МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №5
по курсу «Нейроинформатика»

Сети с обратными связями.

Выполнил: Д. Д. Син
Группа: 8О-407Б
Преподаватели: Н.П Аносова

Москва, 2021

Постановка задачи

Целью работы является исследование свойств сетей Хопфилда, Хэмминга и Элмана, алгорит- мов обучения, а также применение сетей в задачах распознавания статических и динамических образов.

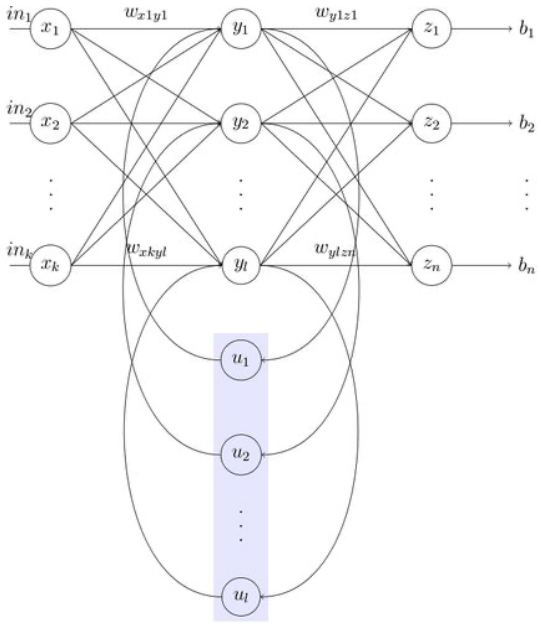
Основные этапы работы:

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.

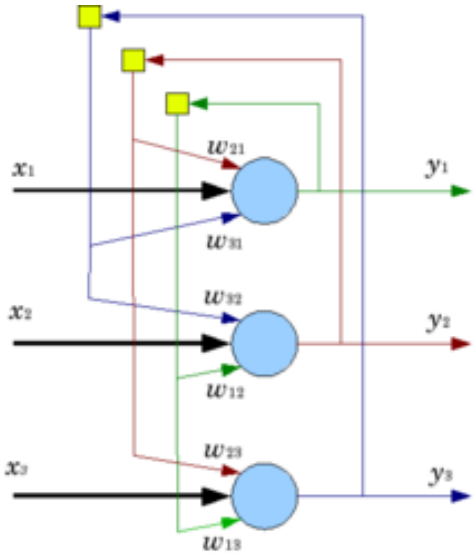
Метод решения

Для решения задачи воспользуемся библиотеками neupy и neirolab. Там уже есть готовые реализации сетей Элмана, Хопфилда и Хэмминга. Сами сети называются рекуррентными, так как имеют обратные связи, которые необходимы для запоминания ранее полученной информации, которая помогает изменять веса лучшим образом. Такие сети хорошо работают при анализе последовательностей, например временных рядов, как в задании 1 или текстов.

