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

ОТЧЕТ

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

Студентка гр. 8382 _____ Ивлева О.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2021

Цель.

Реализовать ИНС для прогноза успеха фильмов по обзорам (Predict Sentiment From Movie Reviews)

Задание.

- Ознакомиться с задачей регрессии
- Изучить способы представления текста для передачи в ИНС
- Достигнуть точность прогноза не менее 95%
- Построить и обучить нейронную сеть для обработки текста.
- Исследовать результаты при различном размере вектора представления текста.
- Написать функцию, которая позволяет ввести пользовательский текст (в отчете привести пример работы сети на пользовательском тексте).

Ход работы.

1. Была создана модель искусственной нейронной сети для прогноза успеха фильмов оп обзорам.

2. Для исследования влияния размера вектора представления текста на точность результата сети выберем следующие размерности: 1000, 3000, 5000, 7000, 9000, 11000, 13000, 15000.

Обучим модель для заданных размеров вектора представления текста и оценим точность нейронной сети. Графики точностей для модели представлены на рис. 1.

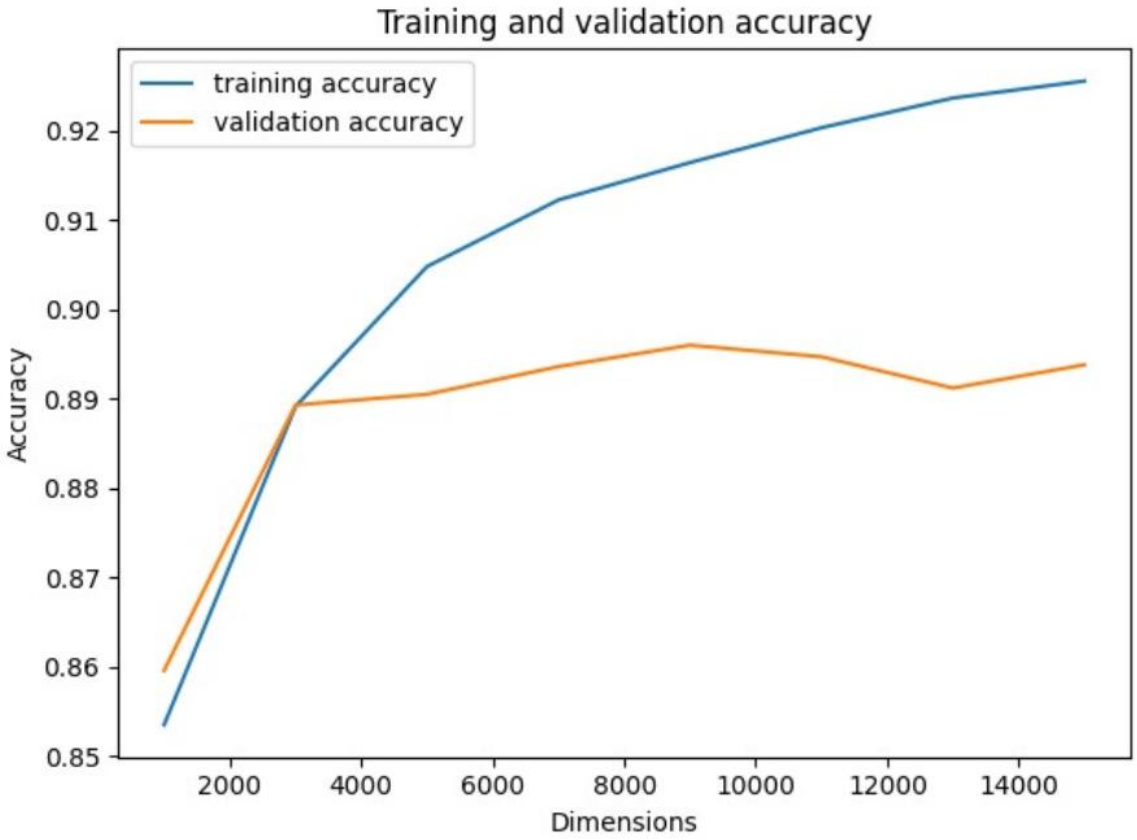

Рисунок 1 – График точности модели

Как видно из графика в обучении модели достигается пиковая точность на проверочных данных, равная около 0.89 при размерности, равной 3000. При дальнейшей увеличении размерности изменения в точности на проверочных данных незначительны.

