Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Нейротехнологии и аффективные вычисления

Распознавание эмоций людей по фото

Выполнил:

Маликов Глеб Игоревич

Группа № P3324

Преподаватель:

Русак Алена Викторовна

Санкт-Петербург

2025

**Содержание**

[Задание 3](#_Toc193898099)

[Реализация 4](#_Toc193898100)

[Код 10](#_Toc193898101)

[Вывод 12](#_Toc193898102)

# Задание

1. Провести исследование представленных моделей классификации эмоций по фото с целью повышения точности распознавания:
   1. Провести подбор гиперпараметров модели сверточной нейронной сети: batch size, количество эпох обучения, алгоритм оптимизации, количество слоев сети и количество нейронов в них, скорость обучения (learning rate). Можно менять все параметры или некоторые их них.
   2. Провести тонкую настройку модели VGG16: разморозить несколько слоев и дообучить модель на представленных данных (см. презентацию Transfer Learning). При этом также можно варьировать гиперпараметры модели: количество полносвязных слоев и количество нейронов в них, алгоритм оптимизации, скорость обучения, batch size, количество эпох обучения (обычно для тонкой настройки требуется меньше эпох, чем при обучении с нуля).
2. Составить отчет с результатами проведенных экспериментов.

# Решение

## Модель сверточной нейронной сети

A screen shot of a computer

AI-generated content may be incorrect.

### Первоначальные результаты:

Epoch 20/20

5/5 ━━━━━━━━━━━━━━━━━━━━ 4s 794ms/step - accuracy: 0.5726 - loss: 0.8334 - val\_accuracy: 0.4359 - val\_loss: 1.0823

A screenshot of a graph

AI-generated content may be incorrect.

Loss: 0.86

Accuracy: 0.64

### Изменённая модель

def se\_block(input\_tensor, ratio=4):  
 channels = input\_tensor.shape[-1]  
 # Squeeze: global spatial information into channel descriptor  
 se = GlobalAveragePooling2D()(input\_tensor)  
 # Excitation: two FC layers with a bottleneck  
 se = Dense(channels // ratio, activation='relu')(se)  
 se = Dense(channels, activation='sigmoid')(se)  
  
 se = Reshape((1, 1, channels))(se)  
 return Multiply()([input\_tensor, se])  
  
def res\_block(x, filters, stride=1, survival=0.9, ratio=2):  
 shortcut = x  
  
 # --- First conv ---  
 x = Conv2D(filters, 3, strides=stride, padding='same',  
 kernel\_initializer='he\_normal', use\_bias=False)(x)  
 x = BatchNormalization()(x)  
 x = Activation('relu')(x)  
  
 # --- Second conv ---  
 x = Conv2D(filters, 3, padding='same',  
 kernel\_initializer='he\_normal', use\_bias=False)(x)  
 x = BatchNormalization()(x)  
  
 # --- Squeeze–Excite ---  
 x = se\_block(x, ratio)  
  
 # ----- Regularisation -----  
 x = SpatialDropout2D(0.05)(x)  
 x = StochasticDepth(1 - survival)(x)  
  
 if shortcut.shape[-1] != filters or stride != 1:  
 shortcut = Conv2D(filters, 1, strides=stride,  
 kernel\_initializer='he\_normal',  
 use\_bias=False)(shortcut)  
 shortcut = BatchNormalization()(shortcut)  
  
 x = Add()([shortcut, x])  
 x = Activation('relu')(x)  
 return x  
  
  
def build\_emotion\_resnet(input\_shape=(224, 224, 3), num\_classes=3):  
 inputs = Input(shape=input\_shape)  
  
 aug = keras.Sequential([  
 RandomZoom(0.15, 0.15), # in/out zoom  
 RandomRotation(0.10), # ±10°  
 ], name="augment")  
  
 x = aug(inputs)  
  
 # Initial conv + pool  
 x = Conv2D(32, 3, padding='same',  
 kernel\_initializer='he\_normal', use\_bias=False)(x)  
 x = BatchNormalization()(x)  
 x = Activation('relu')(x)  
  
