Lab 3: Gesture Recognition using Convolutional Neural Networks

Deadlines: Feb 8, 5:00PM

Late Penalty: There is a penalty-free grace period of one hour past the deadline. Any work that is submitted between 1 hour and 24 hours past the deadline will receive a 20% grade deduction. No other late work is accepted. Quercus submission time will be used, not your local computer time. You can submit your labs as many times as you want before the deadline, so please submit often and early.

Grading TAs: Geoff Donoghue

This lab is based on an assignment developed by Prof. Lisa Zhang.

This lab will be completed in two parts. In Part A you will you will gain experience gathering your own data set (specifically images of hand gestures), and understand the challenges involved in the data cleaning process. In Part B you will train a convolutional neural network to make classifications on different hand gestures. By the end of the lab, you should be able to:

1. Generate and preprocess your own data
2. Load and split data for training, validation and testing
3. Train a Convolutional Neural Network
4. Apply transfer learning to improve your model

Note that for this lab we will not be providing you with any starter code. You should be able to take the code used in previous labs, tutorials and lectures and modify it accordingly to complete the tasks outlined below.

What to submit

Submission for Part A:
Submit a zip file containing your images. Three images each of American Sign Language gestures for letters A - I (total of 27 images). You will be required to clean the images before submitting them. Details are provided under Part A of the handout.

Individual image file names should follow the convention of student-number_Alphabet_file-number.jpg (e.g. 100343434_A_1.jpg).

Submission for Part B:
Submit a PDF file containing all your code, outputs, and write-up from parts 1-5. You can produce a PDF of your Google Colab file by going to **File > Print** and then save as PDF. The Colab instructions has more information. Make sure to review the PDF submission to ensure that your answers are easy to read. Make sure that your text is not cut off at the margins.

Do not submit any other files produced by your code.

Include a link to your colab file in your submission.

Please use Google Colab to complete this assignment. If you want to use Jupyter Notebook, please complete the assignment and upload your Jupyter Notebook file to Google Colab for submission.

Colab Link

Include a link to your colab file here

Colab Link: https://drive.google.com/file/d/1Kw51hAEhvpVSI1QLAyMI6S6tqQrEneXh/view?usp=sharing (https://drive.google.com/file/d/1Kw51hAEhvpVSI1QLAyMI6S6tqQrEneXh/view?usp=sharing)

In [9]:
```
%%shell
jupyter nbconvert --to html /content/Lab_3_Gesture_Recognition_lab.ipynb
```

```
[NbConvertApp] Converting notebook /content/Lab_3_Gesture_Recognition.ipynb t
o html
[NbConvertApp] Writing 301392 bytes to /content/Lab_3_Gesture_Recognition.htm
l
```

Out[9]:

In [7]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Part A. Data Collection [10 pt]

So far, we have worked with data sets that have been collected, cleaned, and curated by machine learning researchers and practitioners. Datasets like MNIST and CIFAR are often used as toy examples, both by students and by researchers testing new machine learning models.

In the real world, getting a clean data set is never that easy. More than half the work in applying machine learning is finding, gathering, cleaning, and formatting your data set.

The purpose of this lab is to help you gain experience gathering your own data set, and understand the challenges involved in the data cleaning process.

American Sign Language

American Sign Language (ASL) is a complete, complex language that employs signs made by moving the hands combined with facial expressions and postures of the body. It is the primary language of many North Americans who are deaf and is one of several communication options used by people who are deaf or hard-of-hearing.

The hand gestures representing English alphabet are shown below. This lab focuses on classifying a subset of these hand gesture images using convolutional neural networks. Specifically, given an image of a hand showing one of the letters A-I, we want to detect which letter is being represented.

Generating Data

We will produce the images required for this lab by ourselves. Each student will collect, clean and submit three images each of Americal Sign Language gestures for letters A - I (total of 27 images) Steps involved in data collection

1. Familiarize yourself with American Sign Language gestures for letters from A - I (9 letters).
2. Take three pictures at slightly different orientation for each letter gesture using your mobile phone.
    - Ensure adequate lighting while you are capturing the images.
    - Use a white wall as your background.
    - Use your right hand to create gestures (for consistency).
    - Keep your right hand fairly apart from your body and any other obstructions.
    - Avoid having shadows on parts of your hand.
3. Transfer the images to your laptop for cleaning.

Cleaning Data

To simplify the machine learning the task, we will standardize the training images. We will make sure that all our images are of the same size (224 x 224 pixels RGB), and have the hand in the center of the cropped regions.

You may use the following applications to crop and resize your images:

Mac

- Use Preview: – Holding down CMD + Shift will keep a square aspect ratio while selecting the hand area. – Resize to 224x224 pixels.

Windows 10

- Use Photos app to edit and crop the image and keep the aspect ratio a square.
- Use Paint to resize the image to the final image size of 224x224 pixels.

Linux

- You can use GIMP, imagemagick, or other tools of your choosing. You may also use online tools such as http://picresize.com (http://picresize.com) All the above steps are illustrative only. You need not follow these steps but following these will ensure that you produce a good quality dataset. You will be judged based on the quality of the images alone. Please do not edit your photos in any other way. You should not need to change the aspect ratio of your image. You also should not digitally remove the background or shadows—instead, take photos with a white background and minimal shadows.

Accepted Images

Images will be accepted and graded based on the criteria below

1. The final image should be size 224x224 pixels (RGB).
2. The file format should be a .jpg file.
3. The hand should be approximately centered on the frame.
4. The hand should not be obscured or cut off.
5. The photos follows the ASL gestures posted earlier.
6. The photos were not edited in any other way (e.g. no electronic removal of shadows or background).

Submission

Submit a zip file containing your images. There should be a total of 27 images (3 for each category)

1. Individual image file names should follow the convention of student-number_Alphabet_file-number.jpg (e.g. 100343434_A_1.jpg)
2. Zip all the images together and name it with the following convention: last-name_student-number.zip (e.g. last-name_100343434.zip).
3. Submit the zipped folder. We will be anonymizing and combining the images that everyone submits. We will announce when the combined data set will be available for download.

Figure 1: Acceptable Images (left) and Unacceptable Images (right)

Part B. Building a CNN [50 pt]

For this lab, we are not going to give you any starter code. You will be writing a convolutional neural network from scratch. You are welcome to use any code from previous labs, lectures and tutorials. You should also write your own code.

You may use the PyTorch documentation freely. You might also find online tutorials helpful. However, all code that you submit must be your own.

Make sure that your code is vectorized, and does not contain obvious inefficiencies (for example, unecessary for loops, or unnecessary calls to unsqueeze()). Ensure enough comments are included in the code so that your TA can understand what you are doing. It is your responsibility to show that you understand what you write.

This is much more challenging and time-consuming than the previous labs. Make sure that you give yourself plenty of time by starting early.

1. Data Loading and Splitting [5 pt]

Download the anonymized data provided on Quercus. To allow you to get a heads start on this project we will provide you with sample data from previous years. Split the data into training, validation, and test sets.

Note: Data splitting is not as trivial in this lab. We want our test set to closely resemble the setting in which our model will be used. In particular, our test set should contain hands that are never seen in training!

Explain how you split the data, either by describing what you did, or by showing the code that you used. Justify your choice of splitting strategy. How many training, validation, and test images do you have?

For loading the data, you can use plt.imread as in Lab 1, or any other method that you choose. You may find torchvision.datasets.ImageFolder helpful. (see https://pytorch.org/docs/stable/torchvision/datasets.html?highlight=image%20folder#torchvision.datasets.ImageFolder (https://pytorch.org/docs/stable/torchvision/datasets.html?highlight=image%20folder#torchvision.datasets.ImageFolder))

In [ ]:
```
# mount our Google Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [ ]:
```
!unzip '/content/drive/My Drive/APS360/TUT and Lab03/Lab_3b_Gesture_Dataset.zip' -d '/root/datasets'
```

```
Archive:  /content/drive/My Drive/APS360/TUT and Lab03/Lab_3b_Gesture_Datase
t.zip
   creating: /root/datasets/test/A/
  inflating: /root/datasets/test/A/100_A_1.jpg
  inflating: /root/datasets/test/A/100_A_2.jpg
  inflating: /root/datasets/test/A/100_A_3.jpg
  inflating: /root/datasets/test/A/101_A_1.jpg
  inflating: /root/datasets/test/A/101_A_2.jpg
  inflating: /root/datasets/test/A/101_A_3.jpg
  inflating: /root/datasets/test/A/102_A_1.jpg
  inflating: /root/datasets/test/A/102_A_2.jpg
  inflating: /root/datasets/test/A/102_A_3.jpg
  inflating: /root/datasets/test/A/83_A_1.jpg
  inflating: /root/datasets/test/A/83_A_2.jpg
  inflating: /root/datasets/test/A/83_A_3.jpg
  inflating: /root/datasets/test/A/84_A_1.jpg
  inflating: /root/datasets/test/A/84_A_2.jpg
  inflating: /root/datasets/test/A/84_A_3.jpg
  inflating: /root/datasets/test/A/85_A_1.jpg
  inflating: /root/datasets/test/A/85_A_2.jpg
  inflating: /root/datasets/test/A/85_A_3.jpg
  inflating: /root/datasets/test/A/86_A_1.jpg
  inflating: /root/datasets/test/A/86_A_2.jpg
  inflating: /root/datasets/test/A/86_A_3.jpg
  inflating: /root/datasets/test/A/87_A_1.jpg
  inflating: /root/datasets/test/A/87_A_2.jpg
  inflating: /root/datasets/test/A/87_A_3.jpg
  inflating: /root/datasets/test/A/88_A_1.jpg
  inflating: /root/datasets/test/A/88_A_2.jpg
  inflating: /root/datasets/test/A/88_A_3.jpg
  inflating: /root/datasets/test/A/90_A_1.jpg
  inflating: /root/datasets/test/A/90_A_2.jpg
  inflating: /root/datasets/test/A/90_A_3.jpg
  inflating: /root/datasets/test/A/91_A_1.jpg
  inflating: /root/datasets/test/A/91_A_2.jpg
  inflating: /root/datasets/test/A/91_A_3.jpg
  inflating: /root/datasets/test/A/92_A_1.jpg
  inflating: /root/datasets/test/A/92_A_2.jpg
  inflating: /root/datasets/test/A/92_A_3.jpg
  inflating: /root/datasets/test/A/93_A_1.jpg
  inflating: /root/datasets/test/A/93_A_2.jpg
  inflating: /root/datasets/test/A/93_A_3.jpg
  inflating: /root/datasets/test/A/94_A_1.jpg
  inflating: /root/datasets/test/A/94_A_2.jpg
  inflating: /root/datasets/test/A/94_A_3.jpg
  inflating: /root/datasets/test/A/95_A_1.jpg
  inflating: /root/datasets/test/A/95_A_2.jpg
  inflating: /root/datasets/test/A/95_A_3.jpg
  inflating: /root/datasets/test/A/96_A_1.jpg
  inflating: /root/datasets/test/A/96_A_2.jpg
  inflating: /root/datasets/test/A/96_A_3.jpg
  inflating: /root/datasets/test/A/97_A_1.jpg
  inflating: /root/datasets/test/A/97_A_2.jpg
  inflating: /root/datasets/test/A/97_A_3.jpg
  inflating: /root/datasets/test/A/98_A_1.jpg
  inflating: /root/datasets/test/A/98_A_2.jpg
  inflating: /root/datasets/test/A/98_A_3.jpg
```

```
inflating: /root/datasets/test/A/99_A_1.jpg
inflating: /root/datasets/test/A/99_A_2.jpg
inflating: /root/datasets/test/A/99_A_3.jpg
 creating: /root/datasets/test/B/
inflating: /root/datasets/test/B/100_B_1.jpg
inflating: /root/datasets/test/B/100_B_2.jpg
inflating: /root/datasets/test/B/100_B_3.jpg
inflating: /root/datasets/test/B/101_B_1.jpg
inflating: /root/datasets/test/B/101_B_2.jpg
inflating: /root/datasets/test/B/101_B_3.jpg
inflating: /root/datasets/test/B/102_B_1.jpg
inflating: /root/datasets/test/B/102_B_2.jpg
inflating: /root/datasets/test/B/102_B_3.jpg
inflating: /root/datasets/test/B/83_B_1.jpg
inflating: /root/datasets/test/B/83_B_2.jpg
inflating: /root/datasets/test/B/83_B_3.jpg
inflating: /root/datasets/test/B/84_B_1.jpg
inflating: /root/datasets/test/B/84_B_2.jpg
inflating: /root/datasets/test/B/84_B_3.jpg
inflating: /root/datasets/test/B/85_B_1.jpg
inflating: /root/datasets/test/B/85_B_2.jpg
inflating: /root/datasets/test/B/85_B_3.jpg
inflating: /root/datasets/test/B/86_B_1.jpg
inflating: /root/datasets/test/B/86_B_2.jpg
inflating: /root/datasets/test/B/86_B_3.jpg
inflating: /root/datasets/test/B/87_B_1.jpg
inflating: /root/datasets/test/B/87_B_2.jpg
inflating: /root/datasets/test/B/87_B_3.jpg
inflating: /root/datasets/test/B/88_B_1.jpg
inflating: /root/datasets/test/B/88_B_2.jpg
inflating: /root/datasets/test/B/88_B_3.jpg
inflating: /root/datasets/test/B/90_B_1.jpg
inflating: /root/datasets/test/B/90_B_2.jpg
inflating: /root/datasets/test/B/90_B_3.jpg
inflating: /root/datasets/test/B/91_B_1.jpg
inflating: /root/datasets/test/B/91_B_2.jpg
inflating: /root/datasets/test/B/91_B_3.jpg
inflating: /root/datasets/test/B/92_B_1.jpg
inflating: /root/datasets/test/B/92_B_2.jpg
inflating: /root/datasets/test/B/92_B_3.jpg
inflating: /root/datasets/test/B/93_B_1.jpg
inflating: /root/datasets/test/B/93_B_2.jpg
inflating: /root/datasets/test/B/93_B_3.jpg
inflating: /root/datasets/test/B/94_B_1.jpg
inflating: /root/datasets/test/B/94_B_2.jpg
inflating: /root/datasets/test/B/94_B_3.jpg
inflating: /root/datasets/test/B/95_B_1.jpg
inflating: /root/datasets/test/B/95_B_2.jpg
inflating: /root/datasets/test/B/95_B_3.jpg
inflating: /root/datasets/test/B/96_B_1.jpg
inflating: /root/datasets/test/B/96_B_2.jpg
inflating: /root/datasets/test/B/96_B_3.jpg
inflating: /root/datasets/test/B/97_B_1.jpg
inflating: /root/datasets/test/B/97_B_2.jpg
inflating: /root/datasets/test/B/97_B_3.jpg
inflating: /root/datasets/test/B/98_B_1.jpg
inflating: /root/datasets/test/B/98_B_2.jpg
```

```
inflating: /root/datasets/test/B/98_B_3.jpg
inflating: /root/datasets/test/B/99_B_1.jpg
inflating: /root/datasets/test/B/99_B_2.jpg
inflating: /root/datasets/test/B/99_B_3.jpg
 creating: /root/datasets/test/C/
inflating: /root/datasets/test/C/100_C_1.jpg
inflating: /root/datasets/test/C/100_C_2.jpg
inflating: /root/datasets/test/C/100_C_3.jpg
inflating: /root/datasets/test/C/101_C_1.jpg
inflating: /root/datasets/test/C/101_C_2.jpg
inflating: /root/datasets/test/C/101_C_3.jpg
inflating: /root/datasets/test/C/102_C_1.jpg
inflating: /root/datasets/test/C/102_C_2.jpg
inflating: /root/datasets/test/C/102_C_3.jpg
inflating: /root/datasets/test/C/83_C_1.jpg
inflating: /root/datasets/test/C/83_C_2.jpg
inflating: /root/datasets/test/C/83_C_3.jpg
inflating: /root/datasets/test/C/84_C_1.jpg
inflating: /root/datasets/test/C/84_C_2.jpg
inflating: /root/datasets/test/C/84_C_3.jpg
inflating: /root/datasets/test/C/85_C_1.jpg
inflating: /root/datasets/test/C/85_C_2.jpg
inflating: /root/datasets/test/C/85_C_3.jpg
inflating: /root/datasets/test/C/86_C_1.jpg
inflating: /root/datasets/test/C/86_C_2.jpg
inflating: /root/datasets/test/C/86_C_3.jpg
inflating: /root/datasets/test/C/87_C_1.jpg
inflating: /root/datasets/test/C/87_C_2.jpg
inflating: /root/datasets/test/C/87_C_3.jpg
inflating: /root/datasets/test/C/88_C_1.jpg
inflating: /root/datasets/test/C/88_C_2.jpg
inflating: /root/datasets/test/C/88_C_3.jpg
inflating: /root/datasets/test/C/90_C_1.jpg
inflating: /root/datasets/test/C/90_C_2.jpg
inflating: /root/datasets/test/C/90_C_3.jpg
inflating: /root/datasets/test/C/91_C_1.jpg
inflating: /root/datasets/test/C/91_C_2.jpg
inflating: /root/datasets/test/C/91_C_3.jpg
inflating: /root/datasets/test/C/92_C_1.jpg
inflating: /root/datasets/test/C/92_C_2.jpg
inflating: /root/datasets/test/C/92_C_3.jpg
inflating: /root/datasets/test/C/93_C_1.jpg
inflating: /root/datasets/test/C/93_C_2.jpg
inflating: /root/datasets/test/C/93_C_3.jpg
inflating: /root/datasets/test/C/94_C_1.jpg
inflating: /root/datasets/test/C/94_C_2.jpg
inflating: /root/datasets/test/C/94_C_3.jpg
inflating: /root/datasets/test/C/95_C_1.jpg
inflating: /root/datasets/test/C/95_C_2.jpg
inflating: /root/datasets/test/C/95_C_3.jpg
inflating: /root/datasets/test/C/96_C_1.jpg
inflating: /root/datasets/test/C/96_C_2.jpg
inflating: /root/datasets/test/C/96_C_3.jpg
inflating: /root/datasets/test/C/97_C_1.jpg
inflating: /root/datasets/test/C/97_C_2.jpg
inflating: /root/datasets/test/C/97_C_3.jpg
inflating: /root/datasets/test/C/98_C_1.jpg
```

```
inflating: /root/datasets/test/C/98_C_2.jpg
inflating: /root/datasets/test/C/98_C_3.jpg
inflating: /root/datasets/test/C/99_C_1.jpg
inflating: /root/datasets/test/C/99_C_2.jpg
inflating: /root/datasets/test/C/99_C_3.jpg
 creating: /root/datasets/test/D/
inflating: /root/datasets/test/D/100_D_1.jpg
inflating: /root/datasets/test/D/100_D_2.jpg
inflating: /root/datasets/test/D/100_D_3.jpg
inflating: /root/datasets/test/D/101_D_1.jpg
inflating: /root/datasets/test/D/101_D_2.jpg
inflating: /root/datasets/test/D/101_D_3.jpg
inflating: /root/datasets/test/D/102_D_1.jpg
inflating: /root/datasets/test/D/102_D_2.jpg
inflating: /root/datasets/test/D/102_D_3.jpg
inflating: /root/datasets/test/D/83_D_1.jpg
inflating: /root/datasets/test/D/83_D_2.jpg
inflating: /root/datasets/test/D/83_D_3.jpg
inflating: /root/datasets/test/D/84_D_1.jpg
inflating: /root/datasets/test/D/84_D_2.jpg
inflating: /root/datasets/test/D/84_D_3.jpg
inflating: /root/datasets/test/D/85_D_1.jpg
inflating: /root/datasets/test/D/85_D_2.jpg
inflating: /root/datasets/test/D/85_D_3.jpg
inflating: /root/datasets/test/D/86_D_1.jpg
inflating: /root/datasets/test/D/86_D_2.jpg
inflating: /root/datasets/test/D/86_D_3.jpg
inflating: /root/datasets/test/D/87_D_1.jpg
inflating: /root/datasets/test/D/87_D_2.jpg
inflating: /root/datasets/test/D/87_D_3.jpg
inflating: /root/datasets/test/D/88_D_1.jpg
inflating: /root/datasets/test/D/88_D_2.jpg
inflating: /root/datasets/test/D/88_D_3.jpg
inflating: /root/datasets/test/D/90_D_1.jpg
inflating: /root/datasets/test/D/90_D_2.jpg
inflating: /root/datasets/test/D/90_D_3.jpg
inflating: /root/datasets/test/D/91_D_1.jpg
inflating: /root/datasets/test/D/91_D_2.jpg
inflating: /root/datasets/test/D/91_D_3.jpg
inflating: /root/datasets/test/D/92_D_1.jpg
inflating: /root/datasets/test/D/92_D_2.jpg
inflating: /root/datasets/test/D/92_D_3.jpg
inflating: /root/datasets/test/D/93_D_1.jpg
inflating: /root/datasets/test/D/93_D_2.jpg
inflating: /root/datasets/test/D/93_D_3.jpg
inflating: /root/datasets/test/D/94_D_1.jpg
inflating: /root/datasets/test/D/94_D_2.jpg
inflating: /root/datasets/test/D/94_D_3.jpg
inflating: /root/datasets/test/D/95_D_1.jpg
inflating: /root/datasets/test/D/95_D_2.jpg
inflating: /root/datasets/test/D/95_D_3.jpg
inflating: /root/datasets/test/D/96_D_1.jpg
inflating: /root/datasets/test/D/96_D_2.jpg
inflating: /root/datasets/test/D/96_D_3.jpg
inflating: /root/datasets/test/D/97_D_1.jpg
inflating: /root/datasets/test/D/97_D_2.jpg
inflating: /root/datasets/test/D/97_D_3.jpg
```

