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

Лабораторная работа №4
По дисциплине: «ОМО»
Тема:” Введение в нейронные сети:
построение многослойного перцептрона”

Выполнил:
Студент 3-го курса
Группы АС-66
Пекун М.С.
Проверил:
Крощенко А.А.

Брест 2025

Цель: построить, обучить и оценить многослойный перцептрон (MLP) для решения задачи классификации.

Вариант 8

Вариант 8 Классификация семян пшеницы
• Seeds
• Задача: классифицировать семена на три сорта (3 класса).
• Архитектура:
входной слой;
один скрытый слой с 7 нейронами (ReLU);
выходной слой с 3 нейронами (Softmax).
• Эксперимент: увеличьте количество нейронов в скрытом слое до 14 и оцените, как это повлияло на точность.

```
import torch
import torch.nn as nn
import torch.optim as optim
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
import pandas as pd
import matplotlib.pyplot as plt
from pathlib import Path


# ============================================
# 1. Загрузка и подготовка данных
# ============================================

FEATURE_COLUMNS = [
    "area", "perimeter", "compactness", "kernel_length",
    "kernel_width", "asymmetry", "groove_length", "class"
]

current_dir = Path(__file__).resolve().parent
dataset_path = (
    current_dir.parent.parent / "lab3" / "src" / "seeds_dataset.txt"
)

df = pd.read_csv(dataset_path, sep=r"\s+", names=FEATURE_COLUMNS)

# Признаки и метки
X = df.iloc[:, :-1].values
y = df.iloc[:, -1].values - 1    # классы: 0, 1, 2

# Стандартизация признаков
```