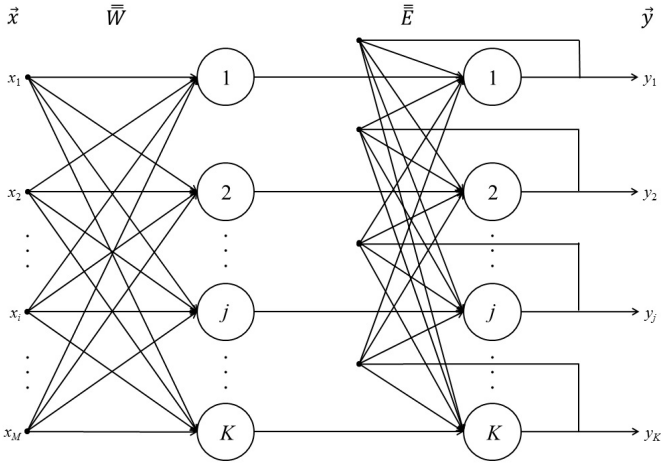
Архитектура сети Элмана



Архитектура сети Хопфилда



Архитектура сети Хэмминга

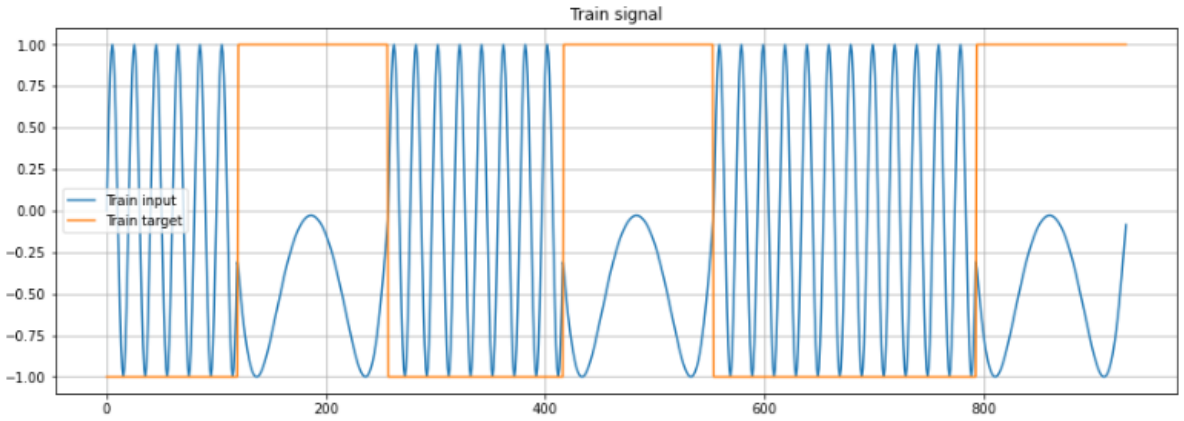


Описание работы программы

Задание 1.

Исходный тренировочный сигнал

```
p, t = gen_train_signal([3, 4, 6])
plot_train_signal(p, t)
```



Обучение сети Элмана

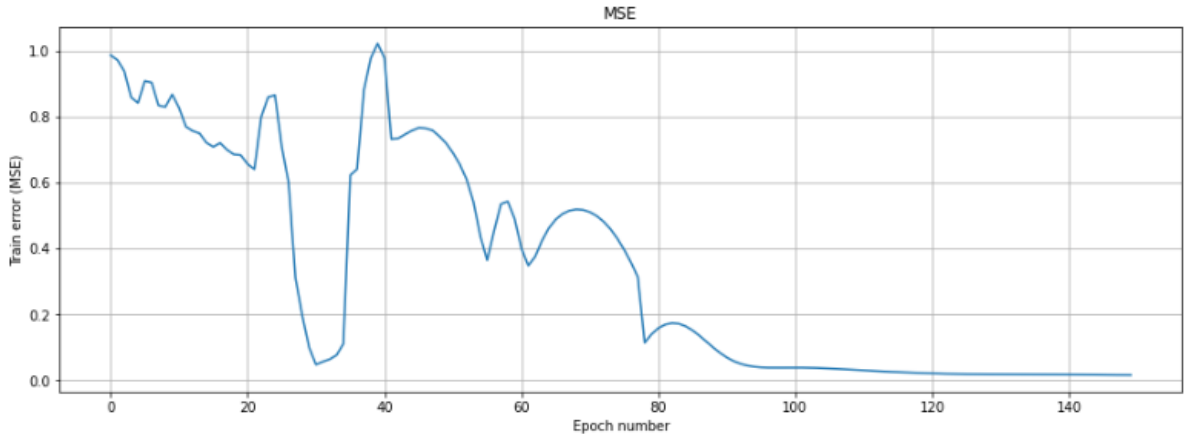
```
result, error = train_elman_net(p, t)
```

