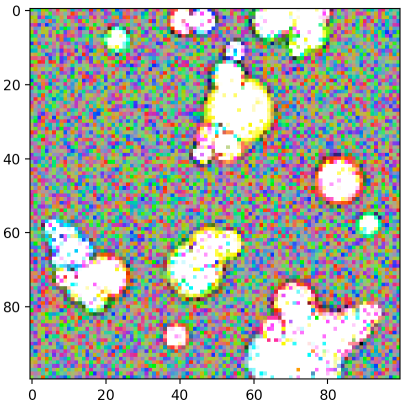
Epoch 40, Learning Rate 30



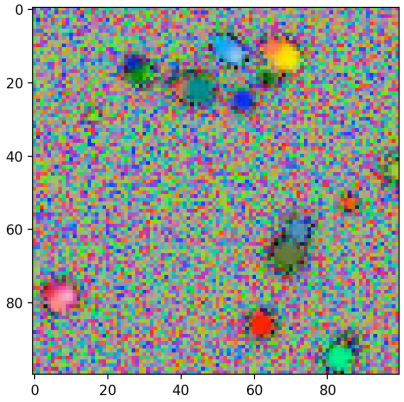
Epoch 40, Learning Rate of 50



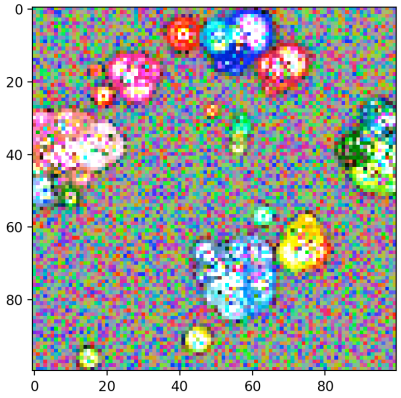
Epoch 40, Learning Rate of 70



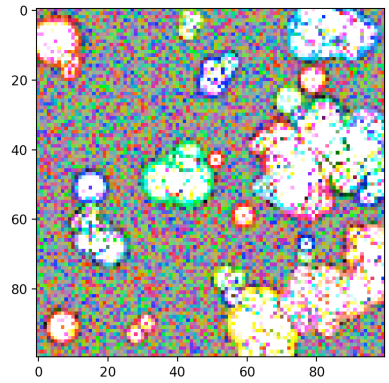
Epoch 100, Learning Rate 1



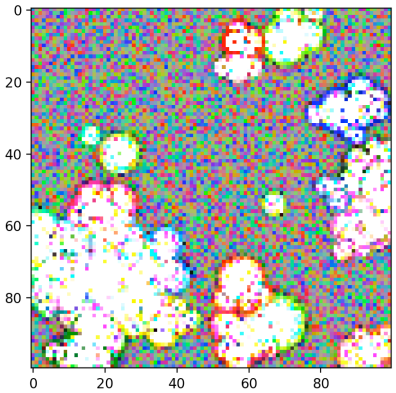
Epoch 100, Learning Rate 10



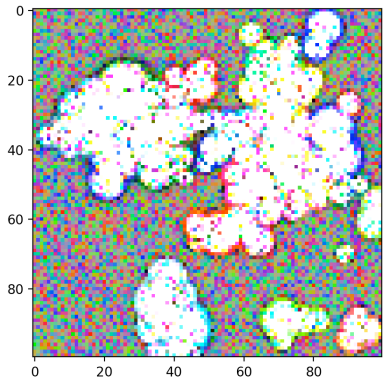
Epoch 100, Learning Rate of 30



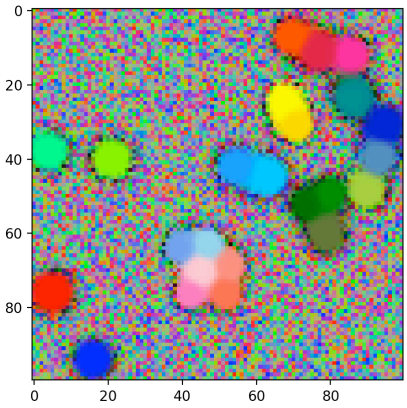
Epoch 100, Learning Rate of 50



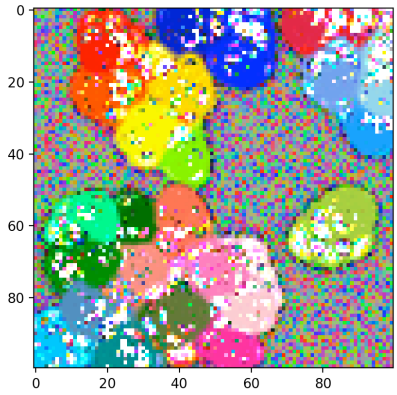
Epoch 100, Learning Rate of 70



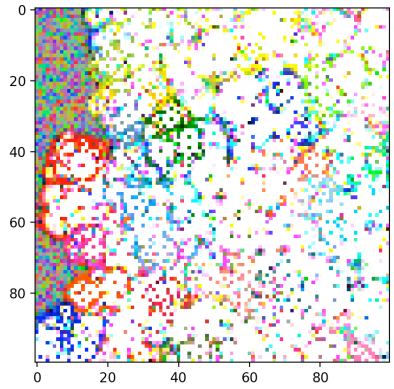
Epoch 1000, Learning Rate 1



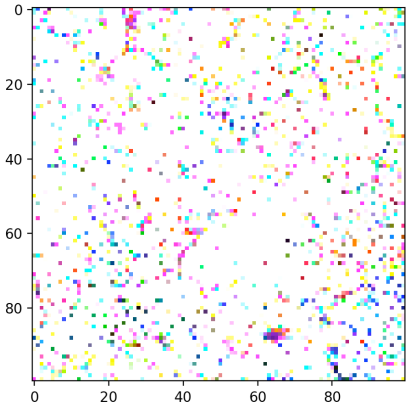
Epoch 1000, Learning Rate 10



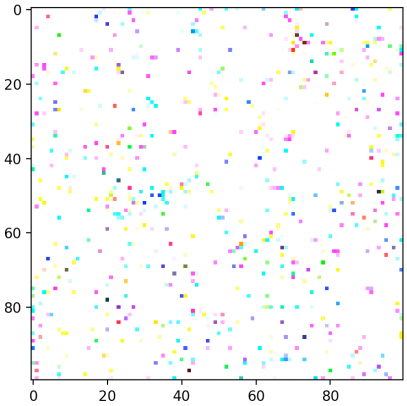
Epoch 1000, Learning Rate 30



Epoch 1000, Learning Rate 50



Epoch 1000, Learning Rate 70



There seems to be more distinguished clusters of colours as the number of epochs increases. Epochs 20-40 provided little to no clusters. Clusters were formed, however, they were not filled with their respective colours. The colours were prevalent around the borders of the clusters and white in the middle. The clusters started to become clearer once the model trained on 100 and 1000 epochs. They were no longer white but had their respective filled colour.

Learning rate also played a role in the formation of these clusters. As the learning rate increased, the number of clusters started to dissolve and become whiter. They basically lost the colour information. For the higher epochs and higher learning rates (30+), the network organized itself into essentially some whitespace with a few colour pixels distributed throughout. As a result, choosing a lower learning rate seems to provide the best formation of clusters. From our model, it seems like the best result occurred using 1000 epochs and a learning rate of 1 and 10.

References

Model used in Question 3:
Vidnerová, Petra. RBF-Keras: an RBF Layer for Keras Library. 2019. Available at https://github.com/PetraVidnerova/rbf_keras

Model used in Question 4:
Vettigli, Giuseppe. MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map. 2018. Available at https://github.com/JustGlowing/minisom/