HW 12

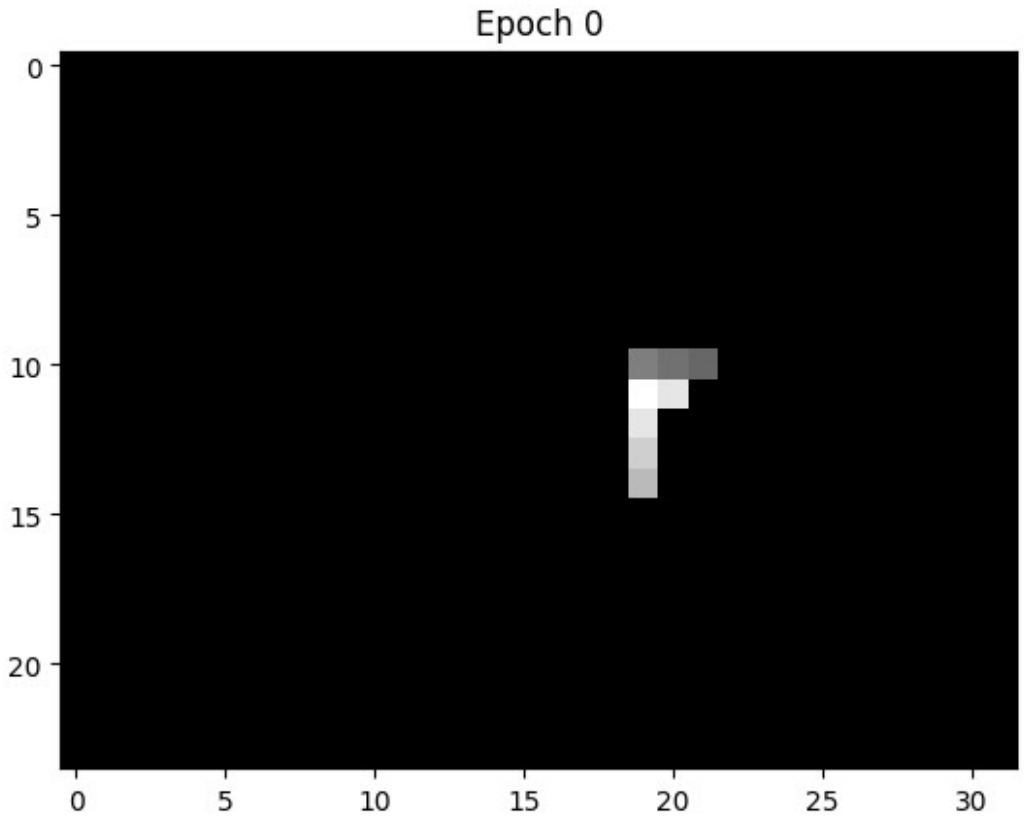
As always start with importing libraries and data.....

