Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Лабораторная работа №6
Дисциплина «Разработка интеллектуальных систем»

Выполнил:
Съестов Дмитрий Вячеславович
Группа P3417

Преподаватель:
Жукова Наталия Александровна

Санкт-Петербург
2020

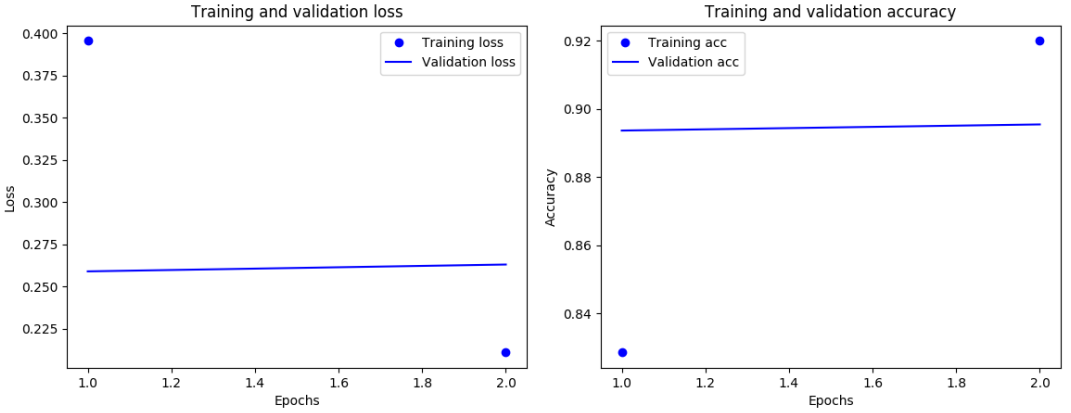
Листинг программы

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

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

(training_data, training_targets), (testing_data, testing_targets) =
imdb.load_data(num_words=10000)

data = np.concatenate((training_data, testing_data), axis=0)
data = vectorize(data)

targets = np.concatenate((training_targets, testing_targets), axis=0)
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(50, activation = "relu", input_shape=(10000, )))

# Hidden - Layers
model.add(layers.Dropout(0.3, noise_shape=None, seed=None))
model.add(layers.Dense(50, activation = "relu"))
model.add(layers.Dropout(0.2, noise_shape=None, seed=None))
model.add(layers.Dense(50, 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 = 500,
    validation_data = (test_x, test_y)
)
```

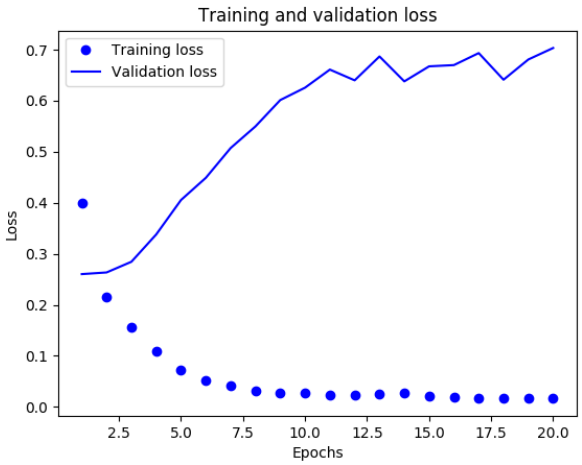
```
loss = H.history['loss']
val_loss = H.history['val_loss']
acc = H.history['accuracy']
val_acc = H.history['val_accuracy']

epochs = range(1, len(loss) + 1)

# Построение графика ошибки
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()

# Построение графика точности
plt.clf()
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

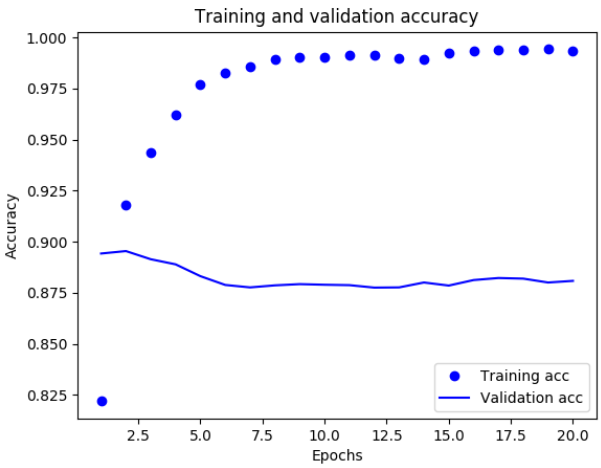
Результаты

2 эпохи, batch size = 500:

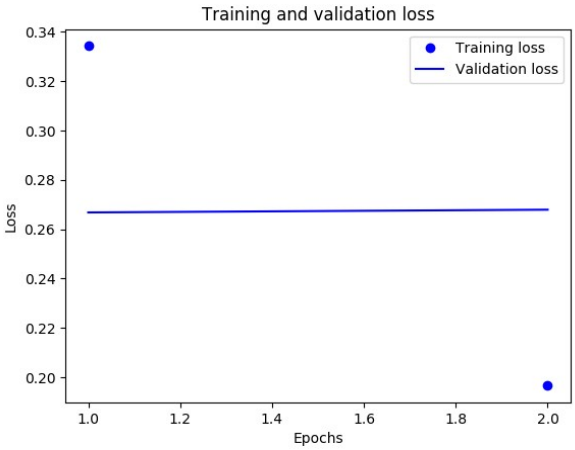


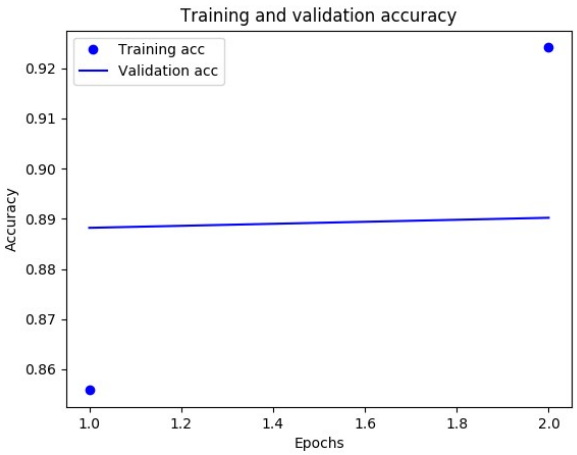
20 эпох, batch size = 500:





2 эпохи, batch size = 100:





Вывод

По графикам видно, что после 2 эпохи наступает переобучение, о чём говорит повышение потерь и снижение точности, так что увеличивать количество эпох бессмысленно. Уменьшение batch size слегка улучшило показатели при том же количестве эпох. Из этого можно сделать вывод, что на данном наборе данных целесообразно уменьшать параметр batch size.