A picture containing text

Description automatically generated**МІНІСТЕРСТВО ОСВІТИ і НАУКИ УКРАЇНИ**

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп’ютерних систем

**Лабораторна робота № 3**  
**з дисципліни «Штучні нейронні мережі»**

Виконали:

студенти групи КВ-81

Поляков Є. А.

Ядуха Б. В.

Завдання

1. Сформувати навчальну вибірку для навчання рекурентної нейронної мережі (РНМ), призначеної для розпізнавання емоційного забарвлення тексту.
2. Розробити програмне забезпечення (ПЗ) для попередньої обробки тексту.
3. Розробити ПЗ, призначене для перетворення сформованих прикладів до виду придатному для подачі на вхід РНМ.
4. Розробити ПЗ для реалізації РНМ, призначеної для розпізнавання емоційного забарвлення тексту.
5. Провести комп’ютерні експерименти спрямовані на верифікацію розробленого ПЗ. ПЗ для ЗНМ1 вважається верифікованим, якщо точність розпізнавання тренувальних прикладів >0,7.

6. Оформити звіт в якому відобразити:

a. Прізвище, ініціали та номер групи виконавця.

b. Назву та завдання лабораторної роботи.

c. Інструкцію для використання розробленого ПЗ.

d. Скріншоти, що демонструють використання розробленого ПЗ.

e. Висновки.

Лабораторну роботу можливо виконувати в складі групи, що складається не більше ніж з 4 студентів.

Репозиторій з кодом і даними лабораторних робіт 1-3:

<https://github.com/Ent71/neural-networks.git>

Для тренування було використано датасет imdb який містить відгуки глядачів і класифікацію good/bad. Програма для обробки даних:

**PreProcessingUnit.py**:

import numpy as np

def load():

    word2index = {}

    max\_index = 0

    resulting\_entries = []

    with open('data.txt', 'r') as f:

        for line in f.readlines():

            label = line[:1]

            words = line[2:].split(" ")

            word\_indices = []

            for word in words:

                if word in word2index:

                    index = word2index[word]

                else:

                    index = max\_index

                    max\_index += 1

                    word2index[word] = max\_index

                word\_indices.append(index)

            resulting\_entries.append([int(label), np.array(word\_indices)])

    return np.array(resulting\_entries)

def shuffle(resulting\_entries\_np):

    np.random.shuffle(resulting\_entries\_np)

    return resulting\_entries\_np

def split(resulting\_entries\_np):

    rate = 5

    x = []

    y = []

    for entry in resulting\_entries\_np:

        y.append(entry[0])

        x.append(entry[1])

    test\_count = len(x) // rate

    x\_train = np.array(x[test\_count:])

    y\_train = np.array(y[test\_count:])

    x\_test = np.array(x[:test\_count])

    y\_test = np.array(y[:test\_count])

    return x\_train, y\_train, x\_test, y\_test

entries\_np = load()

entries\_np\_shuffled = shuffle(entries\_np)

x\_train\_np, y\_train\_np, x\_test\_np, y\_test\_np = split(entries\_np\_shuffled)

type(x\_train\_np)

all\_np = np.array([x\_train\_np, y\_train\_np, x\_test\_np, y\_test\_np])

np.save("data.npy", all\_np)

Програма для тренування рекурентної мережі (LSTM):

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers.embeddings import Embedding

from keras.preprocessing import sequence

# number of words in data

num\_words = 5000

def load\_data():

    all\_data = np.load('data.npy', allow\_pickle=True)

    x\_train\_np = all\_data[0]

    y\_train\_np = all\_data[1]

    x\_test\_np = all\_data[2]

    y\_test\_np = all\_data[3]

    return x\_train\_np, y\_train\_np, x\_test\_np, y\_test\_np

x\_train, y\_train, x\_test, y\_test = load\_data()

max\_review\_length = 500

X\_train = sequence.pad\_sequences(x\_train, maxlen=max\_review\_length)

X\_test = sequence.pad\_sequences(x\_test, maxlen=max\_review\_length)

embedding\_vecor\_length = 32

model = Sequential(

    Embedding(num\_words, embedding\_vecor\_length, input\_length=max\_review\_length),

    LSTM(100),

    Dense(1, activation='sigmoid'))

model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])

print(model.summary())

model.fit(X\_train, y\_train, validation\_data=(X\_test, y\_test), epochs=3, batch\_size=64)

scores = model.evaluate(X\_test, y\_test, verbose=0)

print("Accuracy: %.2f%%" % (scores[1]\*100))

Результат роботи:

Text

Description automatically generated

Точність на тренувальних даних даних: 90.65%

Точність на тестових даних: 87.97%

**Висновок**

Під час виконання даної лабораторної роботи було розроблено програму для загрузки класифікованого тексту, кодування слів і дампу в npy масив. Також було проведено тренування рекурентної нейронної мережі (LSTM).