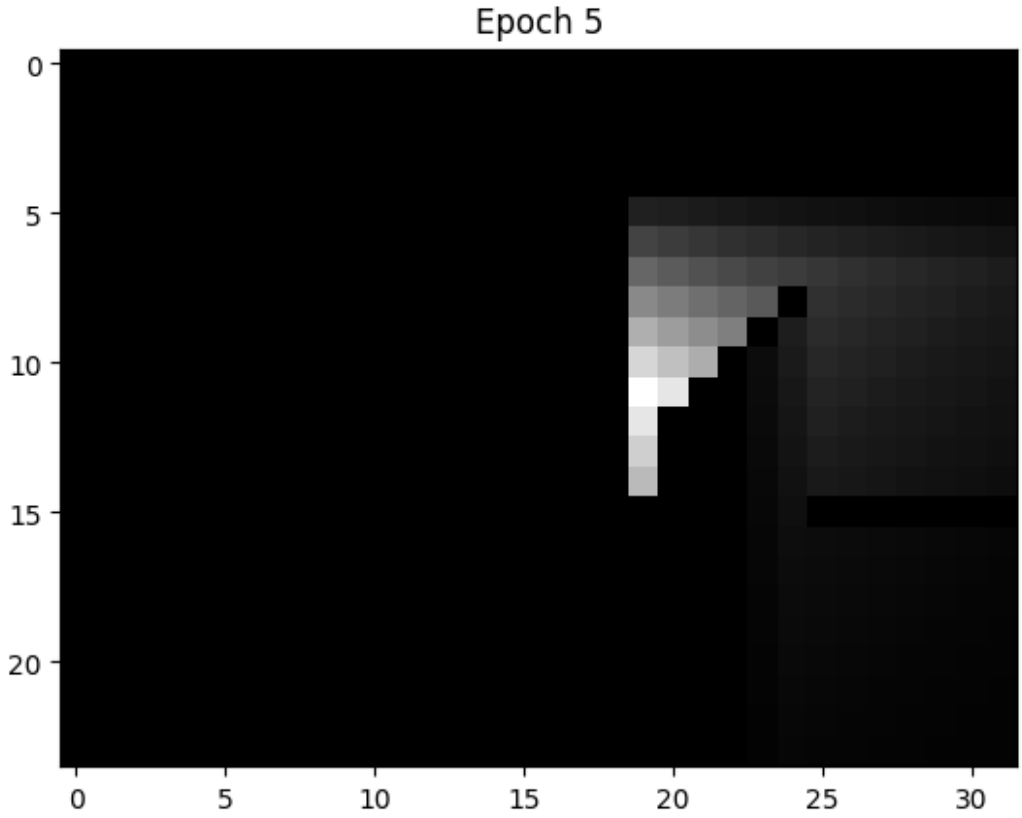
```
import numpy as np
import matplotlib.pyplot as plt

import os
os.chdir("C:/Users/rique/Downloads/datasets")

# Load the map from the CSV file
map_data = np.genfromtxt('map_24x32.csv', delimiter=',')

# Constants, actions and initalize V
gamma = 0.9
max_epochs = 50
valid_locations = map_data >= 0
V = np.copy(map_data)
V[~valid_locations] = 0
actions = [(-1, 0), (1, 0), (0, -1), (0, 1)]
action_symbols = ['L', 'R', 'U', 'D']
```

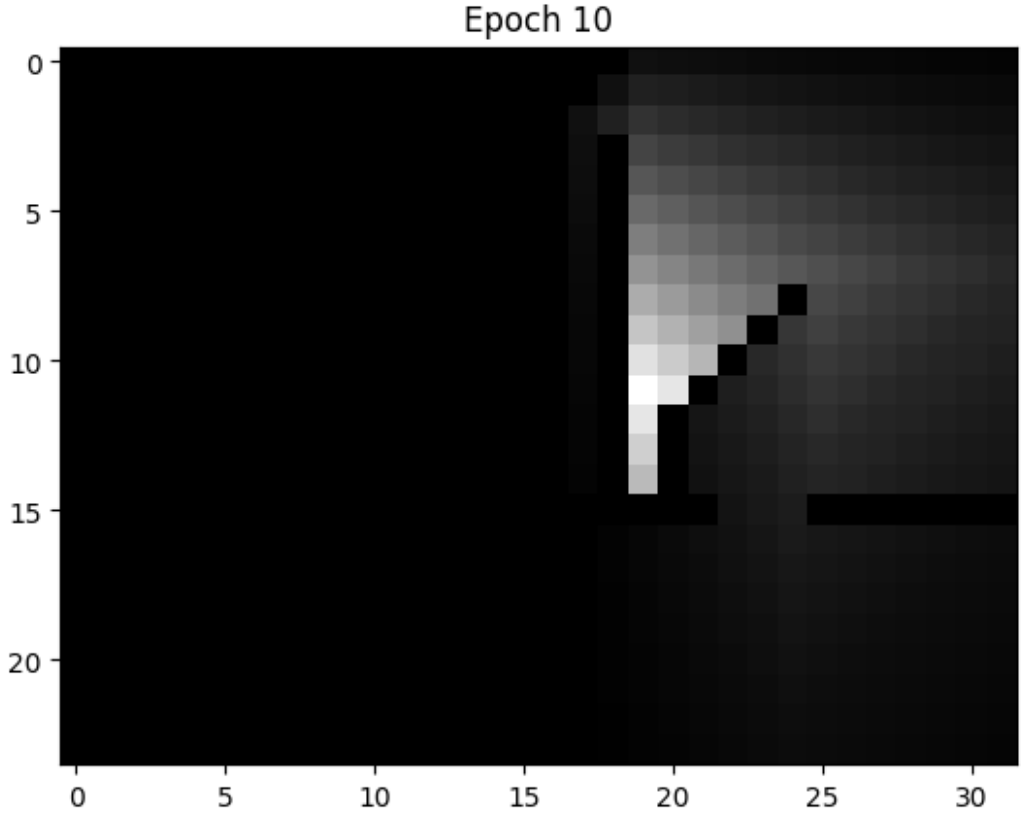
a) Displaying initial and obtained V for every 5 epochs

