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

ОТЧЕТ

по лабораторной работе №6
по дисциплине
«Искусственные нейронные сети»
Тема: Прогноз успеха фильмов по обзорам

Студентка гр. 7382 _____ Лящевская А.П

Преподаватель _____ Жукова Н.А.

Санкт-Петербург
2020

Цель работы.

Прогноз успеха фильмов по обзорам (Predict Sentiment From Movie Reviews).

Порядок выполнения работы.

1. Ознакомиться с задачей регрессии.
2. Изучить способы представления текста для передачи в ИНС.

Требования к выполнению задания.

1. Построить и обучить нейронную сеть для обработки текста.
2. Исследовать результаты при различном размере вектора представления текста.
3. Написать функцию, которая позволяет ввести пользовательский текст (в отчете привести пример работы сети на пользовательском тексте).

Основные теоретические положения.

Датасет IMDb состоит из 50 000 обзоров фильмов от пользователей, помеченных как положительные (1) и отрицательные (0). Это пример бинарной или двуклассовой классификации, важный и широко применяющийся тип задач машинного обучения.

- Рецензии предварительно обрабатываются, и каждая из них кодируется последовательностью индексов слов в виде целых чисел.
- Слова в обзорах индексируются по их общей частоте появления в датасете. Например, целое число «2» кодирует второе наиболее частое используемое слово.
- 50 000 обзоров разделены на два набора: 25 000 для обучения и 25 000 для тестирования.

Ход работы.

1. Была построена и обучена нейронная сеть для обработки текста со следующей архитектурой:

- optimizer="adam",
- loss="binary_crossentropy",
- metrics=["accuracy"],
- epochs=2,
- batch_size = 500,
- Размер вектора представления текста – 10000.
- model:

```
model = Sequential()

# Input - Layer
model.add(Dense(50, activation='relu', input_shape=(10000, )))

# Hidden - Layers
model.add(Dropout(0.3, noise_shape=None, seed=None))
model.add(Dense(50, activation="relu"))
model.add(Dropout(0.2, noise_shape=None, seed=None))
model.add(Dense(50, activation="relu"))

# Output- Layer
model.add(Dense(1, activation="sigmoid"));
model.summary()

model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"]);

H = model.fit( train_x, train_y, epochs=2, batch_size = 500, validation_data = (test_x, test_y))
```

Данная архитектура дает точность: на тренировочных данных ~ 91%, на валидационных ~ 89%. Графики точности и ошибки предоставлены на рис. 1 и рис. 2 соответственно.

