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

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Искусственные нейронные сети»

Тема: Распознавание объектов на фотографиях

Студент гр. 7383 _____ Медведев И.С.

Преподаватель _____ Жукова Н. А.

Санкт-Петербург

2020

Цель работы.

Реализовать классификацию небольших изображений по десяти классам: самолет, автомобиль, птица, кошка, олень, собака, лягушка, лошадь, корабль и грузовик).

Выполнение работы.

Изначально была создана сверточная сеть использующая сверточные слои, слои maxpooling и слои разреживания dropout. Количество слоев и их параметры представлены в коде программы. Код программы представлен в приложении А.

На рис. 1-2 представлены результаты выполнения программы. Можно заметить, что с данной архитектурой сети достигается точность 76%.

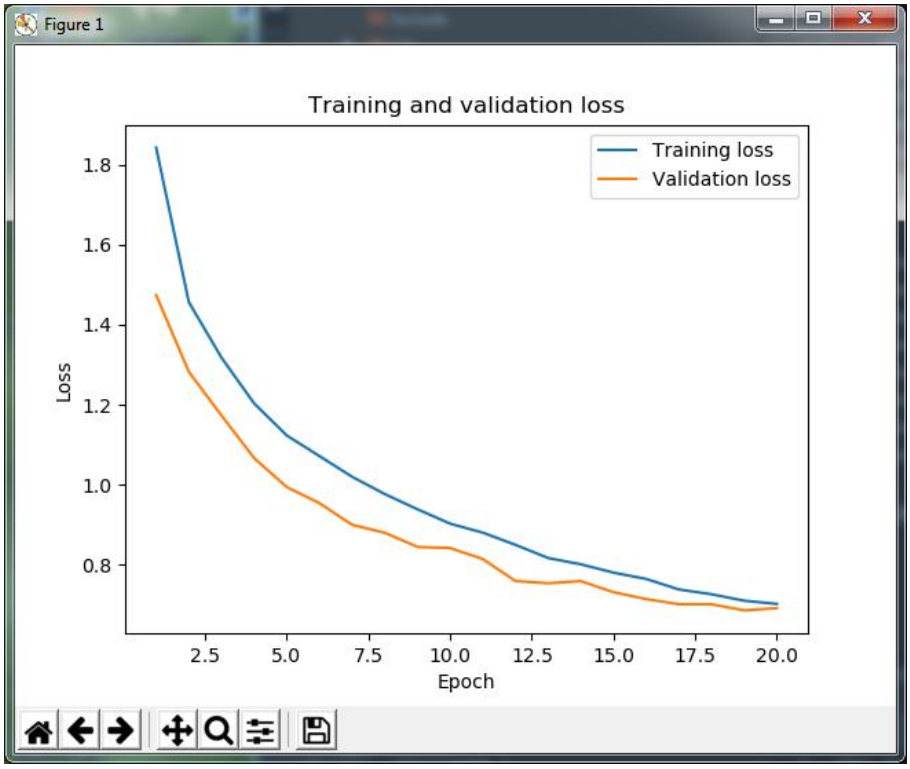

Рисунок 1 – График потерь для базовой модели

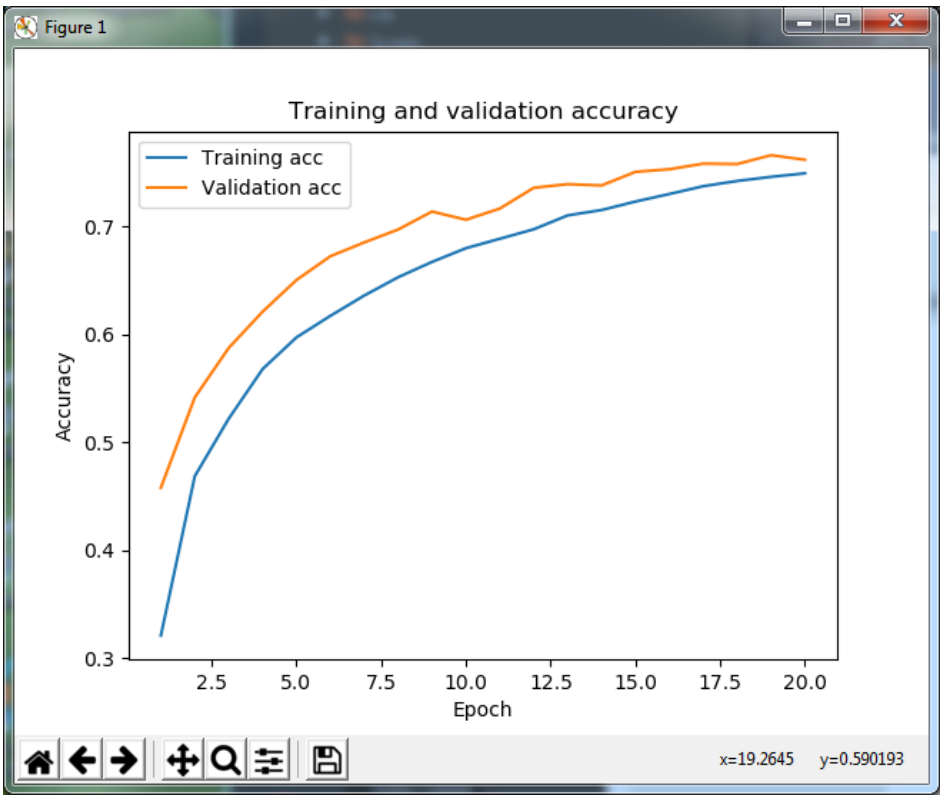

Рисунок 2 – График точности для базовой модели

Далее были убраны слои прореживания dropout, для того чтобы изучить как поведет себя модель без данных слоев. Результаты представлены на рис. 3-4.

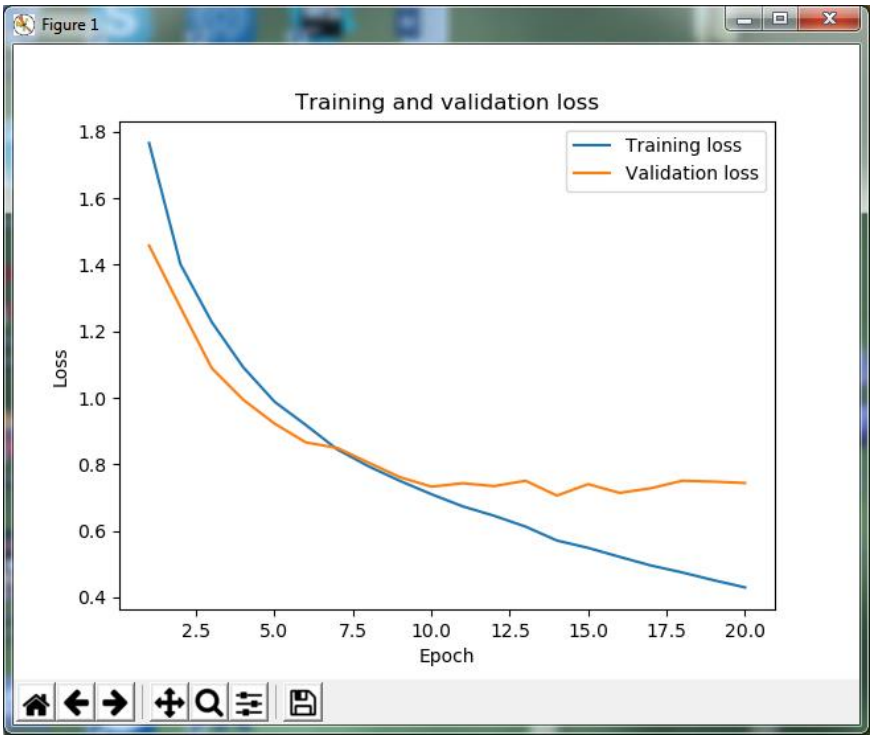

Рисунок 3 – График потерь для модели без dropout слоев

Рисунок 4 – График точности для модели без dropout слоев

Из графиков видно, что после 10 эпохи потери начали расти, а точность перестала повышаться, из чего можно сделать вывод о необходимости слоев прореживания.