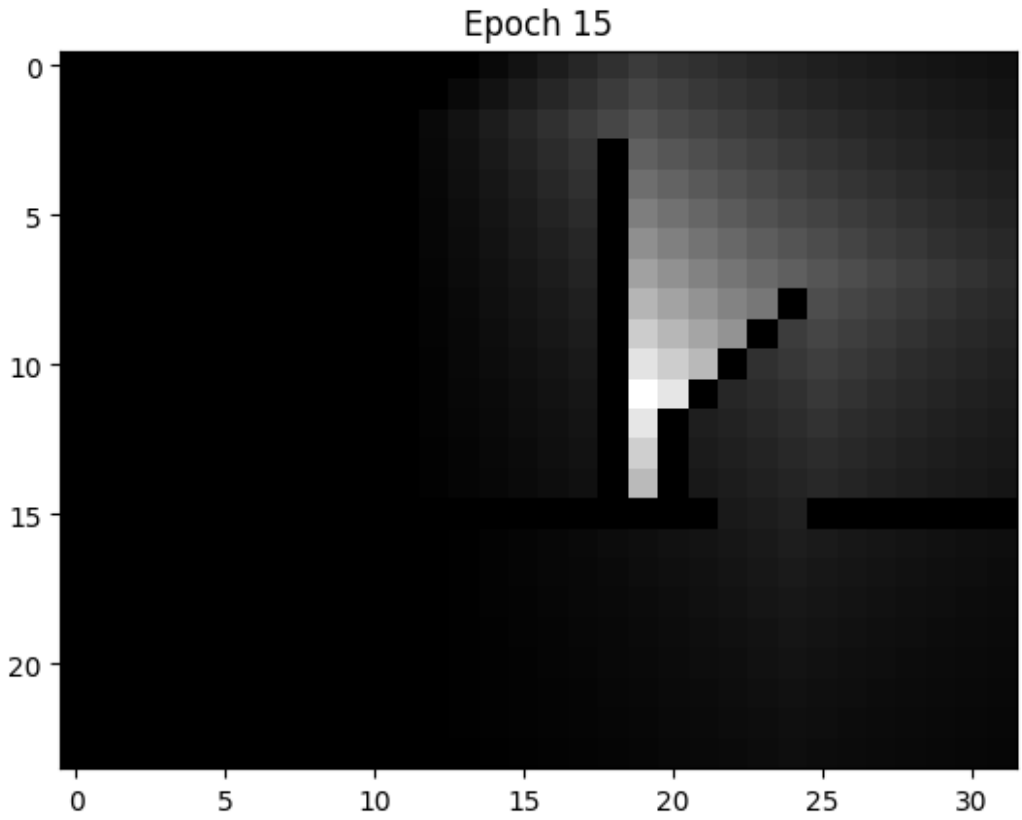
```
for epoch in range(max_epochs):
    for i in range(V.shape[0]):
        for j in range(V.shape[1]):
            if not valid_locations[i, j]:
                continue

            Q_values = []
            for action in actions:
                next_i, next_j = i + action[0], j + action[1]
                next_i = max(0, min(V.shape[0] - 1, next_i))  # Ensure
next_i is within bounds
                next_j = max(0, min(V.shape[1] - 1, next_j))  # Ensure
next_j is within bounds
                Q_values.append(V[next_i, next_j])

            V[i, j] = map_data[i, j] + gamma * max(Q_values)

    if epoch % 5 == 0:
        plt.imshow(V, cmap='gray', interpolation='none')
```