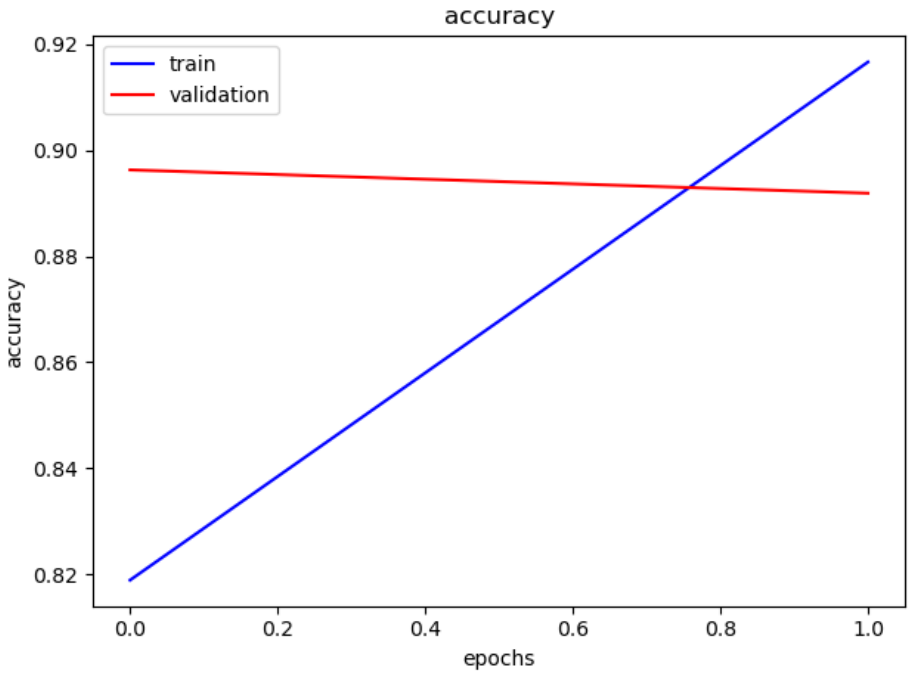

Рисунок 1 – График точности

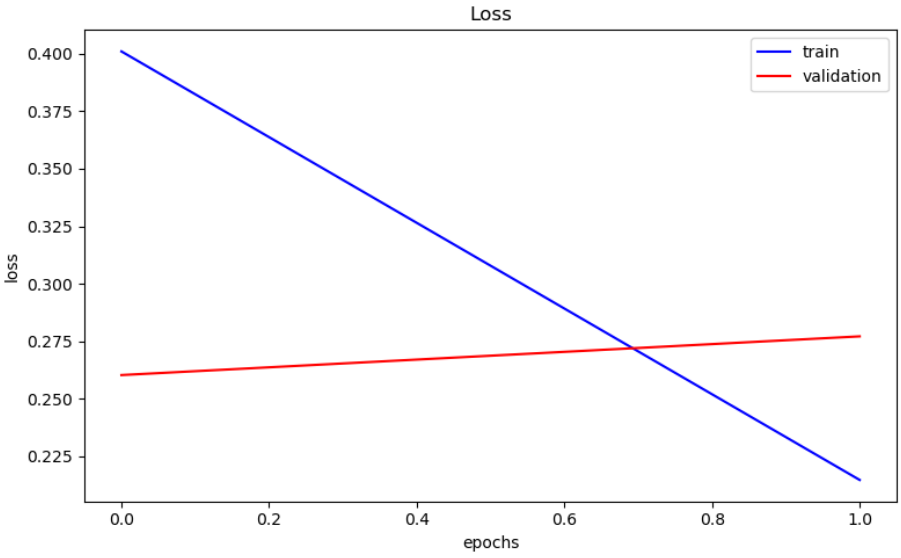

Рисунок 2 – График потерь

2. Исследовать результаты при различном размере вектора представления текста.

При изменении максимального размера вектора представления текста с 10 тыс. до 50 тыс. точность: на тренировочных упала до ~ 90%, на

валидационных также уменьшилась ~ 88%. Графики точности и ошибки предоставлены на рис. 3 и рис. 4 соответственно.

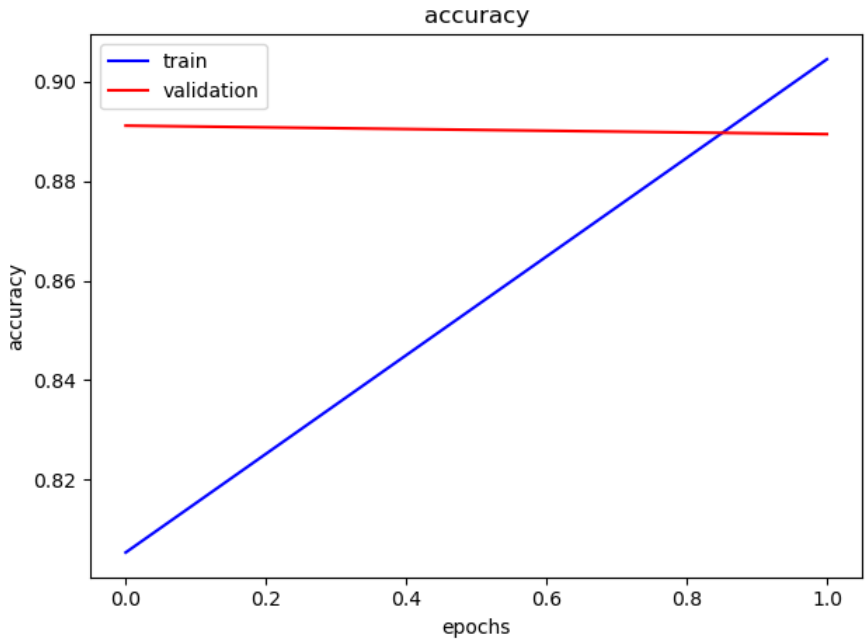

Рисунок 3 – График точности при размере вектора представления текста - 50 тыс. обзоров