```
inflating: /root/datasets/test/D/98_D_1.jpg
inflating: /root/datasets/test/D/98_D_2.jpg
inflating: /root/datasets/test/D/98_D_3.jpg
inflating: /root/datasets/test/D/99_D_1.jpg
inflating: /root/datasets/test/D/99_D_2.jpg
inflating: /root/datasets/test/D/99_D_3.jpg
 creating: /root/datasets/test/E/
inflating: /root/datasets/test/E/100_E_1.jpg
inflating: /root/datasets/test/E/100_E_2.jpg
inflating: /root/datasets/test/E/100_E_3.jpg
inflating: /root/datasets/test/E/101_E_1.jpg
inflating: /root/datasets/test/E/101_E_2.jpg
inflating: /root/datasets/test/E/101_E_3.jpg
inflating: /root/datasets/test/E/102_E_1.jpg
inflating: /root/datasets/test/E/102_E_2.jpg
inflating: /root/datasets/test/E/102_E_3.jpg
inflating: /root/datasets/test/E/83_E_1.jpg
inflating: /root/datasets/test/E/83_E_2.jpg
inflating: /root/datasets/test/E/83_E_3.jpg
inflating: /root/datasets/test/E/84_E_1.jpg
inflating: /root/datasets/test/E/84_E_2.jpg
inflating: /root/datasets/test/E/84_E_3.jpg
inflating: /root/datasets/test/E/85_E_1.jpg
inflating: /root/datasets/test/E/85_E_2.jpg
inflating: /root/datasets/test/E/85_E_3.jpg
inflating: /root/datasets/test/E/86_E_1.jpg
inflating: /root/datasets/test/E/86_E_2.jpg
inflating: /root/datasets/test/E/86_E_3.jpg
inflating: /root/datasets/test/E/87_E_1.jpg
inflating: /root/datasets/test/E/87_E_2.jpg
inflating: /root/datasets/test/E/87_E_3.jpg
inflating: /root/datasets/test/E/88_E_1.jpg
inflating: /root/datasets/test/E/88_E_2.jpg
inflating: /root/datasets/test/E/88_E_3.jpg
inflating: /root/datasets/test/E/90_E_1.jpg
inflating: /root/datasets/test/E/90_E_2.jpg
inflating: /root/datasets/test/E/90_E_3.jpg
inflating: /root/datasets/test/E/91_E_1.jpg
inflating: /root/datasets/test/E/91_E_2.jpg
inflating: /root/datasets/test/E/91_E_3.jpg
inflating: /root/datasets/test/E/92_E_1.jpg
inflating: /root/datasets/test/E/92_E_2.jpg
inflating: /root/datasets/test/E/92_E_3.jpg
inflating: /root/datasets/test/E/93_E_1.jpg
inflating: /root/datasets/test/E/93_E_2.jpg
inflating: /root/datasets/test/E/93_E_3.jpg
inflating: /root/datasets/test/E/94_E_1.jpg
inflating: /root/datasets/test/E/94_E_2.jpg
inflating: /root/datasets/test/E/94_E_3.jpg
inflating: /root/datasets/test/E/95_E_1.jpg
inflating: /root/datasets/test/E/95_E_2.jpg
inflating: /root/datasets/test/E/95_E_3.jpg
inflating: /root/datasets/test/E/96_E_1.jpg
inflating: /root/datasets/test/E/96_E_2.jpg
inflating: /root/datasets/test/E/96_E_3.jpg
inflating: /root/datasets/test/E/97_E_1.jpg
inflating: /root/datasets/test/E/97_E_2.jpg
```

```
inflating: /root/datasets/test/E/97_E_3.jpg
inflating: /root/datasets/test/E/98_E_1.jpg
inflating: /root/datasets/test/E/98_E_2.jpg
inflating: /root/datasets/test/E/98_E_3.jpg
inflating: /root/datasets/test/E/99_E_1.jpg
inflating: /root/datasets/test/E/99_E_2.jpg
inflating: /root/datasets/test/E/99_E_3.jpg
 creating: /root/datasets/test/F/
inflating: /root/datasets/test/F/100_F_1.jpg
inflating: /root/datasets/test/F/100_F_2.jpg
inflating: /root/datasets/test/F/100_F_3.jpg
inflating: /root/datasets/test/F/101_F_1.jpg
inflating: /root/datasets/test/F/101_F_2.jpg
inflating: /root/datasets/test/F/101_F_3.jpg
inflating: /root/datasets/test/F/102_F_1.jpg
inflating: /root/datasets/test/F/102_F_2.jpg
inflating: /root/datasets/test/F/102_F_3.jpg
inflating: /root/datasets/test/F/83_F_1.jpg
inflating: /root/datasets/test/F/83_F_2.jpg
inflating: /root/datasets/test/F/83_F_3.jpg
inflating: /root/datasets/test/F/84_F_1.jpg
inflating: /root/datasets/test/F/84_F_2.jpg
inflating: /root/datasets/test/F/84_F_3.jpg
inflating: /root/datasets/test/F/85_F_1.jpg
inflating: /root/datasets/test/F/85_F_2.jpg
inflating: /root/datasets/test/F/85_F_3.jpg
inflating: /root/datasets/test/F/86_F_1.jpg
inflating: /root/datasets/test/F/86_F_2.jpg
inflating: /root/datasets/test/F/86_F_3.jpg
inflating: /root/datasets/test/F/87_F_1.jpg
inflating: /root/datasets/test/F/87_F_2.jpg
inflating: /root/datasets/test/F/87_F_3.jpg
inflating: /root/datasets/test/F/88_F_1.jpg
inflating: /root/datasets/test/F/88_F_2.jpg
inflating: /root/datasets/test/F/88_F_3.jpg
inflating: /root/datasets/test/F/90_F_1.jpg
inflating: /root/datasets/test/F/90_F_2.jpg
inflating: /root/datasets/test/F/90_F_3.jpg
inflating: /root/datasets/test/F/91_F_1.jpg
inflating: /root/datasets/test/F/91_F_2.jpg
inflating: /root/datasets/test/F/91_F_3.jpg
inflating: /root/datasets/test/F/92_F_1.jpg
inflating: /root/datasets/test/F/92_F_2.jpg
inflating: /root/datasets/test/F/92_F_3.jpg
inflating: /root/datasets/test/F/93_F_1.jpg
inflating: /root/datasets/test/F/93_F_2.jpg
inflating: /root/datasets/test/F/93_F_3.jpg
inflating: /root/datasets/test/F/94_F_1.jpg
inflating: /root/datasets/test/F/94_F_2.jpg
inflating: /root/datasets/test/F/94_F_3.jpg
inflating: /root/datasets/test/F/95_F_1.jpg
inflating: /root/datasets/test/F/95_F_2.jpg
inflating: /root/datasets/test/F/95_F_3.jpg
inflating: /root/datasets/test/F/96_F_1.jpg
inflating: /root/datasets/test/F/96_F_2.jpg
inflating: /root/datasets/test/F/96_F_3.jpg
inflating: /root/datasets/test/F/97_F_1.jpg
```

```
inflating: /root/datasets/test/F/97_F_2.jpg
inflating: /root/datasets/test/F/97_F_3.jpg
inflating: /root/datasets/test/F/98_F_1.jpg
inflating: /root/datasets/test/F/98_F_2.jpg
inflating: /root/datasets/test/F/98_F_3.jpg
inflating: /root/datasets/test/F/99_F_1.jpg
inflating: /root/datasets/test/F/99_F_2.jpg
inflating: /root/datasets/test/F/99_F_3.jpg
 creating: /root/datasets/test/G/
inflating: /root/datasets/test/G/100_G_1.jpg
inflating: /root/datasets/test/G/100_G_2.jpg
inflating: /root/datasets/test/G/100_G_3.jpg
inflating: /root/datasets/test/G/101_G_1.jpg
inflating: /root/datasets/test/G/101_G_2.jpg
inflating: /root/datasets/test/G/101_G_3.jpg
inflating: /root/datasets/test/G/102_G_1.jpg
inflating: /root/datasets/test/G/102_G_2.jpg
inflating: /root/datasets/test/G/102_G_3.jpg
inflating: /root/datasets/test/G/83_G_1.jpg
inflating: /root/datasets/test/G/83_G_2.jpg
inflating: /root/datasets/test/G/83_G_3.jpg
inflating: /root/datasets/test/G/84_G_1.jpg
inflating: /root/datasets/test/G/84_G_2.jpg
inflating: /root/datasets/test/G/84_G_3.jpg
inflating: /root/datasets/test/G/85_G_1.jpg
inflating: /root/datasets/test/G/85_G_2.jpg
inflating: /root/datasets/test/G/85_G_3.jpg
inflating: /root/datasets/test/G/86_G_1.jpg
inflating: /root/datasets/test/G/86_G_2.jpg
inflating: /root/datasets/test/G/86_G_3.jpg
inflating: /root/datasets/test/G/87_G_1.jpg
inflating: /root/datasets/test/G/87_G_2.jpg
inflating: /root/datasets/test/G/87_G_3.jpg
inflating: /root/datasets/test/G/88_G_1.jpg
inflating: /root/datasets/test/G/88_G_2.jpg
inflating: /root/datasets/test/G/88_G_3.jpg
inflating: /root/datasets/test/G/90_G_1.jpg
inflating: /root/datasets/test/G/90_G_2.jpg
inflating: /root/datasets/test/G/90_G_3.jpg
inflating: /root/datasets/test/G/91_G_1.jpg
inflating: /root/datasets/test/G/91_G_2.jpg
inflating: /root/datasets/test/G/91_G_3.jpg
inflating: /root/datasets/test/G/92_G_1.jpg
inflating: /root/datasets/test/G/92_G_2.jpg
inflating: /root/datasets/test/G/92_G_3.jpg
inflating: /root/datasets/test/G/93_G_1.jpg
inflating: /root/datasets/test/G/93_G_2.jpg
inflating: /root/datasets/test/G/93_G_3.jpg
inflating: /root/datasets/test/G/94_G_1.jpg
inflating: /root/datasets/test/G/94_G_2.jpg
inflating: /root/datasets/test/G/94_G_3.jpg
inflating: /root/datasets/test/G/95_G_1.jpg
inflating: /root/datasets/test/G/95_G_2.jpg
inflating: /root/datasets/test/G/95_G_3.jpg
inflating: /root/datasets/test/G/96_G_1.jpg
inflating: /root/datasets/test/G/96_G_2.jpg
inflating: /root/datasets/test/G/96_G_3.jpg
```

```
inflating: /root/datasets/test/G/97_G_1.jpg
inflating: /root/datasets/test/G/97_G_2.jpg
inflating: /root/datasets/test/G/97_G_3.jpg
inflating: /root/datasets/test/G/98_G_1.jpg
inflating: /root/datasets/test/G/98_G_2.jpg
inflating: /root/datasets/test/G/98_G_3.jpg
inflating: /root/datasets/test/G/99_G_1.jpg
inflating: /root/datasets/test/G/99_G_2.jpg
inflating: /root/datasets/test/G/99_G_3.jpg
 creating: /root/datasets/test/H/
inflating: /root/datasets/test/H/100_H_1.jpg
inflating: /root/datasets/test/H/100_H_2.jpg
inflating: /root/datasets/test/H/100_H_3.jpg
inflating: /root/datasets/test/H/101_H_1.jpg
inflating: /root/datasets/test/H/101_H_2.jpg
inflating: /root/datasets/test/H/101_H_3.jpg
inflating: /root/datasets/test/H/102_H_1.jpg
inflating: /root/datasets/test/H/102_H_2.jpg
inflating: /root/datasets/test/H/102_H_3.jpg
inflating: /root/datasets/test/H/83_H_1.jpg
inflating: /root/datasets/test/H/83_H_2.jpg
inflating: /root/datasets/test/H/83_H_3.jpg
inflating: /root/datasets/test/H/84_H_1.jpg
inflating: /root/datasets/test/H/84_H_2.jpg
inflating: /root/datasets/test/H/84_H_3.jpg
inflating: /root/datasets/test/H/85_H_1.jpg
inflating: /root/datasets/test/H/85_H_2.jpg
inflating: /root/datasets/test/H/85_H_3.jpg
inflating: /root/datasets/test/H/86_H_1.jpg
inflating: /root/datasets/test/H/86_H_2.jpg
inflating: /root/datasets/test/H/86_H_3.jpg
inflating: /root/datasets/test/H/87_H_1.jpg
inflating: /root/datasets/test/H/87_H_2.jpg
inflating: /root/datasets/test/H/87_H_3.jpg
inflating: /root/datasets/test/H/88_H_1.jpg
inflating: /root/datasets/test/H/88_H_2.jpg
inflating: /root/datasets/test/H/88_H_3.jpg
inflating: /root/datasets/test/H/90_H_1.jpg
inflating: /root/datasets/test/H/90_H_2.jpg
inflating: /root/datasets/test/H/90_H_3.jpg
inflating: /root/datasets/test/H/91_H_1.jpg
inflating: /root/datasets/test/H/91_H_2.jpg
inflating: /root/datasets/test/H/91_H_3.jpg
inflating: /root/datasets/test/H/92_H_1.jpg
inflating: /root/datasets/test/H/92_H_2.jpg
inflating: /root/datasets/test/H/92_H_3.jpg
inflating: /root/datasets/test/H/93_H_1.jpg
inflating: /root/datasets/test/H/93_H_2.jpg
inflating: /root/datasets/test/H/93_H_3.jpg
inflating: /root/datasets/test/H/94_H_1.jpg
inflating: /root/datasets/test/H/94_H_2.jpg
inflating: /root/datasets/test/H/94_H_3.jpg
inflating: /root/datasets/test/H/95_H_1.jpg
inflating: /root/datasets/test/H/95_H_2.jpg
inflating: /root/datasets/test/H/95_H_3.jpg
inflating: /root/datasets/test/H/96_H_1.jpg
inflating: /root/datasets/test/H/96_H_2.jpg
```

```
inflating: /root/datasets/test/H/96_H_3.jpg
inflating: /root/datasets/test/H/97_H_1.jpg
inflating: /root/datasets/test/H/97_H_2.jpg
inflating: /root/datasets/test/H/97_H_3.jpg
inflating: /root/datasets/test/H/98_H_1.jpg
inflating: /root/datasets/test/H/98_H_2.jpg
inflating: /root/datasets/test/H/98_H_3.jpg
inflating: /root/datasets/test/H/99_H_1.jpg
inflating: /root/datasets/test/H/99_H_2.jpg
inflating: /root/datasets/test/H/99_H_3.jpg
 creating: /root/datasets/test/I/
inflating: /root/datasets/test/I/100_I_1.jpg
inflating: /root/datasets/test/I/100_I_2.jpg
inflating: /root/datasets/test/I/100_I_3.jpg
inflating: /root/datasets/test/I/101_I_1.jpg
inflating: /root/datasets/test/I/101_I_2.jpg
inflating: /root/datasets/test/I/101_I_3.jpg
inflating: /root/datasets/test/I/84_I_1.jpg
inflating: /root/datasets/test/I/84_I_2.jpg
inflating: /root/datasets/test/I/84_I_3.jpg
inflating: /root/datasets/test/I/85_I_1.jpg
inflating: /root/datasets/test/I/85_I_2.jpg
inflating: /root/datasets/test/I/85_I_3.jpg
inflating: /root/datasets/test/I/86_I_1.jpg
inflating: /root/datasets/test/I/86_I_2.jpg
inflating: /root/datasets/test/I/86_I_3.jpg
inflating: /root/datasets/test/I/87_I_1.jpg
inflating: /root/datasets/test/I/87_I_2.jpg
inflating: /root/datasets/test/I/87_I_3.jpg
inflating: /root/datasets/test/I/88_I_1.jpg
inflating: /root/datasets/test/I/88_I_2.jpg
inflating: /root/datasets/test/I/88_I_3.jpg
inflating: /root/datasets/test/I/90_I_1.jpg
inflating: /root/datasets/test/I/90_I_2.jpg
inflating: /root/datasets/test/I/90_I_3.jpg
inflating: /root/datasets/test/I/91_I_1.jpg
inflating: /root/datasets/test/I/91_I_2.jpg
inflating: /root/datasets/test/I/91_I_3.jpg
inflating: /root/datasets/test/I/92_I_1.jpg
inflating: /root/datasets/test/I/92_I_2.jpg
inflating: /root/datasets/test/I/92_I_3.jpg
inflating: /root/datasets/test/I/93_I_1.jpg
inflating: /root/datasets/test/I/93_I_2.jpg
inflating: /root/datasets/test/I/93_I_3.jpg
inflating: /root/datasets/test/I/94_I_1.jpg
inflating: /root/datasets/test/I/94_I_2.jpg
inflating: /root/datasets/test/I/94_I_3.jpg
inflating: /root/datasets/test/I/95_I_1.jpg
inflating: /root/datasets/test/I/95_I_2.jpg
inflating: /root/datasets/test/I/95_I_3.jpg
inflating: /root/datasets/test/I/96_I_1.jpg
inflating: /root/datasets/test/I/96_I_2.jpg
inflating: /root/datasets/test/I/96_I_3.jpg
inflating: /root/datasets/test/I/99_I_1.jpg
inflating: /root/datasets/test/I/99_I_2.jpg
inflating: /root/datasets/test/I/99_I_3.jpg
 creating: /root/datasets/train/A/
```

```
inflating: /root/datasets/train/A/1_A_1.jpg
inflating: /root/datasets/train/A/1_A_2.jpg
inflating: /root/datasets/train/A/1_A_3.jpg
inflating: /root/datasets/train/A/10_A_1.jpg
inflating: /root/datasets/train/A/10_A_2.jpg
inflating: /root/datasets/train/A/10_A_3.jpg
inflating: /root/datasets/train/A/11_A_1.jpg
inflating: /root/datasets/train/A/11_A_2.jpg
inflating: /root/datasets/train/A/11_A_3.jpg
inflating: /root/datasets/train/A/12_A_1.jpg
inflating: /root/datasets/train/A/12_A_2.jpg
inflating: /root/datasets/train/A/12_A_3.jpg
inflating: /root/datasets/train/A/13_A_1.jpg
inflating: /root/datasets/train/A/13_A_2.jpg
inflating: /root/datasets/train/A/13_A_3.jpg
inflating: /root/datasets/train/A/14_A_1.jpg
inflating: /root/datasets/train/A/14_A_2.jpg
inflating: /root/datasets/train/A/14_A_3.jpg
inflating: /root/datasets/train/A/15_A_1.jpg
inflating: /root/datasets/train/A/15_A_2.jpg
inflating: /root/datasets/train/A/15_A_3.jpg
inflating: /root/datasets/train/A/16_A_1.jpg
inflating: /root/datasets/train/A/16_A_2.jpg
inflating: /root/datasets/train/A/16_A_3.jpg
inflating: /root/datasets/train/A/17_A_1.jpg
inflating: /root/datasets/train/A/17_A_2.jpg
inflating: /root/datasets/train/A/17_A_3.jpg
inflating: /root/datasets/train/A/18_A_1.jpg
inflating: /root/datasets/train/A/18_A_2.jpg
inflating: /root/datasets/train/A/18_A_3.jpg
inflating: /root/datasets/train/A/19_A_1.jpg
inflating: /root/datasets/train/A/19_A_2.jpg
inflating: /root/datasets/train/A/19_A_3.jpg
inflating: /root/datasets/train/A/2_A_1.jpg
inflating: /root/datasets/train/A/2_A_2.jpg
inflating: /root/datasets/train/A/2_A_3.jpg
inflating: /root/datasets/train/A/20_A_01.jpg
inflating: /root/datasets/train/A/20_A_02.jpg
inflating: /root/datasets/train/A/20_A_03.jpg
inflating: /root/datasets/train/A/21_A_1.jpg
inflating: /root/datasets/train/A/21_A_2.jpg
inflating: /root/datasets/train/A/21_A_3.jpg
inflating: /root/datasets/train/A/22_A_1.jpg
inflating: /root/datasets/train/A/22_A_2.jpg
inflating: /root/datasets/train/A/22_A_3.jpg
inflating: /root/datasets/train/A/24_A_1.jpg
inflating: /root/datasets/train/A/24_A_2.jpg
inflating: /root/datasets/train/A/24_A_3.jpg
inflating: /root/datasets/train/A/26_A_1.jpg
inflating: /root/datasets/train/A/26_A_2.jpg
inflating: /root/datasets/train/A/26_A_3.jpg
inflating: /root/datasets/train/A/3_A_1.jpg
inflating: /root/datasets/train/A/3_A_2.jpg
inflating: /root/datasets/train/A/3_A_3.jpg
inflating: /root/datasets/train/A/30_A_1.jpg
inflating: /root/datasets/train/A/30_A_2.jpg
inflating: /root/datasets/train/A/30_A_3.jpg
```

