Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Фундаментальных наук

КАФЕДРА Математического моделирования

ЛАБОРАТОРНАЯ РАБОТА

ПО КУРСУ:

«Нейронные сети и компьютерное зрение»

Студент *Шевляков Е.Д.*

Преподаватель *Фетисов Д.А.*

Москва, 2022 г.

1. Описание данных

Имеется набор изображений еды 4-х видов:

- 1724 изображения хлеба в подкаталоге Bread
- 1500 изображений десертов в подкаталоге Dessert
- 1757 изображений мяса в подкаталоге Meat
- 1500 изображений супов подкаталоге Soup

2. Подход к решению задачи

Для классификации изображений будем использовать последовательную модель Sequential. Используя ее, создадим сверточную нейросеть. Разрабатываемая сеть будет выполнять свои уровни последовательно, т.е. будет сетью прямого распространения. Наша нейронная сеть будет иметь несколько сверточныхслоев. В соответствии с концепцией CHC для укрепления масштаба полученных признаков применем метод объединения – max pooling. При применении max-pooling фильтр выбирает максимум из пикселей, покрытых фильтром.Фильтр действует как окно, из которого выбирается только максимальное значение для вывода. Для повышения производительности и стабилизации работы НС будем использовать Batch-normalization. После добавим два полносвязных слоя, чтобы добиться нужной размерности, соответствующей количеству классов.

Тогда сеть имеет вид:

$$
\begin{aligned}
&IN\ P\ U\ T \Rightarrow CON\ V\ 32 \Rightarrow RELU \Rightarrow BN \Rightarrow P\ OOL \Rightarrow \\
&\Rightarrow [CON\ V\ 64 \Rightarrow RELU \Rightarrow BN\ ]^{*}\ 2 \Rightarrow P\ OOL \Rightarrow \\
&\Rightarrow [CON\ V\ 128 \Rightarrow RELU \Rightarrow BN\ ]^{*}\ 3 \Rightarrow P\ OOL \Rightarrow \\
&\Rightarrow F\ C512 \Rightarrow RELU \Rightarrow F\ C3 \Rightarrow SOF\ T\ M\ AX
\end{aligned}
$$

Также для получения лучших результатов обобщения модели используем аугментацию данных – методика создания дополнительных данных из имеющихся.

3. Реализация

3.1 Подключение модулей

In [ ]:
```
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import tensorflow as tf
```

```
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Input, Conv2D, MaxPooling2D, Flatten,
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
```

```
2022-12-13 06:45:21.544023: I tensorflow/core/platform/cpu_feature_guar
d.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Ne
twork Library (oneDNN) to use the following CPU instructions in performa
nce-critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropri
ate compiler flags.
2022-12-13 06:45:21.659196: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcuda
rt.so.11.0'; dlerror: libcudart.so.11.0: cannot open shared object file:
No such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/ver
sions/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:21.659215: I tensorflow/compiler/xla/stream_executor/cu
da/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a G
PU set up on your machine.
2022-12-13 06:45:22.447795: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libnvin
fer.so.7'; dlerror: libnvinfer.so.7: cannot open shared object file: No
such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/versio
ns/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:22.447868: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libnvin
fer_plugin.so.7'; dlerror: libnvinfer_plugin.so.7: cannot open shared ob
ject file: No such file or directory; LD_LIBRARY_PATH: /home/eshevlyako
v/.pyenv/versions/3.10.7/envs/study/lib/python3.10/site-packages/cv
2/../../lib64:
2022-12-13 06:45:22.447877: W tensorflow/compiler/tf2tensorrt/utils/py_u
tils.cc:38] TF-TRT Warning: Cannot dlopen some TensorRT libraries. If yo
u would like to use Nvidia GPU with TensorRT, please make sure the missi
ng libraries mentioned above are installed properly.
```