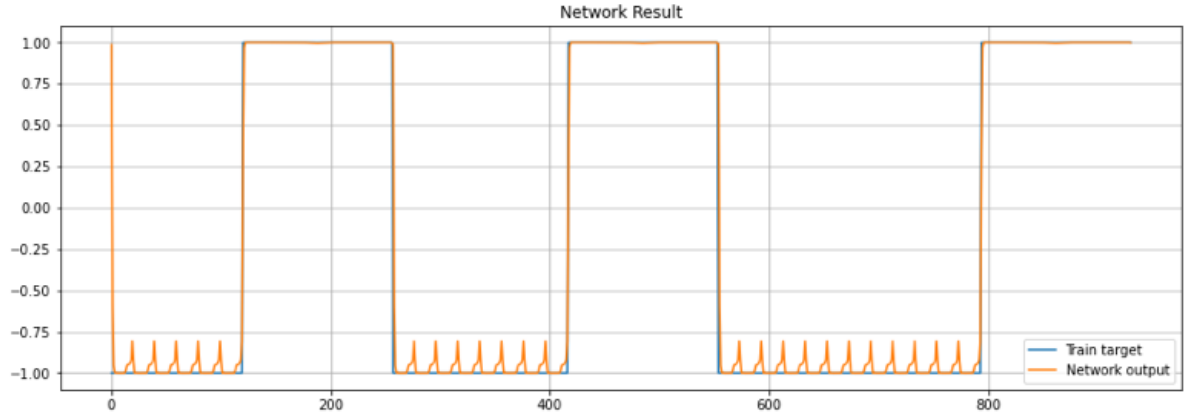
```
Epoch: 10; Error: 0.8685144501087656;
Epoch: 20; Error: 0.6843797132080932;
Epoch: 30; Error: 0.10003009125963479;
Epoch: 40; Error: 1.023002901711216;
Epoch: 50; Error: 0.7204853654490493;
Epoch: 60; Error: 0.4915166523779955;
Epoch: 70; Error: 0.5180251871410508;
Epoch: 80; Error: 0.14036445271436077;
Epoch: 90; Error: 0.08384486580459301;
Epoch: 100; Error: 0.039074680256821134;
Epoch: 110; Error: 0.032111485000421006;
Epoch: 120; Error: 0.02217803512780071;
Epoch: 130; Error: 0.018836707722363434;
Epoch: 140; Error: 0.018111412003323637;
Epoch: 150; Error: 0.016304774950776087;
The maximum number of train epochs is reached
```

Результаты

```
]: plot_result(p, t, result, error)
```

```
MSE = 0.016306637858263882
RMSE = 0.1276974465612523
```

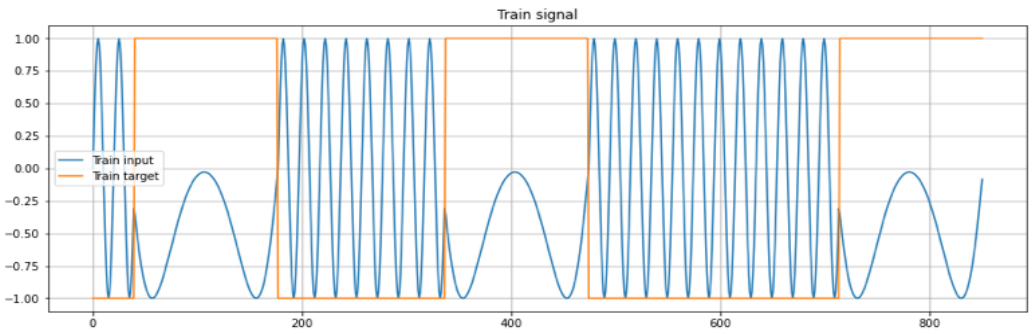





```
]: p, t = gen_train_signal([1, 4, 6])
plot_train_signal(p, t)
```

Изменим входной сигнал и посмотрим на результаты работы сети

```
]: p, t = gen_train_signal([1, 4, 6])
plot_train_signal(p, t)
```