```
inflating: /root/datasets/train/A/31_A_1.jpg
inflating: /root/datasets/train/A/31_A_2.jpg
inflating: /root/datasets/train/A/31_A_3.jpg
inflating: /root/datasets/train/A/32_A_1.jpg
inflating: /root/datasets/train/A/32_A_2.jpg
inflating: /root/datasets/train/A/32_A_3.jpg
inflating: /root/datasets/train/A/33_A_1.jpg
inflating: /root/datasets/train/A/33_A_2.jpg
inflating: /root/datasets/train/A/33_A_3.jpg
inflating: /root/datasets/train/A/34_A_1.jpg
inflating: /root/datasets/train/A/34_A_2.jpg
inflating: /root/datasets/train/A/34_A_3.jpg
inflating: /root/datasets/train/A/35_A_1.jpg
inflating: /root/datasets/train/A/35_A_2.jpg
inflating: /root/datasets/train/A/35_A_3.jpg
inflating: /root/datasets/train/A/36_A_1.jpg
inflating: /root/datasets/train/A/36_A_2.jpg
inflating: /root/datasets/train/A/36_A_3.jpg
inflating: /root/datasets/train/A/37_A_1.jpg
inflating: /root/datasets/train/A/37_A_2.jpg
inflating: /root/datasets/train/A/37_A_3.jpg
inflating: /root/datasets/train/A/38_A_1.jpg
inflating: /root/datasets/train/A/38_A_2.jpg
inflating: /root/datasets/train/A/38_A_3.jpg
inflating: /root/datasets/train/A/39_A_1.jpg
inflating: /root/datasets/train/A/39_A_2.jpg
inflating: /root/datasets/train/A/39_A_3.jpg
inflating: /root/datasets/train/A/4_A_1.jpg
inflating: /root/datasets/train/A/4_A_2.jpg
inflating: /root/datasets/train/A/4_A_3.jpg
inflating: /root/datasets/train/A/40_A_1.jpg
inflating: /root/datasets/train/A/40_A_2.jpg
inflating: /root/datasets/train/A/40_A_3.jpg
inflating: /root/datasets/train/A/41_A_1.jpg
inflating: /root/datasets/train/A/41_A_2.jpg
inflating: /root/datasets/train/A/41_A_3.jpg
inflating: /root/datasets/train/A/42_A_1.jpg
inflating: /root/datasets/train/A/42_A_2.jpg
inflating: /root/datasets/train/A/42_A_3.jpg
inflating: /root/datasets/train/A/43_A_1.jpg
inflating: /root/datasets/train/A/43_A_2.jpg
inflating: /root/datasets/train/A/43_A_3.jpg
inflating: /root/datasets/train/A/44_A_1.jpg
inflating: /root/datasets/train/A/44_A_2.jpg
inflating: /root/datasets/train/A/44_A_3.jpg
inflating: /root/datasets/train/A/45_A_1.jpg
inflating: /root/datasets/train/A/45_A_2.jpg
inflating: /root/datasets/train/A/45_A_3.jpg
inflating: /root/datasets/train/A/47_A_1.jpg
inflating: /root/datasets/train/A/47_A_2.jpg
inflating: /root/datasets/train/A/47_A_3.jpg
inflating: /root/datasets/train/A/48_A_1.jpg
inflating: /root/datasets/train/A/48_A_2.jpg
inflating: /root/datasets/train/A/48_A_3.jpg
inflating: /root/datasets/train/A/49_A_1.jpg
inflating: /root/datasets/train/A/49_A_2.jpg
inflating: /root/datasets/train/A/49_A_3.jpg
```

```
inflating: /root/datasets/train/A/50_A_1.jpg
inflating: /root/datasets/train/A/50_A_2.jpg
inflating: /root/datasets/train/A/50_A_3.jpg
inflating: /root/datasets/train/A/51_A_1.jpg
inflating: /root/datasets/train/A/51_A_2.jpg
inflating: /root/datasets/train/A/51_A_3.jpg
inflating: /root/datasets/train/A/52_A_1.jpg
inflating: /root/datasets/train/A/52_A_2.jpg
inflating: /root/datasets/train/A/52_A_3.jpg
inflating: /root/datasets/train/A/53_A_1.jpg
inflating: /root/datasets/train/A/53_A_2.jpg
inflating: /root/datasets/train/A/53_A_3.jpg
inflating: /root/datasets/train/A/54_A_1.jpg
inflating: /root/datasets/train/A/54_A_2.jpg
inflating: /root/datasets/train/A/54_A_3.jpg
inflating: /root/datasets/train/A/55_A_1.jpg
inflating: /root/datasets/train/A/55_A_2.jpg
inflating: /root/datasets/train/A/55_A_3.jpg
inflating: /root/datasets/train/A/56_A_1.jpg
inflating: /root/datasets/train/A/56_A_2.jpg
inflating: /root/datasets/train/A/56_A_3.jpg
inflating: /root/datasets/train/A/57_A_1.jpg
inflating: /root/datasets/train/A/57_A_2.jpg
inflating: /root/datasets/train/A/57_A_3.jpg
inflating: /root/datasets/train/A/58_A_1.jpg
inflating: /root/datasets/train/A/58_A_2.jpg
inflating: /root/datasets/train/A/58_A_3.jpg
inflating: /root/datasets/train/A/59_A_1.jpg
inflating: /root/datasets/train/A/59_A_2.jpg
inflating: /root/datasets/train/A/59_A_3.jpg
inflating: /root/datasets/train/A/6_A_1.jpg
inflating: /root/datasets/train/A/6_A_2.jpg
inflating: /root/datasets/train/A/6_A_3.jpg
inflating: /root/datasets/train/A/60_A_1.jpg
inflating: /root/datasets/train/A/60_A_2.jpg
inflating: /root/datasets/train/A/60_A_3.jpg
inflating: /root/datasets/train/A/61_A_1.jpg
inflating: /root/datasets/train/A/61_A_2.jpg
inflating: /root/datasets/train/A/61_A_3.jpg
inflating: /root/datasets/train/A/62_A_1.jpg
inflating: /root/datasets/train/A/62_A_2.jpg
inflating: /root/datasets/train/A/62_A_3.jpg
inflating: /root/datasets/train/A/7_A_1.jpg
inflating: /root/datasets/train/A/7_A_2.jpg
inflating: /root/datasets/train/A/7_A_3.jpg
inflating: /root/datasets/train/A/8_A_1.jpg
inflating: /root/datasets/train/A/8_A_2.jpg
inflating: /root/datasets/train/A/8_A_3.jpg
inflating: /root/datasets/train/A/9_A_1.jpg
inflating: /root/datasets/train/A/9_A_2.jpg
inflating: /root/datasets/train/A/9_A_3.jpg
 creating: /root/datasets/train/B/
inflating: /root/datasets/train/B/1_B_1.jpg
inflating: /root/datasets/train/B/1_B_2.jpg
inflating: /root/datasets/train/B/1_B_3.jpg
inflating: /root/datasets/train/B/10_B_4.jpg
inflating: /root/datasets/train/B/10_B_5.jpg
```

```
inflating: /root/datasets/train/B/10_B_6.jpg
inflating: /root/datasets/train/B/11_B_1.jpg
inflating: /root/datasets/train/B/11_B_2.jpg
inflating: /root/datasets/train/B/11_B_3.jpg
inflating: /root/datasets/train/B/12_B_1.jpg
inflating: /root/datasets/train/B/12_B_2.jpg
inflating: /root/datasets/train/B/12_B_3.jpg
inflating: /root/datasets/train/B/13_B_1.jpg
inflating: /root/datasets/train/B/13_B_2.jpg
inflating: /root/datasets/train/B/13_B_3.jpg
inflating: /root/datasets/train/B/14_B_1.jpg
inflating: /root/datasets/train/B/14_B_2.jpg
inflating: /root/datasets/train/B/14_B_3.jpg
inflating: /root/datasets/train/B/15_B_1.jpg
inflating: /root/datasets/train/B/15_B_2.jpg
inflating: /root/datasets/train/B/15_B_3.jpg
inflating: /root/datasets/train/B/16_B_1.jpg
inflating: /root/datasets/train/B/16_B_2.jpg
inflating: /root/datasets/train/B/16_B_3.jpg
inflating: /root/datasets/train/B/17_B_1.jpg
inflating: /root/datasets/train/B/17_B_2.jpg
inflating: /root/datasets/train/B/17_B_3.jpg
inflating: /root/datasets/train/B/18_B_1.jpg
inflating: /root/datasets/train/B/18_B_2.jpg
inflating: /root/datasets/train/B/18_B_3.jpg
inflating: /root/datasets/train/B/19_B_1.jpg
inflating: /root/datasets/train/B/19_B_2.jpg
inflating: /root/datasets/train/B/19_B_3.jpg
inflating: /root/datasets/train/B/2_B_1.jpg
inflating: /root/datasets/train/B/2_B_2.jpg
inflating: /root/datasets/train/B/2_B_3.jpg
inflating: /root/datasets/train/B/20_B_01.jpg
inflating: /root/datasets/train/B/20_B_02.jpg
inflating: /root/datasets/train/B/20_B_03.jpg
inflating: /root/datasets/train/B/21_B_1.jpg
inflating: /root/datasets/train/B/21_B_2.jpg
inflating: /root/datasets/train/B/21_B_3.jpg
inflating: /root/datasets/train/B/22_B_2.jpg
inflating: /root/datasets/train/B/22_B_3.jpg
inflating: /root/datasets/train/B/22_B_4.jpg
inflating: /root/datasets/train/B/24_B_1.jpg
inflating: /root/datasets/train/B/24_B_2.jpg
inflating: /root/datasets/train/B/24_B_3.jpg
inflating: /root/datasets/train/B/26_B_1.jpg
inflating: /root/datasets/train/B/26_B_2.jpg
inflating: /root/datasets/train/B/26_B_3.jpg
inflating: /root/datasets/train/B/3_B_1.jpg
inflating: /root/datasets/train/B/3_B_2.jpg
inflating: /root/datasets/train/B/3_B_3.jpg
inflating: /root/datasets/train/B/30_B_1.jpg
inflating: /root/datasets/train/B/30_B_2.jpg
inflating: /root/datasets/train/B/30_B_3.jpg
inflating: /root/datasets/train/B/31_B_1.jpg
inflating: /root/datasets/train/B/31_B_2.jpg
inflating: /root/datasets/train/B/31_B_3.jpg
inflating: /root/datasets/train/B/32_B_1.jpg
inflating: /root/datasets/train/B/32_B_2.jpg
```

```
inflating: /root/datasets/train/B/32_B_3.jpg
inflating: /root/datasets/train/B/33_B_4.jpg
inflating: /root/datasets/train/B/33_B_5.jpg
inflating: /root/datasets/train/B/33_B_6.jpg
inflating: /root/datasets/train/B/34_B_1.jpg
inflating: /root/datasets/train/B/34_B_2.jpg
inflating: /root/datasets/train/B/34_B_3.jpg
inflating: /root/datasets/train/B/35_B_1.jpg
inflating: /root/datasets/train/B/35_B_2.jpg
inflating: /root/datasets/train/B/35_B_3.jpg
inflating: /root/datasets/train/B/36_B_1.jpg
inflating: /root/datasets/train/B/36_B_2.jpg
inflating: /root/datasets/train/B/36_B_3.jpg
inflating: /root/datasets/train/B/37_B_1.jpg
inflating: /root/datasets/train/B/37_B_2.jpg
inflating: /root/datasets/train/B/37_B_3.jpg
inflating: /root/datasets/train/B/38_B_1.jpg
inflating: /root/datasets/train/B/38_B_2.jpg
inflating: /root/datasets/train/B/38_B_3.jpg
inflating: /root/datasets/train/B/39_B_1.jpg
inflating: /root/datasets/train/B/39_B_2.jpg
inflating: /root/datasets/train/B/39_B_3.jpg
inflating: /root/datasets/train/B/4_B_1.jpg
inflating: /root/datasets/train/B/4_B_2.jpg
inflating: /root/datasets/train/B/4_B_3.jpg
inflating: /root/datasets/train/B/40_B_1.jpg
inflating: /root/datasets/train/B/40_B_2.jpg
inflating: /root/datasets/train/B/40_B_3.jpg
inflating: /root/datasets/train/B/41_B_1.jpg
inflating: /root/datasets/train/B/41_B_2.jpg
inflating: /root/datasets/train/B/41_B_3.jpg
inflating: /root/datasets/train/B/42_B_1.jpg
inflating: /root/datasets/train/B/42_B_2.jpg
inflating: /root/datasets/train/B/42_B_3.jpg
inflating: /root/datasets/train/B/43_B_1.jpg
inflating: /root/datasets/train/B/43_B_2.jpg
inflating: /root/datasets/train/B/43_B_3.jpg
inflating: /root/datasets/train/B/44_B_1.jpg
inflating: /root/datasets/train/B/44_B_2.jpg
inflating: /root/datasets/train/B/44_B_3.jpg
inflating: /root/datasets/train/B/45_B_1.jpg
inflating: /root/datasets/train/B/45_B_2.jpg
inflating: /root/datasets/train/B/45_B_3.jpg
inflating: /root/datasets/train/B/47_B_1.jpg
inflating: /root/datasets/train/B/47_B_2.jpg
inflating: /root/datasets/train/B/47_B_3.jpg
inflating: /root/datasets/train/B/48_B_1.jpg
inflating: /root/datasets/train/B/48_B_2.jpg
inflating: /root/datasets/train/B/48_B_3.jpg
inflating: /root/datasets/train/B/49_B_1.jpg
inflating: /root/datasets/train/B/49_B_2.jpg
inflating: /root/datasets/train/B/49_B_3.jpg
inflating: /root/datasets/train/B/50_B_1.jpg
inflating: /root/datasets/train/B/50_B_2.jpg
inflating: /root/datasets/train/B/50_B_3.jpg
inflating: /root/datasets/train/B/51_B_1.jpg
inflating: /root/datasets/train/B/51_B_2.jpg
```

```
inflating: /root/datasets/train/B/51_B_3.jpg
inflating: /root/datasets/train/B/52_B_1.jpg
inflating: /root/datasets/train/B/52_B_2.jpg
inflating: /root/datasets/train/B/52_B_3.jpg
inflating: /root/datasets/train/B/53_B_1.jpg
inflating: /root/datasets/train/B/53_B_2.jpg
inflating: /root/datasets/train/B/53_B_3.jpg
inflating: /root/datasets/train/B/54_B_1.jpg
inflating: /root/datasets/train/B/54_B_2.jpg
inflating: /root/datasets/train/B/54_B_3.jpg
inflating: /root/datasets/train/B/55_B_4.jpg
inflating: /root/datasets/train/B/55_B_5.jpg
inflating: /root/datasets/train/B/55_B_6.jpg
inflating: /root/datasets/train/B/56_B_1.jpg
inflating: /root/datasets/train/B/56_B_2.jpg
inflating: /root/datasets/train/B/56_B_3.jpg
inflating: /root/datasets/train/B/57_B_1.jpg
inflating: /root/datasets/train/B/57_B_2.jpg
inflating: /root/datasets/train/B/57_B_3.jpg
inflating: /root/datasets/train/B/58_B_1.jpg
inflating: /root/datasets/train/B/58_B_2.jpg
inflating: /root/datasets/train/B/58_B_3.jpg
inflating: /root/datasets/train/B/59_B_1.jpg
inflating: /root/datasets/train/B/59_B_2.jpg
inflating: /root/datasets/train/B/59_B_3.jpg
inflating: /root/datasets/train/B/6_B_1.jpg
inflating: /root/datasets/train/B/6_B_2.jpg
inflating: /root/datasets/train/B/6_B_3.jpg
inflating: /root/datasets/train/B/60_B_1.jpg
inflating: /root/datasets/train/B/60_B_2.jpg
inflating: /root/datasets/train/B/60_B_3.jpg
inflating: /root/datasets/train/B/61_B_1.jpg
inflating: /root/datasets/train/B/61_B_2.jpg
inflating: /root/datasets/train/B/61_B_3.jpg
inflating: /root/datasets/train/B/62_B_1.jpg
inflating: /root/datasets/train/B/62_B_2.jpg
inflating: /root/datasets/train/B/62_B_3.jpg
inflating: /root/datasets/train/B/7_B_1.jpg
inflating: /root/datasets/train/B/7_B_2.jpg
inflating: /root/datasets/train/B/7_B_3.jpg
inflating: /root/datasets/train/B/8_B_1.jpg
inflating: /root/datasets/train/B/8_B_2.jpg
inflating: /root/datasets/train/B/8_B_3.jpg
inflating: /root/datasets/train/B/9_B_1.jpg
inflating: /root/datasets/train/B/9_B_2.jpg
inflating: /root/datasets/train/B/9_B_3.jpg
 creating: /root/datasets/train/C/
inflating: /root/datasets/train/C/1_C_1.jpg
inflating: /root/datasets/train/C/1_C_2.jpg
inflating: /root/datasets/train/C/1_C_3.jpg
inflating: /root/datasets/train/C/10_C_7.jpg
inflating: /root/datasets/train/C/10_C_8.jpg
inflating: /root/datasets/train/C/10_C_9.jpg
inflating: /root/datasets/train/C/11_C_1.jpg
inflating: /root/datasets/train/C/11_C_2.jpg
inflating: /root/datasets/train/C/11_C_3.jpg
inflating: /root/datasets/train/C/12_C_1.jpg
```

```
inflating: /root/datasets/train/C/12_C_2.jpg
inflating: /root/datasets/train/C/12_C_3.jpg
inflating: /root/datasets/train/C/13_C_1.jpg
inflating: /root/datasets/train/C/13_C_2.jpg
inflating: /root/datasets/train/C/13_C_3.jpg
inflating: /root/datasets/train/C/14_C_1.jpg
inflating: /root/datasets/train/C/14_C_2.jpg
inflating: /root/datasets/train/C/14_C_3.jpg
inflating: /root/datasets/train/C/15_C_1.jpg
inflating: /root/datasets/train/C/15_C_2.jpg
inflating: /root/datasets/train/C/15_C_3.jpg
inflating: /root/datasets/train/C/16_C_1.jpg
inflating: /root/datasets/train/C/16_C_2.jpg
inflating: /root/datasets/train/C/16_C_3.jpg
inflating: /root/datasets/train/C/17_C_1.jpg
inflating: /root/datasets/train/C/17_C_2.jpg
inflating: /root/datasets/train/C/17_C_3.jpg
inflating: /root/datasets/train/C/18_C_1.jpg
inflating: /root/datasets/train/C/18_C_2.jpg
inflating: /root/datasets/train/C/18_C_3.jpg
inflating: /root/datasets/train/C/19_C_1.jpg
inflating: /root/datasets/train/C/19_C_2.jpg
inflating: /root/datasets/train/C/19_C_3.jpg
inflating: /root/datasets/train/C/2_C_1.jpg
inflating: /root/datasets/train/C/2_C_2.jpg
inflating: /root/datasets/train/C/2_C_3.jpg
inflating: /root/datasets/train/C/20_C_01.jpg
inflating: /root/datasets/train/C/20_C_02.jpg
inflating: /root/datasets/train/C/20_C_03.jpg
inflating: /root/datasets/train/C/21_C_1.jpg
inflating: /root/datasets/train/C/21_C_2.jpg
inflating: /root/datasets/train/C/21_C_3.jpg
inflating: /root/datasets/train/C/22_C_7.jpg
inflating: /root/datasets/train/C/22_C_8.jpg
inflating: /root/datasets/train/C/22_C_9.jpg
inflating: /root/datasets/train/C/24_C_1.jpg
inflating: /root/datasets/train/C/24_C_2.jpg
inflating: /root/datasets/train/C/24_C_3.jpg
inflating: /root/datasets/train/C/26_C_1.jpg
inflating: /root/datasets/train/C/26_C_2.jpg
inflating: /root/datasets/train/C/26_C_3.jpg
inflating: /root/datasets/train/C/3_C_1.jpg
inflating: /root/datasets/train/C/3_C_2.jpg
inflating: /root/datasets/train/C/3_C_3.jpg
inflating: /root/datasets/train/C/30_C_1.jpg
inflating: /root/datasets/train/C/30_C_2.jpg
inflating: /root/datasets/train/C/30_C_3.jpg
inflating: /root/datasets/train/C/31_C_1.jpg
inflating: /root/datasets/train/C/31_C_2.jpg
inflating: /root/datasets/train/C/31_C_3.jpg
inflating: /root/datasets/train/C/32_C_1.jpg
inflating: /root/datasets/train/C/32_C_2.jpg
inflating: /root/datasets/train/C/32_C_3.jpg
inflating: /root/datasets/train/C/33_C_7.jpg
inflating: /root/datasets/train/C/33_C_8.jpg
inflating: /root/datasets/train/C/33_C_9.jpg
inflating: /root/datasets/train/C/34_C_1.jpg
```

