МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЁТ

по лабораторной работе №7

по дисциплине «Искусственные нейронные сети»

Тема: Классификация обзоров фильмов

Студент гр.8382 _____ Фильцин И.В.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2021

Цель работы

Классификация последовательностей - это проблема прогнозирующего моделирования, когда у вас есть некоторая последовательность входных данных в пространстве или времени, и задача состоит в том, чтобы предсказать категорию для последовательности.

Проблема усложняется тем, что последовательности могут различаться по длине, состоять из очень большого словарного запаса входных символов и могут потребовать от модели изучения долгосрочного контекста или зависимостей между символами во входной последовательности.

В данной лабораторной работе также будет использоваться датасет IMDb, однако обучение будет проводиться с помощью рекуррентной нейронной сети.

Задание

Ознакомиться с рекуррентными нейронными сетями

Изучить способы классификации текста

Ознакомиться с ансамблированием сетей

Построить ансамбль сетей, который позволит получать точность не менее 97

Ход работы

В ходе работы были реализованы 2 модели.

Модель 1:

Layer (**type**)	Output Shape	Param #
embedding_53 (Embedding)	(None, 500, 32)	320000
conv1d_21 (Conv1D)	(None, 500, 32)	3104
max_pooling1d_21 (MaxPooling	(None, 250, 32)	0
lstm_24 (LSTM)	(None, 100)	53200

dense_110 (Dense) (None, 1) 101

==========

Модель 2:

Layer (**type**)	Output Shape	Param #
embedding_54 (Embedding)	(None, 500, 32)	320000
simple_rnn_2 (SimpleRNN)	(None, 500, 32)	2080
simple_rnn_3 (SimpleRNN)	(None, 32)	2080
dense_111 (Dense)	(None, 1)	33

Точность 1ой модели составляет 90.37. Точность 2ой модели составляет 87.93. Из этих моделей был построен ансамбль, точность которого на тестовых данных составила 91.24. Т.е. ансамбль выдает более точный результат, чем каждая из двух моделей по отдельности.

Полученный ансамбль был протестирован на 4 текстах.

Текст 1 - 10 звезд, результат работы сети - 0.99347988

Wow, what a masterpiece!! Really enjoyed **this** movie, full of action, great dialogues **and** perfect acting **not** forgetting the wonderful choice of soundtracks. A very unique movie which has similar taste to the John Wick franchise but more exciting with some family action **and** moments. A little bit **short in length** but worth every scene. Bob did a wonderful job **in this** movie **and** suited him perfectly for the role, wish to see him **in** more action like this. I highly recommend **this** movie to every John Wick fan, action **or** intense **and** realistic fight sequences... With a purchase to the collection

Текст 2 - 1 звезда, результат работы сети - 0.05792282

The movie was a disappointment for me. I saw the trailer for the movie before I saw the actual movie itself. I thought **this** movie would be more like a vigilante movie **and** I love these sort of movies.

But **in** effect **this** was more of a dumb backstory kind of generic Hollywood movies dished out. **This** follows the same case as John Wick. The first John Wick was good, but they killed it with the sequels. They should have stopped after the first one. "I Am Wrath" is also another good movie **and** better than **this** one.

I have already seen good vigilante movies **in** the past. Death Wish **and** The Exterminator are prime examples of good movies that have survived **into** the present era. Vigilante 1983 starring the late Robert Forster **and** Fighting Back 1982 starring Tom Skerrit are also terrific.

In Nobody(2021), the story follows a family man played by Bob Odenkirk(of Breaking Bad fame) who fails to stop some inept burglars from stealing his possessions. After which he has a change of heart **and** lashes **out** vigilante–style to vent **out** his anger. The rest of the movie deals with repercussions of his decisions.

The acting **in** the movie is so–so. The direction **and** background music **and** sound–effects is typical of a generic Hollywood action thriller. Overall **this** movie was a real waste of my time **and** I would never recommend you to watch it. See all of the above movies I have listed above.