3.2 Объявление нужных классов и функций

In [ ]:

```
class Preprocessor:
    """
    Предобработчик изображений
    """
    def __init__(self, width, height, inter=cv2.INTER_AREA):
        self.width = width
        self.height = height
        self.inter = inter

    def preprocess(self, image):
        return cv2.resize(image, (self.width, self.height), interpolation


class DatasetLoader:
    """
    Загрузчик изображений
    """
    def __init__(self, preprocessors=[]):
        self.preprocessors = preprocessors

    def load(self, image_Paths):
```

```
        data, labels = [], []

        for imagePath in image_Paths:
            image = cv2.imread(imagePath)
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            label = imagePath.split (os.path.sep)[-2]

            for p in self.preprocessors:
                image = p.preprocess(image)

            data.append(image)
            labels.append(label)

        return(np.array(data), np.array(labels))


class ConvNetComBNDO:
    """
    Конфигуратор нейросети
    """
    @staticmethod
    def build(width, height, depth, classes):
        model = Sequential()
        inputShape = (height, width, depth)
        model.add(Conv2D(filters = 32, kernel_size = (3, 3), activation =
        model.add(BatchNormalization())
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Conv2D(filters = 64, kernel_size = (3, 3), activation =
        model.add(BatchNormalization())
        model.add(Conv2D(filters = 64, kernel_size = (3, 3), activation =
        model.add(BatchNormalization())
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Conv2D(filters = 128, kernel_size = (3, 3), activation
        model.add(BatchNormalization())
        model.add(Conv2D(filters = 128, kernel_size = (3, 3), activation
        model.add(BatchNormalization())
        model.add(Conv2D(filters = 128, kernel_size = (3, 3), activation
        model.add(BatchNormalization())
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Flatten())
        model.add(Dense(512, activation = 'relu'))
        model.add(BatchNormalization())
        model.add(Dropout(0.5))
        model.add(Dense(classes, activation = 'softmax'))
        return model


def visual(X_train, Y_train):
    """
    Визуализация данных
    """
    index = np.random.choice(np.arange(len(X_train)), 24, replace=False)
    figure, axes = plt.subplots(nrows=4, ncols=6, figsize=(16,9))

    for item in zip(axes.ravel(), X_train[index], Y_train[index]):
        axes, image, target = item
        axes.imshow(image)
```

```
        axes.set_xticks([])
        axes.set_yticks([])
        axes.set_title(target)
    plt.show()


def incorrect(X_test,Y_test, pred):
    """
    Вывод неправильных прогнозов
    """
    food = ('Bread','Dessert','Meat', 'Soup')
    incorrect_predictions = []
    for i, (p, e) in enumerate(zip(pred,Y_test)):
        predicted, expected = np.argmax(p), np.argmax(e)
        if predicted != expected:
            incorrect_predictions.append((i, X_test[i], predicted, expect

    figure, axes = plt.subplots(nrows=4, ncols=6, figsize=(16,12))

    for item in zip(axes.ravel(), incorrect_predictions):
        axes, inc_pred = item
        axes.imshow(inc_pred[1])
        axes.set_xticks([])
        axes.set_yticks([])
        axes.set_title(f'p: {food[inc_pred[2]]}; e: {food[inc_pred[3]]}')

    plt.show()
    confusion = tf.math.confusion_matrix(Y_test.argmax(axis=1), pred.argm
    print(confusion)


def visual_incorrect(Epochs, Hist):
    """
    Отображение результатов
    """
    N = np.arange(0, Epochs)
    plt.style.use("ggplot")
    plt.figure()
    plt.plot(N, Hist.history["loss"], label="train_loss")
    plt.plot(N, Hist.history["val_loss"], label="val_loss")
    plt.plot(N, Hist.history["accuracy"], label="train_acc")
    plt.plot(N, Hist.history["val_accuracy"], label="val_acc")
    plt.title("Training Loss and Accuracy")
    plt.xlabel("Epoch #")
    plt.ylabel("Loss/Accuracy")
    plt.legend()
    plt.show()
```

3.3 Подготовка данных

In [ ]:
```
imagePaths = list(paths.list_images('data/food'))
random.seed(42)
random.shuffle(imagePaths)
```