```
inflating: /root/datasets/train/C/34_C_2.jpg
inflating: /root/datasets/train/C/34_C_3.jpg
inflating: /root/datasets/train/C/35_C_1.jpg
inflating: /root/datasets/train/C/35_C_2.jpg
inflating: /root/datasets/train/C/35_C_3.jpg
inflating: /root/datasets/train/C/36_C_1.jpg
inflating: /root/datasets/train/C/36_C_2.jpg
inflating: /root/datasets/train/C/36_C_3.jpg
inflating: /root/datasets/train/C/37_C_1.jpg
inflating: /root/datasets/train/C/37_C_2.jpg
inflating: /root/datasets/train/C/37_C_3.jpg
inflating: /root/datasets/train/C/38_C_1.jpg
inflating: /root/datasets/train/C/38_C_2.jpg
inflating: /root/datasets/train/C/38_C_3.jpg
inflating: /root/datasets/train/C/39_C_1.jpg
inflating: /root/datasets/train/C/39_C_2.jpg
inflating: /root/datasets/train/C/39_C_3.jpg
inflating: /root/datasets/train/C/4_C_1.jpg
inflating: /root/datasets/train/C/4_C_2.jpg
inflating: /root/datasets/train/C/4_C_3.jpg
inflating: /root/datasets/train/C/40_C_1.jpg
inflating: /root/datasets/train/C/40_C_2.jpg
inflating: /root/datasets/train/C/40_C_3.jpg
inflating: /root/datasets/train/C/41_C_1.jpg
inflating: /root/datasets/train/C/41_C_2.jpg
inflating: /root/datasets/train/C/41_C_3.jpg
inflating: /root/datasets/train/C/42_C_1.jpg
inflating: /root/datasets/train/C/42_C_2.jpg
inflating: /root/datasets/train/C/42_C_3.jpg
inflating: /root/datasets/train/C/43_C_1.jpg
inflating: /root/datasets/train/C/43_C_2.jpg
inflating: /root/datasets/train/C/43_C_3.jpg
inflating: /root/datasets/train/C/44_C_1.jpg
inflating: /root/datasets/train/C/44_C_2.jpg
inflating: /root/datasets/train/C/44_C_3.jpg
inflating: /root/datasets/train/C/45_C_1.jpg
inflating: /root/datasets/train/C/45_C_2.jpg
inflating: /root/datasets/train/C/45_C_3.jpg
inflating: /root/datasets/train/C/47_C_1.jpg
inflating: /root/datasets/train/C/47_C_2.jpg
inflating: /root/datasets/train/C/47_C_3.jpg
inflating: /root/datasets/train/C/48_C_1.jpg
inflating: /root/datasets/train/C/48_C_2.jpg
inflating: /root/datasets/train/C/48_C_3.jpg
inflating: /root/datasets/train/C/49_C_1.jpg
inflating: /root/datasets/train/C/49_C_2.jpg
inflating: /root/datasets/train/C/49_C_3.jpg
inflating: /root/datasets/train/C/50_C_1.jpg
inflating: /root/datasets/train/C/50_C_2.jpg
inflating: /root/datasets/train/C/50_C_3.jpg
inflating: /root/datasets/train/C/51_C_1.jpg
inflating: /root/datasets/train/C/51_C_2.jpg
inflating: /root/datasets/train/C/51_C_3.jpg
inflating: /root/datasets/train/C/52_C_1.jpg
inflating: /root/datasets/train/C/52_C_2.jpg
inflating: /root/datasets/train/C/52_C_3.jpg
inflating: /root/datasets/train/C/53_C_1.jpg
```

```
  inflating: /root/datasets/train/C/53_C_2.jpg
  inflating: /root/datasets/train/C/53_C_3.jpg
  inflating: /root/datasets/train/C/54_C_1.jpg
  inflating: /root/datasets/train/C/54_C_2.jpg
  inflating: /root/datasets/train/C/54_C_3.jpg
  inflating: /root/datasets/train/C/55_C_7.jpg
  inflating: /root/datasets/train/C/55_C_8.jpg
  inflating: /root/datasets/train/C/55_C_9.jpg
  inflating: /root/datasets/train/C/56_C_1.jpg
  inflating: /root/datasets/train/C/56_C_2.jpg
  inflating: /root/datasets/train/C/56_C_3.jpg
  inflating: /root/datasets/train/C/57_C_1.jpg
  inflating: /root/datasets/train/C/57_C_2.jpg
  inflating: /root/datasets/train/C/57_C_3.jpg
  inflating: /root/datasets/train/C/58_C_1.jpg
  inflating: /root/datasets/train/C/58_C_2.jpg
  inflating: /root/datasets/train/C/58_C_3.jpg
  inflating: /root/datasets/train/C/59_C_1.jpg
  inflating: /root/datasets/train/C/59_C_2.jpg
  inflating: /root/datasets/train/C/59_C_3.jpg
  inflating: /root/datasets/train/C/6_C_1.jpg
  inflating: /root/datasets/train/C/6_C_2.jpg
  inflating: /root/datasets/train/C/6_C_3.jpg
  inflating: /root/datasets/train/C/60_C_1.jpg
  inflating: /root/datasets/train/C/60_C_2.jpg
  inflating: /root/datasets/train/C/60_C_3.jpg
  inflating: /root/datasets/train/C/61_C_1.jpg
  inflating: /root/datasets/train/C/61_C_2.jpg
  inflating: /root/datasets/train/C/61_C_3.jpg
  inflating: /root/datasets/train/C/62_C_1.jpg
  inflating: /root/datasets/train/C/62_C_2.jpg
  inflating: /root/datasets/train/C/62_C_3.jpg
  inflating: /root/datasets/train/C/7_C_1.jpg
  inflating: /root/datasets/train/C/7_C_2.jpg
  inflating: /root/datasets/train/C/7_C_3.jpg
  inflating: /root/datasets/train/C/8_C_1.jpg
  inflating: /root/datasets/train/C/8_C_2.jpg
  inflating: /root/datasets/train/C/8_C_3.jpg
  inflating: /root/datasets/train/C/9_C_1.jpg
  inflating: /root/datasets/train/C/9_C_2.jpg
  inflating: /root/datasets/train/C/9_C_3.jpg
   creating: /root/datasets/train/D/
  inflating: /root/datasets/train/D/1_D_1.jpg
  inflating: /root/datasets/train/D/1_D_2.jpg
  inflating: /root/datasets/train/D/1_D_3.jpg
  inflating: /root/datasets/train/D/10_D_10.jpg
  inflating: /root/datasets/train/D/10_D_11.jpg
  inflating: /root/datasets/train/D/10_D_12.jpg
  inflating: /root/datasets/train/D/11_D_1.jpg
  inflating: /root/datasets/train/D/11_D_2.jpg
  inflating: /root/datasets/train/D/11_D_3.jpg
  inflating: /root/datasets/train/D/12_D_1.jpg
  inflating: /root/datasets/train/D/12_D_2.jpg
  inflating: /root/datasets/train/D/12_D_3.jpg
  inflating: /root/datasets/train/D/13_D_1.jpg
  inflating: /root/datasets/train/D/13_D_2.jpg
  inflating: /root/datasets/train/D/13_D_3.jpg
```

```
inflating: /root/datasets/train/D/14_D_1.jpg
inflating: /root/datasets/train/D/14_D_2.jpg
inflating: /root/datasets/train/D/14_D_3.jpg
inflating: /root/datasets/train/D/15_D_1.jpg
inflating: /root/datasets/train/D/15_D_2.jpg
inflating: /root/datasets/train/D/15_D_3.jpg
inflating: /root/datasets/train/D/16_D_1.jpg
inflating: /root/datasets/train/D/16_D_2.jpg
inflating: /root/datasets/train/D/16_D_3.jpg
inflating: /root/datasets/train/D/17_D_1.jpg
inflating: /root/datasets/train/D/17_D_2.jpg
inflating: /root/datasets/train/D/17_D_3.jpg
inflating: /root/datasets/train/D/18_D_1.jpg
inflating: /root/datasets/train/D/18_D_2.jpg
inflating: /root/datasets/train/D/18_D_3.jpg
inflating: /root/datasets/train/D/19_D_1.jpg
inflating: /root/datasets/train/D/19_D_2.jpg
inflating: /root/datasets/train/D/19_D_3.jpg
inflating: /root/datasets/train/D/2_D_1.jpg
inflating: /root/datasets/train/D/2_D_2.jpg
inflating: /root/datasets/train/D/2_D_3.jpg
inflating: /root/datasets/train/D/20_D_01.jpg
inflating: /root/datasets/train/D/20_D_02.jpg
inflating: /root/datasets/train/D/20_D_03.jpg
inflating: /root/datasets/train/D/21_D_1.jpg
inflating: /root/datasets/train/D/21_D_2.jpg
inflating: /root/datasets/train/D/21_D_3.jpg
inflating: /root/datasets/train/D/22_D_10.jpg
inflating: /root/datasets/train/D/22_D_11.jpg
inflating: /root/datasets/train/D/22_D_12.jpg
inflating: /root/datasets/train/D/24_D_1.jpg
inflating: /root/datasets/train/D/24_D_2.jpg
inflating: /root/datasets/train/D/24_D_3.jpg
inflating: /root/datasets/train/D/26_D_1.jpg
inflating: /root/datasets/train/D/26_D_2.jpg
inflating: /root/datasets/train/D/26_D_3.jpg
inflating: /root/datasets/train/D/3_D_1.jpg
inflating: /root/datasets/train/D/3_D_2.jpg
inflating: /root/datasets/train/D/3_D_3.jpg
inflating: /root/datasets/train/D/30_D_1.jpg
inflating: /root/datasets/train/D/30_D_2.jpg
inflating: /root/datasets/train/D/30_D_3.jpg
inflating: /root/datasets/train/D/31_D_1.jpg
inflating: /root/datasets/train/D/31_D_2.jpg
inflating: /root/datasets/train/D/31_D_3.jpg
inflating: /root/datasets/train/D/32_D_1.jpg
inflating: /root/datasets/train/D/32_D_2.jpg
inflating: /root/datasets/train/D/32_D_3.jpg
inflating: /root/datasets/train/D/33_D_10.jpg
inflating: /root/datasets/train/D/33_D_11.jpg
inflating: /root/datasets/train/D/33_D_12.jpg
inflating: /root/datasets/train/D/34_D_1.jpg
inflating: /root/datasets/train/D/34_D_2.jpg
inflating: /root/datasets/train/D/34_D_3.jpg
inflating: /root/datasets/train/D/35_D_1.jpg
inflating: /root/datasets/train/D/35_D_2.jpg
inflating: /root/datasets/train/D/35_D_3.jpg
```

```
inflating: /root/datasets/train/D/36_D_1.jpg
inflating: /root/datasets/train/D/36_D_2.jpg
inflating: /root/datasets/train/D/36_D_3.jpg
inflating: /root/datasets/train/D/37_D_1.jpg
inflating: /root/datasets/train/D/37_D_2.jpg
inflating: /root/datasets/train/D/37_D_3.jpg
inflating: /root/datasets/train/D/38_D_1.jpg
inflating: /root/datasets/train/D/38_D_2.jpg
inflating: /root/datasets/train/D/38_D_3.jpg
inflating: /root/datasets/train/D/39_D_1.jpg
inflating: /root/datasets/train/D/39_D_2.jpg
inflating: /root/datasets/train/D/39_D_3.jpg
inflating: /root/datasets/train/D/4_D_1.jpg
inflating: /root/datasets/train/D/4_D_2.jpg
inflating: /root/datasets/train/D/4_D_3.jpg
inflating: /root/datasets/train/D/40_D_1.jpg
inflating: /root/datasets/train/D/40_D_2.jpg
inflating: /root/datasets/train/D/40_D_3.jpg
inflating: /root/datasets/train/D/41_D_1.jpg
inflating: /root/datasets/train/D/41_D_2.jpg
inflating: /root/datasets/train/D/41_D_3.jpg
inflating: /root/datasets/train/D/42_D_1.jpg
inflating: /root/datasets/train/D/42_D_2.jpg
inflating: /root/datasets/train/D/42_D_3.jpg
inflating: /root/datasets/train/D/43_D_1.jpg
inflating: /root/datasets/train/D/43_D_2.jpg
inflating: /root/datasets/train/D/43_D_3.jpg
inflating: /root/datasets/train/D/44_D_1.jpg
inflating: /root/datasets/train/D/44_D_2.jpg
inflating: /root/datasets/train/D/44_D_3.jpg
inflating: /root/datasets/train/D/45_D_1.jpg
inflating: /root/datasets/train/D/45_D_2.jpg
inflating: /root/datasets/train/D/45_D_3.jpg
inflating: /root/datasets/train/D/47_D_1.jpg
inflating: /root/datasets/train/D/47_D_2.jpg
inflating: /root/datasets/train/D/47_D_3.jpg
inflating: /root/datasets/train/D/48_D_1.jpg
inflating: /root/datasets/train/D/48_D_2.jpg
inflating: /root/datasets/train/D/48_D_3.jpg
inflating: /root/datasets/train/D/49_D_1.jpg
inflating: /root/datasets/train/D/49_D_2.jpg
inflating: /root/datasets/train/D/49_D_3.jpg
inflating: /root/datasets/train/D/50_D_1.jpg
inflating: /root/datasets/train/D/50_D_2.jpg
inflating: /root/datasets/train/D/50_D_3.jpg
inflating: /root/datasets/train/D/51_D_1.jpg
inflating: /root/datasets/train/D/51_D_2.jpg
inflating: /root/datasets/train/D/51_D_3.jpg
inflating: /root/datasets/train/D/52_D_1.jpg
inflating: /root/datasets/train/D/52_D_2.jpg
inflating: /root/datasets/train/D/52_D_3.jpg
inflating: /root/datasets/train/D/53_D_1.jpg
inflating: /root/datasets/train/D/53_D_2.jpg
inflating: /root/datasets/train/D/53_D_3.jpg
inflating: /root/datasets/train/D/54_D_1.jpg
inflating: /root/datasets/train/D/54_D_2.jpg
inflating: /root/datasets/train/D/54_D_3.jpg
```

```
  inflating: /root/datasets/train/D/55_D_10.jpg
  inflating: /root/datasets/train/D/55_D_11.jpg
  inflating: /root/datasets/train/D/55_D_12.jpg
  inflating: /root/datasets/train/D/56_D_1.jpg
  inflating: /root/datasets/train/D/56_D_2.jpg
  inflating: /root/datasets/train/D/56_D_3.jpg
  inflating: /root/datasets/train/D/57_D_1.jpg
  inflating: /root/datasets/train/D/57_D_2.jpg
  inflating: /root/datasets/train/D/57_D_3.jpg
  inflating: /root/datasets/train/D/58_D_1.jpg
  inflating: /root/datasets/train/D/58_D_2.jpg
  inflating: /root/datasets/train/D/58_D_3.jpg
  inflating: /root/datasets/train/D/59_D_1.jpg
  inflating: /root/datasets/train/D/59_D_2.jpg
  inflating: /root/datasets/train/D/59_D_3.jpg
  inflating: /root/datasets/train/D/6_D_1.jpg
  inflating: /root/datasets/train/D/6_D_2.jpg
  inflating: /root/datasets/train/D/6_D_3.jpg
  inflating: /root/datasets/train/D/60_D_1.jpg
  inflating: /root/datasets/train/D/60_D_2.jpg
  inflating: /root/datasets/train/D/60_D_3.jpg
  inflating: /root/datasets/train/D/61_D_1.jpg
  inflating: /root/datasets/train/D/61_D_2.jpg
  inflating: /root/datasets/train/D/61_D_3.jpg
  inflating: /root/datasets/train/D/62_D_1.jpg
  inflating: /root/datasets/train/D/62_D_2.jpg
  inflating: /root/datasets/train/D/62_D_3.jpg
  inflating: /root/datasets/train/D/7_D_1.jpg
  inflating: /root/datasets/train/D/7_D_2.jpg
  inflating: /root/datasets/train/D/7_D_3.jpg
  inflating: /root/datasets/train/D/8_D_1.jpg
  inflating: /root/datasets/train/D/8_D_2.jpg
  inflating: /root/datasets/train/D/8_D_3.jpg
  inflating: /root/datasets/train/D/9_D_1.jpg
  inflating: /root/datasets/train/D/9_D_2.jpg
  inflating: /root/datasets/train/D/9_D_3.jpg
   creating: /root/datasets/train/E/
  inflating: /root/datasets/train/E/1_E_1.jpg
  inflating: /root/datasets/train/E/1_E_2.jpg
  inflating: /root/datasets/train/E/1_E_3.jpg
  inflating: /root/datasets/train/E/10_E_13.jpg
  inflating: /root/datasets/train/E/10_E_14.jpg
  inflating: /root/datasets/train/E/10_E_15.jpg
  inflating: /root/datasets/train/E/11_E_1.jpg
  inflating: /root/datasets/train/E/11_E_2.jpg
  inflating: /root/datasets/train/E/11_E_3.jpg
  inflating: /root/datasets/train/E/12_E_1.jpg
  inflating: /root/datasets/train/E/12_E_2.jpg
  inflating: /root/datasets/train/E/12_E_3.jpg
  inflating: /root/datasets/train/E/13_E_1.jpg
  inflating: /root/datasets/train/E/13_E_2.jpg
  inflating: /root/datasets/train/E/13_E_3.jpg
  inflating: /root/datasets/train/E/14_E_1.jpg
  inflating: /root/datasets/train/E/14_E_2.jpg
  inflating: /root/datasets/train/E/14_E_3.jpg
  inflating: /root/datasets/train/E/15_E_1.jpg
  inflating: /root/datasets/train/E/15_E_2.jpg
```

```
inflating: /root/datasets/train/E/15_E_3.jpg
inflating: /root/datasets/train/E/16_E_1.jpg
inflating: /root/datasets/train/E/16_E_2.jpg
inflating: /root/datasets/train/E/16_E_3.jpg
inflating: /root/datasets/train/E/17_E_1.jpg
inflating: /root/datasets/train/E/17_E_2.jpg
inflating: /root/datasets/train/E/17_E_3.jpg
inflating: /root/datasets/train/E/18_E_1.jpg
inflating: /root/datasets/train/E/18_E_2.jpg
inflating: /root/datasets/train/E/18_E_3.jpg
inflating: /root/datasets/train/E/19_E_1.jpg
inflating: /root/datasets/train/E/19_E_2.jpg
inflating: /root/datasets/train/E/19_E_3.jpg
inflating: /root/datasets/train/E/2_E_1.jpg
inflating: /root/datasets/train/E/2_E_2.jpg
inflating: /root/datasets/train/E/2_E_3.jpg
inflating: /root/datasets/train/E/20_E_01.jpg
inflating: /root/datasets/train/E/20_E_02.jpg
inflating: /root/datasets/train/E/20_E_03.jpg
inflating: /root/datasets/train/E/21_E_1.jpg
inflating: /root/datasets/train/E/21_E_2.jpg
inflating: /root/datasets/train/E/21_E_3.jpg
inflating: /root/datasets/train/E/22_E_13.jpg
inflating: /root/datasets/train/E/22_E_14.jpg
inflating: /root/datasets/train/E/22_E_15.jpg
inflating: /root/datasets/train/E/24_E_1.jpg
inflating: /root/datasets/train/E/24_E_2.jpg
inflating: /root/datasets/train/E/24_E_3.jpg
inflating: /root/datasets/train/E/26_E_1.jpg
inflating: /root/datasets/train/E/26_E_2.jpg
inflating: /root/datasets/train/E/26_E_3.jpg
inflating: /root/datasets/train/E/3_E_1.jpg
inflating: /root/datasets/train/E/3_E_2.jpg
inflating: /root/datasets/train/E/3_E_3.jpg
inflating: /root/datasets/train/E/30_E_1.jpg
inflating: /root/datasets/train/E/30_E_2.jpg
inflating: /root/datasets/train/E/30_E_3.jpg
inflating: /root/datasets/train/E/31_E_1.jpg
inflating: /root/datasets/train/E/31_E_2.jpg
inflating: /root/datasets/train/E/31_E_3.jpg
inflating: /root/datasets/train/E/32_E_1.jpg
inflating: /root/datasets/train/E/32_E_2.jpg
inflating: /root/datasets/train/E/32_E_3.jpg
inflating: /root/datasets/train/E/33_E_13.jpg
inflating: /root/datasets/train/E/33_E_14.jpg
inflating: /root/datasets/train/E/33_E_15.jpg
inflating: /root/datasets/train/E/34_E_1.jpg
inflating: /root/datasets/train/E/34_E_2.jpg
inflating: /root/datasets/train/E/34_E_3.jpg
inflating: /root/datasets/train/E/35_E_1.jpg
inflating: /root/datasets/train/E/35_E_2.jpg
inflating: /root/datasets/train/E/35_E_3.jpg
inflating: /root/datasets/train/E/36_E_1.jpg
inflating: /root/datasets/train/E/36_E_2.jpg
inflating: /root/datasets/train/E/36_E_3.jpg
inflating: /root/datasets/train/E/37_E_1.jpg
inflating: /root/datasets/train/E/37_E_2.jpg
```

