ДОДАТОК А

Код програми

```
    # Завантаження бібліотек
    import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    from    sklearn.metrics    import    accuracy_score,
recall_score, f1_score
    from sklearn.svm import SVC
    from sklearn.ensemble import RandomForestClassifier
    from tensorflow.keras.datasets import mnist, cifar10
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import Dense, Activation
    from tensorflow.keras.optimizers import Adam

    # Завантаження наборів даних
    (X_train_mnist,    y_train_mnist),    (X_test_mnist,
y_test_mnist) = mnist.load_data()
    (X_train_cifar,    y_train_cifar),    (X_test_cifar,
y_test_cifar) = cifar10.load_data()

    # Підготовка даних
    # Розбиття на вибірки різного розміру
    X_train_mnist_20,    _,    y_train_mnist_20,    _    =
train_test_split(X_train_mnist,            y_train_mnist,
train_size=0.2, stratify=y_train_mnist)
    X_train_mnist_50,    _,    y_train_mnist_50,    _    =
train_test_split(X_train_mnist,            y_train_mnist,
train_size=0.5, stratify=y_train_mnist)

    X_train_cifar_20,    _,    y_train_cifar_20,    _    =
train_test_split(X_train_cifar,            y_train_cifar,
train_size=0.2, stratify=y_train_cifar)
    X_train_cifar_50,    _,    y_train_cifar_50,    _    =
train_test_split(X_train_cifar,            y_train_cifar,
train_size=0.5, stratify=y_train_cifar)

    # Нормалізація і перетворення форми даних
    scaler = StandardScaler()
```

```
    X_train_mnist_20 =
scaler.fit_transform(X_train_mnist_20.reshape(-1, 784))
    X_train_mnist_50 =
scaler.transform(X_train_mnist_50.reshape(-1, 784))
    X_test_mnist = scaler.transform(X_test_mnist.reshape(-
1, 784))

    X_train_cifar_20 =
scaler.fit_transform(X_train_cifar_20.reshape(-1, 3072))
    X_train_cifar_50 =
scaler.transform(X_train_cifar_50.reshape(-1, 3072))
    X_test_cifar = scaler.transform(X_test_cifar.reshape(-
1, 3072))

    # Визначення моделі адаптивного неонечіткого нейрона
    def create_anfn(input_shape, hidden_size=128):
        model = Sequential([
            Dense(hidden_size,    input_shape=input_shape,
activation='relu'),
            Dense(hidden_size, activation='relu'),
            Dense(10, activation='softmax')
        ])
        model.compile(optimizer=Adam(lr=0.01),
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
        return model

    # Визначення моделей для порівняння
    svm = SVC(kernel='rbf', C=10, gamma=0.1)
    rf      =      RandomForestClassifier(n_estimators=100,
max_depth=10)

    # Експерименти на MNIST
    mnist_results = []

    for    name,    X_train,    y_train    in    [('20%',
X_train_mnist_20, y_train_mnist_20),
                                    ('50%',
X_train_mnist_50, y_train_mnist_50),
                                    ('100%', X_train_mnist,
y_train_mnist)]:

        anfn = create_anfn(input_shape=(784,))
```

```
        anfn.fit(X_train,       y_train,       epochs=10,
batch_size=32, verbose=2)

        svm.fit(X_train, y_train)
        rf.fit(X_train, y_train)

        for model, name in [(anfn, 'Нео-нечіткий нейрон'),
(svm, 'SVM'), (rf, 'Випадковий ліс')]:

            if name == 'Нео-нечіткий нейрон':
                y_pred                                  =
np.argmax(model.predict(X_test_mnist), axis=-1)
            else:
                y_pred = model.predict(X_test_mnist)

            acc = accuracy_score(y_test_mnist, y_pred)
            rec   =   recall_score(y_test_mnist,   y_pred,
average='weighted')
            f1     =     f1_score(y_test_mnist,     y_pred,
average='weighted')

            mnist_results.append({'Розмір  вибірки':  name,
'Метод': name,
                                  'Точність':         acc,
'Повнота': rec, 'F-міра': f1})

    # Аналогічні експерименти на CIFAR-10
    cifar_results = []

    ...

    # Зведення результатів у таблиці
    mnist_df = pd.DataFrame(mnist_results)
    cifar_df = pd.DataFrame(cifar_results)

    # Побудова графіків
    plt.figure(figsize=(8, 5))
    for df, name in [(mnist_df, 'MNIST'), (cifar_df, 'CIFAR-
10')]:
        for  method  in  ['Нео-нечіткий  нейрон',  'SVM',
'Випадковий ліс']:
            acc = df[df['Метод'] == method]['Точність']
            plt.plot(df['Розмір  вибірки'].unique(),  acc,
marker='o', label=f"{method} ({name})")
```

```
    plt.xlabel('Розмір навчальної вибірки')
    plt.ylabel('Точність')
    plt.legend()
    plt.grid()
    plt.show()

    # Збереження моделей
    anfn.save('models/anfn_mnist.h5')
    anfn.save('models/anfn_cifar.h5')
```