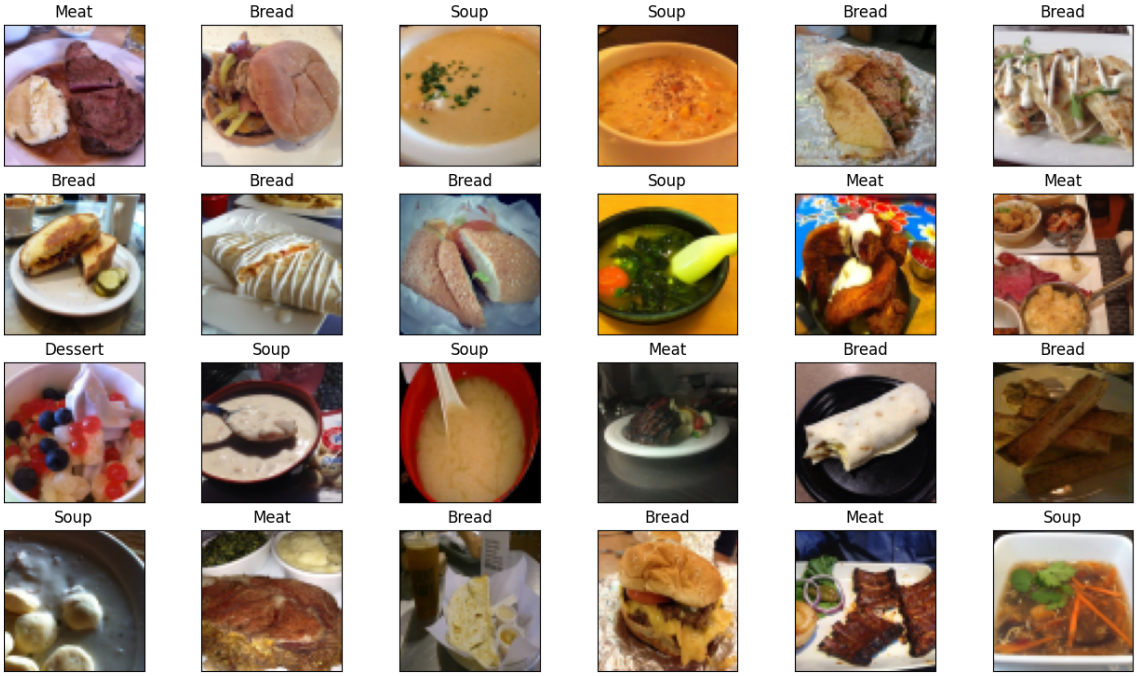
Preprocessor и DatasetLoader

In [ ]:
```
input_width = 64
sp = Preprocessor(input_width, input_width)
```

```
dsl = DatasetLoader(preprocessors=[sp])

(data, labels) = dsl.load(imagePaths)
data = data.astype('float32') / 255
```

Разделение выборки и обработка меток

In [ ]:
```
(trainX, testX, trainY, testY) = train_test_split(data, labels, test_size
visual(trainX, trainY)

lb = LabelBinarizer()
trainY = lb.fit_transform(trainY)
testY = lb.transform(testY)

aug = ImageDataGenerator(rotation_range=30, width_shift_range=0.1, height
                         shear_range=0.2, zoom_range=0.2, horizontal_flip
```