```
inflating: /root/datasets/train/E/37_E_3.jpg
inflating: /root/datasets/train/E/38_E_1.jpg
inflating: /root/datasets/train/E/38_E_2.jpg
inflating: /root/datasets/train/E/38_E_3.jpg
inflating: /root/datasets/train/E/39_E_1.jpg
inflating: /root/datasets/train/E/39_E_2.jpg
inflating: /root/datasets/train/E/39_E_3.jpg
inflating: /root/datasets/train/E/4_E_1.jpg
inflating: /root/datasets/train/E/4_E_2.jpg
inflating: /root/datasets/train/E/4_E_3.jpg
inflating: /root/datasets/train/E/40_E_1.jpg
inflating: /root/datasets/train/E/40_E_2.jpg
inflating: /root/datasets/train/E/40_E_3.jpg
inflating: /root/datasets/train/E/41_E_1.jpg
inflating: /root/datasets/train/E/41_E_2.jpg
inflating: /root/datasets/train/E/41_E_3.jpg
inflating: /root/datasets/train/E/42_E_1.jpg
inflating: /root/datasets/train/E/42_E_2.jpg
inflating: /root/datasets/train/E/42_E_3.jpg
inflating: /root/datasets/train/E/43_E_1.jpg
inflating: /root/datasets/train/E/43_E_2.jpg
inflating: /root/datasets/train/E/43_E_3.jpg
inflating: /root/datasets/train/E/44_E_1.jpg
inflating: /root/datasets/train/E/44_E_2.jpg
inflating: /root/datasets/train/E/44_E_3.jpg
inflating: /root/datasets/train/E/45_E_1.jpg
inflating: /root/datasets/train/E/45_E_2.jpg
inflating: /root/datasets/train/E/45_E_3.jpg
inflating: /root/datasets/train/E/47_E_1.jpg
inflating: /root/datasets/train/E/47_E_2.jpg
inflating: /root/datasets/train/E/47_E_3.jpg
inflating: /root/datasets/train/E/48_E_1.jpg
inflating: /root/datasets/train/E/48_E_2.jpg
inflating: /root/datasets/train/E/48_E_3.jpg
inflating: /root/datasets/train/E/49_E_1.jpg
inflating: /root/datasets/train/E/49_E_2.jpg
inflating: /root/datasets/train/E/49_E_3.jpg
inflating: /root/datasets/train/E/50_E_1.jpg
inflating: /root/datasets/train/E/50_E_2.jpg
inflating: /root/datasets/train/E/50_E_3.jpg
inflating: /root/datasets/train/E/51_E_1.jpg
inflating: /root/datasets/train/E/51_E_2.jpg
inflating: /root/datasets/train/E/51_E_3.jpg
inflating: /root/datasets/train/E/52_E_1.jpg
inflating: /root/datasets/train/E/52_E_2.jpg
inflating: /root/datasets/train/E/52_E_3.jpg
inflating: /root/datasets/train/E/53_E_1.jpg
inflating: /root/datasets/train/E/53_E_2.jpg
inflating: /root/datasets/train/E/53_E_3.jpg
inflating: /root/datasets/train/E/54_E_1.jpg
inflating: /root/datasets/train/E/54_E_2.jpg
inflating: /root/datasets/train/E/54_E_3.jpg
inflating: /root/datasets/train/E/55_E_13.jpg
inflating: /root/datasets/train/E/55_E_14.jpg
inflating: /root/datasets/train/E/55_E_15.jpg
inflating: /root/datasets/train/E/56_E_1.jpg
inflating: /root/datasets/train/E/56_E_2.jpg
```

```
inflating: /root/datasets/train/E/56_E_3.jpg
inflating: /root/datasets/train/E/57_E_1.jpg
inflating: /root/datasets/train/E/57_E_2.jpg
inflating: /root/datasets/train/E/57_E_3.jpg
inflating: /root/datasets/train/E/58_E_1.jpg
inflating: /root/datasets/train/E/58_E_2.jpg
inflating: /root/datasets/train/E/58_E_3.jpg
inflating: /root/datasets/train/E/59_E_1.jpg
inflating: /root/datasets/train/E/59_E_2.jpg
inflating: /root/datasets/train/E/59_E_3.jpg
inflating: /root/datasets/train/E/6_E_1.jpg
inflating: /root/datasets/train/E/6_E_2.jpg
inflating: /root/datasets/train/E/6_E_3.jpg
inflating: /root/datasets/train/E/60_E_1.jpg
inflating: /root/datasets/train/E/60_E_2.jpg
inflating: /root/datasets/train/E/60_E_3.jpg
inflating: /root/datasets/train/E/61_E_1.jpg
inflating: /root/datasets/train/E/61_E_2.jpg
inflating: /root/datasets/train/E/61_E_3.jpg
inflating: /root/datasets/train/E/62_E_2.jpg
inflating: /root/datasets/train/E/62_E_3.jpg
inflating: /root/datasets/train/E/7_E_1.jpg
inflating: /root/datasets/train/E/7_E_2.jpg
inflating: /root/datasets/train/E/7_E_3.jpg
inflating: /root/datasets/train/E/8_E_1.jpg
inflating: /root/datasets/train/E/8_E_2.jpg
inflating: /root/datasets/train/E/8_E_3.jpg
inflating: /root/datasets/train/E/9_E_1.jpg
inflating: /root/datasets/train/E/9_E_2.jpg
inflating: /root/datasets/train/E/9_E_3.jpg
 creating: /root/datasets/train/F/
inflating: /root/datasets/train/F/1_F_1.jpg
inflating: /root/datasets/train/F/1_F_2.jpg
inflating: /root/datasets/train/F/1_F_3.jpg
inflating: /root/datasets/train/F/10_F_16.jpg
inflating: /root/datasets/train/F/10_F_17.jpg
inflating: /root/datasets/train/F/10_F_18.jpg
inflating: /root/datasets/train/F/11_F_1.jpg
inflating: /root/datasets/train/F/11_F_2.jpg
inflating: /root/datasets/train/F/11_F_3.jpg
inflating: /root/datasets/train/F/12_F_1.jpg
inflating: /root/datasets/train/F/12_F_2.jpg
inflating: /root/datasets/train/F/12_F_3.jpg
inflating: /root/datasets/train/F/13_F_1.jpg
inflating: /root/datasets/train/F/13_F_2.jpg
inflating: /root/datasets/train/F/13_F_3.jpg
inflating: /root/datasets/train/F/14_F_1.jpg
inflating: /root/datasets/train/F/14_F_2.jpg
inflating: /root/datasets/train/F/14_F_3.jpg
inflating: /root/datasets/train/F/15_F_1.jpg
inflating: /root/datasets/train/F/15_F_2.jpg
inflating: /root/datasets/train/F/15_F_3.jpg
inflating: /root/datasets/train/F/16_F_1.jpg
inflating: /root/datasets/train/F/16_F_2.jpg
inflating: /root/datasets/train/F/16_F_3.jpg
inflating: /root/datasets/train/F/17_F_1.jpg
inflating: /root/datasets/train/F/17_F_2.jpg
```

```
inflating: /root/datasets/train/F/17_F_3.jpg
inflating: /root/datasets/train/F/18_F_1.jpg
inflating: /root/datasets/train/F/18_F_2.jpg
inflating: /root/datasets/train/F/18_F_3.jpg
inflating: /root/datasets/train/F/19_F_1.jpg
inflating: /root/datasets/train/F/19_F_2.jpg
inflating: /root/datasets/train/F/19_F_3.jpg
inflating: /root/datasets/train/F/2_F_1.jpg
inflating: /root/datasets/train/F/2_F_2.jpg
inflating: /root/datasets/train/F/2_F_3.jpg
inflating: /root/datasets/train/F/20_F_01.jpg
inflating: /root/datasets/train/F/20_F_02.jpg
inflating: /root/datasets/train/F/20_F_03.jpg
inflating: /root/datasets/train/F/21_F_1.jpg
inflating: /root/datasets/train/F/21_F_2.jpg
inflating: /root/datasets/train/F/21_F_3.jpg
inflating: /root/datasets/train/F/22_F_16.jpg
inflating: /root/datasets/train/F/22_F_17.jpg
inflating: /root/datasets/train/F/22_F_18.jpg
inflating: /root/datasets/train/F/24_F_1.jpg
inflating: /root/datasets/train/F/24_F_2.jpg
inflating: /root/datasets/train/F/24_F_3.jpg
inflating: /root/datasets/train/F/26_F_1.jpg
inflating: /root/datasets/train/F/26_F_2.jpg
inflating: /root/datasets/train/F/26_F_3.jpg
inflating: /root/datasets/train/F/3_F_1.jpg
inflating: /root/datasets/train/F/3_F_2.jpg
inflating: /root/datasets/train/F/3_F_3.jpg
inflating: /root/datasets/train/F/30_F_1.jpg
inflating: /root/datasets/train/F/30_F_2.jpg
inflating: /root/datasets/train/F/30_F_3.jpg
inflating: /root/datasets/train/F/31_F_1.jpg
inflating: /root/datasets/train/F/31_F_2.jpg
inflating: /root/datasets/train/F/31_F_3.jpg
inflating: /root/datasets/train/F/32_F_1.jpg
inflating: /root/datasets/train/F/32_F_2.jpg
inflating: /root/datasets/train/F/32_F_3.jpg
inflating: /root/datasets/train/F/33_F_16.jpg
inflating: /root/datasets/train/F/33_F_17.jpg
inflating: /root/datasets/train/F/33_F_18.jpg
inflating: /root/datasets/train/F/34_F_1.jpg
inflating: /root/datasets/train/F/34_F_2.jpg
inflating: /root/datasets/train/F/34_F_3.jpg
inflating: /root/datasets/train/F/35_F_1.jpg
inflating: /root/datasets/train/F/35_F_2.jpg
inflating: /root/datasets/train/F/35_F_3.jpg
inflating: /root/datasets/train/F/36_F_1.jpg
inflating: /root/datasets/train/F/36_F_2.jpg
inflating: /root/datasets/train/F/36_F_3.jpg
inflating: /root/datasets/train/F/37_F_1.jpg
inflating: /root/datasets/train/F/37_F_2.jpg
inflating: /root/datasets/train/F/37_F_3.jpg
inflating: /root/datasets/train/F/38_F_1.jpg
inflating: /root/datasets/train/F/38_F_2.jpg
inflating: /root/datasets/train/F/38_F_3.jpg
inflating: /root/datasets/train/F/39_F_1.jpg
inflating: /root/datasets/train/F/39_F_2.jpg
```

```
inflating: /root/datasets/train/F/39_F_3.jpg
inflating: /root/datasets/train/F/4_F_1.jpg
inflating: /root/datasets/train/F/4_F_2.jpg
inflating: /root/datasets/train/F/4_F_3.jpg
inflating: /root/datasets/train/F/40_F_1.jpg
inflating: /root/datasets/train/F/40_F_2.jpg
inflating: /root/datasets/train/F/40_F_3.jpg
inflating: /root/datasets/train/F/41_F_1.jpg
inflating: /root/datasets/train/F/41_F_2.jpg
inflating: /root/datasets/train/F/41_F_3.jpg
inflating: /root/datasets/train/F/42_F_1.jpg
inflating: /root/datasets/train/F/42_F_2.jpg
inflating: /root/datasets/train/F/42_F_3.jpg
inflating: /root/datasets/train/F/43_F_1.jpg
inflating: /root/datasets/train/F/43_F_2.jpg
inflating: /root/datasets/train/F/43_F_3.jpg
inflating: /root/datasets/train/F/44_F_1.jpg
inflating: /root/datasets/train/F/44_F_2.jpg
inflating: /root/datasets/train/F/44_F_3.jpg
inflating: /root/datasets/train/F/45_F_1.jpg
inflating: /root/datasets/train/F/45_F_2.jpg
inflating: /root/datasets/train/F/45_F_3.jpg
inflating: /root/datasets/train/F/47_F_1.jpg
inflating: /root/datasets/train/F/47_F_2.jpg
inflating: /root/datasets/train/F/47_F_3.jpg
inflating: /root/datasets/train/F/48_F_1.jpg
inflating: /root/datasets/train/F/48_F_2.jpg
inflating: /root/datasets/train/F/48_F_3.jpg
inflating: /root/datasets/train/F/49_F_1.jpg
inflating: /root/datasets/train/F/49_F_2.jpg
inflating: /root/datasets/train/F/49_F_3.jpg
inflating: /root/datasets/train/F/50_F_1.jpg
inflating: /root/datasets/train/F/50_F_2.jpg
inflating: /root/datasets/train/F/50_F_3.jpg
inflating: /root/datasets/train/F/51_F_1.jpg
inflating: /root/datasets/train/F/51_F_2.jpg
inflating: /root/datasets/train/F/51_F_3.jpg
inflating: /root/datasets/train/F/52_F_1.jpg
inflating: /root/datasets/train/F/52_F_2.jpg
inflating: /root/datasets/train/F/52_F_3.jpg
inflating: /root/datasets/train/F/53_F_1.jpg
inflating: /root/datasets/train/F/53_F_2.jpg
inflating: /root/datasets/train/F/53_F_3.jpg
inflating: /root/datasets/train/F/54_F_1.jpg
inflating: /root/datasets/train/F/54_F_2.jpg
inflating: /root/datasets/train/F/54_F_3.jpg
inflating: /root/datasets/train/F/55_F_16.jpg
inflating: /root/datasets/train/F/55_F_17.jpg
inflating: /root/datasets/train/F/55_F_18.jpg
inflating: /root/datasets/train/F/56_F_1.jpg
inflating: /root/datasets/train/F/56_F_2.jpg
inflating: /root/datasets/train/F/56_F_3.jpg
inflating: /root/datasets/train/F/57_F_1.jpg
inflating: /root/datasets/train/F/57_F_2.jpg
inflating: /root/datasets/train/F/57_F_3.jpg
inflating: /root/datasets/train/F/58_F_1.jpg
inflating: /root/datasets/train/F/58_F_2.jpg
```

```
inflating: /root/datasets/train/F/58_F_3.jpg
inflating: /root/datasets/train/F/59_F_1.jpg
inflating: /root/datasets/train/F/59_F_2.jpg
inflating: /root/datasets/train/F/59_F_3.jpg
inflating: /root/datasets/train/F/6_F_1.jpg
inflating: /root/datasets/train/F/6_F_2.jpg
inflating: /root/datasets/train/F/6_F_3.jpg
inflating: /root/datasets/train/F/60_F_1.jpg
inflating: /root/datasets/train/F/60_F_2.jpg
inflating: /root/datasets/train/F/60_F_3.jpg
inflating: /root/datasets/train/F/61_F_1.jpg
inflating: /root/datasets/train/F/61_F_2.jpg
inflating: /root/datasets/train/F/61_F_3.jpg
inflating: /root/datasets/train/F/62_F_1.jpg
inflating: /root/datasets/train/F/62_F_2.jpg
inflating: /root/datasets/train/F/62_F_3.jpg
inflating: /root/datasets/train/F/7_F_1.jpg
inflating: /root/datasets/train/F/7_F_2.jpg
inflating: /root/datasets/train/F/7_F_3.jpg
inflating: /root/datasets/train/F/8_F_1.jpg
inflating: /root/datasets/train/F/8_F_2.jpg
inflating: /root/datasets/train/F/8_F_3.jpg
inflating: /root/datasets/train/F/9_F_1.jpg
inflating: /root/datasets/train/F/9_F_2.jpg
inflating: /root/datasets/train/F/9_F_3.jpg
 creating: /root/datasets/train/G/
inflating: /root/datasets/train/G/1_G_1.jpg
inflating: /root/datasets/train/G/1_G_2.jpg
inflating: /root/datasets/train/G/1_G_3.jpg
inflating: /root/datasets/train/G/10_G_19.jpg
inflating: /root/datasets/train/G/10_G_20.jpg
inflating: /root/datasets/train/G/10_G_21.jpg
inflating: /root/datasets/train/G/11_G_1.jpg
inflating: /root/datasets/train/G/11_G_2.jpg
inflating: /root/datasets/train/G/11_G_3.jpg
inflating: /root/datasets/train/G/12_G_1.jpg
inflating: /root/datasets/train/G/12_G_2.jpg
inflating: /root/datasets/train/G/12_G_3.jpg
inflating: /root/datasets/train/G/13_G_1.jpg
inflating: /root/datasets/train/G/13_G_2.jpg
inflating: /root/datasets/train/G/13_G_3.jpg
inflating: /root/datasets/train/G/14_G_1.jpg
inflating: /root/datasets/train/G/14_G_2.jpg
inflating: /root/datasets/train/G/14_G_3.jpg
inflating: /root/datasets/train/G/15_G_1.jpg
inflating: /root/datasets/train/G/15_G_2.jpg
inflating: /root/datasets/train/G/15_G_3.jpg
inflating: /root/datasets/train/G/16_G_1.jpg
inflating: /root/datasets/train/G/16_G_2.jpg
inflating: /root/datasets/train/G/16_G_3.jpg
inflating: /root/datasets/train/G/17_G_1.jpg
inflating: /root/datasets/train/G/17_G_2.jpg
inflating: /root/datasets/train/G/17_G_3.jpg
inflating: /root/datasets/train/G/18_G_1.jpg
inflating: /root/datasets/train/G/18_G_2.jpg
inflating: /root/datasets/train/G/18_G_3.jpg
inflating: /root/datasets/train/G/19_G_1.jpg
```

```
inflating: /root/datasets/train/G/19_G_2.jpg
inflating: /root/datasets/train/G/19_G_3.jpg
inflating: /root/datasets/train/G/2_G_1.jpg
inflating: /root/datasets/train/G/2_G_2.jpg
inflating: /root/datasets/train/G/2_G_3.jpg
inflating: /root/datasets/train/G/20_G_01.jpg
inflating: /root/datasets/train/G/20_G_02.jpg
inflating: /root/datasets/train/G/20_G_03.jpg
inflating: /root/datasets/train/G/21_G_1.jpg
inflating: /root/datasets/train/G/21_G_2.jpg
inflating: /root/datasets/train/G/21_G_3.jpg
inflating: /root/datasets/train/G/22_G_19.jpg
inflating: /root/datasets/train/G/22_G_20.jpg
inflating: /root/datasets/train/G/22_G_21.jpg
inflating: /root/datasets/train/G/24_G_1.jpg
inflating: /root/datasets/train/G/24_G_2.jpg
inflating: /root/datasets/train/G/24_G_3.jpg
inflating: /root/datasets/train/G/26_G_1.jpg
inflating: /root/datasets/train/G/26_G_2.jpg
inflating: /root/datasets/train/G/26_G_3.jpg
inflating: /root/datasets/train/G/3_G_1.jpg
inflating: /root/datasets/train/G/3_G_2.jpg
inflating: /root/datasets/train/G/3_G_3.jpg
inflating: /root/datasets/train/G/30_G_1.jpg
inflating: /root/datasets/train/G/30_G_2.jpg
inflating: /root/datasets/train/G/30_G_3.jpg
inflating: /root/datasets/train/G/31_G_1.jpg
inflating: /root/datasets/train/G/31_G_2.jpg
inflating: /root/datasets/train/G/31_G_3.jpg
inflating: /root/datasets/train/G/32_G_1.jpg
inflating: /root/datasets/train/G/32_G_2.jpg
inflating: /root/datasets/train/G/32_G_3.jpg
inflating: /root/datasets/train/G/33_G_19.jpg
inflating: /root/datasets/train/G/33_G_20.jpg
inflating: /root/datasets/train/G/33_G_21.jpg
inflating: /root/datasets/train/G/34_G_1.jpg
inflating: /root/datasets/train/G/34_G_2.jpg
inflating: /root/datasets/train/G/34_G_3.jpg
inflating: /root/datasets/train/G/35_G_1.jpg
inflating: /root/datasets/train/G/35_G_2.jpg
inflating: /root/datasets/train/G/35_G_3.jpg
inflating: /root/datasets/train/G/36_G_1.jpg
inflating: /root/datasets/train/G/36_G_2.jpg
inflating: /root/datasets/train/G/36_G_3.jpg
inflating: /root/datasets/train/G/37_G_1.jpg
inflating: /root/datasets/train/G/37_G_2.jpg
inflating: /root/datasets/train/G/37_G_3.jpg
inflating: /root/datasets/train/G/38_G_1.jpg
inflating: /root/datasets/train/G/38_G_2.jpg
inflating: /root/datasets/train/G/38_G_3.jpg
inflating: /root/datasets/train/G/39_G_1.jpg
inflating: /root/datasets/train/G/39_G_2.jpg
inflating: /root/datasets/train/G/39_G_3.jpg
inflating: /root/datasets/train/G/4_G_1.jpg
inflating: /root/datasets/train/G/4_G_2.jpg
inflating: /root/datasets/train/G/4_G_3.jpg
inflating: /root/datasets/train/G/40_G_1.jpg
```

