МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №7
по дисциплине «Искусственные нейронные сети»
Тема: Классификация обзоров фильмов

Студентка гр. 7383 _____ Тян Е.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург
2020

Цель работы.

Реализовать классификацию обзоров фильмов.

Задание.

1. Ознакомиться с рекуррентными нейронными сетями
2. Изучить способы классификации текста
3. Ознакомиться с ансамблированием сетей

Ход работы.

Импортируем необходимые для работы зависимости для предварительной обработки данных и построения модели:

```
import numpy as np
from tensorflow.keras.datasets import imdb
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Conv1D
from tensorflow.keras.layers import MaxPooling1D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Embedding
from tensorflow.keras.preprocessing import sequence
import matplotlib.pyplot as plt
```

Загрузим датасет IMDb, который уже встроен в Keras. Поскольку мы не хотим иметь данные обучения и тестирования в пропорции 50/50, мы сразу же объединим эти данные после загрузки для последующего разделения в пропорции 80/20:

```
(training_data, training_targets), (testing_data, testing_targets) =
imdb.load_data(num_words=10000)
data = np.concatenate((training_data, testing_data), axis=0)
targets = np.concatenate((training_targets, testing_targets), axis=0)
```

Обрежем и дополним входные последовательности так, чтобы они были одинаковой длины для моделирования. Модель узнает, что нулевые значения не содержат никакой информации, поэтому на самом деле последовательности имеют разную длину с точки зрения содержания, но для выполнения вычислений в Керасе требуются векторы одинаковой длины:

```
max_review_length = 500
training_data = sequence.pad_sequences(training_data, maxlen=max_review_length)
```

```
testing_data = sequence.pad_sequences(testing_data, maxlen=max_review_length)
```

Построим три модели нейронной сети. Первая модель будет задана функцией premiere_m():

```
def premier_m():
    model = Sequential()
          model.add(Embedding(top_words, embedding_vecor_length,
input_length=max_review_length))
    model.add(LSTM(100))
    model.add(Dropout(0.3))
    model.add(Dense(50, activation = 'relu'))
    model.add(Dropout(0.25))
    model.add(Dense(1, activation='sigmoid'))
    train_m(model, 'model1.h5')
    return model
```

Вторая модель будет задана функцией deuxieme_m():

```
def deuxieme_m():
    model = Sequential()
          model.add(Embedding(top_words, embedding_vecor_length,
input_length=max_review_length))
          model.add(Conv1D(filters=32, kernel_size=3, padding='same',
activation='relu'))
    model.add(MaxPooling1D(pool_size=2))
    model.add(Dropout(0.3))
    model.add(LSTM(50))
    model.add(Dropout(0.3))
    model.add(Dense(1, activation='sigmoid'))
    train_m(model, 'model2.h5')
    return model
```

Третья модель будет задана функцией troisieme_m():

```
def troisieme_m():
    model = Sequential()
          model.add(Embedding(top_words, embedding_vecor_length,
input_length=max_review_length))
          model.add(Conv1D(filters=32, kernel_size=3, padding='same',
activation='relu'))
    model.add(MaxPooling1D(pool_size=2))
    model.add(Dropout(0.3))
          model.add(Conv1D(filters = 64, kernel_size=3, padding='same',
activation='relu'))
    model.add(MaxPooling1D(pool_size=2))
    model.add(Dropout(0.4))
    model.add(Flatten())
    model.add(Dense(1, activation='sigmoid'))
    train_m(model, 'model3.h5')
    return model
```

Обучим каждую сеть. Первая сеть имеет точность 86.66%. Вторая – 87.48%. Третья – 88.85%. Графики точности и ошибки представлены на рис. 1 – 6.