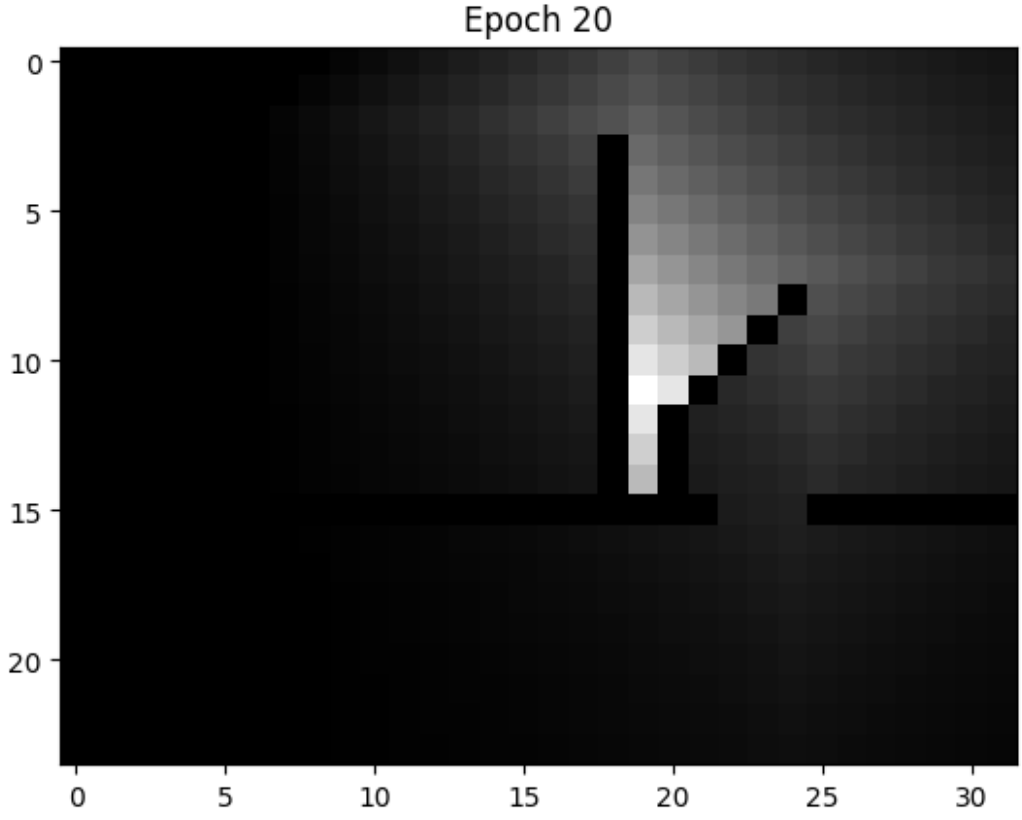
```
        plt.title(f'Epoch {epoch}')
        plt.show()
```

