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

ОТЧЕТ

по лабораторной работе №6

по дисциплине «Искусственные нейронные сети»

Тема: «Прогноз успеха фильма по обзорам»

Студентка гр. 7381 _____ Давкаева В.С.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург

2020

Цель

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

Задачи

- Ознакомиться с задачей регрессии
- Изучить способы представления текста для передачи в ИНС
- Достигнуть точность прогноза не менее 95%

Требования

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)

Ход работы

1)Была построена и обучена нейронная сеть. Код программы представлен в приложении А. Точность данной модели равна 0.89. Ниже представлены графики точности и ошибок.

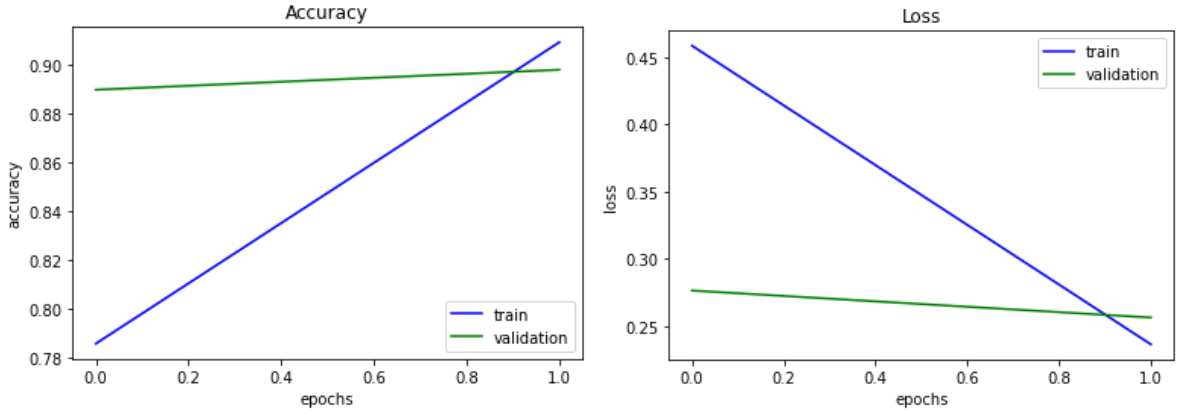

Рисунок 1 – График точности и ошибок.

2)Для исследования результатов при различном размере вектора возьмем значения 3000, 1000 и 300. Ниже представлены соответственные графики точности и ошибок (рис.2, рис.3, рис.4).

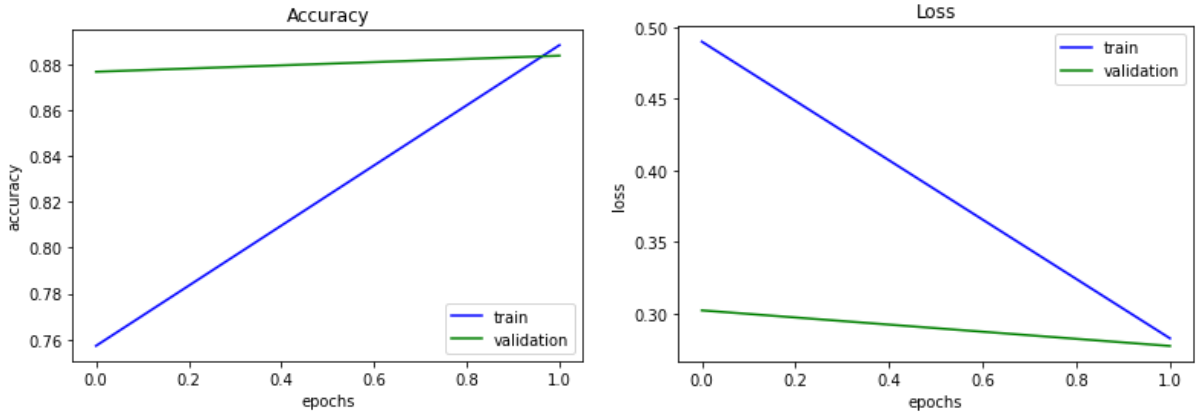

Рисунок 2 – График точности и ошибок (длина вектора = 3000, точность = 0.88).

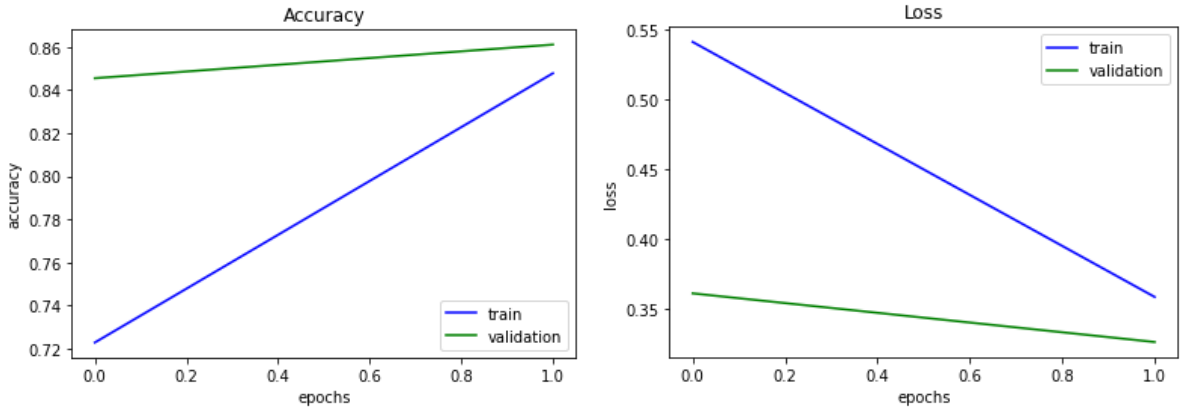

Рисунок 3 – График точности и ошибок (длина вектора = 1000, точность = 0.86).