Текст 3 - 5 звезд, результат работы сети - 0.87129948

It’s a good time while watching but like most movies these days (ok boomer) you dislike more things about it as you think about it. Which is fine I guess.

Do yourself a favor and rent "A History of Violence" it’s the same story but with much deeper exploration of the characters **and** what something like **this** does to them

Текст 4 - 9 звезд, результат работы сети - 0.97708505

Finally got to see some amazing movie **in** the cinema. To be honest **this** film is filled with lots of terrific moments **and** surprises (**not** twists) with its badass characters, especially the protagonist "Bob Odenkirk" he completely nailed it.

From the start to the **end** of the film was a BANG! ... for all those who love movies like John Wick, Equalizer shouldn’t miss this movie at all ... It worth the money

Таким образом, можно сказать, что сеть выдает верные результаты.

Вывод

В ходе лабораторной работы был реализован ансамбль сетей.

Приложение А.

Исходный код

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Embedding, Conv1D, MaxPooling1D, Dropout, SimpleRNN, GRU
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.datasets import imdb

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

train_length = (data.shape[0] // 10) * 8

X_train = data[:train_length]
Y_train = targets[:train_length]
X_test = data[train_length:]
Y_test = targets[train_length:]

max_review_length = 500
X_train = sequence.pad_sequences(X_train, maxlen=max_review_length)
X_test = sequence.pad_sequences(X_test, maxlen=max_review_length)

embedding_vecor_length = 32
model1 = Sequential()
model1.add(Embedding(num_words, embedding_vecor_length, input_length=max_review_length))
model1.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model1.add(MaxPooling1D(pool_size=2))
model1.add(LSTM(100))
model1.add(Dense(1, activation='sigmoid'))

model1.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

print(model1.summary())

model1.fit(X_train, Y_train, validation_data=(X_test, Y_test), epochs=3, batch_size=64)
scores = model1.evaluate(X_test, Y_test, verbose=0)

print("Accuracy: %.2f%%" % (scores[1]*100))

model2 = Sequential()
model2.add(Embedding(num_words, embedding_vecor_length, input_length=max_review_length))
```

```
model2.add(SimpleRNN(32, return_sequences=True))
model2.add(SimpleRNN(32))
model2.add(Dense(1, activation="sigmoid"))
model2.compile(loss='binary_crossentropy',  optimizer='adam',  metrics=['accuracy'])

print(model2.summary())

model2.compile(loss='binary_crossentropy',  optimizer='adam',  metrics=['accuracy'])

print(model2.summary())

model2.fit(X_train,  Y_train,  validation_data=(X_test,  Y_test),  epochs=3, batch_size=64)
scores = model2.evaluate(X_test,  Y_test,  verbose=0)

print("Accuracy:  %.2f%%" % (scores[1]*100))

models = [model1, model2]


def ensemble_prediction(data):
    predictions = np.array([])
    for model in models:
        predictions = np.append(predictions,  model.predict(data))
    predictions = np.mean(predictions,  0)
    return predictions.flatten()


def ensemble_accuracy(prediction,  label):
    return np.count_nonzero(np.round(prediction)  == label) / len(label)


print("Ensemble accuracy:  %.2f%%" % ensemble_accuracy(ensemble_prediction(X_test), Y_test))


def load_text(file):
    str = ""
    with open(file,  'r')  as fd:
        str = fd.read()
    result = []
    index = imdb.get_word_index()

    for w in str.split():
        i = index.get(w.lower())
        if i is not None and i < num_words:
            result = np.append(result,  i + 3)
```

```
    return  result


def prediction_text(file):
    vec = load_text(file)
    pad = sequence.pad_sequences([vec],    maxlen=max_review_length)
    return  ensemble_prediction(pad)


print(prediction_text("1"))
print(prediction_text("2"))
print(prediction_text("3"))
print(prediction_text("4"))
```