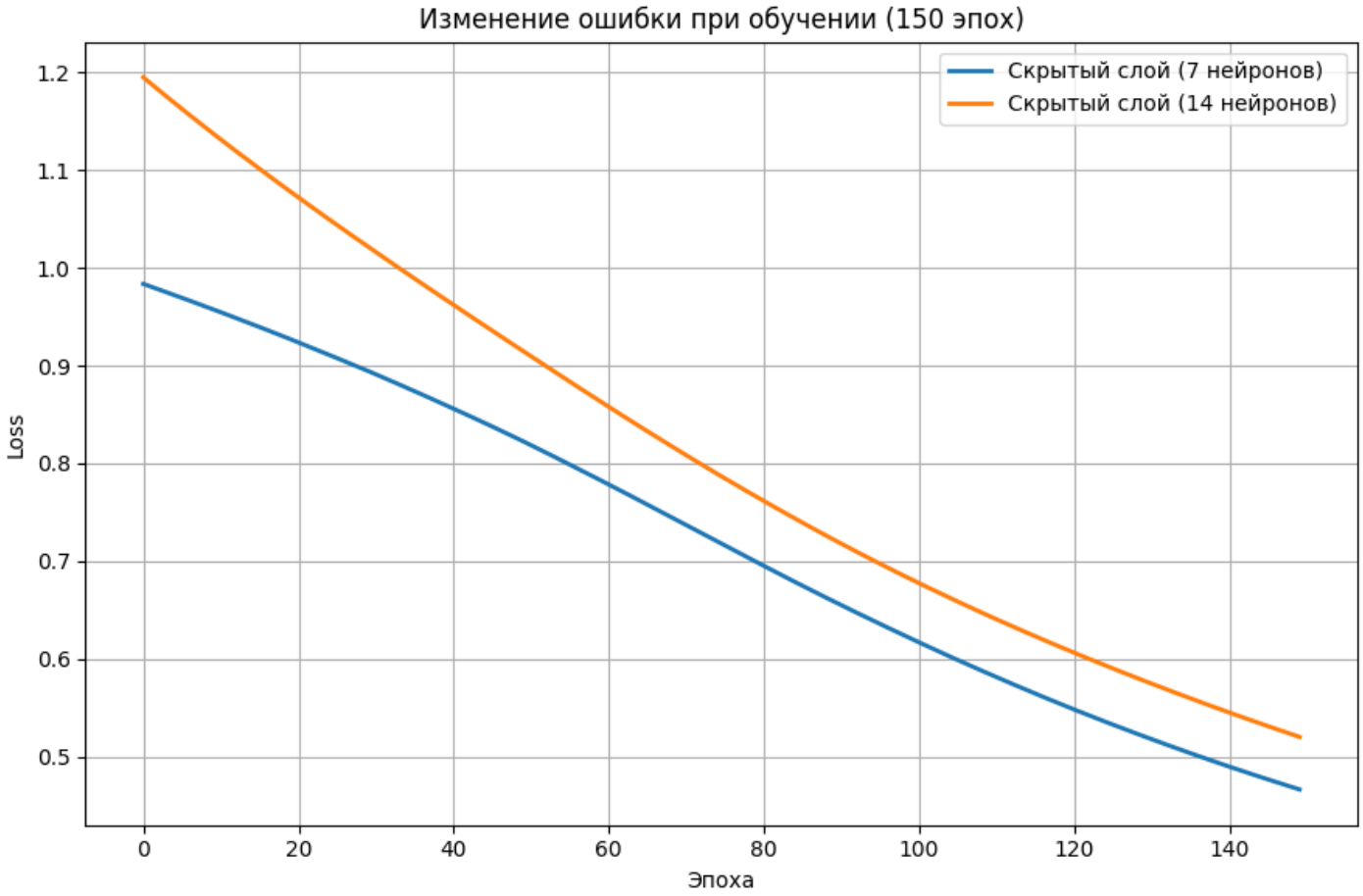
```
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Разделение выборки
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42
)

# Преобразование данных в тензоры
X_train = torch.tensor(X_train, dtype=torch.float32)
X_test = torch.tensor(X_test, dtype=torch.float32)
y_train = torch.tensor(y_train, dtype=torch.long)
y_test = torch.tensor(y_test, dtype=torch.long)


# ============================================
# 2. Архитектура многослойного перцептрона
# ============================================

class MLP(nn.Module):
    """Многослойный перцептрон для классификации Seeds."""

    def __init__(self, hidden_units: int = 7):
        super().__init__()
        self.fc1 = nn.Linear(7, hidden_units)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(hidden_units, 3)

    def forward(self, x):
        x = self.relu(self.fc1(x))
        return self.fc2(x)


# ============================================
# 3. Обучение модели с сохранением истории ошибок
# ============================================

def train_model(hidden_units: int, epochs: int = 150):
    """Обучает модель MLP и возвращает точность и историю ошибок."""
    model = MLP(hidden_units)
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.Adam(model.parameters(), lr=0.001)

    loss_history = []

    for epoch in range(epochs):
        model.train()

        optimizer.zero_grad()
```

```
        predictions = model(X_train)
        loss = criterion(predictions, y_train)

        loss.backward()
        optimizer.step()

        loss_history.append(loss.item())

    # Оценка модели
    model.eval()
    with torch.no_grad():
        logits = model(X_test)
        pred_labels = torch.argmax(logits, dim=1)

    accuracy = accuracy_score(y_test, pred_labels)

    return accuracy, loss_history


# ============================================
# 4. Запуск эксперимента
# ============================================

acc_7, loss_7 = train_model(hidden_units=7)
acc_14, loss_14 = train_model(hidden_units=14)

print(f"Accuracy (7 нейронов):  {acc_7:.4f}")
print(f"Accuracy (14 нейронов): {acc_14:.4f}")


# ============================================
# 5. График изменения ошибки
# ============================================

epochs_range = range(len(loss_7))

plt.figure(figsize=(9, 6))
plt.plot(epochs_range, loss_7, label="Скрытый слой (7 нейронов)",
linewidth=2)
plt.plot(epochs_range, loss_14, label="Скрытый слой (14 нейронов)",
linewidth=2)

plt.title("Изменение ошибки при обучении (150 эпох)")
plt.xlabel("Эпоха")
plt.ylabel("Loss")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

Результат:

```
Accuracy (7 нейронов):  0.9245
Accuracy (14 нейронов): 0.9245

Process finished with exit code 0
```

График изменения ошибок:



Вывод: На практике построили, обучили и оценили многослойный перцептрон (MLP) для решения задачи классификации.