Обучение сети Элмана

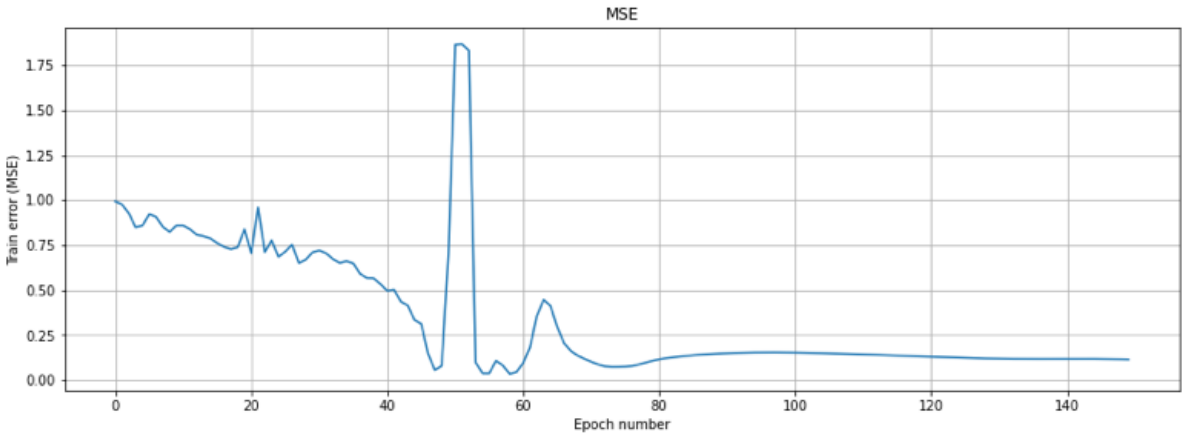
```
result, error = train_elman_net(p, t)
```

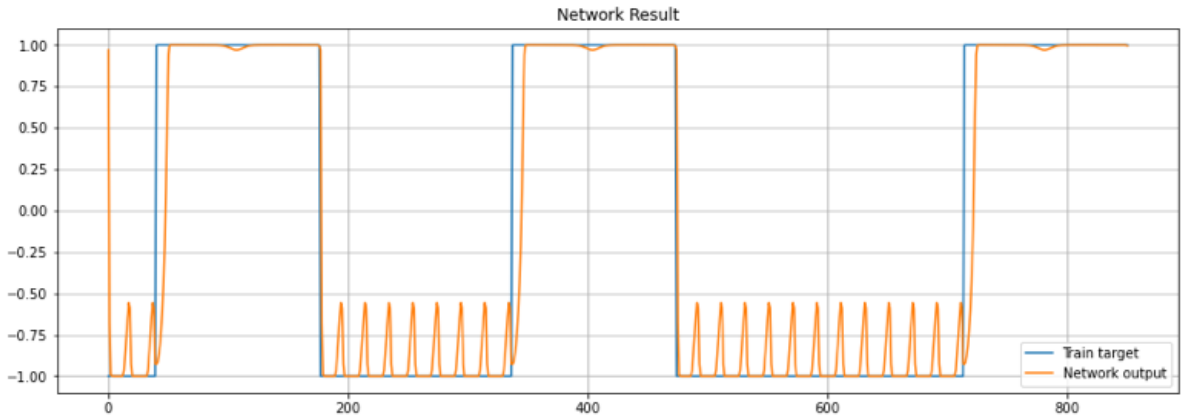
```
Epoch: 10; Error: 0.8585271213020547;
Epoch: 20; Error: 0.8378132026760929;
Epoch: 30; Error: 0.7094947389617914;
Epoch: 40; Error: 0.5331993913229047;
Epoch: 50; Error: 0.7015161244598049;
Epoch: 60; Error: 0.04460532079610133;
Epoch: 70; Error: 0.11753325346146534;
Epoch: 80; Error: 0.10701311765995333;
Epoch: 90; Error: 0.1475274238127273;
Epoch: 100; Error: 0.1530108452686428;
Epoch: 110; Error: 0.14364674633243507;
Epoch: 120; Error: 0.1301622606345474;
Epoch: 130; Error: 0.11887171941916798;
Epoch: 140; Error: 0.11726773269271393;
Epoch: 150; Error: 0.11429564648116101;
The maximum number of train epochs is reached
```

Результаты

```
plot_result(p, t, result, error)
```

```
MSE = 0.11429493404264376
RMSE = 0.3380753378207937
```





Задание 2.

