

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Московский авиационный институт

(Национальный исследовательский университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра 806: «Вычислительная математика и программирование»

Лабораторная работа № 5

по курсу «Нейроинформатика»

Студент: А. О. Тояков

Преподаватель: Н. П. Аносова

Группа: М8о-4076-18

Дата:

Оценка:

Подпись:

Москва, 2021

## СЕТИ С ОБРАТНЫМИ СВЯЗЯМИ

*Цель работы:* Исследование свойств сетей Хопфилда, Хэмминга и Элмана, алгоритмов обучения, а также применение сетей в задачах распознавания статических и динамических образов.

*Основные этапы работы:*

1. Использовать сеть Элмана для распознавания динамических образов.  
Проверить качество распознавания.
2. Использовать сеть Хопфилда для распознавания статических образов.  
Проверить качество распознавания.
3. Использовать сеть Хэмминга для распознавания статических образов.  
Проверить качество распознавания.

*Вариант № 26:*

1.  $g(k) = \sin(k^2 - 10k + 3)$ , for all  $k$  in  $[2.5, 4.84]$ ,  $R = [1, 2, 3]$
2.  $[4, 0, 6]$

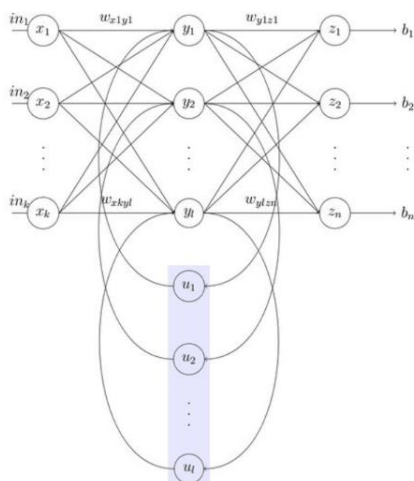
## СТРУКТУРА МОДЕЛИ

*Нейронная сеть Элмана:*

Сеть обладает скрытым слоем и производит расчет выходного слоя для  $t$ -го входного вектора по следующим формулам:

$$y_t = \sigma_y(W_y x_t + U_y y_{t-1} + b_y)$$

$$z_t = \sigma_z(W_z y_t + b_z)$$

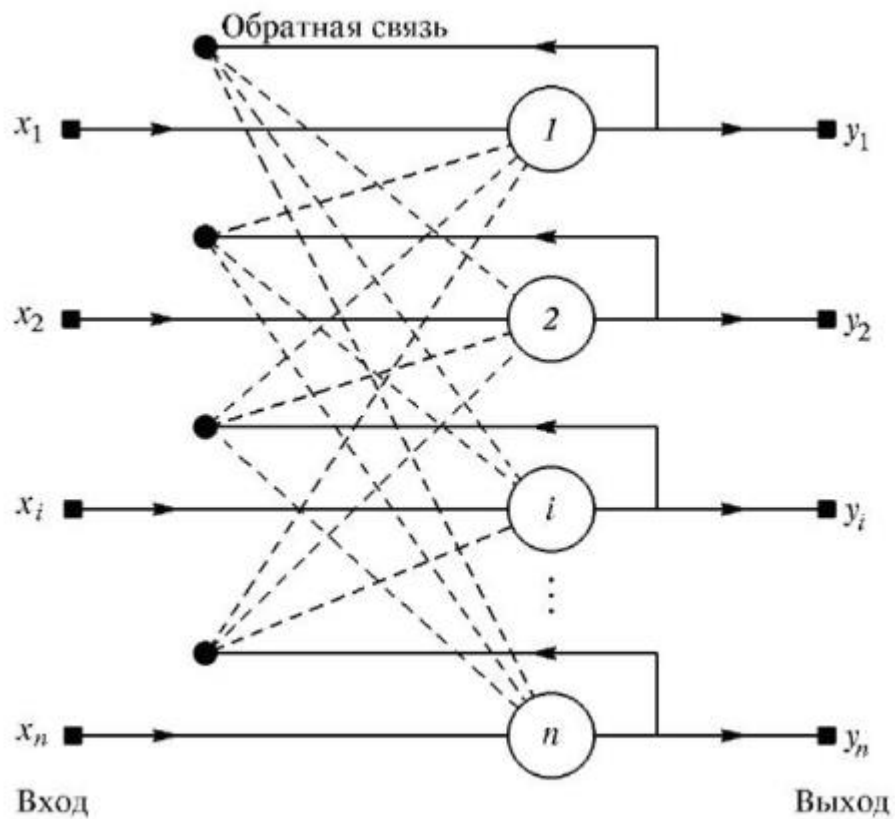


### Нейронная сеть Хопфилда:

Каждый нейрон слоя из  $n$  нейронов связывается обратными связями с остальными  $n - 1$  нейронами этого слоя. Сеть позволяет корректировать искаженные образы.