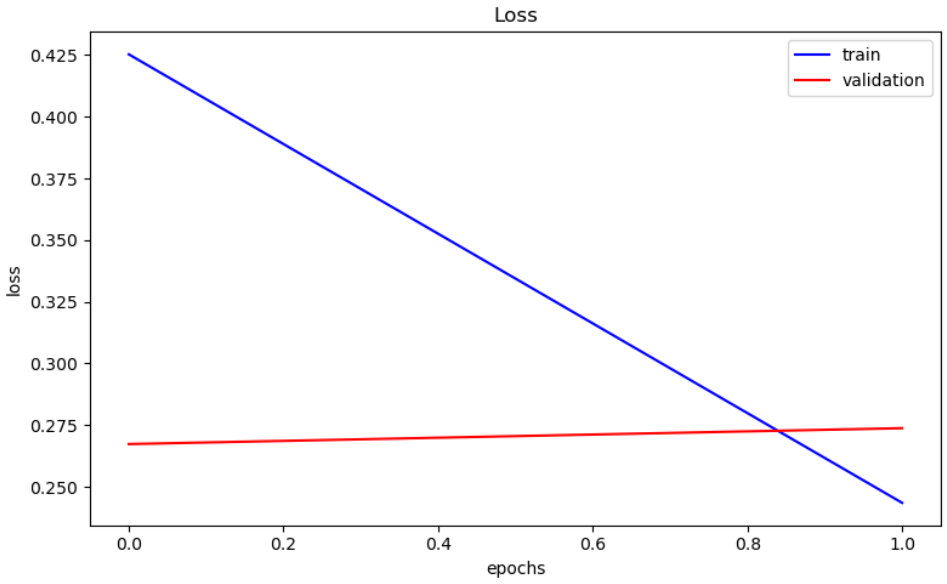

Рисунок 4 – График потерь при размере вектора представления текста - 50 тыс. обзоров

При изменении максимального размера вектора представления текста - до 1 тыс.. Точность: на тренировочных упала ~ 85%, на валидационных

также уменьшилась до ~ 86%. Графики точности и ошибки предоставлены на рис. 5 и рис. 6 соответственно.

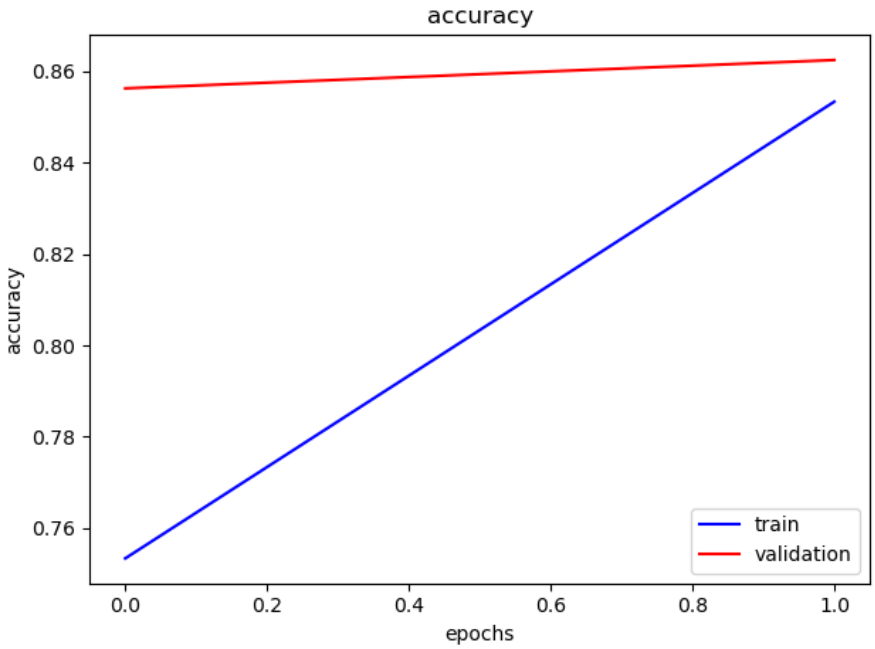

Рисунок 5 – График точности при размере вектора представления текста 1 тыс. обзоров

