CSE354N - ASSIGNMENT 10

220002018 - Arnav Jain

Question 1

```python
import numpy as np

class HopfieldNN:
    def __init__(self, data, activation, threshold = 0):
        self.threshold = threshold
        self.act = activation
        self.weights = (data.T@data)
        for i in range(len(self.weights)):
            self.weights[i,i] = 0

    def __call__(self, pattern, update_order = [1,4,3,2]):
        if not update_order:
            update_order = [i for i in range(len(pattern))]
        change = True
        state = pattern
        it = 0
        while change:
            it += 1
            change = False
            for i in update_order:
                i -= 1
                state[i] = self.weights[i]@state + pattern[i] - self.threshold
                state[i] = self.act(state[i])
                if state[i] - pattern[i]:
                    change = True
        print(f"Done in {it} iterations")
        return state
```

```python
def sign(n):
    if n > 0:
        return 1
    if n < 0:
        return -1
    return 0
```

```
[22]
data = np.array([
    [1,1,1,1],
    [1,1,-1,-1]
])

print(len(data))
```

```
2
```

```
[23]
hnn = HopfieldNN(data, sign, 0)
```

```
[24]
hnn([1,1,1,1])
```

```
Done in 1 iterations

[1, 1, 1, 1]
```

```
[25]
hnn([1,1,-1,-1])
```

```
Done in 1 iterations

[1, 1, -1, -1]
```

```
[26]
hnn([1,1,1,-1])
```

```
Done in 1 iterations

[1, 1, 1, 1]
```

Question 2

```python
import numpy as np

class HopfieldNN:
    def __init__(self, data, activation, threshold = 0):
        self.threshold = threshold
        self.act = activation
        self.weights = (data.T@data)
        for i in range(len(self.weights)):
            self.weights[i,i] = 0

    def __call__(self, pattern, update_order = [1,4,3,2]):
        if not update_order:
            update_order = [i for i in range(len(pattern))]
        change = True
        state = pattern
        it = 0
        while change:
            it += 1
            change = False
            for i in update_order:
                i -= 1
                state[i] = self.weights[i]@state + pattern[i] - self.threshold
                state[i] = self.act(state[i])
                if state[i] - pattern[i]:
                    change = True
        print(f"Done in {it} iterations")
        return state
```

```python
def sign(n):
    if n > 0:
        return 1
    if n < 0:
        return -1
    return 0
```

```
data = np.array([
    [1,1,1,0],
])

print(len(data))
```

```
1
```

```
hnn = HopfieldNN(data, sign, 0)
```

```

```

```
hnn([1,0,0,0])
```

```
Done in 1 iterations

[1, 1, 1, 0]
```

```
hnn([0,0,1,0])
```

```
Done in 1 iterations

[1, 1, 1, 0]
```

```
hnn([0,0,0,1])
```

```
Done in 1 iterations

[0, 0, 0, 1]
```

GitHub

https://github.com/arnavjain2710/Computational-Intelligence-Lab-CS354N/tree/main/LAB%209