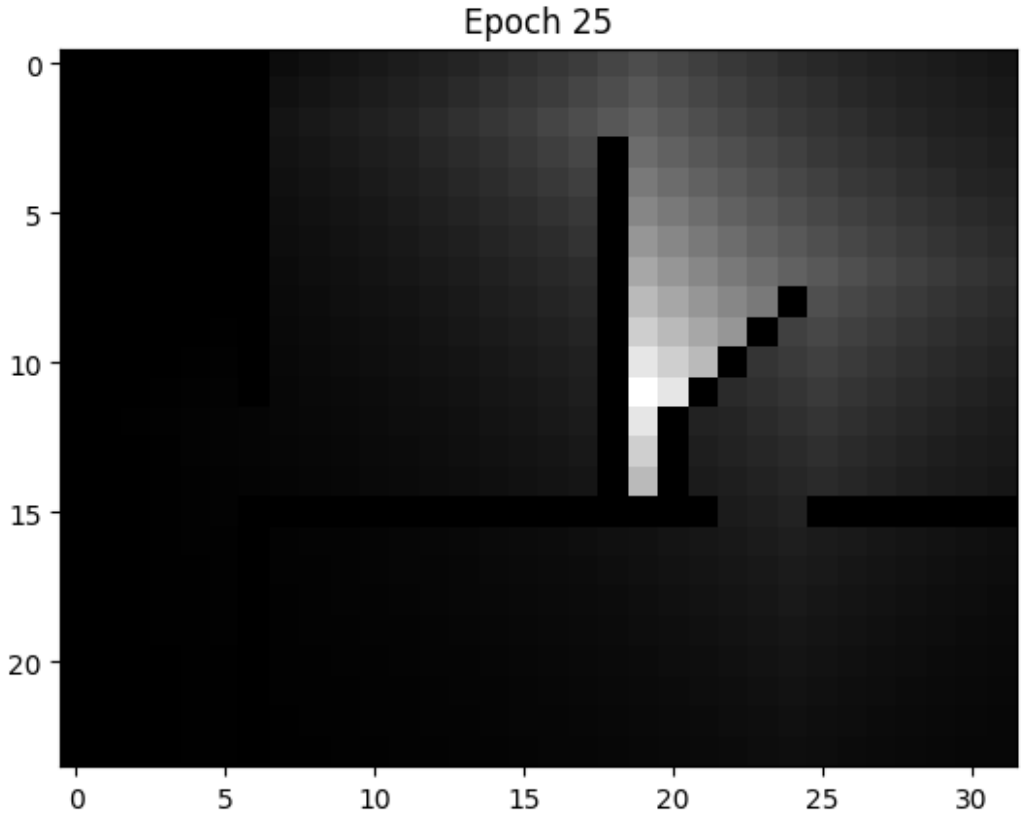


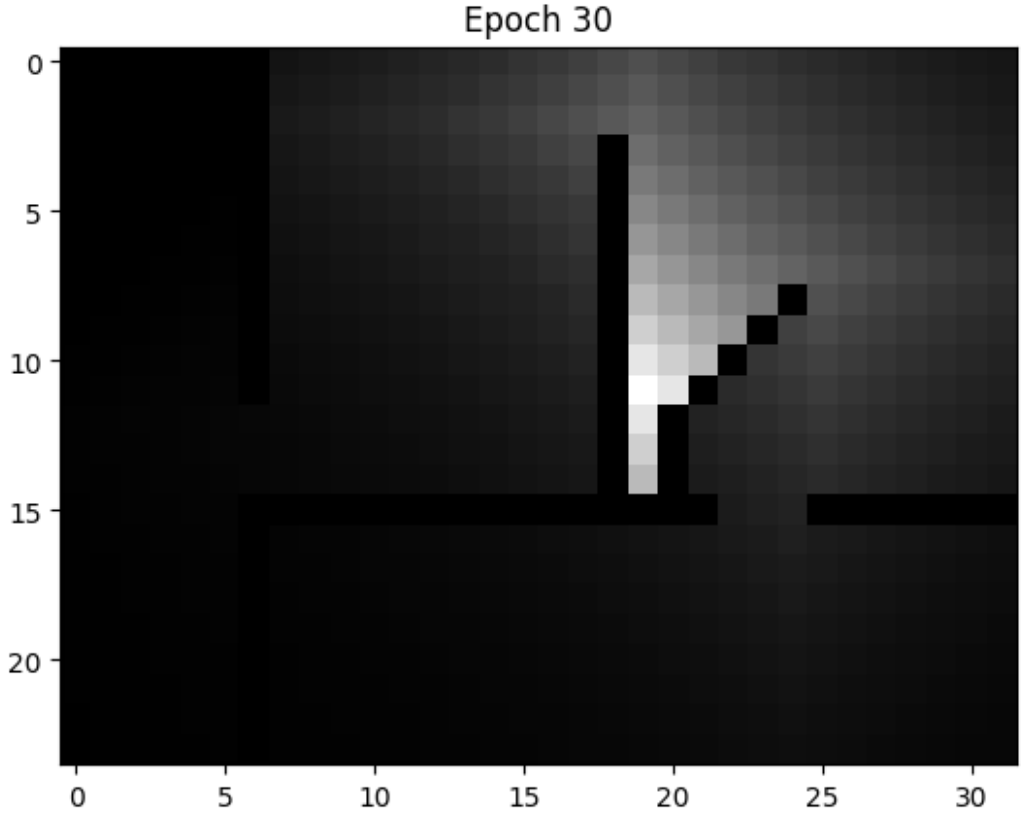

Epoch 5
0
5
10
15
20
0
5
10
15
20
25
30

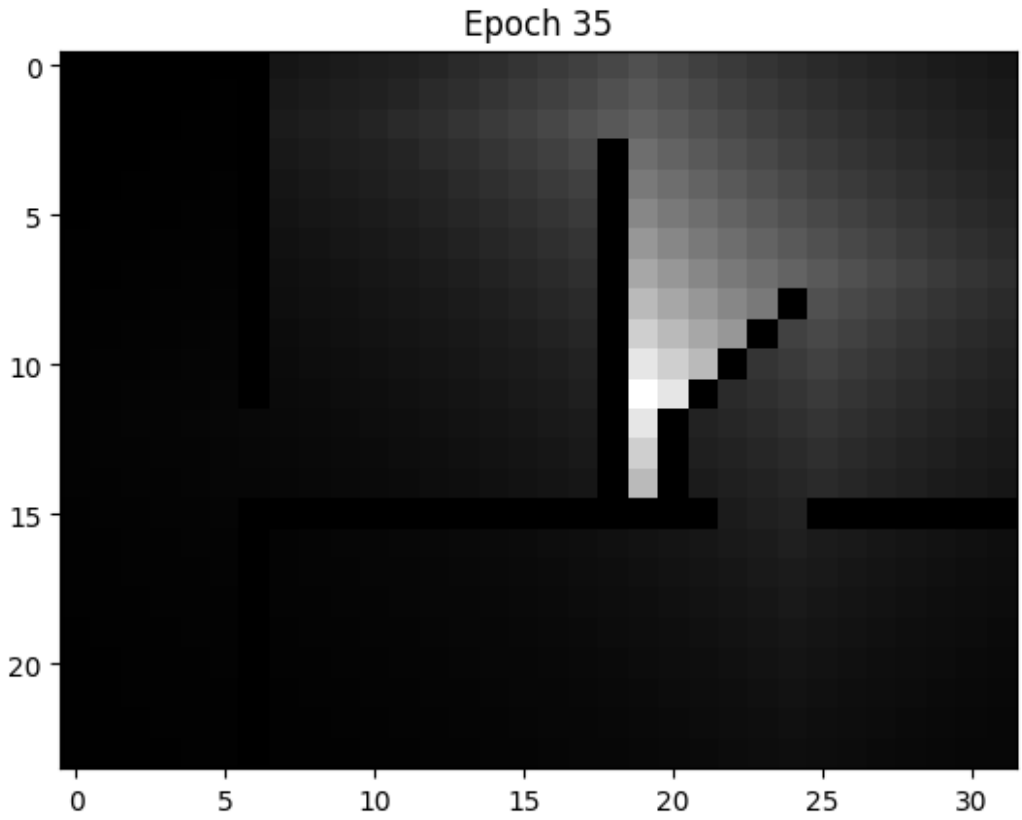

Epoch 10
0
5
10
15
20
0
5
10
15
20
25
30

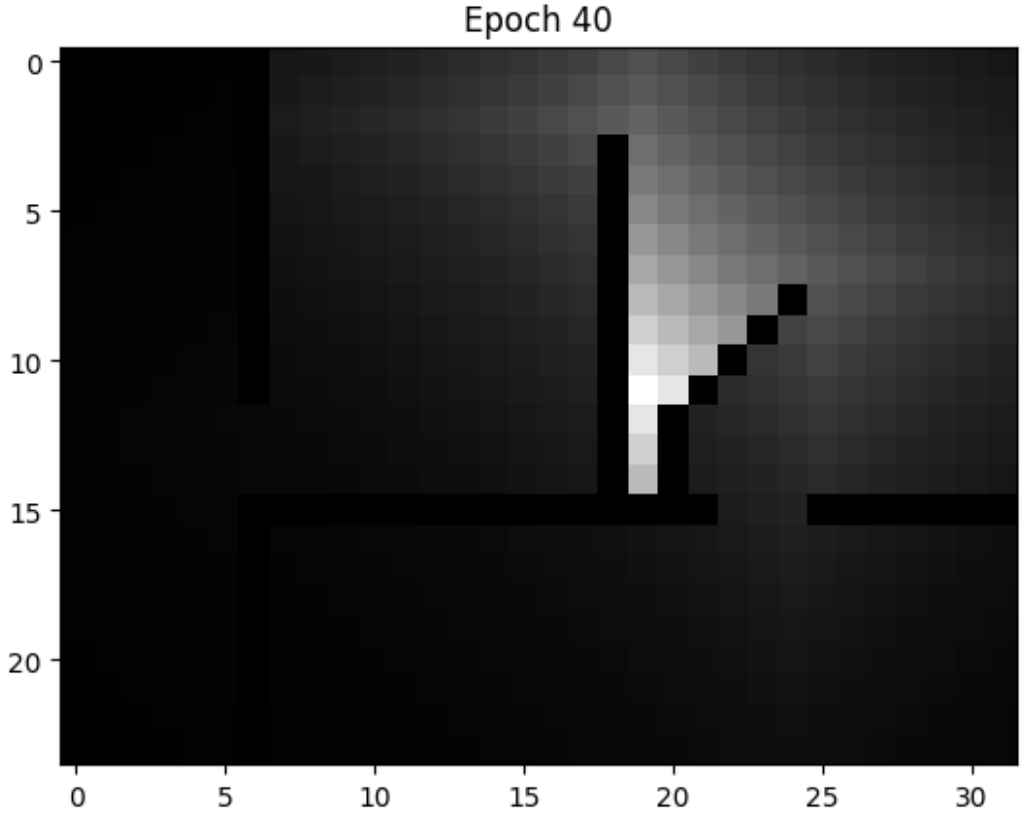

Epoch 15
0
5
10
15
20
0
5
10
15
20
25
30

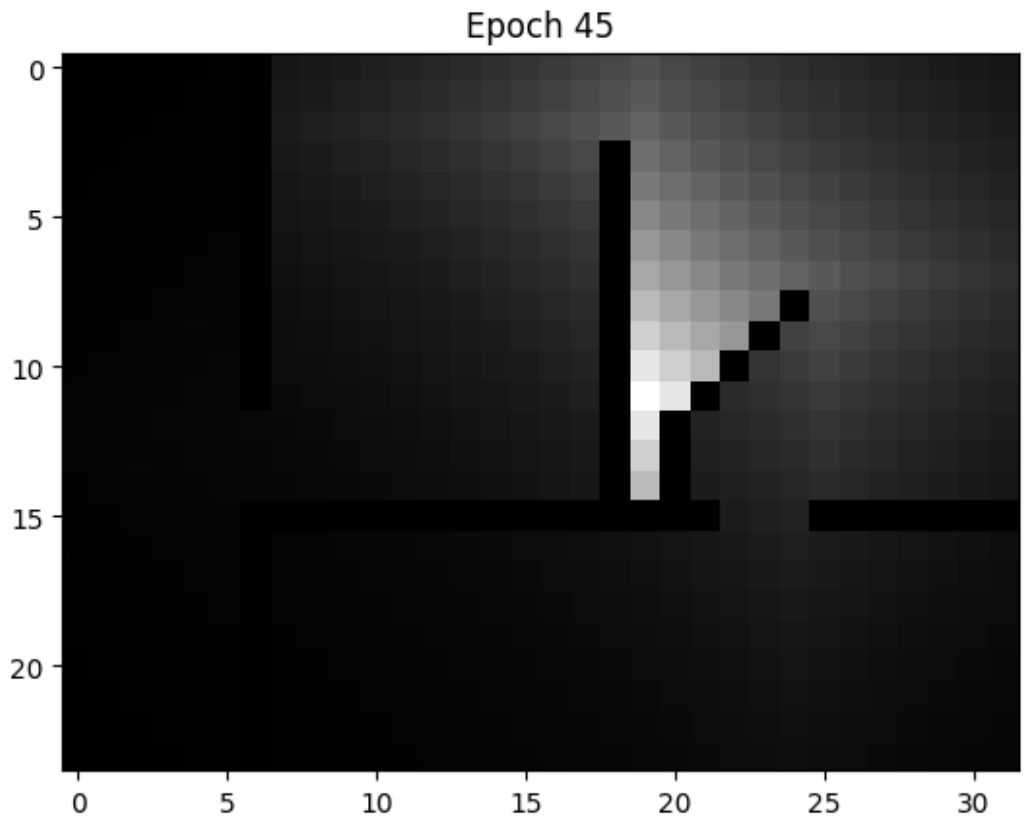

Epoch 20
0
5
10
15
20
0
5
10
15
20
25
30


Epoch 25
0
5
10
15
20
0
5
10
15
20
25
30


Epoch 30
0
5
10
15
20
0
5
10
15
20
25
30


Epoch 35
0
5
10
15
20
0
5
10
15
20
25
30


Epoch 40
0
5
10
15
20
0
5
10
15
20
25
30



b) final learned policy table

```
policy_table = np.empty_like(map_data, dtype='<U1')