Веса этого слоя могут быть получены с помощью аналитической формулы:

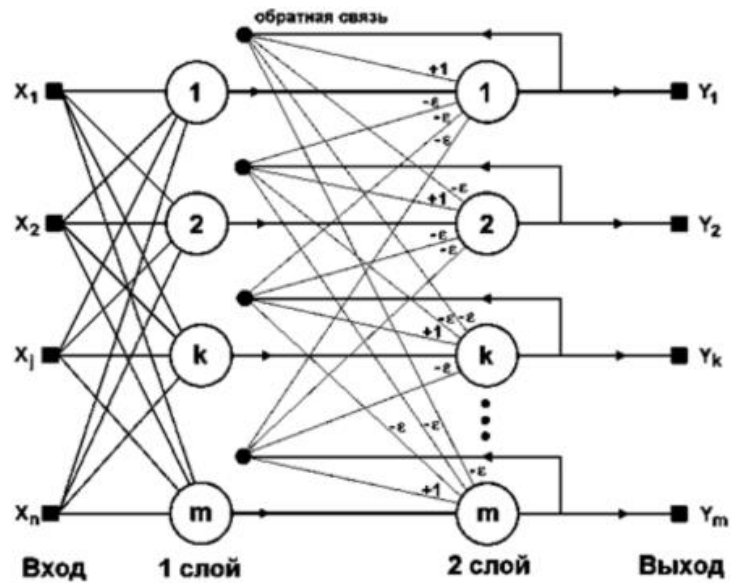
$$W = \frac{1}{N} \sum_{i=1}^N \langle x_i, x_i \rangle$$



### Нейронная сеть Хэмминга:

Сеть состоит из \*полносвязного слоя\* вместе со \*слоем сети Хопфилда\*. Сеть позволяет классифицировать образы, поданные на вход.

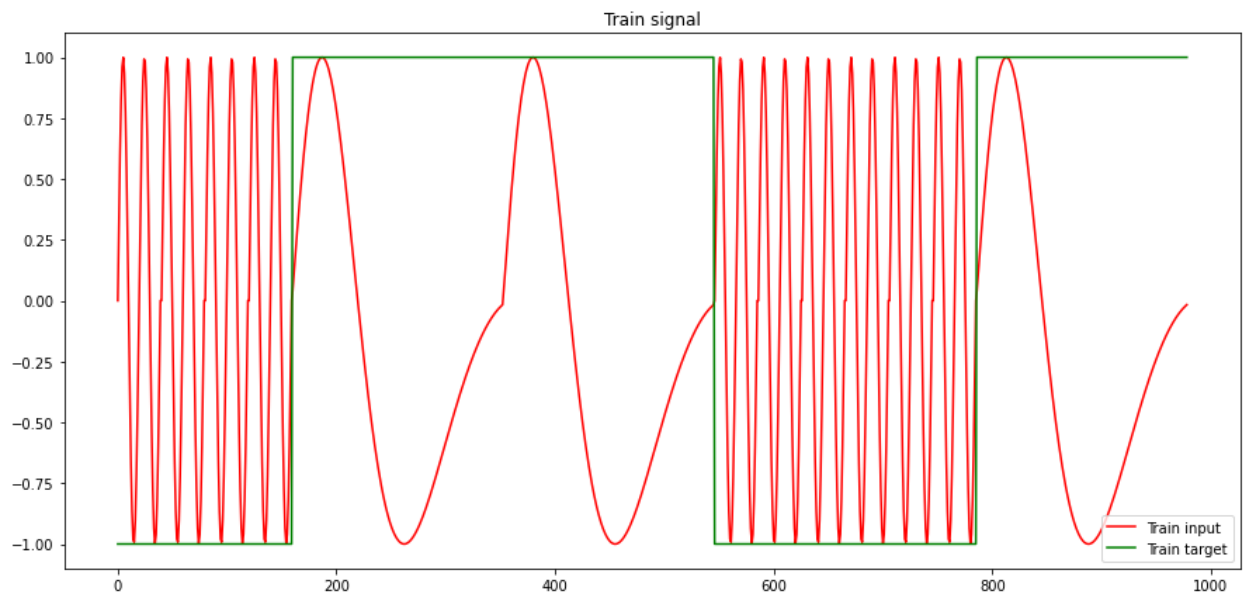
Веса полносвязного слоя могут быть найдены также аналитическим путем.