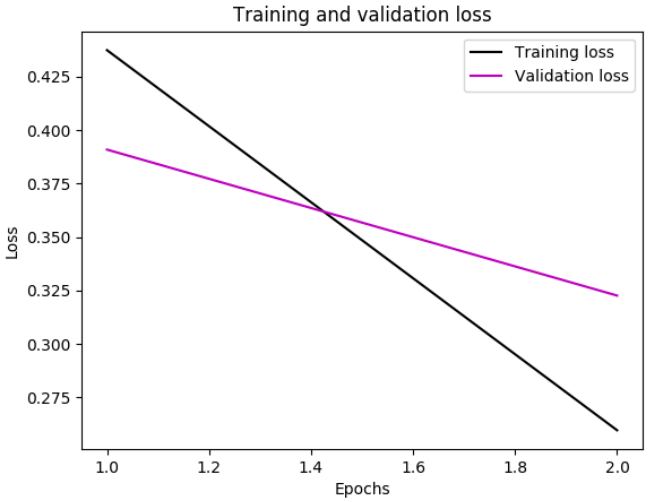

Рисунок 1 – График ошибки первой модели

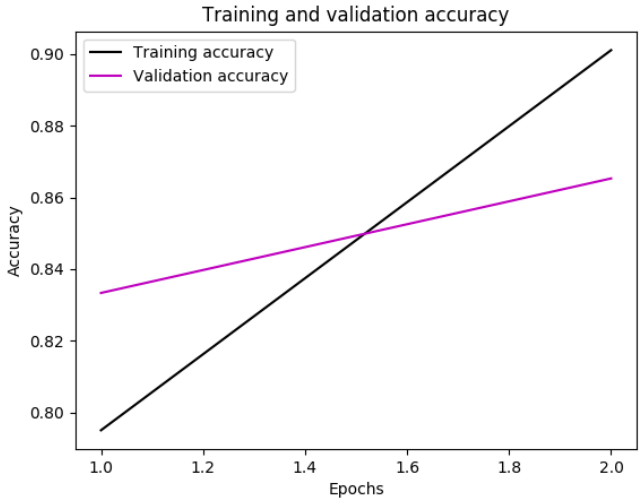

Рисунок 2 – График точности первой модели

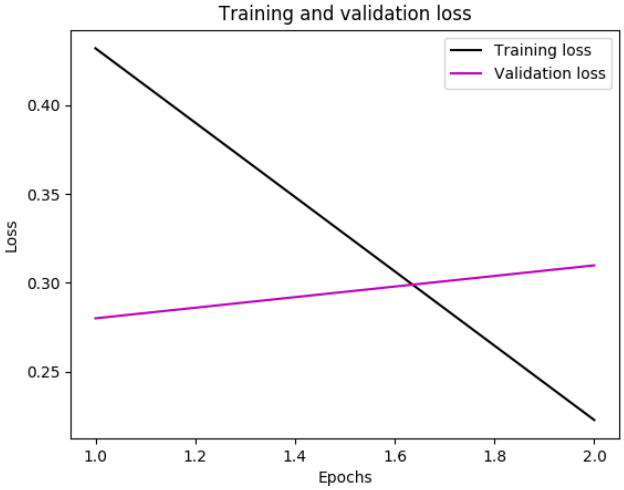

Рисунок 3 – График ошибки второй модели

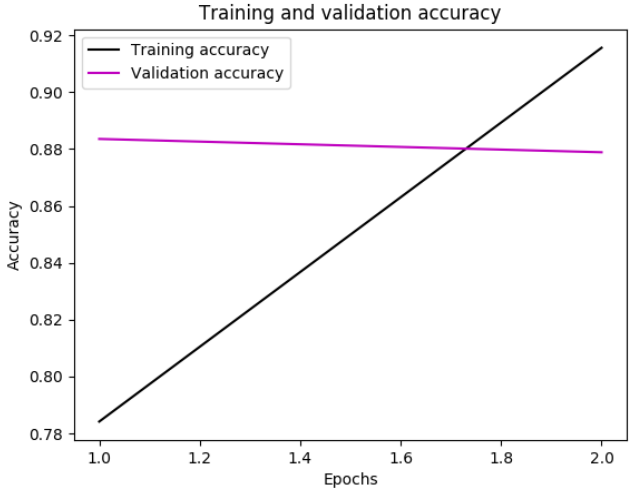

Рисунок 4 – График точности второй модели

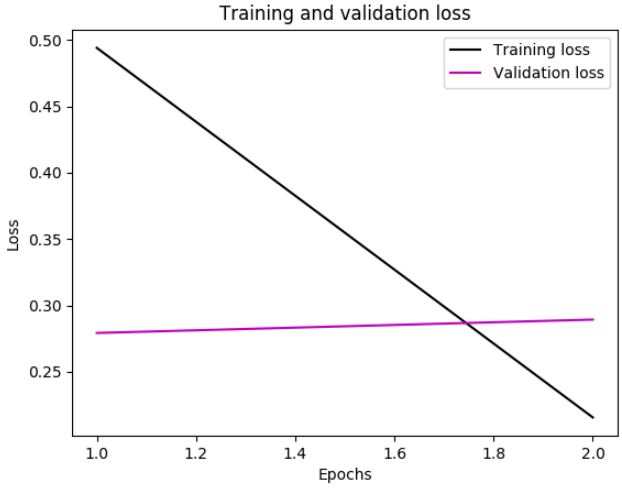

Рисунок 5 – График ошибки третьей модели

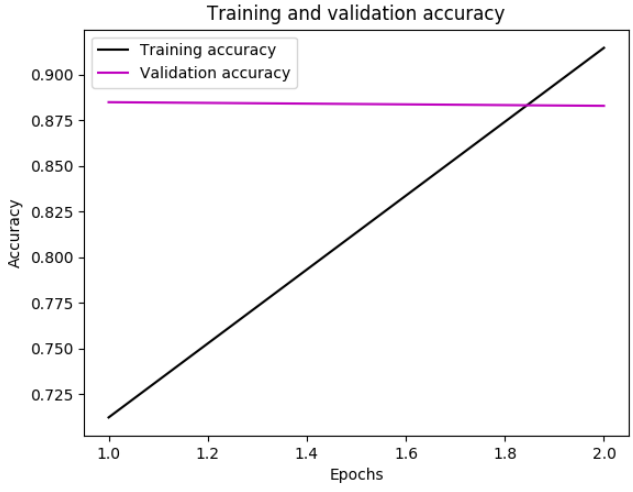

Рисунок 6 – График точности третьей модели

Проверим набор построенных ИНС для классификации текста. В результате применения нескольких моделей получили, что первая и вторая лают

точность – 0.88, первая и третья – 0.879, вторая и третья – 0.889, первая, вторая и третья – 0.8875

Проведем ансамблирование сетей:

```
def ensemble_m(models):
    results = []
    accs = []
    labels = []
    for model in models:
        results.append(model.predict(testing_data))
        result = np.array(results[-1])
        result = np.reshape(result, result.shape[0])
        result = np.greater_equal(result, np.array([0.5]), dtype=np.float64)
        acc = 1 - np.abs(testing_targets-result).mean(axis=0)
        accs.append(acc)
        labels.append(str(len(results)))
    pairs = [(0, 1), (1, 2), (0, 2)]
    for (i, j) in pairs:
        result = np.array([results[i], results[j]]).mean(axis=0)
        result = np.reshape(result, result.shape[0])
        result = np.greater_equal(result, np.array([0.5]), dtype=np.float64)
        acc = 1 - np.abs(testing_targets-result).mean(axis=0)
        accs.append(acc)