```
inflating: /root/datasets/train/G/40_G_2.jpg
inflating: /root/datasets/train/G/40_G_3.jpg
inflating: /root/datasets/train/G/41_G_1.jpg
inflating: /root/datasets/train/G/41_G_2.jpg
inflating: /root/datasets/train/G/41_G_3.jpg
inflating: /root/datasets/train/G/42_G_1.jpg
inflating: /root/datasets/train/G/42_G_2.jpg
inflating: /root/datasets/train/G/42_G_3.jpg
inflating: /root/datasets/train/G/43_G_1.jpg
inflating: /root/datasets/train/G/43_G_2.jpg
inflating: /root/datasets/train/G/43_G_3.jpg
inflating: /root/datasets/train/G/44_G_1.jpg
inflating: /root/datasets/train/G/44_G_2.jpg
inflating: /root/datasets/train/G/44_G_3.jpg
inflating: /root/datasets/train/G/45_G_1.jpg
inflating: /root/datasets/train/G/45_G_2.jpg
inflating: /root/datasets/train/G/45_G_3.jpg
inflating: /root/datasets/train/G/47_G_1.jpg
inflating: /root/datasets/train/G/47_G_2.jpg
inflating: /root/datasets/train/G/47_G_3.jpg
inflating: /root/datasets/train/G/48_G_1.jpg
inflating: /root/datasets/train/G/48_G_2.jpg
inflating: /root/datasets/train/G/48_G_3.jpg
inflating: /root/datasets/train/G/49_G_1.jpg
inflating: /root/datasets/train/G/49_G_2.jpg
inflating: /root/datasets/train/G/49_G_3.jpg
inflating: /root/datasets/train/G/50_G_1.jpg
inflating: /root/datasets/train/G/50_G_2.jpg
inflating: /root/datasets/train/G/50_G_3.jpg
inflating: /root/datasets/train/G/51_G_1.jpg
inflating: /root/datasets/train/G/51_G_2.jpg
inflating: /root/datasets/train/G/51_G_3.jpg
inflating: /root/datasets/train/G/52_G_1.jpg
inflating: /root/datasets/train/G/52_G_2.jpg
inflating: /root/datasets/train/G/52_G_3.jpg
inflating: /root/datasets/train/G/53_G_1.jpg
inflating: /root/datasets/train/G/53_G_2.jpg
inflating: /root/datasets/train/G/53_G_3.jpg
inflating: /root/datasets/train/G/54_G_1.jpg
inflating: /root/datasets/train/G/54_G_2.jpg
inflating: /root/datasets/train/G/54_G_3.jpg
inflating: /root/datasets/train/G/55_G_19.jpg
inflating: /root/datasets/train/G/55_G_20.jpg
inflating: /root/datasets/train/G/55_G_21.jpg
inflating: /root/datasets/train/G/56_G_1.jpg
inflating: /root/datasets/train/G/56_G_2.jpg
inflating: /root/datasets/train/G/56_G_3.jpg
inflating: /root/datasets/train/G/57_G_1.jpg
inflating: /root/datasets/train/G/57_G_2.jpg
inflating: /root/datasets/train/G/57_G_3.jpg
inflating: /root/datasets/train/G/58_G_1.jpg
inflating: /root/datasets/train/G/58_G_2.jpg
inflating: /root/datasets/train/G/58_G_3.jpg
inflating: /root/datasets/train/G/59_G_1.jpg
inflating: /root/datasets/train/G/59_G_2.jpg
inflating: /root/datasets/train/G/59_G_3.jpg
inflating: /root/datasets/train/G/6_G_1.jpg
```

```
inflating: /root/datasets/train/G/6_G_2.jpg
inflating: /root/datasets/train/G/6_G_3.jpg
inflating: /root/datasets/train/G/60_G_1.jpg
inflating: /root/datasets/train/G/60_G_2.jpg
inflating: /root/datasets/train/G/60_G_3.jpg
inflating: /root/datasets/train/G/61_G_1.jpg
inflating: /root/datasets/train/G/61_G_2.jpg
inflating: /root/datasets/train/G/61_G_3.jpg
inflating: /root/datasets/train/G/62_G_1.jpg
inflating: /root/datasets/train/G/62_G_2.jpg
inflating: /root/datasets/train/G/62_G_3.jpg
inflating: /root/datasets/train/G/7_G_1.jpg
inflating: /root/datasets/train/G/7_G_2.jpg
inflating: /root/datasets/train/G/7_G_3.jpg
inflating: /root/datasets/train/G/8_G_1.jpg
inflating: /root/datasets/train/G/8_G_2.jpg
inflating: /root/datasets/train/G/8_G_3.jpg
inflating: /root/datasets/train/G/9_G_1.jpg
inflating: /root/datasets/train/G/9_G_2.jpg
inflating: /root/datasets/train/G/9_G_3.jpg
 creating: /root/datasets/train/H/
inflating: /root/datasets/train/H/1_H_1.jpg
inflating: /root/datasets/train/H/1_H_2.jpg
inflating: /root/datasets/train/H/1_H_3.jpg
inflating: /root/datasets/train/H/10_H_22.jpg
inflating: /root/datasets/train/H/10_H_23.jpg
inflating: /root/datasets/train/H/10_H_24.jpg
inflating: /root/datasets/train/H/11_H_1.jpg
inflating: /root/datasets/train/H/11_H_2.jpg
inflating: /root/datasets/train/H/11_H_3.jpg
inflating: /root/datasets/train/H/12_H_1.jpg
inflating: /root/datasets/train/H/12_H_2.jpg
inflating: /root/datasets/train/H/12_H_3.jpg
inflating: /root/datasets/train/H/13_H_1.jpg
inflating: /root/datasets/train/H/13_H_2.jpg
inflating: /root/datasets/train/H/13_H_3.jpg
inflating: /root/datasets/train/H/14_H_1.jpg
inflating: /root/datasets/train/H/14_H_2.jpg
inflating: /root/datasets/train/H/14_H_3.jpg
inflating: /root/datasets/train/H/15_H_1.jpg
inflating: /root/datasets/train/H/15_H_2.jpg
inflating: /root/datasets/train/H/15_H_3.jpg
inflating: /root/datasets/train/H/16_H_1.jpg
inflating: /root/datasets/train/H/16_H_2.jpg
inflating: /root/datasets/train/H/16_H_3.jpg
inflating: /root/datasets/train/H/17_H_1.jpg
inflating: /root/datasets/train/H/17_H_2.jpg
inflating: /root/datasets/train/H/17_H_3.jpg
inflating: /root/datasets/train/H/18_H_1.jpg
inflating: /root/datasets/train/H/18_H_2.jpg
inflating: /root/datasets/train/H/18_H_3.jpg
inflating: /root/datasets/train/H/19_H_1.jpg
inflating: /root/datasets/train/H/19_H_2.jpg
inflating: /root/datasets/train/H/19_H_3.jpg
inflating: /root/datasets/train/H/2_H_1.jpg
inflating: /root/datasets/train/H/2_H_2.jpg
inflating: /root/datasets/train/H/2_H_3.jpg
```

```
inflating: /root/datasets/train/H/20_H_01.jpg
inflating: /root/datasets/train/H/20_H_02.jpg
inflating: /root/datasets/train/H/20_H_03.jpg
inflating: /root/datasets/train/H/21_H_1.jpg
inflating: /root/datasets/train/H/21_H_2.jpg
inflating: /root/datasets/train/H/21_H_3.jpg
inflating: /root/datasets/train/H/22_H_22.jpg
inflating: /root/datasets/train/H/22_H_23.jpg
inflating: /root/datasets/train/H/22_H_24.jpg
inflating: /root/datasets/train/H/24_H_1.jpg
inflating: /root/datasets/train/H/24_H_2.jpg
inflating: /root/datasets/train/H/24_H_3.jpg
inflating: /root/datasets/train/H/26_H_1.jpg
inflating: /root/datasets/train/H/26_H_2.jpg
inflating: /root/datasets/train/H/26_H_3.jpg
inflating: /root/datasets/train/H/3_H_1.jpg
inflating: /root/datasets/train/H/3_H_2.jpg
inflating: /root/datasets/train/H/3_H_3.jpg
inflating: /root/datasets/train/H/30_H_1.jpg
inflating: /root/datasets/train/H/30_H_2.jpg
inflating: /root/datasets/train/H/30_H_3.jpg
inflating: /root/datasets/train/H/31_H_1.jpg
inflating: /root/datasets/train/H/31_H_2.jpg
inflating: /root/datasets/train/H/31_H_3.jpg
inflating: /root/datasets/train/H/32_H_1.jpg
inflating: /root/datasets/train/H/32_H_2.jpg
inflating: /root/datasets/train/H/32_H_3.jpg
inflating: /root/datasets/train/H/33_H_22.jpg
inflating: /root/datasets/train/H/33_H_23.jpg
inflating: /root/datasets/train/H/33_H_24.jpg
inflating: /root/datasets/train/H/34_H_1.jpg
inflating: /root/datasets/train/H/34_H_2.jpg
inflating: /root/datasets/train/H/34_H_3.jpg
inflating: /root/datasets/train/H/35_H_1.jpg
inflating: /root/datasets/train/H/35_H_2.jpg
inflating: /root/datasets/train/H/35_H_3.jpg
inflating: /root/datasets/train/H/36_H_1.jpg
inflating: /root/datasets/train/H/36_H_2.jpg
inflating: /root/datasets/train/H/36_H_3.jpg
inflating: /root/datasets/train/H/37_H_1.jpg
inflating: /root/datasets/train/H/37_H_2.jpg
inflating: /root/datasets/train/H/37_H_3.jpg
inflating: /root/datasets/train/H/38_H_1.jpg
inflating: /root/datasets/train/H/38_H_2.jpg
inflating: /root/datasets/train/H/38_H_3.jpg
inflating: /root/datasets/train/H/39_H_1.jpg
inflating: /root/datasets/train/H/39_H_2.jpg
inflating: /root/datasets/train/H/39_H_3.jpg
inflating: /root/datasets/train/H/4_H_1.jpg
inflating: /root/datasets/train/H/4_H_2.jpg
inflating: /root/datasets/train/H/4_H_3.jpg
inflating: /root/datasets/train/H/40_H_1.jpg
inflating: /root/datasets/train/H/40_H_2.jpg
inflating: /root/datasets/train/H/40_H_3.jpg
inflating: /root/datasets/train/H/41_H_1.jpg
inflating: /root/datasets/train/H/41_H_2.jpg
inflating: /root/datasets/train/H/41_H_3.jpg
```

```
inflating: /root/datasets/train/H/42_H_1.jpg
inflating: /root/datasets/train/H/42_H_2.jpg
inflating: /root/datasets/train/H/42_H_3.jpg
inflating: /root/datasets/train/H/43_H_1.jpg
inflating: /root/datasets/train/H/43_H_2.jpg
inflating: /root/datasets/train/H/43_H_3.jpg
inflating: /root/datasets/train/H/44_H_1.jpg
inflating: /root/datasets/train/H/44_H_2.jpg
inflating: /root/datasets/train/H/44_H_3.jpg
inflating: /root/datasets/train/H/45_H_1.jpg
inflating: /root/datasets/train/H/45_H_2.jpg
inflating: /root/datasets/train/H/45_H_3.jpg
inflating: /root/datasets/train/H/47_H_1.jpg
inflating: /root/datasets/train/H/47_H_2.jpg
inflating: /root/datasets/train/H/47_H_3.jpg
inflating: /root/datasets/train/H/48_H_1.jpg
inflating: /root/datasets/train/H/48_H_2.jpg
inflating: /root/datasets/train/H/48_H_3.jpg
inflating: /root/datasets/train/H/49_H_1.jpg
inflating: /root/datasets/train/H/49_H_2.jpg
inflating: /root/datasets/train/H/49_H_3.jpg
inflating: /root/datasets/train/H/50_H_1.jpg
inflating: /root/datasets/train/H/50_H_2.jpg
inflating: /root/datasets/train/H/50_H_3.jpg
inflating: /root/datasets/train/H/51_H_1.jpg
inflating: /root/datasets/train/H/51_H_2.jpg
inflating: /root/datasets/train/H/51_H_3.jpg
inflating: /root/datasets/train/H/52_H_1.jpg
inflating: /root/datasets/train/H/52_H_2.jpg
inflating: /root/datasets/train/H/52_H_3.jpg
inflating: /root/datasets/train/H/53_H_1.jpg
inflating: /root/datasets/train/H/53_H_2.jpg
inflating: /root/datasets/train/H/53_H_3.jpg
inflating: /root/datasets/train/H/54_H_1.jpg
inflating: /root/datasets/train/H/54_H_2.jpg
inflating: /root/datasets/train/H/54_H_3.jpg
inflating: /root/datasets/train/H/55_H_22.jpg
inflating: /root/datasets/train/H/55_H_23.jpg
inflating: /root/datasets/train/H/55_H_24.jpg
inflating: /root/datasets/train/H/56_H_1.jpg
inflating: /root/datasets/train/H/56_H_2.jpg
inflating: /root/datasets/train/H/56_H_3.jpg
inflating: /root/datasets/train/H/57_H_1.jpg
inflating: /root/datasets/train/H/57_H_2.jpg
inflating: /root/datasets/train/H/57_H_3.jpg
inflating: /root/datasets/train/H/58_H_1.jpg
inflating: /root/datasets/train/H/58_H_2.jpg
inflating: /root/datasets/train/H/58_H_3.jpg
inflating: /root/datasets/train/H/59_H_1.jpg
inflating: /root/datasets/train/H/59_H_2.jpg
inflating: /root/datasets/train/H/59_H_3.jpg
inflating: /root/datasets/train/H/6_H_1.jpg
inflating: /root/datasets/train/H/6_H_2.jpg
inflating: /root/datasets/train/H/6_H_3.jpg
inflating: /root/datasets/train/H/60_H_1.jpg
inflating: /root/datasets/train/H/60_H_2.jpg
inflating: /root/datasets/train/H/60_H_3.jpg
```

```
inflating: /root/datasets/train/H/61_H_1.jpg
inflating: /root/datasets/train/H/61_H_2.jpg
inflating: /root/datasets/train/H/61_H_3.jpg
inflating: /root/datasets/train/H/62_H_1.jpg
inflating: /root/datasets/train/H/62_H_2.jpg
inflating: /root/datasets/train/H/62_H_3.jpg
inflating: /root/datasets/train/H/7_H_1.jpg
inflating: /root/datasets/train/H/7_H_2.jpg
inflating: /root/datasets/train/H/7_H_3.jpg
inflating: /root/datasets/train/H/8_H_1.jpg
inflating: /root/datasets/train/H/8_H_2.jpg
inflating: /root/datasets/train/H/8_H_3.jpg
inflating: /root/datasets/train/H/9_H_1.jpg
inflating: /root/datasets/train/H/9_H_2.jpg
inflating: /root/datasets/train/H/9_H_3.jpg
 creating: /root/datasets/train/I/
inflating: /root/datasets/train/I/1_I_1.jpg
inflating: /root/datasets/train/I/1_I_2.jpg
inflating: /root/datasets/train/I/1_I_3.jpg
inflating: /root/datasets/train/I/10_I_25.jpg
inflating: /root/datasets/train/I/10_I_26.jpg
inflating: /root/datasets/train/I/10_I_27.jpg
inflating: /root/datasets/train/I/11_I_1.jpg
inflating: /root/datasets/train/I/11_I_2.jpg
inflating: /root/datasets/train/I/11_I_3.jpg
inflating: /root/datasets/train/I/12_I_1.jpg
inflating: /root/datasets/train/I/12_I_2.jpg
inflating: /root/datasets/train/I/12_I_3.jpg
inflating: /root/datasets/train/I/13_I_1.jpg
inflating: /root/datasets/train/I/13_I_2.jpg
inflating: /root/datasets/train/I/13_I_3.jpg
inflating: /root/datasets/train/I/14_I_1.jpg
inflating: /root/datasets/train/I/14_I_2.jpg
inflating: /root/datasets/train/I/14_I_3.jpg
inflating: /root/datasets/train/I/15_I_1.jpg
inflating: /root/datasets/train/I/15_I_2.jpg
inflating: /root/datasets/train/I/15_I_3.jpg
inflating: /root/datasets/train/I/16_I_1.jpg
inflating: /root/datasets/train/I/16_I_2.jpg
inflating: /root/datasets/train/I/16_I_3.jpg
inflating: /root/datasets/train/I/17_I_1.jpg
inflating: /root/datasets/train/I/17_I_2.jpg
inflating: /root/datasets/train/I/17_I_3.jpg
inflating: /root/datasets/train/I/18_I_1.jpg
inflating: /root/datasets/train/I/18_I_2.jpg
inflating: /root/datasets/train/I/18_I_3.jpg
inflating: /root/datasets/train/I/19_I_1.jpg
inflating: /root/datasets/train/I/19_I_2.jpg
inflating: /root/datasets/train/I/19_I_3.jpg
inflating: /root/datasets/train/I/2_I_1.jpg
inflating: /root/datasets/train/I/2_I_2.jpg
inflating: /root/datasets/train/I/2_I_3.jpg
inflating: /root/datasets/train/I/20_I_01.jpg
inflating: /root/datasets/train/I/20_I_02.jpg
inflating: /root/datasets/train/I/20_I_03.jpg
inflating: /root/datasets/train/I/21_I_1.jpg
inflating: /root/datasets/train/I/21_I_2.jpg
```

```
inflating: /root/datasets/train/I/21_I_3.jpg
inflating: /root/datasets/train/I/24_I_1.jpg
inflating: /root/datasets/train/I/24_I_2.jpg
inflating: /root/datasets/train/I/24_I_3.jpg
inflating: /root/datasets/train/I/26_I_1.jpg
inflating: /root/datasets/train/I/26_I_2.jpg
inflating: /root/datasets/train/I/26_I_3.jpg
inflating: /root/datasets/train/I/3_I_1.jpg
inflating: /root/datasets/train/I/3_I_2.jpg
inflating: /root/datasets/train/I/3_I_3.jpg
inflating: /root/datasets/train/I/30_I_1.jpg
inflating: /root/datasets/train/I/30_I_2.jpg
inflating: /root/datasets/train/I/30_I_3.jpg
inflating: /root/datasets/train/I/31_I_1.jpg
inflating: /root/datasets/train/I/31_I_2.jpg
inflating: /root/datasets/train/I/31_I_3.jpg
inflating: /root/datasets/train/I/32_I_1.jpg
inflating: /root/datasets/train/I/32_I_2.jpg
inflating: /root/datasets/train/I/32_I_3.jpg
inflating: /root/datasets/train/I/33_I_25.jpg
inflating: /root/datasets/train/I/33_I_26.jpg
inflating: /root/datasets/train/I/33_I_27.jpg
inflating: /root/datasets/train/I/34_I_1.jpg
inflating: /root/datasets/train/I/34_I_2.jpg
inflating: /root/datasets/train/I/34_I_3.jpg
inflating: /root/datasets/train/I/35_I_1.jpg
inflating: /root/datasets/train/I/35_I_2.jpg
inflating: /root/datasets/train/I/35_I_3.jpg
inflating: /root/datasets/train/I/36_I_1.jpg
inflating: /root/datasets/train/I/36_I_2.jpg
inflating: /root/datasets/train/I/36_I_3.jpg
inflating: /root/datasets/train/I/37_I_1.jpg
inflating: /root/datasets/train/I/37_I_2.jpg
inflating: /root/datasets/train/I/37_I_3.jpg
inflating: /root/datasets/train/I/38_I_1.jpg
inflating: /root/datasets/train/I/38_I_2.jpg
inflating: /root/datasets/train/I/38_I_3.jpg
inflating: /root/datasets/train/I/39_I_1.jpg
inflating: /root/datasets/train/I/39_I_2.jpg
inflating: /root/datasets/train/I/39_I_3.jpg
inflating: /root/datasets/train/I/4_I_1.jpg
inflating: /root/datasets/train/I/4_I_2.jpg
inflating: /root/datasets/train/I/4_I_3.jpg
inflating: /root/datasets/train/I/40_I_1.jpg
inflating: /root/datasets/train/I/40_I_2.jpg
inflating: /root/datasets/train/I/40_I_3.jpg
inflating: /root/datasets/train/I/41_I_1.jpg
inflating: /root/datasets/train/I/41_I_2.jpg
inflating: /root/datasets/train/I/41_I_3.jpg
inflating: /root/datasets/train/I/42_I_1.jpg
inflating: /root/datasets/train/I/42_I_2.jpg
inflating: /root/datasets/train/I/42_I_3.jpg
inflating: /root/datasets/train/I/43_I_1.jpg
inflating: /root/datasets/train/I/43_I_2.jpg
inflating: /root/datasets/train/I/43_I_3.jpg
inflating: /root/datasets/train/I/45_I_1.jpg
inflating: /root/datasets/train/I/45_I_2.jpg
```