## ХОД РАБОТЫ

Этап № 1.

Я сгенерировал и отобразил обучающее множество:

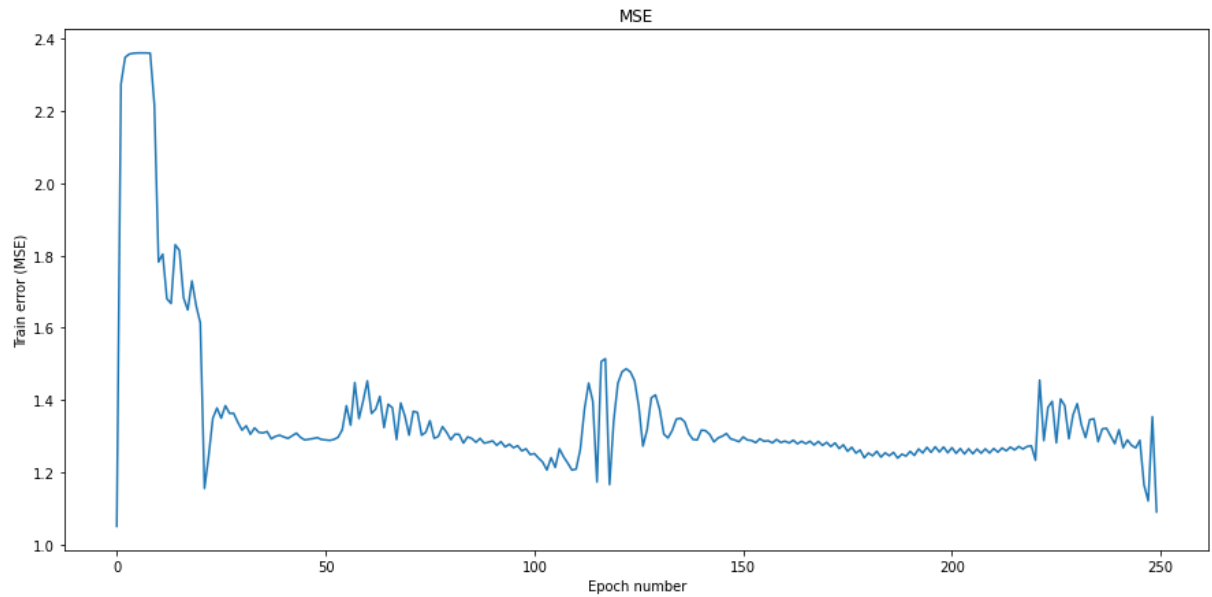


Создал и обучил Нейронную сеть Элмана и сделал предсказание:

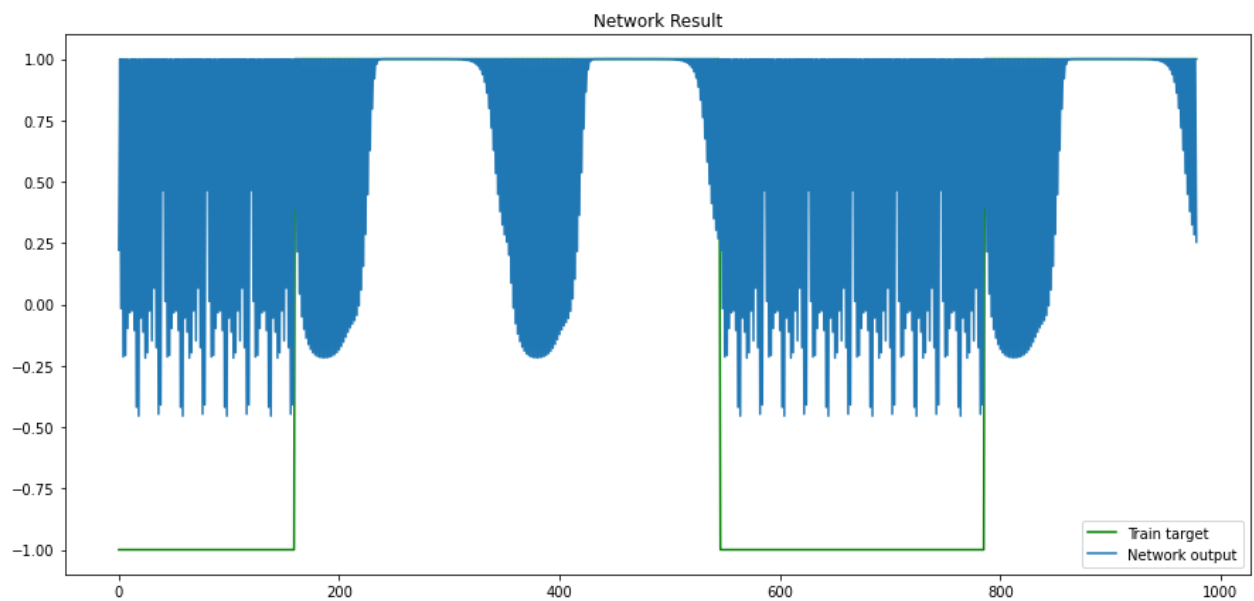
```
net = nl.net.newelm([[ -10, 10]], [8, 1], [nl.trans.TanSig(), nl.trans.TanSig()])
net.layers[0].np['w'][:] = 1
net.layers[0].np['b'][:] = 0
net.init()

error = net.train(P, T, epochs=250, show=50, goal=0.0001)
output = net.sim(P)
```

