In [ ]:
```
from __future__ import print_function

from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers import Dense, Embedding
from keras.layers import LSTM
from keras.datasets import imdb
from keras.layers import Conv1D
from keras.layers import MaxPooling1D
from keras.layers import Dropout
from keras.losses import mean_squared_error

max_features = 20000

# обрезание текстов после данного количества слов (среди top max_features на
иболее используемые слова)
maxlen = 80
batch_size = 128 # увеличьте значение для ускорения обучения

print('Загрузка данных...')
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=max_feature
s)
print(len(x_train), 'тренировочные последовательности')
print(len(x_test), 'тестовые последовательности')

print('Pad последовательности (примеров в x единицу времени)')
x_train = sequence.pad_sequences(x_train, maxlen=maxlen)
x_test = sequence.pad_sequences(x_test, maxlen=maxlen)
print('x_train shape:', x_train.shape)
print('x_test shape:', x_test.shape)

print('Построение модели...')
model = Sequential()

#model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='rel
u'))
#model.add(MaxPooling1D(pool_size=2))
# model.add(keras.layers.Dropout(0.3))

model.add(Embedding(max_features, 128))
model.add(LSTM(256, dropout=0.2, recurrent_dropout=0.2))

model.add(Dense(1, activation='sigmoid'))

# стоит попробовать использовать другие оптимайзер и другие конфигурации опт
имайзеров
model.compile(loss='mean_squared_error', # loss='binary_crossentropy' прогон
ы 1-4
              optimizer='adam', # optimizer='adam' прогон 4
              metrics=['accuracy']) # metrics=['accuracy'])

print('Процесс обучения...')
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=50, # увеличьте при необходимости
          validation_data=(x_test, y_test))
score, acc = model.evaluate(x_test, y_test,
                            batch_size=batch_size)
print('Результат при тестировании:', score)
print('Тестовая точность:', acc)
```

```
Загрузка данных...
25000 тренировочные последовательности
25000 тестовые последовательности
Pad последовательности (примеров в x единицу времени)
x_train shape: (25000, 80)
x_test shape: (25000, 80)
Построение модели...
Процесс обучения...
```

```
/home/roman/anaconda3/lib/python3.7/site-packages/tensorflow_core/python/fram
ework/indexed_slices.py:433: UserWarning: Converting sparse IndexedSlices to
a dense Tensor of unknown shape. This may consume a large amount of memory.
  "Converting sparse IndexedSlices to a dense Tensor of unknown shape. "
```

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/50
25000/25000 [==============================] - 208s 8ms/step - loss: 0.1597 -
accuracy: 0.7669 - val_loss: 0.1164 - val_accuracy: 0.8381
Epoch 2/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0918 -
accuracy: 0.8774 - val_loss: 0.1173 - val_accuracy: 0.8374
Epoch 3/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0714 -
accuracy: 0.9076 - val_loss: 0.1271 - val_accuracy: 0.8273
Epoch 4/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0610 -
accuracy: 0.9222 - val_loss: 0.1311 - val_accuracy: 0.8278
Epoch 5/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0510 -
accuracy: 0.9362 - val_loss: 0.1363 - val_accuracy: 0.8188
Epoch 6/50
25000/25000 [==============================] - 210s 8ms/step - loss: 0.0412 -
accuracy: 0.9501 - val_loss: 0.1434 - val_accuracy: 0.8172
Epoch 7/50
25000/25000 [==============================] - 215s 9ms/step - loss: 0.0313 -
accuracy: 0.9625 - val_loss: 0.1455 - val_accuracy: 0.8200
Epoch 8/50
25000/25000 [==============================] - 213s 9ms/step - loss: 0.0261 -
accuracy: 0.9691 - val_loss: 0.1491 - val_accuracy: 0.8160
Epoch 9/50
25000/25000 [==============================] - 209s 8ms/step - loss: 0.0257 -
accuracy: 0.9699 - val_loss: 0.1535 - val_accuracy: 0.8110
Epoch 10/50
25000/25000 [==============================] - 209s 8ms/step - loss: 0.0216 -
accuracy: 0.9755 - val_loss: 0.1540 - val_accuracy: 0.8173
Epoch 11/50
25000/25000 [==============================] - 211s 8ms/step - loss: 0.0196 -
accuracy: 0.9780 - val_loss: 0.1590 - val_accuracy: 0.8130
Epoch 12/50
25000/25000 [==============================] - 210s 8ms/step - loss: 0.0193 -
accuracy: 0.9778 - val_loss: 0.1622 - val_accuracy: 0.8125
Epoch 13/50
25000/25000 [==============================] - 217s 9ms/step - loss: 0.0175 -
accuracy: 0.9800 - val_loss: 0.1632 - val_accuracy: 0.8128
Epoch 14/50
25000/25000 [==============================] - 212s 8ms/step - loss: 0.0158 -
accuracy: 0.9819 - val_loss: 0.1670 - val_accuracy: 0.8080
Epoch 15/50
25000/25000 [==============================] - 217s 9ms/step - loss: 0.0157 -
accuracy: 0.9819 - val_loss: 0.1701 - val_accuracy: 0.8068
Epoch 16/50
25000/25000 [==============================] - 212s 8ms/step - loss: 0.0147 -
accuracy: 0.9833 - val_loss: 0.1679 - val_accuracy: 0.8114
Epoch 17/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0153 -
accuracy: 0.9824 - val_loss: 0.1655 - val_accuracy: 0.8121
Epoch 18/50
25000/25000 [==============================] - 207s 8ms/step - loss: 0.0116 -
accuracy: 0.9867 - val_loss: 0.1614 - val_accuracy: 0.8136
Epoch 19/50
25000/25000 [==============================] - 218s 9ms/step - loss: 0.0106 -
accuracy: 0.9884 - val_loss: 0.1645 - val_accuracy: 0.8132
Epoch 20/50
25000/25000 [==============================] - 215s 9ms/step - loss: 0.0105 -
accuracy: 0.9884 - val_loss: 0.1632 - val_accuracy: 0.8146
Epoch 21/50
25000/25000 [==============================] - 208s 8ms/step - loss: 0.0084 -
accuracy: 0.9912 - val_loss: 0.1710 - val_accuracy: 0.8095
Epoch 22/50
25000/25000 [==============================] - 213s 9ms/step - loss: 0.0088 -
accuracy: 0.9904 - val_loss: 0.1673 - val_accuracy: 0.8114
Epoch 23/50
```

```
In [ ]:
```