```
inflating: /root/datasets/train/I/45_I_3.jpg
inflating: /root/datasets/train/I/47_I_1.jpg
inflating: /root/datasets/train/I/47_I_2.jpg
inflating: /root/datasets/train/I/47_I_3.jpg
inflating: /root/datasets/train/I/48_I_1.jpg
inflating: /root/datasets/train/I/48_I_2.jpg
inflating: /root/datasets/train/I/48_I_3.jpg
inflating: /root/datasets/train/I/49_I_1.jpg
inflating: /root/datasets/train/I/49_I_2.jpg
inflating: /root/datasets/train/I/49_I_3.jpg
inflating: /root/datasets/train/I/50_I_1.jpg
inflating: /root/datasets/train/I/50_I_2.jpg
inflating: /root/datasets/train/I/50_I_3.jpg
inflating: /root/datasets/train/I/51_I_1.jpg
inflating: /root/datasets/train/I/51_I_2.jpg
inflating: /root/datasets/train/I/51_I_3.jpg
inflating: /root/datasets/train/I/52_I_1.jpg
inflating: /root/datasets/train/I/52_I_2.jpg
inflating: /root/datasets/train/I/52_I_3.jpg
inflating: /root/datasets/train/I/53_I_1.jpg
inflating: /root/datasets/train/I/53_I_2.jpg
inflating: /root/datasets/train/I/53_I_3.jpg
inflating: /root/datasets/train/I/54_I_1.jpg
inflating: /root/datasets/train/I/54_I_2.jpg
inflating: /root/datasets/train/I/54_I_3.jpg
inflating: /root/datasets/train/I/55_I_25.jpg
inflating: /root/datasets/train/I/55_I_26.jpg
inflating: /root/datasets/train/I/55_I_27.jpg
inflating: /root/datasets/train/I/57_I_1.jpg
inflating: /root/datasets/train/I/57_I_2.jpg
inflating: /root/datasets/train/I/57_I_3.jpg
inflating: /root/datasets/train/I/58_I_1.jpg
inflating: /root/datasets/train/I/58_I_2.jpg
inflating: /root/datasets/train/I/58_I_3.jpg
inflating: /root/datasets/train/I/59_I_1.jpg
inflating: /root/datasets/train/I/59_I_2.jpg
inflating: /root/datasets/train/I/59_I_3.jpg
inflating: /root/datasets/train/I/6_I_1.jpg
inflating: /root/datasets/train/I/6_I_2.jpg
inflating: /root/datasets/train/I/6_I_3.jpg
inflating: /root/datasets/train/I/60_I_1.jpg
inflating: /root/datasets/train/I/60_I_2.jpg
inflating: /root/datasets/train/I/60_I_3.jpg
inflating: /root/datasets/train/I/61_I_1.jpg
inflating: /root/datasets/train/I/61_I_2.jpg
inflating: /root/datasets/train/I/61_I_3.jpg
inflating: /root/datasets/train/I/62_I_1.jpg
inflating: /root/datasets/train/I/62_I_2.jpg
inflating: /root/datasets/train/I/62_I_3.jpg
inflating: /root/datasets/train/I/7_I_1.jpg
inflating: /root/datasets/train/I/7_I_2.jpg
inflating: /root/datasets/train/I/7_I_3.jpg
inflating: /root/datasets/train/I/8_I_1.jpg
inflating: /root/datasets/train/I/8_I_2.jpg
inflating: /root/datasets/train/I/8_I_3.jpg
inflating: /root/datasets/train/I/9_I_1.jpg
inflating: /root/datasets/train/I/9_I_2.jpg
```

```
inflating: /root/datasets/train/I/9_I_3.jpg
 creating: /root/datasets/val/A/
inflating: /root/datasets/val/A/63_A_1.jpg
inflating: /root/datasets/val/A/63_A_2.jpg
inflating: /root/datasets/val/A/63_A_3.jpg
inflating: /root/datasets/val/A/65_A_1.jpg
inflating: /root/datasets/val/A/65_A_2.jpg
inflating: /root/datasets/val/A/65_A_3.jpg
inflating: /root/datasets/val/A/66_A_1.jpg
inflating: /root/datasets/val/A/66_A_2.jpg
inflating: /root/datasets/val/A/66_A_3.jpg
inflating: /root/datasets/val/A/67_A_1.jpg
inflating: /root/datasets/val/A/67_A_2.jpg
inflating: /root/datasets/val/A/67_A_3.jpg
inflating: /root/datasets/val/A/68_A_1.jpg
inflating: /root/datasets/val/A/68_A_2.jpg
inflating: /root/datasets/val/A/68_A_3.jpg
inflating: /root/datasets/val/A/70_A_1.jpg
inflating: /root/datasets/val/A/70_A_2.jpg
inflating: /root/datasets/val/A/70_A_3.jpg
inflating: /root/datasets/val/A/71_A_1.jpg
inflating: /root/datasets/val/A/71_A_2.jpg
inflating: /root/datasets/val/A/71_A_3.jpg
inflating: /root/datasets/val/A/72_A_1.jpg
inflating: /root/datasets/val/A/72_A_2.jpg
inflating: /root/datasets/val/A/72_A_3.jpg
inflating: /root/datasets/val/A/73_A_1.jpg
inflating: /root/datasets/val/A/73_A_2.jpg
inflating: /root/datasets/val/A/73_A_3.jpg
inflating: /root/datasets/val/A/74_A_1.jpg
inflating: /root/datasets/val/A/74_A_2.jpg
inflating: /root/datasets/val/A/74_A_3.jpg
inflating: /root/datasets/val/A/75_A_1.jpg
inflating: /root/datasets/val/A/75_A_2.jpg
inflating: /root/datasets/val/A/75_A_3.jpg
inflating: /root/datasets/val/A/76_A_1.jpg
inflating: /root/datasets/val/A/76_A_2.jpg
inflating: /root/datasets/val/A/76_A_3.jpg
inflating: /root/datasets/val/A/77_A_1.jpg
inflating: /root/datasets/val/A/77_A_2.jpg
inflating: /root/datasets/val/A/77_A_3.jpg
inflating: /root/datasets/val/A/78_A_1.jpg
inflating: /root/datasets/val/A/78_A_2.jpg
inflating: /root/datasets/val/A/78_A_3.jpg
inflating: /root/datasets/val/A/80_A_1.jpg
inflating: /root/datasets/val/A/80_A_2.jpg
inflating: /root/datasets/val/A/80_A_3.jpg
inflating: /root/datasets/val/A/81_A_1.jpg
inflating: /root/datasets/val/A/81_A_2.jpg
inflating: /root/datasets/val/A/81_A_3.jpg
inflating: /root/datasets/val/A/82_A_1.jpg
inflating: /root/datasets/val/A/82_A_3.jpg
 creating: /root/datasets/val/B/
inflating: /root/datasets/val/B/63_B_1.jpg
inflating: /root/datasets/val/B/63_B_2.jpg
inflating: /root/datasets/val/B/63_B_3.jpg
inflating: /root/datasets/val/B/65_B_1.jpg
```

```
inflating: /root/datasets/val/B/65_B_2.jpg
inflating: /root/datasets/val/B/65_B_3.jpg
inflating: /root/datasets/val/B/66_B_1.jpg
inflating: /root/datasets/val/B/66_B_2.jpg
inflating: /root/datasets/val/B/66_B_3.jpg
inflating: /root/datasets/val/B/67_B_1.jpg
inflating: /root/datasets/val/B/67_B_2.jpg
inflating: /root/datasets/val/B/67_B_3.jpg
inflating: /root/datasets/val/B/68_B_1.jpg
inflating: /root/datasets/val/B/68_B_2.jpg
inflating: /root/datasets/val/B/68_B_3.jpg
inflating: /root/datasets/val/B/70_B_1.jpg
inflating: /root/datasets/val/B/70_B_2.jpg
inflating: /root/datasets/val/B/70_B_3.jpg
inflating: /root/datasets/val/B/71_B_1.jpg
inflating: /root/datasets/val/B/71_B_2.jpg
inflating: /root/datasets/val/B/71_B_3.jpg
inflating: /root/datasets/val/B/72_B_1.jpg
inflating: /root/datasets/val/B/72_B_2.jpg
inflating: /root/datasets/val/B/72_B_3.jpg
inflating: /root/datasets/val/B/73_B_1.jpg
inflating: /root/datasets/val/B/73_B_2.jpg
inflating: /root/datasets/val/B/73_B_3.jpg
inflating: /root/datasets/val/B/74_B_1.jpg
inflating: /root/datasets/val/B/74_B_2.jpg
inflating: /root/datasets/val/B/74_B_3.jpg
inflating: /root/datasets/val/B/75_B_1.jpg
inflating: /root/datasets/val/B/75_B_2.jpg
inflating: /root/datasets/val/B/75_B_3.jpg
inflating: /root/datasets/val/B/76_B_1.jpg
inflating: /root/datasets/val/B/76_B_2.jpg
inflating: /root/datasets/val/B/76_B_3.jpg
inflating: /root/datasets/val/B/77_B_1.jpg
inflating: /root/datasets/val/B/77_B_2.jpg
inflating: /root/datasets/val/B/77_B_3.jpg
inflating: /root/datasets/val/B/78_B_1.jpg
inflating: /root/datasets/val/B/78_B_2.jpg
inflating: /root/datasets/val/B/78_B_3.jpg
inflating: /root/datasets/val/B/80_B_1.jpg
inflating: /root/datasets/val/B/80_B_2.jpg
inflating: /root/datasets/val/B/80_B_3.jpg
inflating: /root/datasets/val/B/81_B_1.jpg
inflating: /root/datasets/val/B/81_B_2.jpg
inflating: /root/datasets/val/B/81_B_3.jpg
inflating: /root/datasets/val/B/82_B_1.jpg
inflating: /root/datasets/val/B/82_B_2.jpg
inflating: /root/datasets/val/B/82_B_3.jpg
 creating: /root/datasets/val/C/
inflating: /root/datasets/val/C/63_C_1.jpg
inflating: /root/datasets/val/C/63_C_2.jpg
inflating: /root/datasets/val/C/63_C_3.jpg
inflating: /root/datasets/val/C/65_C_1.jpg
inflating: /root/datasets/val/C/65_C_2.jpg
inflating: /root/datasets/val/C/65_C_3.jpg
inflating: /root/datasets/val/C/66_C_1.jpg
inflating: /root/datasets/val/C/66_C_2.jpg
inflating: /root/datasets/val/C/66_C_3.jpg
```

```
inflating: /root/datasets/val/C/67_C_1.jpg
inflating: /root/datasets/val/C/67_C_2.jpg
inflating: /root/datasets/val/C/67_C_3.jpg
inflating: /root/datasets/val/C/68_C_1.jpg
inflating: /root/datasets/val/C/68_C_2.jpg
inflating: /root/datasets/val/C/68_C_3.jpg
inflating: /root/datasets/val/C/70_C_1.jpg
inflating: /root/datasets/val/C/70_C_2.jpg
inflating: /root/datasets/val/C/70_C_3.jpg
inflating: /root/datasets/val/C/71_C_1.jpg
inflating: /root/datasets/val/C/71_C_2.jpg
inflating: /root/datasets/val/C/71_C_3.jpg
inflating: /root/datasets/val/C/72_C_1.jpg
inflating: /root/datasets/val/C/72_C_2.jpg
inflating: /root/datasets/val/C/72_C_3.jpg
inflating: /root/datasets/val/C/73_C_1.jpg
inflating: /root/datasets/val/C/73_C_2.jpg
inflating: /root/datasets/val/C/73_C_3.jpg
inflating: /root/datasets/val/C/74_C_1.jpg
inflating: /root/datasets/val/C/74_C_2.jpg
inflating: /root/datasets/val/C/74_C_3.jpg
inflating: /root/datasets/val/C/75_C_1.jpg
inflating: /root/datasets/val/C/75_C_2.jpg
inflating: /root/datasets/val/C/75_C_3.jpg
inflating: /root/datasets/val/C/76_C_1.jpg
inflating: /root/datasets/val/C/76_C_2.jpg
inflating: /root/datasets/val/C/76_C_3.jpg
inflating: /root/datasets/val/C/77_C_1.jpg
inflating: /root/datasets/val/C/77_C_2.jpg
inflating: /root/datasets/val/C/77_C_3.jpg
inflating: /root/datasets/val/C/78_C_1.jpg
inflating: /root/datasets/val/C/78_C_2.jpg
inflating: /root/datasets/val/C/78_C_3.jpg
inflating: /root/datasets/val/C/80_C_1.jpg
inflating: /root/datasets/val/C/80_C_2.jpg
inflating: /root/datasets/val/C/80_C_3.jpg
inflating: /root/datasets/val/C/81_C_1.jpg
inflating: /root/datasets/val/C/81_C_2.jpg
inflating: /root/datasets/val/C/81_C_3.jpg
inflating: /root/datasets/val/C/82_C_1.jpg
inflating: /root/datasets/val/C/82_C_2.jpg
inflating: /root/datasets/val/C/82_C_3.jpg
 creating: /root/datasets/val/D/
inflating: /root/datasets/val/D/63_D_1.jpg
inflating: /root/datasets/val/D/63_D_2.jpg
inflating: /root/datasets/val/D/63_D_3.jpg
inflating: /root/datasets/val/D/65_D_1.jpg
inflating: /root/datasets/val/D/65_D_2.jpg
inflating: /root/datasets/val/D/65_D_3.jpg
inflating: /root/datasets/val/D/66_D_1.jpg
inflating: /root/datasets/val/D/66_D_2.jpg
inflating: /root/datasets/val/D/66_D_3.jpg
inflating: /root/datasets/val/D/67_D_1.jpg
inflating: /root/datasets/val/D/67_D_2.jpg
inflating: /root/datasets/val/D/67_D_3.jpg
inflating: /root/datasets/val/D/68_D_1.jpg
inflating: /root/datasets/val/D/68_D_2.jpg
```

```
inflating: /root/datasets/val/D/68_D_3.jpg
inflating: /root/datasets/val/D/70_D_1.jpg
inflating: /root/datasets/val/D/70_D_2.jpg
inflating: /root/datasets/val/D/70_D_3.jpg
inflating: /root/datasets/val/D/71_D_1.jpg
inflating: /root/datasets/val/D/71_D_2.jpg
inflating: /root/datasets/val/D/71_D_3.jpg
inflating: /root/datasets/val/D/72_D_1.jpg
inflating: /root/datasets/val/D/72_D_2.jpg
inflating: /root/datasets/val/D/72_D_3.jpg
inflating: /root/datasets/val/D/73_D_1.jpg
inflating: /root/datasets/val/D/73_D_2.jpg
inflating: /root/datasets/val/D/73_D_3.jpg
inflating: /root/datasets/val/D/74_D_1.jpg
inflating: /root/datasets/val/D/74_D_2.jpg
inflating: /root/datasets/val/D/74_D_3.jpg
inflating: /root/datasets/val/D/75_D_1.jpg
inflating: /root/datasets/val/D/75_D_2.jpg
inflating: /root/datasets/val/D/75_D_3.jpg
inflating: /root/datasets/val/D/76_D_1.jpg
inflating: /root/datasets/val/D/76_D_2.jpg
inflating: /root/datasets/val/D/76_D_3.jpg
inflating: /root/datasets/val/D/77_D_1.jpg
inflating: /root/datasets/val/D/77_D_2.jpg
inflating: /root/datasets/val/D/77_D_3.jpg
inflating: /root/datasets/val/D/78_D_1.jpg
inflating: /root/datasets/val/D/78_D_2.jpg
inflating: /root/datasets/val/D/78_D_3.jpg
inflating: /root/datasets/val/D/80_D_1.jpg
inflating: /root/datasets/val/D/80_D_2.jpg
inflating: /root/datasets/val/D/80_D_3.jpg
inflating: /root/datasets/val/D/81_D_1.jpg
inflating: /root/datasets/val/D/81_D_2.jpg
inflating: /root/datasets/val/D/81_D_3.jpg
inflating: /root/datasets/val/D/82_D_1.jpg
inflating: /root/datasets/val/D/82_D_2.jpg
inflating: /root/datasets/val/D/82_D_3.jpg
 creating: /root/datasets/val/E/
inflating: /root/datasets/val/E/63_E_1.jpg
inflating: /root/datasets/val/E/63_E_2.jpg
inflating: /root/datasets/val/E/63_E_3.jpg
inflating: /root/datasets/val/E/65_E_1.jpg
inflating: /root/datasets/val/E/65_E_2.jpg
inflating: /root/datasets/val/E/65_E_3.jpg
inflating: /root/datasets/val/E/66_E_1.jpg
inflating: /root/datasets/val/E/66_E_2.jpg
inflating: /root/datasets/val/E/66_E_3.jpg
inflating: /root/datasets/val/E/67_E_1.jpg
inflating: /root/datasets/val/E/67_E_2.jpg
inflating: /root/datasets/val/E/67_E_3.jpg
inflating: /root/datasets/val/E/68_E_1.jpg
inflating: /root/datasets/val/E/68_E_2.jpg
inflating: /root/datasets/val/E/68_E_3.jpg
inflating: /root/datasets/val/E/70_E_1.jpg
inflating: /root/datasets/val/E/70_E_2.jpg
inflating: /root/datasets/val/E/70_E_3.jpg
inflating: /root/datasets/val/E/71_E_1.jpg
```

```
inflating: /root/datasets/val/E/71_E_2.jpg
inflating: /root/datasets/val/E/71_E_3.jpg
inflating: /root/datasets/val/E/72_E_1.jpg
inflating: /root/datasets/val/E/72_E_2.jpg
inflating: /root/datasets/val/E/72_E_3.jpg
inflating: /root/datasets/val/E/73_E_1.jpg
inflating: /root/datasets/val/E/73_E_2.jpg
inflating: /root/datasets/val/E/73_E_3.jpg
inflating: /root/datasets/val/E/74_E_1.jpg
inflating: /root/datasets/val/E/74_E_2.jpg
inflating: /root/datasets/val/E/74_E_3.jpg
inflating: /root/datasets/val/E/75_E_1.jpg
inflating: /root/datasets/val/E/75_E_2.jpg
inflating: /root/datasets/val/E/75_E_3.jpg
inflating: /root/datasets/val/E/76_E_1.jpg
inflating: /root/datasets/val/E/76_E_2.jpg
inflating: /root/datasets/val/E/76_E_3.jpg
inflating: /root/datasets/val/E/77_E_1.jpg
inflating: /root/datasets/val/E/77_E_2.jpg
inflating: /root/datasets/val/E/77_E_3.jpg
inflating: /root/datasets/val/E/78_E_1.jpg
inflating: /root/datasets/val/E/78_E_2.jpg
inflating: /root/datasets/val/E/78_E_3.jpg
inflating: /root/datasets/val/E/80_E_1.jpg
inflating: /root/datasets/val/E/80_E_2.jpg
inflating: /root/datasets/val/E/80_E_3.jpg
inflating: /root/datasets/val/E/81_E_1.jpg
inflating: /root/datasets/val/E/81_E_2.jpg
inflating: /root/datasets/val/E/81_E_3.jpg
inflating: /root/datasets/val/E/82_E_1.jpg
inflating: /root/datasets/val/E/82_E_2.jpg
inflating: /root/datasets/val/E/82_E_3.jpg
 creating: /root/datasets/val/F/
inflating: /root/datasets/val/F/63_F_1.jpg
inflating: /root/datasets/val/F/63_F_2.jpg
inflating: /root/datasets/val/F/63_F_3.jpg
inflating: /root/datasets/val/F/65_F_1.jpg
inflating: /root/datasets/val/F/65_F_2.jpg
inflating: /root/datasets/val/F/65_F_3.jpg
inflating: /root/datasets/val/F/66_F_1.jpg
inflating: /root/datasets/val/F/66_F_2.jpg
inflating: /root/datasets/val/F/66_F_3.jpg
inflating: /root/datasets/val/F/67_F_1.jpg
inflating: /root/datasets/val/F/67_F_2.jpg
inflating: /root/datasets/val/F/67_F_3.jpg
inflating: /root/datasets/val/F/68_F_1.jpg
inflating: /root/datasets/val/F/68_F_2.jpg
inflating: /root/datasets/val/F/68_F_3.jpg
inflating: /root/datasets/val/F/70_F_1.jpg
inflating: /root/datasets/val/F/70_F_2.jpg
inflating: /root/datasets/val/F/70_F_3.jpg
inflating: /root/datasets/val/F/71_F_1.jpg
inflating: /root/datasets/val/F/71_F_2.jpg
inflating: /root/datasets/val/F/71_F_3.jpg
inflating: /root/datasets/val/F/72_F_1.jpg
inflating: /root/datasets/val/F/72_F_2.jpg
inflating: /root/datasets/val/F/72_F_3.jpg
```

