Аверина О.С. 8383

Практическое задание 6

Постановка задачи.

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:

- Тензор с изображениями ранга 3
- 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 1

Классификация изображений с прямоугольником или кругом.

Выполнение работы.

Была выполнена генерация датасета, нормализация данных и подготовлены метки.

```
img_size = 40
size = 1000

data, labels = gen_data(size, img_size)
print(data[0])

data_len, height, width = data.shape

data /= np.max(data)
encoder = LabelEncoder()
encoder.fit(labels)
labels = encoder.fit_transform(labels.ravel())

rand_index = np.random.permutation(len(labels))
data, labels = data[rand_index], labels[rand_index]
```

Далее были подготовлены тренировочные и валидационные данные.

```
val_size = 0.2

train_data, val_data = data[: int(data_len * (1 - val_size))],
data[int(data_len*(1 - val_size)):]
train_labels, val_labels = labels[: int(data_len * (1 - val_size))],
labels[int(data_len*(1 - val_size)):]

train_data = train_data.reshape(train_data.shape[0], width, height,
1)
val_data = val_data.reshape(val_data.shape[0], width, height, 1)
```

Была определена модель. Она состоит из четырех сверточных слоев(2 с глубиной 16 и два с глубиной 32), двух слоев субдискретизации, 3х слоев Dropout, слоя Flatten(слой, преобразующий 2D-данные в 1D-данные), двух полносвязных слоев и выходного слоя с функцией активации sigmoid.

```
model = Sequential()
```

```
model.add(Convolution2D(conv_depth_1, (kernel_size, kernel_size),
padding="same", activation='relu',
                        input_shape=(width, height, 1)))
model.add(Convolution2D(conv_depth_1, (kernel_size, kernel_size),
padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(pool_size, pool_size)))
model.add(Dropout(drop_prob_1))

model.add(Convolution2D(conv_depth_2, (kernel_size, kernel_size),
padding="same", activation='relu'))
model.add(Convolution2D(conv_depth_2, (kernel_size, kernel_size),
padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(pool_size, pool_size)))
model.add(Dropout(drop_prob_1))

model.add(Flatten())
model.add(Dense(hidden_size, activation='relu'))
model.add(Dropout(drop_prob_2))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

history = model.fit(train_data, train_labels,
              batch_size=batch_size, epochs=num_epochs,
              verbose=1, validation_split=0.2)
```

Было выполнено обучение модели. Результат представлен на рис. 1-2.