3.4 Создание модели

In [ ]:
```
model = ConvNetComBNDO.build(input_width, input_width, 3, 4)
model.summary()
model.compile(loss="categorical_crossentropy", optimizer='adam', metrics=
```

```
2022-12-13 06:45:48.179903: I tensorflow/compiler/xla/stream_executor/cu
da/cuda_gpu_executor.cc:981] successful NUMA node read from SysFS had ne
gative value (-1), but there must be at least one NUMA node, so returnin
g NUMA node zero
2022-12-13 06:45:48.180144: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcuda
rt.so.11.0'; dlerror: libcudart.so.11.0: cannot open shared object file:
No such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/ver
sions/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180233: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcubl
as.so.11'; dlerror: libcublas.so.11: cannot open shared object file: No
such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/versio
ns/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
```

```
2022-12-13 06:45:48.180299: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcubl
asLt.so.11'; dlerror: libcublasLt.so.11: cannot open shared object file:
No such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/ver
sions/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180359: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcuff
t.so.10'; dlerror: libcufft.so.10: cannot open shared object file: No su
ch file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/version
s/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180417: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcura
nd.so.10'; dlerror: libcurand.so.10: cannot open shared object file: No
such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/versio
ns/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180491: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcuso
lver.so.11'; dlerror: libcusolver.so.11: cannot open shared object file:
No such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/ver
sions/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180547: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcusp
arse.so.11'; dlerror: libcusparse.so.11: cannot open shared object file:
No such file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/ver
sions/3.10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180605: W tensorflow/compiler/xla/stream_executor/pl
atform/default/dso_loader.cc:64] Could not load dynamic library 'libcudn
n.so.8'; dlerror: libcudnn.so.8: cannot open shared object file: No such
file or directory; LD_LIBRARY_PATH: /home/eshevlyakov/.pyenv/versions/3.
10.7/envs/study/lib/python3.10/site-packages/cv2/../../lib64:
2022-12-13 06:45:48.180616: W tensorflow/core/common_runtime/gpu/gpu_dev
ice.cc:1934] Cannot dlopen some GPU libraries. Please make sure the miss
ing libraries mentioned above are installed properly if you would like t
o use GPU. Follow the guide at https://www.tensorflow.org/install/gpu fo
r how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2022-12-13 06:45:48.181212: I tensorflow/core/platform/cpu_feature_guar
d.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Ne
twork Library (oneDNN) to use the following CPU instructions in performa
nce-critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropri
ate compiler flags.
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 32)        896

 batch_normalization (BatchN  (None, 64, 64, 32)       128
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 32, 32, 32)       0
 )

 dropout (Dropout)           (None, 32, 32, 32)        0

 conv2d_1 (Conv2D)           (None, 32, 32, 64)        18496

 batch_normalization_1 (Batc  (None, 32, 32, 64)       256
 hNormalization)
```

```
 conv2d_2 (Conv2D)           (None, 32, 32, 64)        36928

 batch_normalization_2 (Batc  (None, 32, 32, 64)       256
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 16, 16, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 16, 16, 64)        0

 conv2d_3 (Conv2D)           (None, 16, 16, 128)       73856

 batch_normalization_3 (Batc  (None, 16, 16, 128)      512
 hNormalization)

 conv2d_4 (Conv2D)           (None, 16, 16, 128)       147584

 batch_normalization_4 (Batc  (None, 16, 16, 128)      512
 hNormalization)

 conv2d_5 (Conv2D)           (None, 16, 16, 128)       147584

 batch_normalization_5 (Batc  (None, 16, 16, 128)      512
 hNormalization)

 max_pooling2d_2 (MaxPooling  (None, 8, 8, 128)        0
 2D)

 dropout_2 (Dropout)         (None, 8, 8, 128)         0

 flatten (Flatten)           (None, 8192)              0

 dense (Dense)               (None, 512)               4194816

 batch_normalization_6 (Batc  (None, 512)              2048
 hNormalization)

 dropout_3 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 4)                 2052

=================================================================
Total params: 4,626,436
Trainable params: 4,624,324
Non-trainable params: 2,112
_________________________________________________________________
```

