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

Студент гр. 8383 _____ Бессуднов Г. И.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург
2021

Цель работы

Прогноз успеха фильмов по обзорам.

Основные теоретические положения

С помощью анализа настроений можно определить отношение (например, настроение) человека к тексту, взаимодействию или событию. Поэтому сентимент-анализ относится к области обработки естественного языка, в которой смысл текста должен быть расшифрован для извлечения из него тональности и настроений.

Датасет IMDb состоит из 50 000 обзоров фильмов от пользователей, помеченных как положительные (1) и отрицательные (0).

- 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.

Датасет был создан исследователями Стэнфордского университета и представлен в статье 2011 года, в котором достигнутая точность предсказаний была равна 88,89%. Датасет также использовался в рамках конкурса сообщества Kaggle «Bag of Words Meets Bags of Popcorn» в 2011 году.

Выполнение работы

Модель для работы была построена на основании теоретических данных с некоторыми изменениями, были добавлены новые слои и изменены параметры уже существующих. Код программы представлен в Приложении А. Ниже представлена схема модели:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 128)               1280128
_________________________________________________________________
```

```
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 256)               33024
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 129
=================================================================
Total params: 1,362,689
Trainable params: 1,362,689
Non-trainable params: 0
_________________________________________________________________
```

В ходе тренировки была получена точность 89.46%.

Поменяем размер вектора представления текста. При его длине равной 5000 получаем точность 88.84%, что меньше стандартной размерности. При длине вектора 20000 имеем точность 89.73%, что будет несколько больше. При размерности 30000 точность 89.44%, для больших размерностей эксперименты провести не удалось, так как не хватало вычислительных мощностей.

Написанная модель была сохранена и далее использована в программе `semantic_analyzer.py` (код представлен в Приложении Б) для возможности ввода пользовательского обзора и предсказании его настроения. Далее были проведены 3 эксперимента:

1. Положительный обзор. Программе был указан файл с следующим текстом:

   Today I've watched American Psycho seven times in a row and can't be

bored of it. This movie is very great and intense with interesting characters, thought provoking and aesthetic scenes. Christian Bale play is top notch and this film features very good soundtrack.

Из содержания понятно, что обзор положительный, программа успешно смогла это определить.

2. Отрицательный обзор. Программе был указан файл с следующим текстом:

   Yesterday I've watched Revolver and this film is very very bad and meaningless. I didn't understand anything and everything seems to be out of context. I felt like this film was pathos for the sake of pathos and that's it. Didn't like it at all, worst film by Guy Ritchie.

   Из содержания понятно, что обзор отрицательный, программа успешно смогла это определить.

3. Обзор-ловушка. Была проведена попытка обмануть сеть. Программе был указан файл с следующим текстом:

   I've wanted to watch this film so bad and after sitting through it I can say that it was way beyond what expected. Never I've seen such diverse cast of characters and deep storyline as in the No Country For Old Man. Film catches you from the first frame and never let you down. It is all here: evil, pleasantly disgusting and cruel main villain, cowboy-like protagonist and, of course, the old man himself, who can't handle the events of this story. I can't recommend this film enough.

   Данный текст наполнен негативными словами, но несет в себе положительный смысл. Сеть не смогла правильно определить настроение обзора.

Выводы

В ходе работы была реализована сеть, прогнозирующая успех фильмов по обзорам. Было исследовано влияние размера вектора представления текста на

точность сети, а также была написана программа для предсказания пользовательских обзоров.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ MAIN.PY

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from keras.utils import to_categorical
from keras import models
from keras import layers
from keras.datasets import imdb

vector_size = 10000

def vectorize(sequences, dimension = 10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1
    return results



if __name__ == "__main__":
    (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)




    data = vectorize(data, vector_size)
    targets = np.array(targets).astype("float32")

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

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


        model = models.Sequential()
        # Input - Layer
        model.add(layers.Dense(128,       activation       =       "relu",
input_shape=(vector_size, )))

        # Hidden - Layers
        model.add(layers.Dropout(0.2, noise_shape=None, seed=None))
        model.add(layers.Dense(128, activation = "sigmoid"))
        model.add(layers.Dense(256, activation = "relu"))
        model.add(layers.Dropout(0.35, noise_shape=None, seed=None))
        model.add(layers.Dense(128, activation = "relu"))

        # Output- Layer
        model.add(layers.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  =  750,
validation_data = (test_x, test_y), verbose=0)

        print(np.mean(H.history["val_accuracy"]))

        # model.save("model_lb6.h5")
```

ПРИЛОЖЕНИЕ Б

КОД ПРОГРАММЫ SEMANTIC_ANALYZER.PY

```
import numpy as np
from keras.models import load_model
from pathlib import Path
from tensorflow.keras.datasets import imdb
from main import vectorize

def predictReview(filename):
    path = Path(filename)
    if (not path.exists()):
        print("Oops! Can't find file", filename)
        return

    reviewStr = ""
    with open(path.absolute()) as f:
        reviewStr = f.read()

    reviewStr = "".join(char for char in reviewStr if char.isalpha()
or char.isspace() or char == "'").lower().strip().split()
    indicies = []
    wordsIndicies = imdb.get_word_index()

    for word in reviewStr:
        i = wordsIndicies.get(word)
        if i is not None and i < 10000:
            indicies.append(i + 3)

    reviewVector = vectorize(np.asarray([indicies]))
    res = model.predict(reviewVector)
    return res
```

```
model = load_model("model_lb6.h5")
print("Enter file name with review: ")
inp = str(input())

res = predictReview(inp)
if res >= 0.5:
    print("good")
else:
    print("bad")
```