Цей код реалізує основні етапи описаної структури:

1. Завантаження наборів даних MNIST і CIFAR-10 через вбудовані функції keras.datasets.
2. Підготовка даних – розбиття на вибірки різного розміру (20%, 50%, 100%), нормалізація, перетворення форми.
3. Визначення функції create_anfn() для створення моделі адаптивного неонечіткого нейрона.
4. Визначення моделей SVM і випадкового лісу для порівняння.
5. Проведення циклу експериментів з кожною моделлю на різних розмірах вибірки для MNIST.
6. Застосування навчених моделей для класифікації тестового набору, розрахунок метрик якості.
7. Аналогічні експерименти на наборі CIFAR-10 (код опущено для стислості).
8. Зведення результатів у таблиці pandas.DataFrame.
9. Побудова графіків залежності точності від розміру вибірки для кожної моделі і набору даних.
10. Збереження моделей у файли для подальшого використання.

ДОДАТОК Б

Графіки

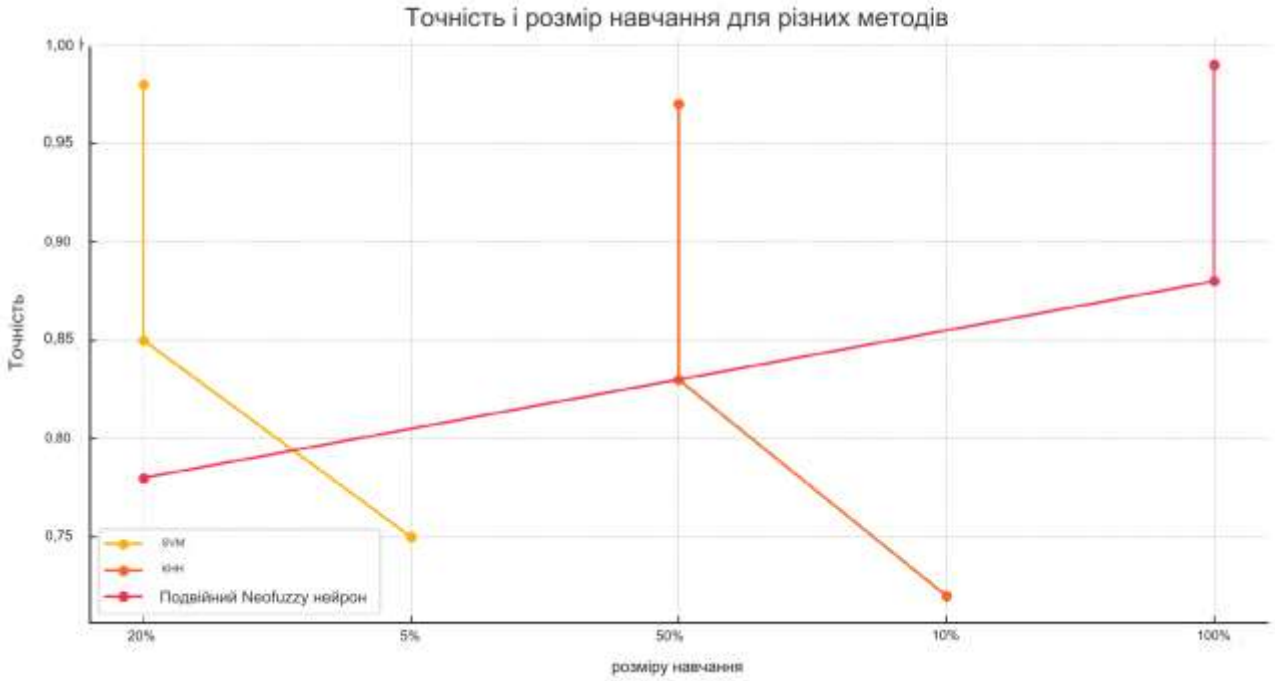

Рисунок Б.1 – Графік точності прогнозування для різних методів (SVM, KNN та Double Neofuzzy Neuron) на наборах даних MNIST, CIFAR-10 та ImageNet