Функция ошибки:



Предсказание:



Расчёт ошибки:

```
output[output >= 0] = 1.0
output[output < 0] = -1.0

MSE = mean_squared_error(T, output)
print('MSE = {}'.format(MSE))
print('RMSE = {}'.format(np.sqrt(MSE)))
```

```
MSE = 1.2625127681307458
RMSE = 1.1236159344414558
```

## Этап № 2.

Входные данные для сети Хопфилда - черно-белые образы чисел.

Обучим нейронную сеть и сделаем предсказание:

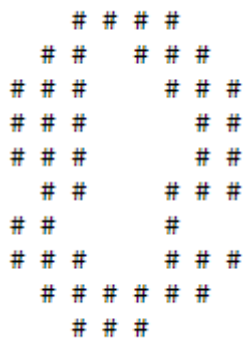
```
data = np.concatenate([four, zero, six], axis=0)

hopf = algorithms.DiscreteHopfieldNetwork(mode='async', n_times=600)
hopf.train(data)

result = hopf.predict(four)
```

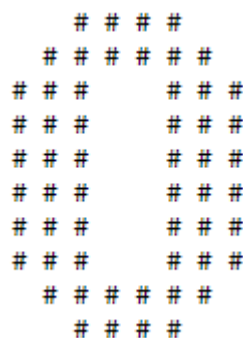
Сравнение входных зашумленных образов и скорректированных нейронной сетью:

Зашумление второго образа на 20%:



A 10x10 grid of characters representing a noisy digit '4'. The grid is composed of '#' characters (black) and spaces (white). The digit '4' is formed by a vertical column of '#' on the left, a horizontal row of '#' in the middle, and a vertical column of '#' on the right. There are several '#' characters scattered in the background, creating a noisy effect.

Результат коррекции сети:



A 10x10 grid of characters representing a clean digit '4'. The grid is composed of '#' characters (black) and spaces (white). The digit '4' is formed by a vertical column of '#' on the left, a horizontal row of '#' in the middle, and a vertical column of '#' on the right. The background is mostly white, indicating that the noise has been removed.

## Этап № 3.

Входные данные для сети Хэмминга - черно-белые образы чисел.

Зададим веса полносвязного слоя аналитически:

```

Q = 7
patterns = np.array([zero, one, two, three, four, six, nine])
nums = [0, 1, 2, 3, 4, 6, 9]
eps = 1 / (Q - 3)
shape = 10 * 12
IW = np.array([zero.T, one.T, two.T, three.T, four.T, six.T, nine.T])
b = shape * np.ones((Q, 1))
a = np.zeros((Q, Q))
for i in range(Q):
    a[i] = IW[i] @ patterns[i] + b[i]

```

Рассчитаем выход (предсказанный класс) сети:

```

A = IW @ four + b
res = network.sim(A)

```

Посмотрим на полученный ответ для зашумленных данных.

```

answer_class = np.argmax(res[0])
print('Result class: {}'.format(nums[answer_class]))

number = patterns[answer_class]
number[number == -1] = 0
draw_image(number.reshape(12, 10))

```

```

Result class: 4
# #      # #
# #      # #
# #      # #
# #      # #
# #      # #
# # # # # # # #
# # # # # # # #
# #      # #
# #      # #
# #      # #
# #      # #
# #      # #

```

## ВЫВОДЫ

Выполнив пятую лабораторную работу по курсу Нейроинформатика, я узнал о рекуррентных нейронных сетях, использующих обратные связи, которые зачастую позволяют лучше запоминать уже просмотренные последовательности.

Я рассмотрел несколько вариантов использования рекуррентных нейронных сетей, таких как предсказание следующего шага временной последовательности, и классификация, и корректировка поврежденных образов.