Образы цифр

```
plot_number(NUMBERS[0])
print()
plot_number(NUMBERS[1])
print()
plot_number(NUMBERS[6])
```

```
    # # # #
  # # # # # #
# # #     # # #
# # #     # # #
# # #     # # #
# # #     # # #
# # #     # # #
# # #     # # #
  # # # # # #
    # # # #


    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #
    # # # #

# # # # # #
# # # # # #
#
#
# # # # # #
# # # # # #
# #     # #
# #     # #
# #     # #
# #     # #
# # # # # #
# # # # # #
```

Обучение сети Хопфилда

```
train_data = np.concatenate(list(NUMBERS.values()), axis=0)
```

```
hopfield = algorithms.DiscreteHopfieldNetwork(mode='async', n_times=600)
hopfield.train(train_data)
```

Результаты работы сети

```
six_noise_20 = put_noise(NUMBERS[6], 20)
plot_number(six_noise_20)
```

```
# # # # # #
# # # # # #


# # # # #
  # # # # #
  #     #
# #     # #
# #     # #
# #
# # # # # #
  # # #   #
```

```
plot_number(hopfield.predict(six_noise_20))
```

```
# # # # # #
# # # # # #
#
#
# # # # # #
# # # # # #
# #     # #
# #     # #
# #     # #
# #     # #
# # # # # #
# # # # # #
```

```
one_noise_30 = put_noise(NUMBERS[1], 30)
plot_number(one_noise_30)
```

```
        # #
        # # #
        # # #
        # # #
      # # # #
      # #   #
      # # # #
      # #   #
      #   # #
      # # #
        # # #
      # # # #
```

```
plot_number(hopfield.predict(one_noise_30))
```

```
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
      # # # #
```

Задание 3.

Обучение сети Хэмминга

```
Q = 3
patterns = np.array(list(NUMBERS.values()))
nums = [0, 1, 6]
eps = 1 / (Q - 1)
shape = 10 * 12
IW = np.array([NUMBERS[0].T, NUMBERS[1].T, NUMBERS[6].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]
LW = np.eye(Q)
LW[LW == 0.0] = -eps
hop = nl.net.newhop(a, max_init=600)
hop.layers[0].np['w'][:] = LW
hop.layers[0].np['b'][:] = 0
```

Результаты работы сети

```
predict(NUMBERS[0])
```

```
Result:  [ 1. -1. -1.]
Predicted number = 0
```

```
predict(NUMBERS[1])
```

```
Result:  [-1.  1. -1.]
Predicted number = 1
```

```
predict(NUMBERS[6])
```

```
Result:  [-1. -1.  1.]
Predicted number = 6
```

```
predict(make_noise(NUMBERS[6], 0.2))
```

```
Result:  [-1. -1.  1.]
Predicted number = 6
```

```
predict(make_noise(NUMBERS[0], 0.3))
```

```
Result:  [ 1. -1. -1.]
Predicted number = 0
```

```
predict(make_noise(NUMBERS[1], 0.4))
```

```
Result:  [-1.  1. -1.]
Predicted number = 1
```

Выводы

В данной лабораторной работе реализовали 3 рекуррентных нейросети. Сеть Элмана, Хопфилда и Хэмминга. По результатам работы можно сказать, что сеть Элмана очень хорошо подходит для анализа временных рядов, так как идея с дополнительным нейроном для запоминания очень хорошо помогает анализировать последовательности, да и в целом ассоциативная память в сетях Хэмминга и Хопфилда также хорошо показывает себя для анализа статических образов, но при смещении входного образа конечно сеть угадывать не будет, для этого необходимо использовать сверточные нейросети.