3. Была написана функция, которая загружала пользовательский текст из файла и позволяла оценить отзыв. Для проверки был оценен следующий отзыв: «Truly a masterpiece, The Best Hollywood film of 2019, one of the Best filmsof the decade.» Результат оценки ИНС — 0.6191652, что говорит о том, что данный отзыв — положительный.

Выводы.

В ходе выполнения данной работы была создана ИНС для прогноза успеха фильмов по отзывам. Также были исследованы результаты при различном размере вектора представления текста. По результатам можно сделать вывод о том, что оптимальная точность достигается при длине вектора, равной 3000 и выше, однако, при увеличении длины точность на проверочных данных меняется незначительно и не улучшается. Также была написана функция, загружающая пользовательский отзыв из файла. Результаты оценивания отзыва показали, что данная функция работает корректно.

ПРИЛОЖЕНИЕ А

Исходный код программы. Файл lr4.py

```
import numpy as np
import matplotlib.pyplot as plt

from keras.datasets import cifar10
from keras.layers import Input, Convolution2D, MaxPooling2D, Dense, Dropout,
Flatten
from keras.models import Model
from keras.utils import np_utils

# Setting hyper-parameters:
batch_size = 150
num_epochs = 10
kernel_size = 4
pool_size = 2
conv_depth_1 = 32
conv_depth_2 = 64
hidden_size = 512
with_dropout = True
if with_dropout:
    drop_prob_1 = 0.25
    drop_prob_2 = 0.5

# Loading data:
(X_train, y_train), (X_test, y_test) = cifar10.load_data()
num_train, depth, height, width = X_train.shape
num_test = X_test.shape[0]
num_classes = np.unique(y_train).shape[0]

# Normalizing data:
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= 255.0
X_test /= 255.0

# Converting labels:
Y_train = np_utils.to_categorical(y_train, num_classes)
Y_test = np_utils.to_categorical(y_test, num_classes)

# Building model:
inp = Input(shape=(depth, height, width))
conv_1 = Convolution2D(conv_depth_1, (kernel_size, kernel_size), padding='same',
activation='relu')(inp)
conv_2 = Convolution2D(conv_depth_1, (kernel_size, kernel_size), padding='same',
activation='relu')(conv_1)
```

```
pool_1 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_2)
if with_dropout:
    drop_1 = Dropout(drop_prob_1)(pool_1)
conv_3 = Convolution2D(conv_depth_2, (kernel_size, kernel_size), padding='same',
activation='relu')(drop_1 if with_dropout else pool_1)
conv_4 = Convolution2D(conv_depth_2, (kernel_size, kernel_size), padding='same',
activation='relu')(conv_3)
pool_2 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_4)
if with_dropout:
    drop_2 = Dropout(drop_prob_1)(pool_2)
flat = Flatten()(drop_2 if with_dropout else pool_2)
hidden = Dense(hidden_size, activation='relu')(flat)
if with_dropout:
    drop_3 = Dropout(drop_prob_2)(hidden)
out = Dense(num_classes, activation='softmax')(drop_3 if with_dropout else
hidden)

model = Model(inp, out)
model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])

# Fitting model:
h = model.fit(X_train, Y_train, batch_size=batch_size, epochs=num_epochs,
verbose=1, validation_data=(X_test, Y_test))

print(model.evaluate(X_test, Y_test))

# Plotting:
loss = h.history['loss']
val_loss = h.history['val_loss']
acc = h.history['accuracy']
val_acc = h.history['val_accuracy']
epochs = range(1, len(loss) + 1)

plt.plot(range(1, num_epochs + 1), loss, 'b', label='Training loss')
plt.plot(range(1, num_epochs + 1), val_loss, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
plt.clf()

plt.plot(range(1, num_epochs + 1), acc, 'b', label='Training acc')
plt.plot(range(1, num_epochs + 1), val_acc, 'r', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
```

```
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```