Муковский Д.В. 8383

Задание

Необходимо построить сверточную нейронную сеть, которая будет классифицировать черно-белые изображения с простыми геометрическими фигурами на них.

К каждому варианту прилагается код, который генерирует изображения.

Для генерации данных необходимо вызвать функцию gen_data, которая возвращает два тензора:

1. Тензор с изображениями ранга 3
2. Тензор с метками классов

Обратите внимание:

- Выборки не перемешаны, то есть наблюдения классов идут по порядку
- Классы характеризуются строковой меткой
- Выборка изначально не разбита на обучающую, контрольную и тестовую
- Скачивать необходимо оба файла. Подключать файл, который начинается с var (в нем и находится функция gen_data)

Вариант 2

Классификация изображений с закрашенным или не закрашенным кругом.

Генерация и подготовка данных

```
def prepare_data(X, Y):
    X, Y = sklearn.utils.shuffle(X, Y)
    encoder = LabelEncoder()
    encoder.fit(Y)
    Y = encoder.transform(Y)

    val_split = 0.1
    test_num = round((1 - val_split) * len(X))
    val_num = round(test_num * (1 - val_split))
    train_data, train_labels = X[:val_num], Y[:val_num]
    val_data, val_labels = X[val_num:test_num], Y[val_num:test_num]
    test_data, test_labels = X[test_num:], Y[test_num:]
    train_data = np.expand_dims(train_data, axis=3)
    val_data = np.expand_dims(val_data, axis=3)
    test_data = np.expand_dims(test_data, axis=3)
    return (train_data, val_data, test_data), (train_labels, val_labels,
test_labels)
```

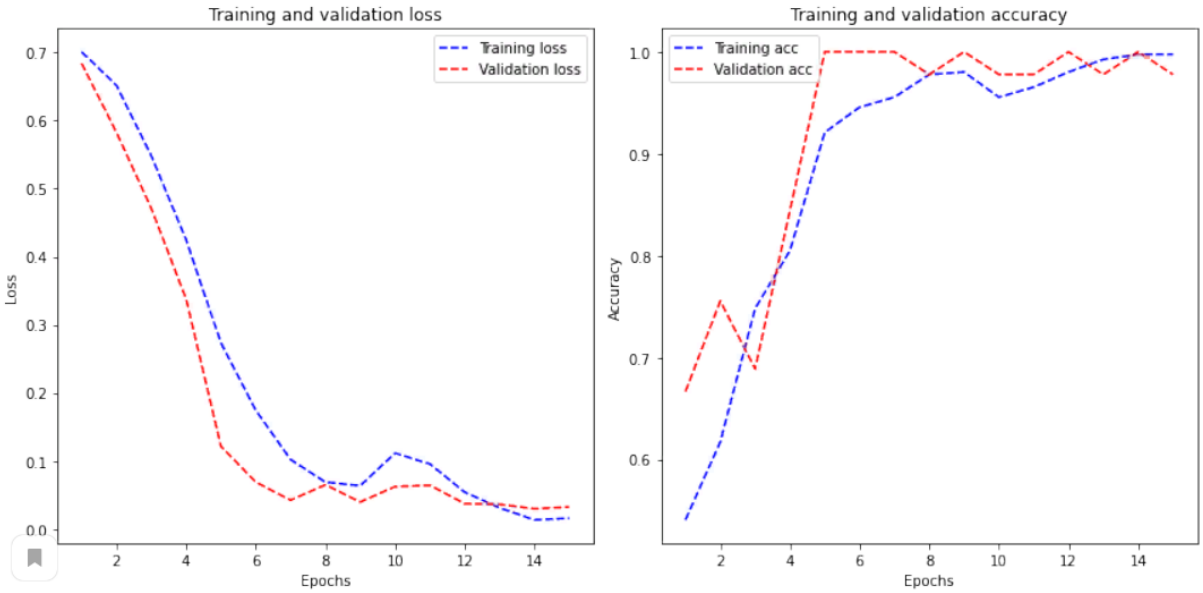
Параметры и архитектура сети:

```
batch_size = 15
num_epochs = 15
kernel_size = 3
pool_size = 2
conv_depth_1 = 16
conv_depth_2 = 32
drop_prob_1 = 0.25
drop_prob_2 = 0.5
hidden_size = 64

inp = Input(shape=(height, width, 1))

conv_1 = Convolution2D(conv_depth_1, (kernel_size, kernel_size),
                       padding='same', activation='relu')(inp)
pool_1 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_1)
drop_1 = Dropout(drop_prob_1)(pool_1)
conv_2 = Convolution2D(conv_depth_2, (kernel_size, kernel_size),
                       padding='same', activation='relu')(drop_1)
pool_2 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_2)
conv_3 = Convolution2D(conv_depth_2, (kernel_size, kernel_size),
                       padding='same', activation='relu')(pool_2)
drop_1 = Dropout(drop_prob_1)(conv_3)
flat = Flatten()(drop_1)
hidden = Dense(hidden_size, activation='relu')(flat)
drop_1 = Dropout(drop_prob_2)(hidden)
out = Dense(1, activation='sigmoid')(drop_1)
```

Результаты обучения представлены на графиках ниже



Ошибка и точность на тестовых данных:

```
loss: 0.0317 - accuracy: 1.0000
```