Лабораторная работа № **6** по Нейроинформатике

Сети Кохонена

Выполнила**:** Тимофеева Наталья М**8**О**-408**Б**-19**

Вариант № **16**

Карта Кохонена

In [45]:

```
import numpy as np
import matplotlib.pyplot as plt
from timeit import default_timer as timer
from IPython.display import Image
import imageio
```

Создадим класс для карты Кохонена

In [58]:

```
class SOM():
    def __init__(self, width, height, in_features):
        self.nodes = np.random.randn(width * height, in_features)
        self.indices = np.array([[x, y] for x in range(height) for y in range(width)])
        self.initial_radius = self.radius = max(width, height)
        self.initial_lr = self.lr = 1
    def update(self, input):
        dist = np.linalg.norm(self.nodes - input, axis=1)
        min_dist_index = np.argmin(dist)
        dist_from_min = np.linalg.norm(self.indices - self.indices[min_dist_index], axis
=1)
        for dist, node in zip(dist_from_min, self.nodes):
            if dist < self.radius:
                influence = np.exp(-dist / (2 * self.radius))
                node += self.lr * influence * (input - node)
    def update_parameters(self, epoch, epochs):
        self.radius = self.initial_radius * np.exp(-epoch / (epochs / np.log(self.initia
l_radius)))
        self.lr = self.initial_lr * np.exp(-epoch / epochs)
```

Задаём точки

In [47]:

```
train_data = []
train_data += [[-1.3, -0.3]]
train_data += [[-0.8, -0.7]]
train_data += [[-0.3, -1.2]]
train_data += [[-0.1, 0.6]]
train_data += [[-0.8, 0.8]]
train_data += [[-1.3, -0.4]]
train_data += [[-1.5, -0.9]]
train_data += [[-1.5, -1]]
train_data += [[-1.1, -0.7]]
train_data += [[-1, -1.1]]
train_data += [[0.2, 1.2]]
train_data += [[1.3, -0.9]]
```

Создаём модель

In [48]:

```
epochs = 250
```

```
width =  64
heigth = 48
```

In [49]:

```
model = SOM(width=width, height=heigth, in_features=2)
```

Создаём функцию для визуализации обучения

In [50]:

```
def visual(data, epoch, images):
    image_to_show = np.dot(model.nodes[..., :2], [0.5, 0.5]).astype(np.float32).reshape(
(heigth, width, 1))
    plt.imshow(image_to_show)
    plt.savefig(f'./epoch_{epoch}.png', transparent=False, facecolor='white')
    images.append(imageio.v2.imread(f'./epoch_{epoch}.png'))
    plt.close()
```

Обучаем

In [57]:

```
def train(train_data, epochs):
    images = []
    for epoch in range(epochs):
        np.random.shuffle(train_data)
        progress = enumerate(train_data)
        for _, input_data in progress:
            model.update(input_data)
        visual(model.nodes, epoch + 1, images)
        model.update_parameters(epoch=epoch, epochs=epochs)
    imageio.mimsave('lab6.gif', images)
```

In [52]:

```
start_time = timer()
train(train_data, epochs=epochs)
end_time = timer()
print('Время обучения = {0} секунд'.format(int(end_time - start_time)))
print('Количество эпох = {0}'.format(epochs))
```

```
Время обучения = 50 секунд
Количество эпох = 250
```

In [53]:

