Ивлева Олеся, группа 8382

Вариант 7

## Задание.

Необходимо реализовать нейронную сеть вычисляющую результат заданной логической операции. Затем реализовать функции, которые будут симулировать работу построенной модели. Функции должны принимать тензор входных данных и список весов. Должно быть реализовано 2 функции:

Функция, в которой все операции реализованы как поэлементные операции над тензорами Функция, в которой все операции реализованы с использованием операций над тензорами из NumPy. Для проверки корректности работы функций необходимо:

Инициализировать модель и получить из нее веса. Прогнать датасет через не обученную модель и реализованные 2 функции. Сравнить результат. Обучить модель и получить веса после обучения. Прогнать датасет через обученную модель и реализованные 2 функции. Сравнить результат.

## Загрузка данных и создание ИНС.

Была создана простая модель с тремя слоями: первый и второй имеют 16 нейрона с функцией активации Relu, третий – 1 нейрона с функцией активации Sigmoid.

```
# Создание модели
model = Sequential()
model.add(Dense(16, activation='relu', input_shape=(3,)))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam',loss='binary_crossentropy',
metrics=['accuracy'])
```

Для симуляции работы нейронной сети на основе весов были реализованы две функции:

- sim операции над тензорами реализованы без использования numpy
- numpy sim операции над тензорами реализованы с помощью numpy

Входные данные (number.csv):

```
0, 0, 0
0, 0, 1
0, 1, 0
0, 1, 1
1, 0, 0
1, 0, 1
1, 1, 0
```

Результат прогона через необученную ИНС:

```
model predict
```

[0.5]

[0.54935133]

[0.46795145]

[0.45557195]

[0.5458182]

[0.5501789]

[0.5639292]

[0.5446008]]

numpy

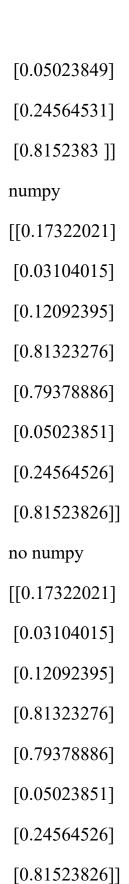
[[0.5]

[0.54935137]

[0.46795146]

```
[0.45557194]
      [0.54581822]
      [0.55017886]
      [0.56392921]
      [0.54460078]]
     no numpy
     [[0.5]]
      [0.54935137]
      [0.46795146]
      [0.45557194]
      [0.54581822]
      [0.55017886]
      [0.56392921]
      [0.54460078]]
     Проведем обучение:
H = model.fit(data, res, epochs=100, batch_size=1)
Результаты после обучения:
model predict
[[0.17322019]
[0.03104019]
[0.12092397]
[0.8132328]
```

[0.7937889]



Как до обучения, так и после обучения значения, полученные симуляциями и реальные практически совпадают. Значения, полученные симуляциями с помощью numpy и без него полностью совпадают.