        labels.append(str(i+1) + '+' + str(j+1))
    result = np.array(results).mean(axis=0)
    result = np.reshape(result, result.shape[0])
    result = np.greater_equal(result, np.array([0.5]), dtype=np.float64)
    acc = 1 - np.abs(testing_targets-result).mean(axis=0)
    accs.append(acc)
    labels.append('1+2+3')
    print(accs)
    plot_acc(accs, labels)
```

Напишем функцию, которая позволяет ввести пользовательский текст:

```
def user_func(review):
    index = imdb.get_word_index()
    test_x = []
    words = []
    for line in review:
        lines = line.translate(str.maketrans('', '', `,.?!:;()')).lower()
        for chars in lines:
            chars = index.get(chars)
            if chars is not None and chars < 10000:
                words.append(chars)
        test_x.append(words)
    return test_x
```

Результат работы программы на примере: "My God People, Really?? Get Your heads out of the sand. Movie is over rated. Its an average movie. Don't know why it created this much buzz." – 0.49, что на самом деле подтверждает нейтрально-негативное настроение отзыва.

Выводы.

В ходе выполнения лабораторной работы была построена и обучена нейронная сеть для обработки текста. Был определен набор оптимальных ИНС для классификации текста: вторая и третья. Было проведено ансамблирование моделей. Написаны функции, которые позводят загружать текст и получать результат ансамбля сетей. Проведено тестирование сетей на тексте: "My God People, Really?? Get Your heads out of the sand. Movie is over rated. Its an average movie. Don't know why it created this much buzz.».