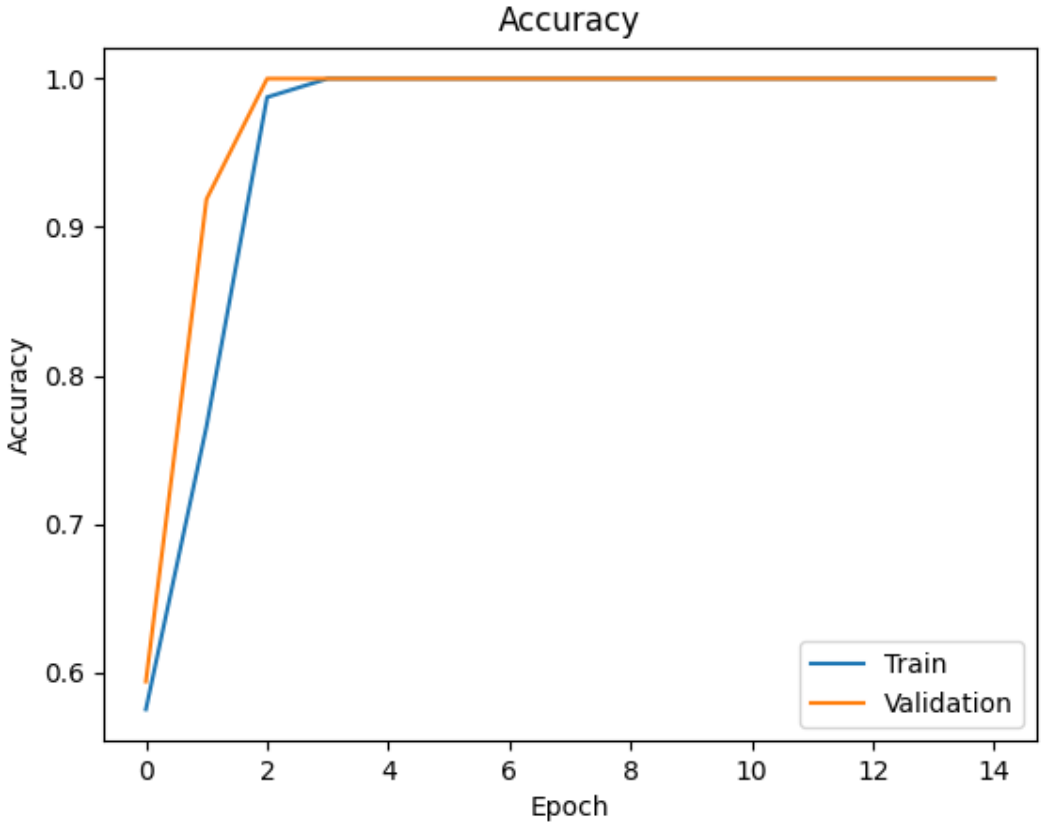

Рис. 1 - точность модели

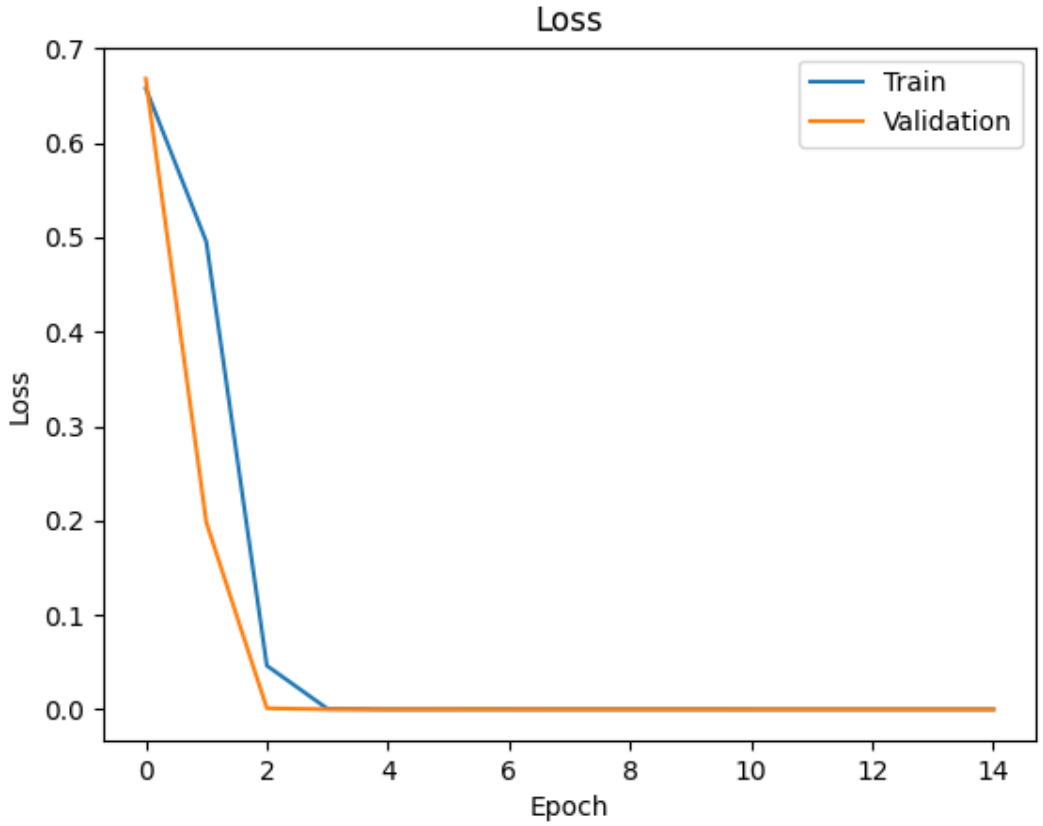

Рис. 2 - потери модели

На тестовых данных была достигнута 100% точность:

Epoch 15/15

20/20 [==============================] - 2s 101ms/step - loss: 2.0007e-05 - accuracy: 1.0000 - val_loss: 1.0187e-05 - val_accuracy: 1.0000

7/7 - 0s - loss: 3.8186e-04 - accuracy: 1.0000

Вывод.

Таким образом, в ходе выполнения практического задания была построена сверточная нейронная сеть, которая классифицирует черно-белые изображения с простыми геометрическими фигурами на них.