Далее были изучены архитектуры, у которых в сверточных слоях размер ядра свертки имеет форму (5,5) и (7,7) соответственно. Результаты представлены на рис. 5-8.

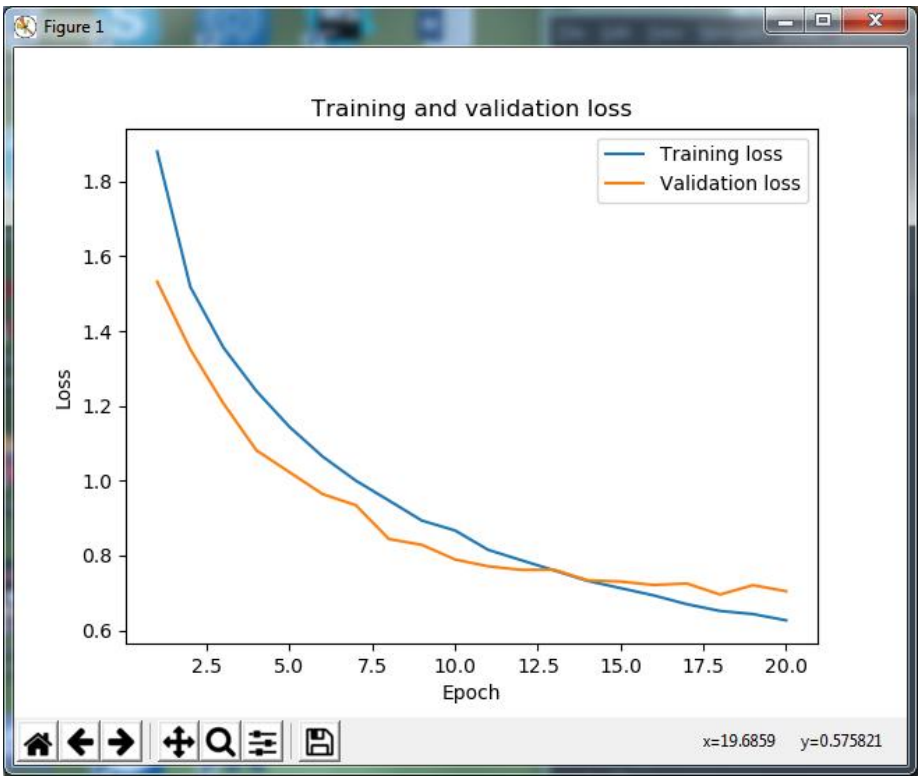

Рисунок 5 – График потерь для модели с размером ядра свертки (5,5)