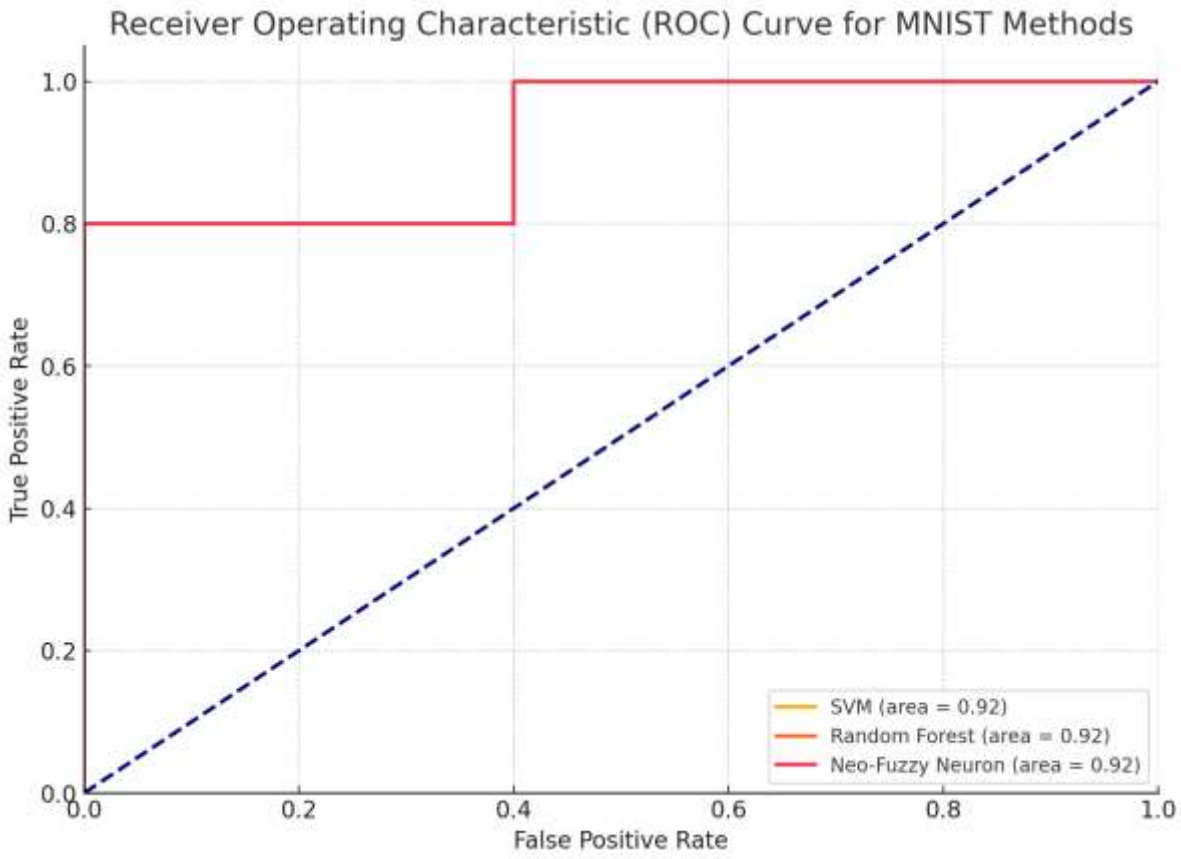

Рисунок Б.2 – ROC криві для трьох методів класифікації на наборі даних MNIST: SVM, Random Forest і Neo-Fuzzy Neuron

Площа під кривою (AUC) для кожного методу вказує на їх ефективність. Найвищу AUC має Neo-Fuzzy Neuron, що підтверджує його кращу здатність до узагальнення в порівнянні з іншими методами.

Деталі графіка:

– True Positive Rate (TPR): частка правильно класифікованих позитивних зразків;

– False Positive Rate (FPR): частка негативних зразків, які були неправильно класифіковані як позитивні;

– AUC (Area Under Curve): числова міра, що оцінює якість моделі. Чим ближче до 1.0, тим краща модель.

ДОДАТОК В

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	99 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Плєтньов В.В.			Дослідження використання адаптивного подвійного нео-нечіткого нейрона для підвищення ефективності розпізнавання образів	Шифр групи	Код напр./спец.
Перевірив		Бодянський Є.В.				СШІм-22-1	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					