**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Звіт по лабораторній роботі № 1

Парцептрон

з дисципліни: «Програмні засоби проектування та реалізації нейромережевих систем»

Студент: Ільйов Д. А.

Група: ІО-12

Перевірив: доц. Шимкович В.М.

Київ, 2024

## Завдання

Написати програму, що реалізує нейронну мережу Перцептрон та навчити її виконувати функцію XOR для 4-х змінних.

## Хід роботи

### Визначення структури нейронної мережі

*Зовнішня структура* нейромережі визначається кількістю вхідних та вихідних даних, тобто враховуючи завдання зрозуміло що вхідний шар нейронів буде складатися з 4 нейронів, а вихидній - з 1-го, адже вирішується задача бінарної класифікації. У вихідного нейрона активаційною функцією буде сигмоїда, яка повертає значення в діапазоні [0;1].

*Внутрішня структура* представлятиме з себе чотирьох-нейронний шар. Оскільки для того щоб нейромережа могла навчитися виявляти закономірності у данних, а сам виконувати задачу класифікації XOR потрібно використовувати нелінійну функцію. У даній лаб-роботі в якості активаційної функції використано гіперболічний тангенс який видає значення в діапазоні [-1;1].

### Розробка програми

Функція `prepare_data()` повертає датасет для навчання персептрону.

Далі використовуючи API Keras будую послідовну мережу яка буде мати три шари: вхідний шар, внутрішній та вихідний.

Вхідний шар представляє із себе 4-вимірний вектор даних, якій подається на вхід у нейромережу.

Внутрішній шар має 4 нейрони , він отримує 4 вхідні значення і використовує активаційну функцію гіперболічний тангенс (`tanh`). Це дозволяє моделі обробляти нелінійні залежності.

Другий шар є вихідним і має 1 нейрон. В якості активаційної функції використовується сигмоїда(`sigmoid`).

Далі компілюю описану модель з оптимізатором Adam задаючи параметри швидкості навчання (`learning_rate=0.05`), який впливає на кількісну зміну вагів, встановлюю стандартну функцію втрат для двійкової класифікації (вона

вимірює різницю між результатами моделі та фактичними відповідями і мінімізує різницю) це також впливає на ваги моделі і допомагає їх коригувати, та відмічаю що потрібно рахувати точність класифікування (`accuracy`) під час навчання моделі.

Далі модель тренується на раніше згенерованих даних після чого виводиться значення втрат та точність моделі і наостанок модель контрольний раз прогоняються по датасету, після чого результати виводяться.

## Програмний код

```
import tensorflow as tf
import numpy as np


def prepare_data(split=True):
   variables = np.array(np.meshgrid([0, 1], [0, 1], [0, 1], [0, 1])).T.reshape(-1,
4)

   xor_result = np.bitwise_xor(np.bitwise_xor(variables[:, 0], variables[:, 1]),
                               np.bitwise_xor(variables[:, 2], variables[:, 3]))

   indexes = np.random.permutation(len(variables))
   variables = np.array(variables)[indexes]
   xor_result = np.array(xor_result)[indexes]

   if split:
       return variables[:-4], xor_result[:-4], variables[-4:], xor_result[-4:]
   return np.array(variables), np.array(xor_result)


train_data, train_answers = prepare_data(split=False)
model = tf.keras.models.Sequential([
   tf.keras.Input(shape=(4,)),
   tf.keras.layers.Dense(4, activation='tanh'),
   tf.keras.layers.Dense(1, activation='sigmoid'),
])

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.05),
loss='binary_crossentropy', metrics=['accuracy'])
model.fit(train_data, train_answers, epochs=100)

loss, accuracy = model.evaluate(train_data, train_answers)
print("loss", loss)
print("accuracy", accuracy)

prediction = model.predict(train_data)
for inp, pred in zip(train_data, prediction):
   print(inp, round(pred[0]))
```

## Результат виконання

```
loss 0.18403099477291107
accuracy 0.9375
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
[0 0 0 0] 0
[0 1 0 0] 1
[1 0 0 0] 1
[1 1 0 0] 0
[0 0 1 0] 1
[0 1 1 0] 0
[1 0 1 0] 0
[1 1 1 0] 0
[0 0 0 1] 1
[0 1 0 1] 0
[1 0 0 1] 0
[1 1 0 1] 1
[0 0 1 1] 0
[0 1 1 1] 1
[1 0 1 1] 1
[1 1 1 1] 0
```