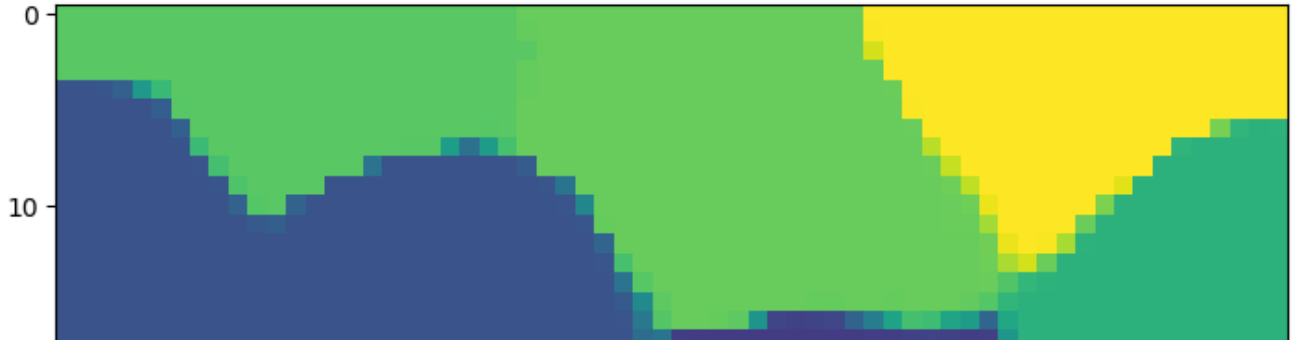
```
image_to_show = np.dot(model.nodes[..., :2], [0.5, 0.5]).astype(np.float32).reshape((hei
gth, width, 1))
```

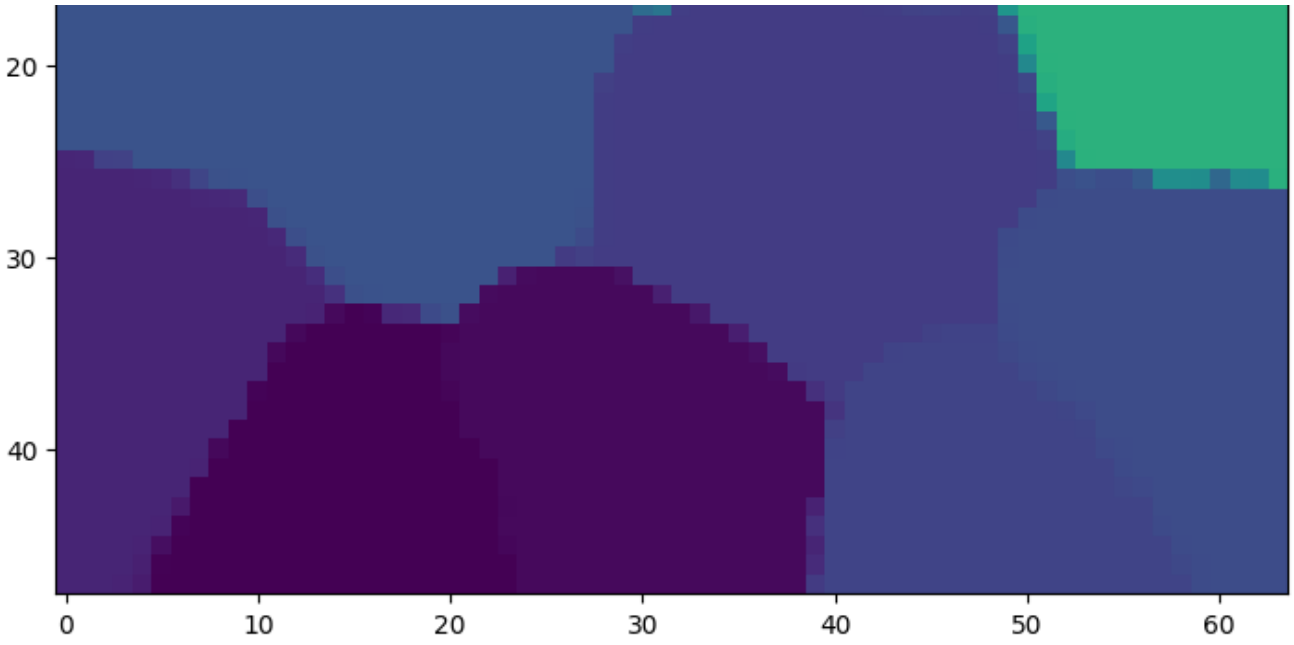
In [54]:

```
plt.figure(figsize=(10, 6.5))
plt.imshow(image_to_show)
```

Out[54]:

```
<matplotlib.image.AxesImage at 0x1b2beadf3a0>
```




20
30
40
0
10
20
30
40
50
60