Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №6
По дисциплине: «ОМО»

Выполнил:
Студент 3-го курса
Группы АС-66
Горобец А.В.
Проверил:
Крощенко А.А.

Брест 2025

Цель работы: По вариантам предыдущей лабораторной работы реализовать предложенный вариант рекуррентной нейронной сети.

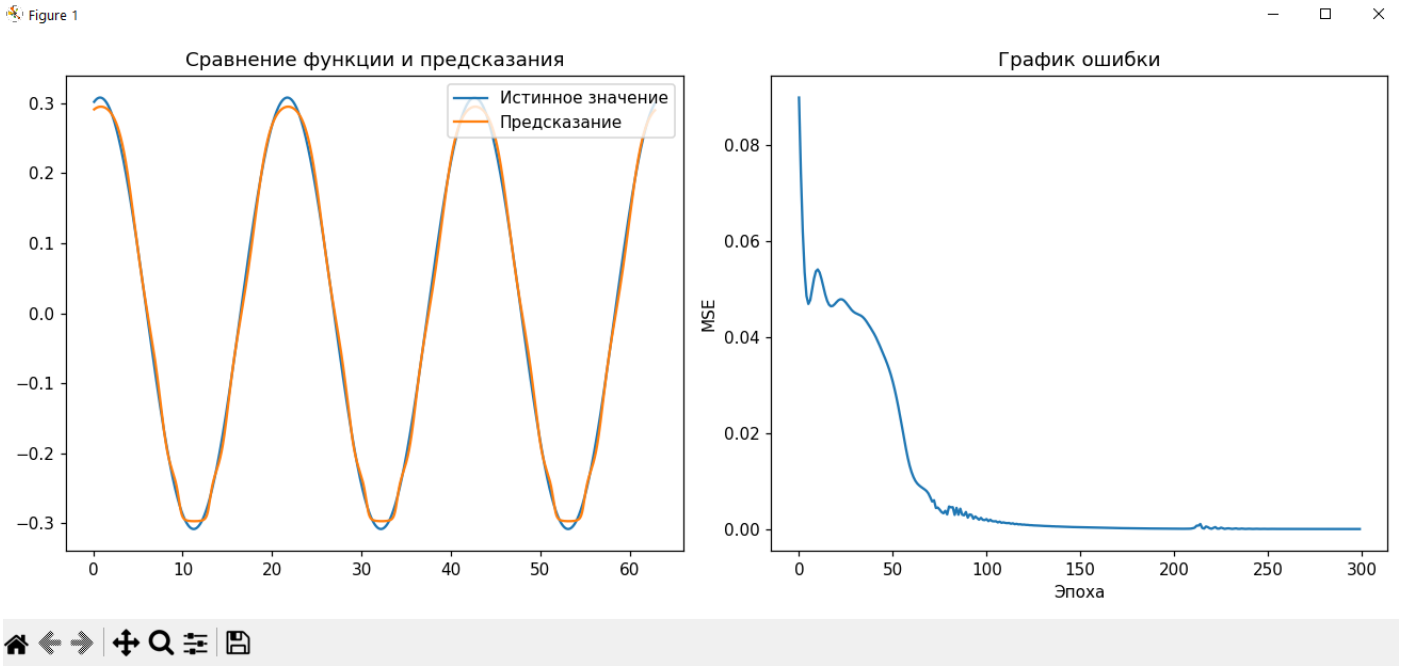
Ход работы

Вариант 3

a	b	c	d	Кол-во входов ИНС	Кол-во НЭ в скрытом слое	Тип РНС
0.3	0.3	0.07	0.3	10	4	Мультирекуррентная

```
import torch
import torch.nn as nn
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 20*np.pi, 0.01)
y = 0.3 * np.cos(0.3 * x) + 0.07 * np.sin(0.3 * x)

window = 10
X, Y = [], []
for i in range(len(x) - window):
    X.append(y[i:i+window])
    Y.append(y[i+window])
X = np.array(X)
Y = np.array(Y).reshape(-1, 1)

X_tensor = torch.tensor(X, dtype=torch.float32)
Y_tensor = torch.tensor(Y, dtype=torch.float32)

class MultiRecurrentNet(nn.Module):
    def __init__(self, input_size, hidden_size):
        super().__init__()
        self.rnn1 = nn.RNN(input_size, hidden_size, batch_first=True)
        self.rnn2 = nn.RNN(hidden_size, hidden_size, batch_first=True)
        self.fc = nn.Linear(hidden_size, 1)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = x.unsqueeze(2)
        out1, _ = self.rnn1(x)
        out1 = self.sigmoid(out1)
        out2, _ = self.rnn2(out1)
        out2 = self.sigmoid(out2)
        out = self.fc(out2[:, -1, :])
        return out

model = MultiRecurrentNet(input_size=1, hidden_size=4)
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.01)

losses = []
```

```
for epoch in range(300):
    optimizer.zero_grad()
    output = model(X_tensor)
    loss = criterion(output, Y_tensor)
    loss.backward()
    optimizer.step()
    losses.append(loss.item())

with torch.no_grad():
    prediction = model(X_tensor).numpy()

plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
plt.plot(x[window:], Y, label='Истинное значение')
plt.plot(x[window:], prediction, label='Предсказание')
plt.legend(loc='upper right')
plt.title('Сравнение функции и предсказания')

plt.subplot(1,2,2)
plt.plot(losses)
plt.title('График ошибки')
plt.xlabel('Эпоха')
plt.ylabel('MSE')
plt.tight_layout()
plt.show()
```



Точность нелинейной ИНС:
MAE = 0.027972, RMSE = 0.030927
Точность мультирекуррентной НС:
Средняя абсолютная ошибка (MAE): 0.008839
Корень из среднеквадратичной ошибки (RMSE): 0.011134

Вывод: я реализовал предложенный вариант рекуррентной нейронной сети.