 # stage 1 – 2× residual, 32 filters  
 for \_ in range(2):  
 x = res\_block(x, 32)  
  
 # stage 2 – 64 filters  
 x = res\_block(x, 64, stride=2)  
 x = res\_block(x, 64)  
  
 # stage 3 – 128 filters  
 x = res\_block(x, 128, stride=2)  
  
 x = Dense(32, activation='relu')(x)  
 x = Flatten()(x)  
 x = Dropout(0.4)(x)  
 x = Dense(64, activation='relu')(x)  
 outputs = Dense(num\_classes, activation='softmax')(x)  
  
 return Model(inputs, outputs)

### Результаты

Epoch 20/20

5/5 ━━━━━━━━━━━━━━━━━━━━ 3s 388ms/step - accuracy: 0.6184 - loss: 0.8374 - val\_accuracy: 0.4359 - val\_loss: 1.0269 - learning\_rate: 5.9667e-04

A screenshot of a graph

AI-generated content may be incorrect.

Loss: 0.91

Accuracy: 0.68

## Модель с тонкой настройкой VGG16

A black screen with white text and green text

AI-generated content may be incorrect.

### Первоначальные результаты:

Epoch 20/20

5/5 ━━━━━━━━━━━━━━━━━━━━ 5s 1s/step - accuracy: 0.8240 - loss: 0.4430 - val\_accuracy: 0.5897 - val\_loss: 1.3431

A screenshot of a graph

AI-generated content may be incorrect.

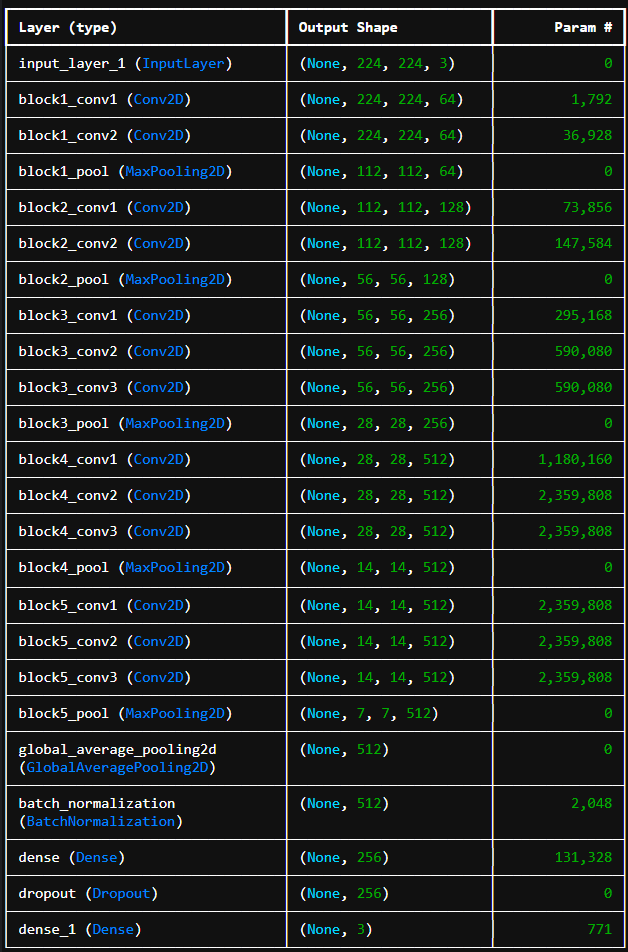
Loss: 1.33

Accuracy: 0.41

A screenshot of a computer screen

AI-generated content may be incorrect.

### Изменённая модель



### Результаты

Epoch 20/20

5/5 ━━━━━━━━━━━━━━━━━━━━ 3s 722ms/step - accuracy: 0.8069 - loss: 0.5114 - prec: 0.8626 - rec: 0.7577 - val\_accuracy: 0.5385 - val\_loss: 1.1443 - val\_prec: 0.6250 - val\_rec: 0.5128 - learning\_rate: 5.9667e-04

A screenshot of a graph

AI-generated content may be incorrect.

Loss: 1.17

Accuracy: 0.64

A screenshot of a computer screen

AI-generated content may be incorrect.

# Вывод

В ходе лабораторной работы были изменены модели для распознавания эмоций по изображению с помощью добавления новых слоев, изменения архитектуры и гиперпараметров моделей. Точность моделей после изменений оказалась значительно выше, но изменённые модели также оказались тяжелее из-за большего количества параметров.