Вариант 7

Классификация изображений по количеству линий на них. Может быть 1, 2 или 3

Выполнение работы.

Данные для обучения были загружены, перемешаны и преобразованы для работы.

```
X, Y = file2.gen_data(size= 1000, img_size=50)
X, Y = shuffle(X, Y)
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
encoded_Y = np_utils.to_categorical(encoded_Y, 3)
```

Гиперпараметры реализованной сети.

```
num_train, height, width = X.shape
batch_size = 32
num_epochs = 10
kernel_size = 6
pool_size = 4
conv_depth_1 = 32
conv_depth_2 = 64
drop_prob_1 = 0.25
drop_prob_2 = 0.5
hidden_size = 512
```

Сама модель.

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

conv_1 = Convolution2D(conv_depth_1, (kernel_size, kernel_size),
padding='same', activation='relu', data_format='channels_last')(inp)
conv_2 = Convolution2D(conv_depth_1, (kernel_size, kernel_size),
padding='same', activation='relu', data_format='channels_last')(conv_1)
pool_1 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_2)
drop_1 = Dropout(drop_prob_1)(pool_1)

conv_3 = Convolution2D(conv_depth_2, (kernel_size, kernel_size),
padding='same', activation='relu', data_format='channels_last')(drop_1)
conv_4 = Convolution2D(conv_depth_2, (kernel_size, kernel_size),
padding='same', activation='relu', data_format='channels_last')(conv_3)
pool_2 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_4)
drop_2 = Dropout(drop_prob_1)(pool_2)

flat = Flatten()(drop_2)
hidden = Dense(hidden_size, activation='relu')(flat)
drop_3 = Dropout(drop_prob_2)(hidden)
out = Dense(3, activation='softmax')(drop_3)
```

Параметры обучения.

```
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

Обучение сети.

```
history = model.fit(X, encoded_Y,
          batch_size= batch_size, epochs= num_epochs,
          validation_split= 0.1)
```

Тестирование сети.

```
X_test, Y_test = file2.gen_data(size= 1000, img_size=50)
encoder = LabelEncoder()
encoder.fit(Y_test)
Y_test = encoder.transform(Y_test)
Y_test = np_utils.to_categorical(Y_test, 3)
model.evaluate(X_test, Y_test)
```

В ходе тестирования точность сети была около 68%. На обучающих и валидационных данных точности были 72% и 71% соответственно.