In [ ]:

```
EPOCHS = 80
H = model.fit(aug.flow(trainX, trainY), epochs=EPOCHS, validation_data=(t
```

```
Epoch 1/80
152/152 [==============================] - ETA: 0s - loss: 1.6839 - accu
racy: 0.4130
2022-12-13 06:46:44.751651: W tensorflow/tsl/framework/cpu_allocator_imp
l.cc:82] Allocation of 79675392 exceeds 10% of free system memory.
152/152 [==============================] - 59s 373ms/step - loss: 1.6839
- accuracy: 0.4130 - val_loss: 2.7502 - val_accuracy: 0.2215
Epoch 2/80
152/152 [==============================] - 52s 344ms/step - loss: 1.4573
```

```
- accuracy: 0.4568 - val_loss: 1.9231 - val_accuracy: 0.3159
Epoch 3/80
152/152 [==============================] - 52s 343ms/step - loss: 1.2392
- accuracy: 0.5243 - val_loss: 1.2855 - val_accuracy: 0.4985
Epoch 4/80
152/152 [==============================] - 54s 352ms/step - loss: 1.1373
- accuracy: 0.5537 - val_loss: 1.1153 - val_accuracy: 0.5472
Epoch 5/80
152/152 [==============================] - 55s 363ms/step - loss: 1.0740
- accuracy: 0.5728 - val_loss: 1.3545 - val_accuracy: 0.5089
Epoch 6/80
152/152 [==============================] - 56s 367ms/step - loss: 0.9963
- accuracy: 0.6095 - val_loss: 1.0057 - val_accuracy: 0.5941
Epoch 7/80
152/152 [==============================] - 52s 344ms/step - loss: 0.9570
- accuracy: 0.6169 - val_loss: 1.3184 - val_accuracy: 0.5318
Epoch 8/80
152/152 [==============================] - 48s 315ms/step - loss: 0.9191
- accuracy: 0.6309 - val_loss: 1.7665 - val_accuracy: 0.4109
Epoch 9/80
152/152 [==============================] - 48s 315ms/step - loss: 0.8795
- accuracy: 0.6523 - val_loss: 1.2343 - val_accuracy: 0.5250
Epoch 10/80
152/152 [==============================] - 48s 314ms/step - loss: 0.8462
- accuracy: 0.6667 - val_loss: 0.9913 - val_accuracy: 0.6163
Epoch 11/80
152/152 [==============================] - 48s 314ms/step - loss: 0.8603
- accuracy: 0.6588 - val_loss: 1.2302 - val_accuracy: 0.5657
Epoch 12/80
152/152 [==============================] - 48s 315ms/step - loss: 0.8172
- accuracy: 0.6759 - val_loss: 1.2418 - val_accuracy: 0.5503
Epoch 13/80
152/152 [==============================] - 48s 315ms/step - loss: 0.7897
- accuracy: 0.6887 - val_loss: 0.8874 - val_accuracy: 0.6434
Epoch 14/80
152/152 [==============================] - 48s 314ms/step - loss: 0.7847
- accuracy: 0.6916 - val_loss: 1.4349 - val_accuracy: 0.5238
Epoch 15/80
152/152 [==============================] - 49s 319ms/step - loss: 0.7820
- accuracy: 0.6961 - val_loss: 0.8761 - val_accuracy: 0.6453
Epoch 16/80
152/152 [==============================] - 48s 316ms/step - loss: 0.7747
- accuracy: 0.7000 - val_loss: 0.8794 - val_accuracy: 0.6619
Epoch 17/80
152/152 [==============================] - 48s 315ms/step - loss: 0.7324
- accuracy: 0.7121 - val_loss: 1.0095 - val_accuracy: 0.6126
Epoch 18/80
152/152 [==============================] - 48s 317ms/step - loss: 0.7573
- accuracy: 0.7019 - val_loss: 1.0312 - val_accuracy: 0.6101
Epoch 19/80
152/152 [==============================] - 48s 315ms/step - loss: 0.7409
- accuracy: 0.7148 - val_loss: 1.0507 - val_accuracy: 0.5972
Epoch 20/80
152/152 [==============================] - 49s 320ms/step - loss: 0.7429
- accuracy: 0.7041 - val_loss: 0.9940 - val_accuracy: 0.6292
Epoch 21/80
152/152 [==============================] - 49s 321ms/step - loss: 0.7001
- accuracy: 0.7340 - val_loss: 0.8078 - val_accuracy: 0.7039
Epoch 22/80
152/152 [==============================] - 48s 315ms/step - loss: 0.6967
```

```
- accuracy: 0.7323 - val_loss: 0.8590 - val_accuracy: 0.6922
Epoch 23/80
152/152 [==============================] - 48s 313ms/step - loss: 0.7219
- accuracy: 0.7140 - val_loss: 1.2406 - val_accuracy: 0.5429
Epoch 24/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6630
- accuracy: 0.7453 - val_loss: 0.7614 - val_accuracy: 0.7119
Epoch 25/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6523
- accuracy: 0.7500 - val_loss: 0.9117 - val_accuracy: 0.6576
Epoch 26/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6587
- accuracy: 0.7444 - val_loss: 0.9137 - val_accuracy: 0.6527