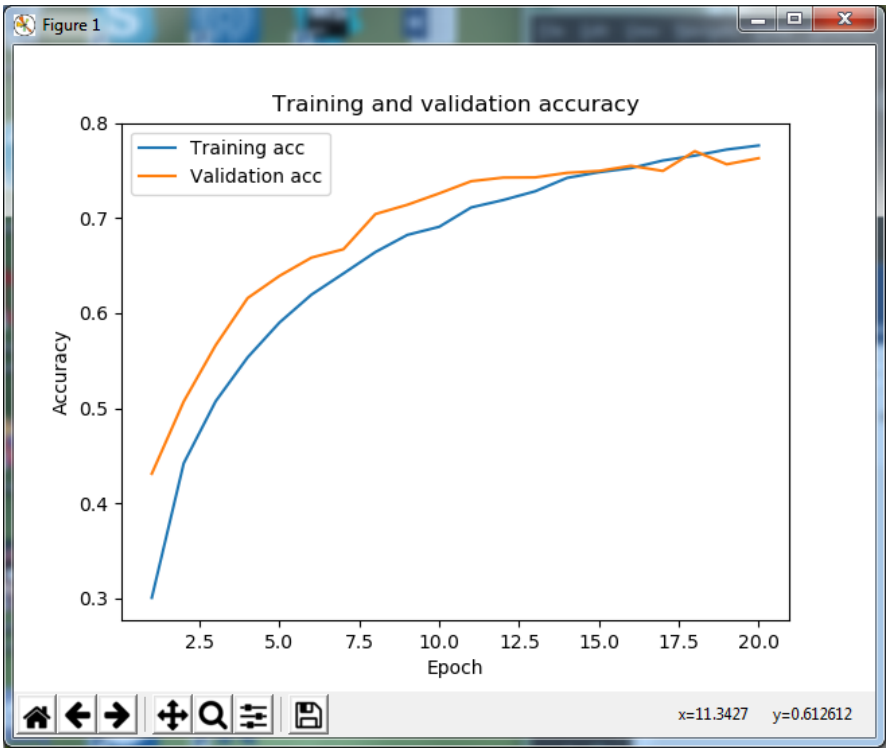

Рисунок 6 – График точности для модели с размером ядра свертки (5,5)

Рисунок 7 – График потерь для модели с размером ядра свертки (7,7)

Рисунок 8 – График точности для модели с размером ядра свертки (7,7)

По графикам на рис. 5-8 заметно, что при увеличении размера ядра свертки, падает точность (75% и 70%) и возрастает ошибка.

Выводы.

В ходе выполнения данной работы была создана сеть для классификации изображений, были более подробно изучены сверточные сети, влияние слоев разреживания и ядра свертки на результаты обучения.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А: КОД ПРОГРАММЫ

```
from tensorflow.keras.datasets import cifar10
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Convolution2D, MaxPooling2D,
Dense, Dropout, Flatten
from tensorflow.keras.utils import to_categorical
import matplotlib.pyplot as plt
import numpy as np

batch_size = 256
num_epochs = 20
kernel_size =7
pool_size = 2
conv_depth_1 = 32
conv_depth_2 = 64
drop_prob_1 = 0.25
drop_prob_2 = 0.5
hidden_size = 512

(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]
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= np.max(X_train)
X_test /= np.max(X_train)


Y_train = to_categorical(y_train, num_classes)
Y_test = to_categorical(y_test, num_classes)

def build_model(dropout = True):
    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 dropout:
        drop_1 = Dropout(drop_prob_1)(pool_1)
    else:
        drop_1 = pool_1
    conv_3  =  Convolution2D(conv_depth_2,  (kernel_size,  kernel_size),
padding='same', activation='relu')(drop_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 dropout:
        drop_2 = Dropout(drop_prob_1)(pool_2)
    else:
        drop_2 = pool_2
    flat = Flatten()(drop_2)
    hidden = Dense(hidden_size, activation='relu')(flat)
    drop_3 = Dropout(drop_prob_2)(hidden)
    out = Dense(num_classes, activation='softmax')(drop_3)
    model = Model(inp, out)
    model.compile(loss='categorical_crossentropy',      optimizer='adam',
metrics=['accuracy'])
    history   =   model.fit(X_train,   Y_train,     batch_size=batch_size,
epochs=num_epochs, verbose=1, validation_split=0.1)
    print(model.evaluate(X_test, Y_test, verbose=1))

    x = range(1, num_epochs + 1)
    plt.plot(x, history.history['loss'], label='Training loss')
    plt.plot(x, history.history['val_loss'], label='Validation loss')
    plt.title('Training and validation loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.legend()
    plt.show()
    plt.clf()

    plt.plot(x, history.history['acc'], label='Training acc')
    plt.plot(x, history.history['val_acc'], label='Validation acc')
    plt.title('Training and validation accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()


build_model()
```