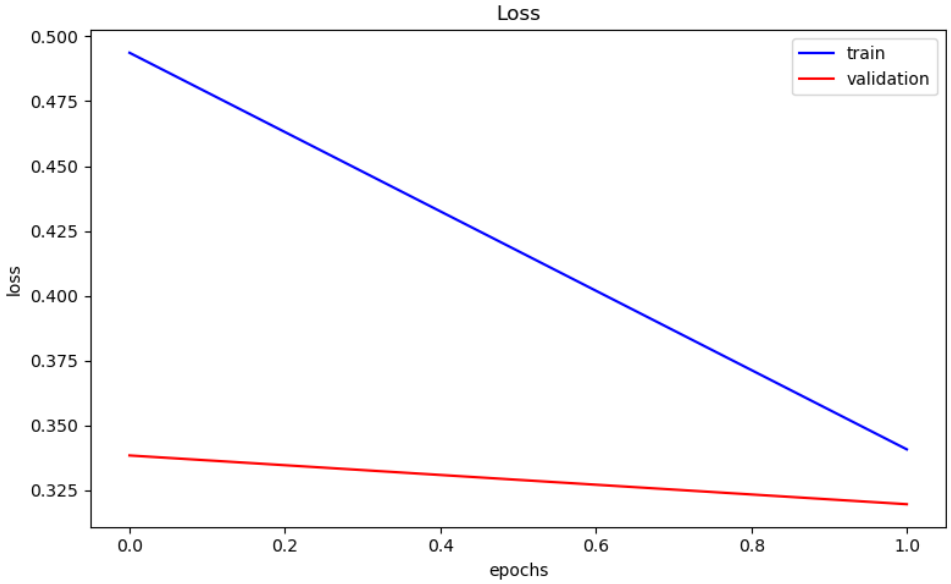

Рисунок 6 – График потерь при размере вектора представления текста 1 тыс. обзоров

Из графиков на рис. 1 - 5, можно сделать вывод, что точность падает с уменьшением размера словаря, так как мы убираем часть слов из обзоров.

Из-за этого оценка некоторых обзоров меняется, поэтому сеть не может их точно классифицировать.

3. Функция, которая позволяющая ввести пользовательский текст:

```
def gen_custom_x(custom_x, word_index):
    def get_index(a, index):
        new_list = a.split()
        for i, v in enumerate(new_list):
            ind = index.get(v.lower());
            new_list[i] = ind
        return new_list
    for i in range(len(custom_x)):
        custom_x[i] = get_index(custom_x[i], word_index)
    for index_j, i in enumerate(custom_x):
        for index, value in enumerate(i):
            if value is None:
                custom_x[index_j][index] = 0
            elif value > 10000:
                custom_x[index_j][index] = 0
    return custom_x
```

При помощи данной функции можно получить из массива строк (обзоров) массив представлений в виде индексов слов в imdb датасете и подготовленные для прогона через модель. График точности оценки фильма, при прогоне через написанный датасет из 5 обзоров (см. рис. 7), предоставлена на рис. 8.

```
custom_x = [
        "There is something much more in this movie than meets the eye. Watch it and find for yourself",
        "I had the privilege of seeing this movie before it came out, it blew me away.",
        "Bottom Line, this is a movie i could write a novel about, and it certainly deserves that attention.",
        "This movie is a disaster on many levels, but where it fails most miserably is at attempting to put dreams
        "Okay, maybe this movie isn't awful."
]
custom_y = [1., 1., 1., 0., 0.]
```

Рисунок 7 – Пользовательский текст

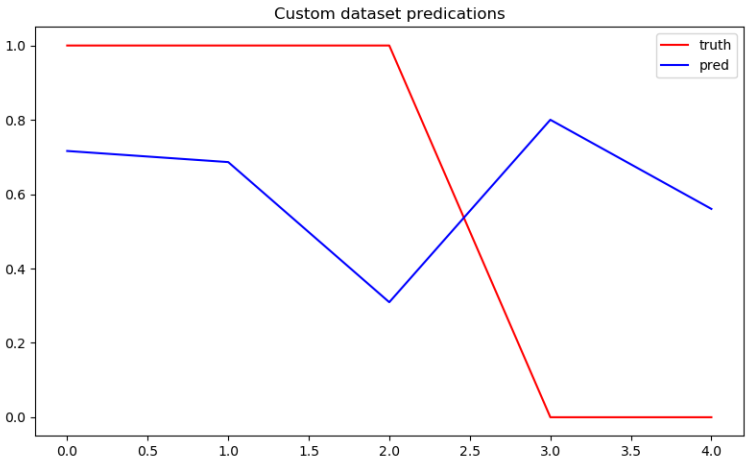

Рисунок 8 – График точности оценки фильма

Выводы.

В ходе работы была изучена задача классификация обзоров из датасета IMDB. Найдена архитектура, дающая точность 89%. Было выяснено, что при уменьшении максимального размера словаря точность уменьшается, вероятно из-за того, что отсекается часть «словарного запаса». Решен вопрос проверки на успех фильмов среди пользовательских обзоров.