policy_table[~valid_locations] = 'X'  # Mark invalid locations with
'X'

for i in range(V.shape[0]):
    for j in range(V.shape[1]):
        if valid_locations[i, j]:
            # Determine the best action for each valid location
            best_action_index = np.argmax([V[max(0, min(V.shape[0] -
1, i + a[0])), max(0, min(V.shape[1] - 1, j + a[1]))] for a in
actions])
            policy_table[i, j] = action_symbols[best_action_index]

print("Final Policy:")
print(policy_table)
```

```
Final Policy:
[['R' 'R' 'R' 'R' 'R' 'R' 'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R'
'R'
  'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R'
```

```
'R'
  'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R' 'R']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'R' 'U' 'U' 'U' 'U' 'U' 'U' 'U' 'U']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'R' 'U' 'X' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'R' 'U' 'X' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'R' 'U' 'X' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['R' 'R' 'R' 'R' 'R' 'R' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'R' 'U' 'X' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'L' 'X' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'L' 'X' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'L'
  'X' 'L' 'X' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'X' 'X' 'X' 'X' 'X' 'X' 'X' 'X' 'X' 'X'
'X'
  'X' 'X' 'X' 'X' 'D' 'D' 'L' 'X' 'X' 'X' 'X' 'X' 'X' 'X']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'U' 'U' 'U' 'U' 'U' 'U' 'U']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
```

```
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']
 ['D' 'D' 'D' 'D' 'D' 'L' 'X' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D' 'D'
'D'
  'D' 'D' 'D' 'D' 'D' 'D' 'L' 'L' 'L' 'L' 'L' 'L' 'L' 'L']]
```