Epoch 27/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6605
- accuracy: 0.7481 - val_loss: 1.2923 - val_accuracy: 0.5484
Epoch 28/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6489
- accuracy: 0.7463 - val_loss: 0.7221 - val_accuracy: 0.7267
Epoch 29/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6434
- accuracy: 0.7535 - val_loss: 0.8765 - val_accuracy: 0.6582
Epoch 30/80
152/152 [==============================] - 48s 315ms/step - loss: 0.6551
- accuracy: 0.7451 - val_loss: 1.1158 - val_accuracy: 0.5731
Epoch 31/80
152/152 [==============================] - 48s 314ms/step - loss: 0.6301
- accuracy: 0.7601 - val_loss: 0.8712 - val_accuracy: 0.6835
Epoch 32/80
152/152 [==============================] - 48s 315ms/step - loss: 0.6318
- accuracy: 0.7588 - val_loss: 0.7519 - val_accuracy: 0.7070
Epoch 33/80
152/152 [==============================] - 48s 315ms/step - loss: 0.6080
- accuracy: 0.7660 - val_loss: 0.8278 - val_accuracy: 0.6774
Epoch 34/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5917
- accuracy: 0.7708 - val_loss: 0.8723 - val_accuracy: 0.6669
Epoch 35/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5954
- accuracy: 0.7739 - val_loss: 0.8945 - val_accuracy: 0.6922
Epoch 36/80
152/152 [==============================] - 48s 315ms/step - loss: 0.5917
- accuracy: 0.7753 - val_loss: 0.7332 - val_accuracy: 0.7353
Epoch 37/80
152/152 [==============================] - 48s 316ms/step - loss: 0.5722
- accuracy: 0.7809 - val_loss: 0.8092 - val_accuracy: 0.7094
Epoch 38/80
152/152 [==============================] - 46s 303ms/step - loss: 0.5843
- accuracy: 0.7747 - val_loss: 1.1748 - val_accuracy: 0.6058
Epoch 39/80
152/152 [==============================] - 48s 318ms/step - loss: 0.5795
- accuracy: 0.7774 - val_loss: 0.8997 - val_accuracy: 0.6761
Epoch 40/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5544
- accuracy: 0.7887 - val_loss: 0.7032 - val_accuracy: 0.7434
Epoch 41/80
152/152 [==============================] - 49s 320ms/step - loss: 0.5585
- accuracy: 0.7846 - val_loss: 0.8419 - val_accuracy: 0.6700
Epoch 42/80
152/152 [==============================] - 48s 313ms/step - loss: 0.5666
```

```
- accuracy: 0.7786 - val_loss: 0.8264 - val_accuracy: 0.6952
Epoch 43/80
152/152 [==============================] - 48s 318ms/step - loss: 0.5465
- accuracy: 0.7975 - val_loss: 0.7926 - val_accuracy: 0.7119
Epoch 44/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5253
- accuracy: 0.7969 - val_loss: 0.7530 - val_accuracy: 0.7477
Epoch 45/80
152/152 [==============================] - 48s 315ms/step - loss: 0.5398
- accuracy: 0.7932 - val_loss: 0.8250 - val_accuracy: 0.7101
Epoch 46/80
152/152 [==============================] - 48s 313ms/step - loss: 0.5132
- accuracy: 0.8016 - val_loss: 0.7517 - val_accuracy: 0.7292
Epoch 47/80
152/152 [==============================] - 48s 315ms/step - loss: 0.5323
- accuracy: 0.7938 - val_loss: 0.6314 - val_accuracy: 0.7693
Epoch 48/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5204
- accuracy: 0.8012 - val_loss: 0.9693 - val_accuracy: 0.6422
Epoch 49/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5108
- accuracy: 0.8062 - val_loss: 0.7549 - val_accuracy: 0.7495
Epoch 50/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4743
- accuracy: 0.8204 - val_loss: 0.7089 - val_accuracy: 0.7471
Epoch 51/80
152/152 [==============================] - 48s 314ms/step - loss: 0.5378
- accuracy: 0.7959 - val_loss: 1.2631 - val_accuracy: 0.5750
Epoch 52/80
152/152 [==============================] - 48s 315ms/step - loss: 0.5267
- accuracy: 0.7981 - val_loss: 0.7480 - val_accuracy: 0.7458
Epoch 53/80
152/152 [==============================] - 48s 315ms/step - loss: 0.4558
- accuracy: 0.8251 - val_loss: 0.8981 - val_accuracy: 0.6891
Epoch 54/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4753
- accuracy: 0.8158 - val_loss: 0.9073 - val_accuracy: 0.7107
Epoch 55/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4383
- accuracy: 0.8340 - val_loss: 0.8624 - val_accuracy: 0.6860
Epoch 56/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4369