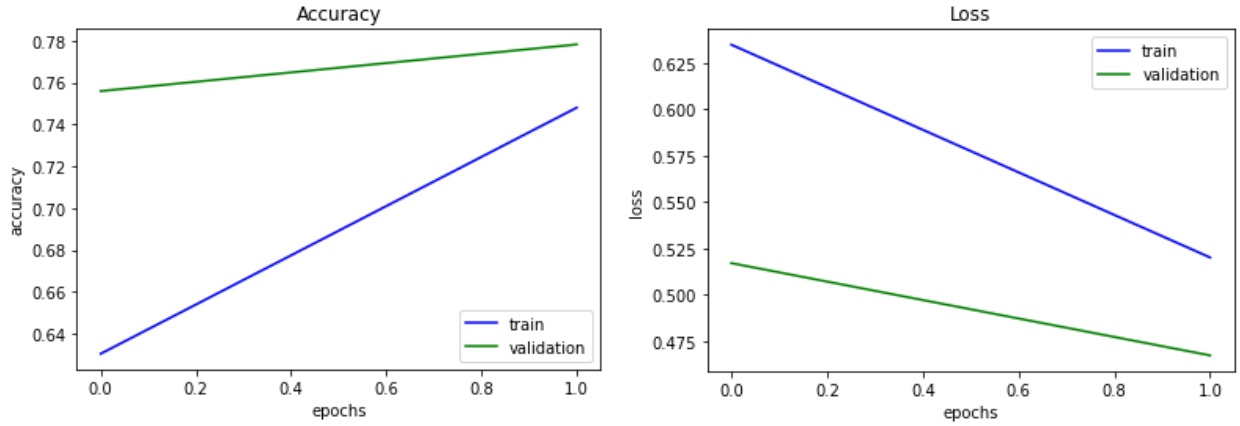

Рисунок 4 – График точности и ошибок (длина вектора = 300, точность = 0.77).

Сделаем вывод, что при уменьшении длины вектора уменьшается точность.

3)Для проверки работы сети на пользовательском тексте были приведены результаты программы на положительный и отрицательный отзыв

Первый файл с положительным отзывом сеть оценила в 0.81, что говорит о его положительности.

“The platform is a film of discovery. He left a certain imprint and a bunch of thoughts inside me, and they scare the hell out of me. I recommend this film to absolutely everyone, but I also warn everyone, be prepared for the consequences. Adjust yourself and your perception.”

Файл с отрицательным отзывом сеть оценила в 0.18, что говорит о его отрицательности.

“I do not advise anyone to watch this film. This is a disgrace. SHAME! Such works disgrace domestic cinema.”

Вывод

В ходе выполнения лабораторной работы была построена и обучена нейронная сеть для обработки текста, были исследованы результаты при различном размере вектора представления текста, была написана функция, которая позволяет ввести пользовательский текст.

ПРИЛОЖЕНИЕ А

Исходный код

```
import matplotlib.pyplot as plt
import numpy as np
from keras import layers, Sequentials
from keras.datasets import imdb

vector_size = 10000

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

index = imdb.get_word_index()
reverse_index = dict([(value, key) for (key, value) in index.items()])
decoded = " ".join( [reverse_index.get(i - 3, "#") for i in data[0]] )
print(decoded)

def load_text(filename):
    punctuation = ['.',',',':',';','!','?','(',')']
    text = []
    with open(filename, 'r') as f:
        for line in f.readlines():
            text += [s.strip(''.join(punctuation)).lower() for s in line.s
trip().split()]
    print(text)
    indexes = imdb.get_word_index()
    encoded = []
    for w in text:
        if w in indexes and indexes[w] <10000:
            encoded.append(indexes[w])
    return np.array(encoded)

def vectorize(sequences, dimension=vector_size):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1
    return results

text = load_text('text.txt')
data = vectorize(data)
targets = np.array(targets).astype("float32")

test_x = data[:10000]
test_y = targets[:10000]
train_x = data[10000:]
train_y = targets[10000:]
```

```
model = Sequential()
model.add(layers.Dense(32, activation="relu", input_shape=(vector_size,)))
model.add(layers.Dropout(0.3, noise_shape=None, seed=None))
model.add(layers.Dense(32, activation="relu"))
model.add(layers.Dropout(0.2, noise_shape=None, seed=None))
model.add(layers.Dense(32, activation="relu"))
model.add(layers.Dense(1, activation="sigmoid"))

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

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

)

text = vectorize([text])
res = model.predict(text)
print(res)

plt.plot(results.history['loss'], 'b', label='train')
plt.plot(results.history['val_loss'], 'g', label='validation')
plt.title('Loss')
plt.ylabel('loss')
plt.xlabel('epochs')
plt.legend()
plt.show()
plt.clf()

plt.plot(results.history['accuracy'], 'b', label='train')
plt.plot(results.history['val_accuracy'], 'g', label='validation')
plt.title('Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epochs')
plt.legend()
plt.show()
plt.clf()
```