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

ОТЧЕТ

по лабораторной работе №7

по дисциплине «Искусственные нейронные сети»

Тема: Классификация обзоров фильмов

Студент гр. 8383 _____ Шишкин 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.

Задачи.

- Ознакомиться с рекуррентными нейронными сетями
- Изучить способы классификации текста
- Ознакомиться с ансамблированием сетей
- Построить ансамбль сетей, который позволит получать точность не менее 97%

Выполнение работы.

Построена нейронная сеть, представленная в листинге 1.

Листинг 1 – Модель №1

```
model = Sequential()
model.add(Embedding(max_words, embedding_vector_length, input_length=max_review_length))
model.add(LSTM(100))
model.add(Dense(1, activation='sigmoid'))
```

Результат работы модели №1 с 3 эпохами представлен в листинге 2.

Листинг 2 – Результат модели №1

```
Epoch 1/3
625/625 [==============================] - 225s 357ms/step - loss:
0.5418 - accuracy: 0.6999 - val_loss: 0.3137 - val_accuracy: 0.8714
Epoch 2/3
625/625 [==============================] - 227s 364ms/step - loss:
0.2669 - accuracy: 0.8924 - val_loss: 0.2695 - val_accuracy: 0.8876
Epoch 3/3
625/625 [==============================] - 237s 378ms/step - loss:
0.2102 - accuracy: 0.9216 - val_loss: 0.2858 - val_accuracy: 0.8817
Accuracy: 88.17%
```

Точность у модели №1 – 92% на обучающих и 88% на тестовых.

Для модели №2 добавим одномерный слой CNN и максимальный пул после слоя Embedding (листинг 3).

Листинг 3 – Модель №2

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

Результат работы модели №2 с 3 эпохами представлен в листинге 4.

Листинг 4 – Результат модели №2

```
Epoch 1/3
625/625 [==============================] - 104s 164ms/step - loss:
0.5445 - accuracy: 0.6807 - val_loss: 0.2728 - val_accuracy: 0.8923
Epoch 2/3
625/625 [==============================] - 100s 160ms/step - loss:
0.2393 - accuracy: 0.9086 - val_loss: 0.2543 - val_accuracy: 0.8977
Epoch 3/3
625/625 [==============================] - 99s 158ms/step - loss: 0.1697
- accuracy: 0.9395 - val_loss: 0.2863 - val_accuracy: 0.8860
Accuracy: 88.60%
```

Точность у модели №2 – 94% на обучающих и 89% на тестовых.

Добавим слои Dropout для модели №3 (листинг 5).

Листинг 5 – Модель №3

```
model = Sequential()
model.add(Embedding(max_words, embedding_vector_length, input_length=max_review_length))
model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Dropout(0.2))
model.add(LSTM(100))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
```

Результат работы модели №3 с 3 эпохами представлен в листинге 6.

Листинг 6 – Результат модели №3

```
Epoch 1/3
625/625 [==============================] - 104s 164ms/step - loss: 0.4936 - accuracy: 0.7215 - val_loss: 0.2546 - val_accuracy: 0.8963
Epoch 2/3
625/625 [==============================] - 100s 160ms/step - loss: 0.2020 - accuracy: 0.9253 - val_loss: 0.2653 - val_accuracy: 0.8989
Epoch 3/3
625/625 [==============================] - 101s 162ms/step - loss: 0.1535 - accuracy: 0.9453 - val_loss: 0.2676 - val_accuracy: 0.8918
Accuracy: 89.18%
```

Точность у модели №3 – 95% на обучающих и 89 % на тестовых.

Так как модель №3 показала лучшие результаты, применим для нее ансамблирование. Создадим 5 разных моделей №3 с разными данными (листинг 7).

Листинг 7 – Ансамблирование

```
models = []
scores = []
m = 4
```

```
for i in range(m):
    x_train = train_x[int(len(train_x) / m) * i: int(len(train_x) / m)
* (i + 1)]
    y_train = train_y[int(len(train_y) / m) * i: int(len(train_y) / m)
* (i + 1)]

    models.append(Sequential())
    models[i].add(Embedding(max_words,         embedding_vector_length,
input_length=max_review_length))
    models[i].add(Conv1D(filters=32,   kernel_size=3,   padding='same',
activation='relu'))
    models[i].add(MaxPooling1D(pool_size=2))
    models[i].add(Dropout(0.2))
    models[i].add(LSTM(100))
    models[i].add(Dropout(0.2))
    models[i].add(Dense(1, activation='sigmoid'))

    models[i].compile(loss='binary_crossentropy',     optimizer='adam',
metrics=['accuracy'])
    models[i].fit(x_train, y_train, validation_data=(x_train, y_train),
epochs=3, batch_size=64)
    print(models[i].evaluate(x_train, y_train, verbose=0))
    scores.append(models[i].evaluate(x_train, y_train, verbose=0)[1])

print("Mean accuracy: %.2f%%" % (np.mean(scores) * 100))
```

Программа выдала следующий результат:

```
Mean accuracy: 97.22%
```

Были написаны функции для пользовательского текста.

Вывод программы для текста “it is not good film. i am upset”:

```
Mean predictions 53.35%
Good
```

Вывод программы для текста “It’s not so bad, but I cannot advise him to others”:

```
Mean predictions 19.73%
Bad
```

Вывод программы для текста “It's bad film”:

```
Mean predictions 27.98%
Bad
```

Вывод программы для текста “it's very good film!”:

```
Mean predictions 79.96%
Good
```

Вывод программы для текста “it is not horrible film, the acting is not bad”:

```
Mean predictions 6.02%
Bad
```

Вывод программы для текста “it is good film. i am not upset”:

```
Mean predictions 59.46%
Good
```

Выводы.

Был найден оптимальный набор ИНС для классификации текста. Проведено ансамблирование моделей. Написана функция, которая позволяет загружать текст и получать результат ансамбля сетей. Проведено тестирование сетей на своих текстах.