- accuracy: 0.8385 - val_loss: 0.8665 - val_accuracy: 0.7113
Epoch 57/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4530
- accuracy: 0.8344 - val_loss: 1.1301 - val_accuracy: 0.6163
Epoch 58/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4270
- accuracy: 0.8391 - val_loss: 0.7707 - val_accuracy: 0.7347
Epoch 59/80
152/152 [==============================] - 48s 315ms/step - loss: 0.4217
- accuracy: 0.8383 - val_loss: 0.7598 - val_accuracy: 0.7619
Epoch 60/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4480
- accuracy: 0.8340 - val_loss: 0.7042 - val_accuracy: 0.7446
Epoch 61/80
152/152 [==============================] - 48s 315ms/step - loss: 0.4372
- accuracy: 0.8315 - val_loss: 0.7196 - val_accuracy: 0.7230
Epoch 62/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4587
```

```
- accuracy: 0.8257 - val_loss: 0.7075 - val_accuracy: 0.7514
Epoch 63/80
152/152 [==============================] - 48s 315ms/step - loss: 0.4342
- accuracy: 0.8342 - val_loss: 0.8320 - val_accuracy: 0.7199
Epoch 64/80
152/152 [==============================] - 48s 315ms/step - loss: 0.4106
- accuracy: 0.8469 - val_loss: 0.6794 - val_accuracy: 0.7600
Epoch 65/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3856
- accuracy: 0.8514 - val_loss: 1.0216 - val_accuracy: 0.6767
Epoch 66/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4126
- accuracy: 0.8403 - val_loss: 0.7752 - val_accuracy: 0.7502
Epoch 67/80
152/152 [==============================] - 49s 320ms/step - loss: 0.3847
- accuracy: 0.8531 - val_loss: 0.8635 - val_accuracy: 0.7230
Epoch 68/80
152/152 [==============================] - 48s 313ms/step - loss: 0.3701
- accuracy: 0.8603 - val_loss: 0.7116 - val_accuracy: 0.7508
Epoch 69/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3614
- accuracy: 0.8634 - val_loss: 0.6831 - val_accuracy: 0.7582
Epoch 70/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3953
- accuracy: 0.8504 - val_loss: 0.6563 - val_accuracy: 0.7711
Epoch 71/80
152/152 [==============================] - 48s 315ms/step - loss: 0.3813
- accuracy: 0.8533 - val_loss: 0.9407 - val_accuracy: 0.7076
Epoch 72/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4001
- accuracy: 0.8426 - val_loss: 0.8467 - val_accuracy: 0.7421
Epoch 73/80
152/152 [==============================] - 48s 314ms/step - loss: 0.4185
- accuracy: 0.8424 - val_loss: 0.7620 - val_accuracy: 0.7415
Epoch 74/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3507
- accuracy: 0.8642 - val_loss: 0.9361 - val_accuracy: 0.7101
Epoch 75/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3772
- accuracy: 0.8595 - val_loss: 0.7156 - val_accuracy: 0.7767
Epoch 76/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3634
- accuracy: 0.8597 - val_loss: 0.8050 - val_accuracy: 0.7298
Epoch 77/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3678
- accuracy: 0.8611 - val_loss: 0.8863 - val_accuracy: 0.7520
Epoch 78/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3478
- accuracy: 0.8698 - val_loss: 0.8136 - val_accuracy: 0.7329
Epoch 79/80
152/152 [==============================] - 48s 314ms/step - loss: 0.3543
- accuracy: 0.8714 - val_loss: 0.9728 - val_accuracy: 0.7033
Epoch 80/80
152/152 [==============================] - 48s 315ms/step - loss: 0.3328
- accuracy: 0.8739 - val_loss: 0.9129 - val_accuracy: 0.7255
```

3.5 Анализ результатов

In [ ]:
```
predictions = model.predict(testX)
```

```
for index, probability in enumerate(predictions[0]):
    print(f'{index}: {probability:.10%}')

incorrect(testX, testY, predictions)

visual_incorrect(EPOCHS, H)
```

```
2022-12-13 07:50:22.746907: W tensorflow/tsl/framework/cpu_allocator_impl.cc:82] Allocation of 79675392 exceeds 10% of free system memory.
51/51 [==============================] - 2s 45ms/step
0: 0.0000043641%
1: 0.0029009783%
2: 0.0000058882%
3: 99.9970853329%
```


```
tf.Tensor(
[[258 123  29  19]
 [ 23 345  18  29]
 [ 21 118 268  13]
 [  9  40   3 305]], shape=(4, 4), dtype=int32)
```


Training Loss and Accuracy
train_loss
val_loss
train_acc
val_acc
Loss/Accuracy
2.5
2.0
1.5
1.0
0.5
0
10
20
30
40
50
60
70
80
Epoch #