```
inflating: /root/datasets/val/F/73_F_1.jpg
inflating: /root/datasets/val/F/73_F_2.jpg
inflating: /root/datasets/val/F/73_F_3.jpg
inflating: /root/datasets/val/F/74_F_1.jpg
inflating: /root/datasets/val/F/74_F_2.jpg
inflating: /root/datasets/val/F/74_F_3.jpg
inflating: /root/datasets/val/F/75_F_1.jpg
inflating: /root/datasets/val/F/75_F_2.jpg
inflating: /root/datasets/val/F/75_F_3.jpg
inflating: /root/datasets/val/F/76_F_1.jpg
inflating: /root/datasets/val/F/76_F_2.jpg
inflating: /root/datasets/val/F/76_F_3.jpg
inflating: /root/datasets/val/F/77_F_1.jpg
inflating: /root/datasets/val/F/77_F_2.jpg
inflating: /root/datasets/val/F/77_F_3.jpg
inflating: /root/datasets/val/F/78_F_1.jpg
inflating: /root/datasets/val/F/78_F_2.jpg
inflating: /root/datasets/val/F/78_F_3.jpg
inflating: /root/datasets/val/F/80_F_1.jpg
inflating: /root/datasets/val/F/80_F_2.jpg
inflating: /root/datasets/val/F/80_F_3.jpg
inflating: /root/datasets/val/F/81_F_1.jpg
inflating: /root/datasets/val/F/81_F_2.jpg
inflating: /root/datasets/val/F/81_F_3.jpg
inflating: /root/datasets/val/F/82_F_1.jpg
inflating: /root/datasets/val/F/82_F_2.jpg
inflating: /root/datasets/val/F/82_F_3.jpg
 creating: /root/datasets/val/G/
inflating: /root/datasets/val/G/63_G_1.jpg
inflating: /root/datasets/val/G/63_G_2.jpg
inflating: /root/datasets/val/G/63_G_3.jpg
inflating: /root/datasets/val/G/65_G_1.jpg
inflating: /root/datasets/val/G/65_G_2.jpg
inflating: /root/datasets/val/G/65_G_3.jpg
inflating: /root/datasets/val/G/66_G_1.jpg
inflating: /root/datasets/val/G/66_G_2.jpg
inflating: /root/datasets/val/G/66_G_3.jpg
inflating: /root/datasets/val/G/67_G_1.jpg
inflating: /root/datasets/val/G/67_G_2.jpg
inflating: /root/datasets/val/G/67_G_3.jpg
inflating: /root/datasets/val/G/68_G_1.jpg
inflating: /root/datasets/val/G/68_G_2.jpg
inflating: /root/datasets/val/G/68_G_3.jpg
inflating: /root/datasets/val/G/70_G_1.jpg
inflating: /root/datasets/val/G/70_G_2.jpg
inflating: /root/datasets/val/G/70_G_3.jpg
inflating: /root/datasets/val/G/71_G_1.jpg
inflating: /root/datasets/val/G/71_G_2.jpg
inflating: /root/datasets/val/G/71_G_3.jpg
inflating: /root/datasets/val/G/72_G_1.jpg
inflating: /root/datasets/val/G/72_G_2.jpg
inflating: /root/datasets/val/G/72_G_3.jpg
inflating: /root/datasets/val/G/73_G_1.jpg
inflating: /root/datasets/val/G/73_G_2.jpg
inflating: /root/datasets/val/G/73_G_3.jpg
inflating: /root/datasets/val/G/74_G_1.jpg
inflating: /root/datasets/val/G/74_G_2.jpg
```

```
inflating: /root/datasets/val/G/74_G_3.jpg
inflating: /root/datasets/val/G/75_G_1.jpg
inflating: /root/datasets/val/G/75_G_2.jpg
inflating: /root/datasets/val/G/75_G_3.jpg
inflating: /root/datasets/val/G/76_G_1.jpg
inflating: /root/datasets/val/G/76_G_2.jpg
inflating: /root/datasets/val/G/76_G_3.jpg
inflating: /root/datasets/val/G/77_G_1.jpg
inflating: /root/datasets/val/G/77_G_2.jpg
inflating: /root/datasets/val/G/77_G_3.jpg
inflating: /root/datasets/val/G/78_G_1.jpg
inflating: /root/datasets/val/G/78_G_2.jpg
inflating: /root/datasets/val/G/78_G_3.jpg
inflating: /root/datasets/val/G/80_G_1.jpg
inflating: /root/datasets/val/G/80_G_2.jpg
inflating: /root/datasets/val/G/80_G_3.jpg
inflating: /root/datasets/val/G/81_G_1.jpg
inflating: /root/datasets/val/G/81_G_2.jpg
inflating: /root/datasets/val/G/81_G_3.jpg
inflating: /root/datasets/val/G/82_G_1.jpg
inflating: /root/datasets/val/G/82_G_2.jpg
inflating: /root/datasets/val/G/82_G_3.jpg
 creating: /root/datasets/val/H/
inflating: /root/datasets/val/H/63_H_1.jpg
inflating: /root/datasets/val/H/63_H_2.jpg
inflating: /root/datasets/val/H/63_H_3.jpg
inflating: /root/datasets/val/H/65_H_1.jpg
inflating: /root/datasets/val/H/65_H_2.jpg
inflating: /root/datasets/val/H/65_H_3.jpg
inflating: /root/datasets/val/H/66_H_1.jpg
inflating: /root/datasets/val/H/66_H_2.jpg
inflating: /root/datasets/val/H/66_H_3.jpg
inflating: /root/datasets/val/H/67_H_1.jpg
inflating: /root/datasets/val/H/67_H_2.jpg
inflating: /root/datasets/val/H/67_H_3.jpg
inflating: /root/datasets/val/H/68_H_1.jpg
inflating: /root/datasets/val/H/68_H_2.jpg
inflating: /root/datasets/val/H/68_H_3.jpg
inflating: /root/datasets/val/H/70_H_1.jpg
inflating: /root/datasets/val/H/70_H_2.jpg
inflating: /root/datasets/val/H/70_H_3.jpg
inflating: /root/datasets/val/H/71_H_1.jpg
inflating: /root/datasets/val/H/71_H_2.jpg
inflating: /root/datasets/val/H/71_H_3.jpg
inflating: /root/datasets/val/H/72_H_1.jpg
inflating: /root/datasets/val/H/72_H_2.jpg
inflating: /root/datasets/val/H/72_H_3.jpg
inflating: /root/datasets/val/H/73_H_1.jpg
inflating: /root/datasets/val/H/73_H_2.jpg
inflating: /root/datasets/val/H/73_H_3.jpg
inflating: /root/datasets/val/H/74_H_1.jpg
inflating: /root/datasets/val/H/74_H_2.jpg
inflating: /root/datasets/val/H/74_H_3.jpg
inflating: /root/datasets/val/H/75_H_1.jpg
inflating: /root/datasets/val/H/75_H_2.jpg
inflating: /root/datasets/val/H/75_H_3.jpg
inflating: /root/datasets/val/H/76_H_1.jpg
```

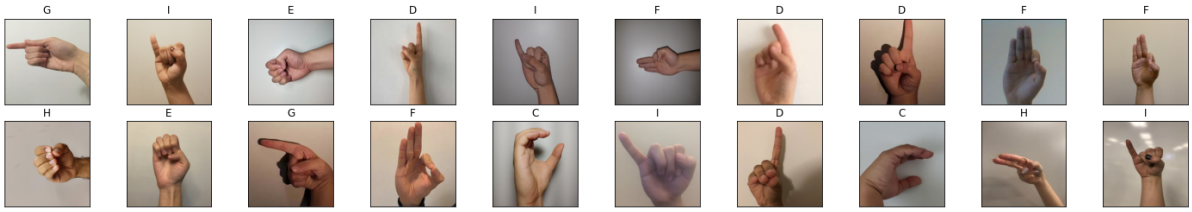
```
inflating: /root/datasets/val/H/76_H_2.jpg
inflating: /root/datasets/val/H/76_H_3.jpg
inflating: /root/datasets/val/H/77_H_1.jpg
inflating: /root/datasets/val/H/77_H_2.jpg
inflating: /root/datasets/val/H/77_H_3.jpg
inflating: /root/datasets/val/H/78_H_1.jpg
inflating: /root/datasets/val/H/78_H_2.jpg
inflating: /root/datasets/val/H/78_H_3.jpg
inflating: /root/datasets/val/H/80_H_1.jpg
inflating: /root/datasets/val/H/80_H_2.jpg
inflating: /root/datasets/val/H/80_H_3.jpg
inflating: /root/datasets/val/H/81_H_1.jpg
inflating: /root/datasets/val/H/81_H_2.jpg
inflating: /root/datasets/val/H/81_H_3.jpg
inflating: /root/datasets/val/H/82_H_1.jpg
inflating: /root/datasets/val/H/82_H_2.jpg
inflating: /root/datasets/val/H/82_H_3.jpg
 creating: /root/datasets/val/I/
inflating: /root/datasets/val/I/63_I_1.jpg
inflating: /root/datasets/val/I/63_I_2.jpg
inflating: /root/datasets/val/I/63_I_3.jpg
inflating: /root/datasets/val/I/65_I_1.jpg
inflating: /root/datasets/val/I/65_I_2.jpg
inflating: /root/datasets/val/I/65_I_3.jpg
inflating: /root/datasets/val/I/66_I_1.jpg
inflating: /root/datasets/val/I/66_I_2.jpg
inflating: /root/datasets/val/I/66_I_3.jpg
inflating: /root/datasets/val/I/67_I_1.jpg
inflating: /root/datasets/val/I/67_I_2.jpg
inflating: /root/datasets/val/I/67_I_3.jpg
inflating: /root/datasets/val/I/68_I_1.jpg
inflating: /root/datasets/val/I/68_I_2.jpg
inflating: /root/datasets/val/I/68_I_3.jpg
inflating: /root/datasets/val/I/71_I_1.jpg
inflating: /root/datasets/val/I/71_I_2.jpg
inflating: /root/datasets/val/I/71_I_3.jpg
inflating: /root/datasets/val/I/72_I_1.jpg
inflating: /root/datasets/val/I/72_I_2.jpg
inflating: /root/datasets/val/I/72_I_3.jpg
inflating: /root/datasets/val/I/73_I_1.jpg
inflating: /root/datasets/val/I/73_I_2.jpg
inflating: /root/datasets/val/I/73_I_3.jpg
inflating: /root/datasets/val/I/74_I_1.jpg
inflating: /root/datasets/val/I/74_I_2.jpg
inflating: /root/datasets/val/I/74_I_3.jpg
inflating: /root/datasets/val/I/75_I_1.jpg
inflating: /root/datasets/val/I/75_I_2.jpg
inflating: /root/datasets/val/I/75_I_3.jpg
inflating: /root/datasets/val/I/76_I_1.jpg
inflating: /root/datasets/val/I/76_I_2.jpg
inflating: /root/datasets/val/I/76_I_3.jpg
inflating: /root/datasets/val/I/77_I_1.jpg
inflating: /root/datasets/val/I/77_I_2.jpg
inflating: /root/datasets/val/I/77_I_3.jpg
inflating: /root/datasets/val/I/78_I_1.jpg
inflating: /root/datasets/val/I/78_I_2.jpg
inflating: /root/datasets/val/I/78_I_3.jpg
```

```
inflating: /root/datasets/val/I/80_I_1.jpg
inflating: /root/datasets/val/I/80_I_2.jpg
inflating: /root/datasets/val/I/80_I_3.jpg
inflating: /root/datasets/val/I/81_I_1.jpg
inflating: /root/datasets/val/I/81_I_2.jpg
inflating: /root/datasets/val/I/81_I_3.jpg
inflating: /root/datasets/val/I/83_I_1.jpg
inflating: /root/datasets/val/I/83_I_2.jpg
inflating: /root/datasets/val/I/83_I_3.jpg
 creating: /root/datasets/Small/A/
inflating: /root/datasets/Small/A/1_A_1.jpg
inflating: /root/datasets/Small/A/1_A_2.jpg
inflating: /root/datasets/Small/A/1_A_3.jpg
 creating: /root/datasets/Small/B/
inflating: /root/datasets/Small/B/1_B_1.jpg
inflating: /root/datasets/Small/B/1_B_2.jpg
inflating: /root/datasets/Small/B/1_B_3.jpg
 creating: /root/datasets/Small/C/
inflating: /root/datasets/Small/C/1_C_1.jpg
inflating: /root/datasets/Small/C/1_C_2.jpg
inflating: /root/datasets/Small/C/1_C_3.jpg
 creating: /root/datasets/Small/D/
inflating: /root/datasets/Small/D/1_D_1.jpg
inflating: /root/datasets/Small/D/1_D_2.jpg
inflating: /root/datasets/Small/D/1_D_3.jpg
 creating: /root/datasets/Small/E/
inflating: /root/datasets/Small/E/1_E_1.jpg
inflating: /root/datasets/Small/E/1_E_2.jpg
inflating: /root/datasets/Small/E/1_E_3.jpg
 creating: /root/datasets/Small/F/
inflating: /root/datasets/Small/F/1_F_1.jpg
inflating: /root/datasets/Small/F/1_F_2.jpg
inflating: /root/datasets/Small/F/1_F_3.jpg
 creating: /root/datasets/Small/G/
inflating: /root/datasets/Small/G/2_G_1.jpg
inflating: /root/datasets/Small/G/2_G_2.jpg
inflating: /root/datasets/Small/G/2_G_3.jpg
 creating: /root/datasets/Small/H/
inflating: /root/datasets/Small/H/1_H_1.jpg
inflating: /root/datasets/Small/H/1_H_2.jpg
inflating: /root/datasets/Small/H/1_H_3.jpg
 creating: /root/datasets/Small/I/
inflating: /root/datasets/Small/I/1_I_1.jpg
inflating: /root/datasets/Small/I/1_I_2.jpg
inflating: /root/datasets/Small/I/1_I_3.jpg
```

In [ ]:
```
import time
import os
import numpy as np
import torch

import torchvision
from torchvision import datasets, models, transforms
import matplotlib.pyplot as plt


# define training and test data directories
data_dir = '/root/datasets/'
train_dir = os.path.join(data_dir, 'train/')
val_dir = os.path.join(data_dir, 'val/')
test_dir = os.path.join(data_dir, 'test/')


# load and transform data using ImageFolder
# Data was already split randomly into train, validation, test data sets with
 approximately 60%, 20% and 20% of the total samples

# resize and ensure all images to 224 x 224 pixels, random crop is not because
the images are created consistently
data_transform = transforms.Compose([transforms.Resize((224,224)),
                                     transforms.ToTensor()])

train_data = datasets.ImageFolder(train_dir, transform=data_transform)
val_data = datasets.ImageFolder(val_dir, transform=data_transform)
test_data = datasets.ImageFolder(test_dir, transform=data_transform)

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# print out some data stats
print('Num training images: ', len(train_data))
print('Num validation images: ', len(val_data))
print('Num test images: ', len(test_data))

# define dataloader parameters
batch_size  = 32 # process 32 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders
train_loader = torch.utils.data.DataLoader(train_data, batch_size=batch_size,
                                           num_workers=num_workers, shuffle=Tr
ue)
val_loader = torch.utils.data.DataLoader(val_data, batch_size=batch_size,
                                         num_workers=num_workers, shuffle=Tru
e)
test_loader = torch.utils.data.DataLoader(test_data, batch_size=batch_size,
                                          num_workers=num_workers, shuffle=Tru
e)

# Visualize some sample data

# obtain one batch of training images
```

```
dataiter = iter(train_loader)
images, labels = dataiter.next()
images = images.numpy() # convert images to numpy for display

# plot the images in the batch, along with the corresponding labels
fig = plt.figure(figsize=(25, 4))
for idx in np.arange(20):
    ax = fig.add_subplot(2, 20/2, idx+1, xticks=[], yticks=[])
    plt.imshow(np.transpose(images[idx], (1, 2, 0)))
    ax.set_title(classes[labels[idx]])

# The data were prepared by random selections of the data and put them in thre
e separate folders that are train, val, and test. About 60% of randomly select
ed data
# are in training dataset, and 20% of the data in val or test data set. The sp
litting strategy is used because it ensures a complete separation of data base
d on usage and
# reduce bias during data preparation since all data were chosen randomly not
 selectively. There are 1475 training images, 455 validation images and 501 te
st images.
```

```
Num training images:  1475
Num validation images:  455
Num test images:  501
```



2. Model Building and Sanity Checking [15 pt]

Part (a) Convolutional Network - 5 pt

Build a convolutional neural network model that takes the (224x224 RGB) image as input, and predicts the gesture letter. Your model should be a subclass of nn.Module. Explain your choice of neural network architecture: how many layers did you choose? What types of layers did you use? Were they fully-connected or convolutional? What about other decisions like pooling layers, activation functions, number of channels / hidden units?

In [ ]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F

import matplotlib.pyplot as plt # for plotting
import torch.optim as optim #for gradient descent

torch.manual_seed(1) # set the random seed

class HandGestureClassifier(nn.Module):
    def __init__(self):
        super(HandGestureClassifier, self).__init__()
        self.conv1 = nn.Conv2d(3, 5, 5) #3 in_channels, 5 out_chanels, 5 kerne
l_size 224*224*3 -> 220*220*5
        #self.pool2 = nn.MaxPool2d(4, 4) #kernel_size, stride 220*220*5 -> 55*
55*5
        self.conv2 = nn.Conv2d(5, 10, 6) #5 in_channels, 10 out_chanels, 6 ker
nel_size 55*55*5 -> 50*50*10
        self.pool = nn.MaxPool2d(2, 2) #2 kernel_size, 2 stride 50*50*10 -> 25
*25*10
        self.conv3 = nn.Conv2d(10, 10, 10) #10 in_channels, 10 out_chanels, 10
kernel_size 25*25*10 -> 16*16*10
        self.pool2 = nn.MaxPool2d(4, 4) #4 kernel_size, 4 stride 16*16*10 -> 4
*4*10
        self.fc1 = nn.Linear(160, 32) #Fully connected layer from now on, hidd
en units = 32 neurons with 1 hidden layer
        self.fc2 = nn.Linear(32, 9)  # Softmax activation functions will be us
ed with 9 ouputs (9 Probabilities)

    def forward(self, x):
        x = self.pool2(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = self.pool2(F.relu(self.conv3(x)))
        x = x.view(-1, 160) # reshape the results, feed them into an fully con
nected network, and apply relu activation functions
        x = F.relu(self.fc1(x)) # relu activation function
        x = self.fc2(x) # Softmax will be applied at crossentropyloss at the o
utput to get the 9 Probabilities
        return x
```

CNN is used because it is powerful in solving the image classification problem. The CNN will be used to extract important features and then feed them into the fully connected network that will classify the inputs. I used 3 convolutional layers and two fully connected layers; in total 5 layers have been used. Pooling layer is applied at the end of each conv layer with different pooling amounts, refer to comments for details. The types of activation functions and number of hidden units and etc are explained in details in the comments in the code above.

Part (b) Training Code - 5 pt

Write code that trains your neural network given some training data. Your training code should make it easy to tweak the usual hyperparameters, like batch size, learning rate, and the model object itself. Make sure that you are checkpointing your models from time to time (the frequency is up to you). Explain your choice of loss function and optimizer.

In [ ]:
```
def get_accuracy(model, data_loader):

    correct = 0
    total = 0
    for imgs, labels in data_loader:


        ############################################
        #To Enable GPU Usage
        if use_cuda and torch.cuda.is_available():
          imgs = imgs.cuda()
          labels = labels.cuda()
        ############################################

        output = model(imgs)

        #select index with maximum prediction score
        pred = output.max(1, keepdim=True)[1]
        correct += pred.eq(labels.view_as(pred)).sum().item()
        total += imgs.shape[0]
    return correct / total
```

GPU is enabled during calculations to increase efficiency and reduce calculation time.

In [ ]:
```
def train(model, train_load,val_load, batch_size=32, num_epochs=1,lr=0.01 ):
    torch.manual_seed(1000)
    criterion = nn.CrossEntropyLoss() # softmax activation function is applied
and cross entropy loss is used for classification problem.
    optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9) # stochasti
c gradient descent is used with momentum to improve efficiency and a learning
 rate of 0.01. Moreover,
    # it is also because that we have a large and complex image processing pro
blem and thus stochastic gradient descent can reduce the computation burden to
achive faster convergence
    # by replacing the gradient calculated with the entire dataset with an est
imated value that is calculated from a randomly selected data subset.

    iteration, n_epochs, train_acc, val_acc = [], [], [], []

    # training
    n = 0 # the number of iterations
    start_time=time.time()
    for epoch in range(num_epochs):
        nepoch = epoch + 1
        n_epochs.append(nepoch)

        for imgs, labels in iter(train_load):


            ##############################################
            #To Enable GPU Usage
            if use_cuda and torch.cuda.is_available():
              imgs = imgs.cuda()
              labels = labels.cuda()
            ##############################################


            out = model(imgs)             # forward pass
            loss = criterion(out, labels) # compute the total loss
            loss.backward()               # backward pass (compute parameter u
pdates)
            optimizer.step()              # make the updates for each paramete
r
            optimizer.zero_grad()         # a clean up step for PyTorch
            n += 1
            iteration.append(n)


            # save the current training information
        val_acc.append(get_accuracy(model, val_load))  # compute validation ac
curacy
        train_acc.append(get_accuracy(model, train_load))  # compute validatio
n accuracy

        print("Iteration: ",n,'Progress: % 6.2f ' % ((epoch * len(train_load))
/ (num_epochs * len(train_load))*100),'%', "Time Elapsed: % 6.2f s " % (time.t
ime()-start_time))

        print ("Epoch %d Finished. " % epoch ,"Time per Epoch: % 6.2f s "% ((t
ime.time()-start_time) / (epoch +1)))
```

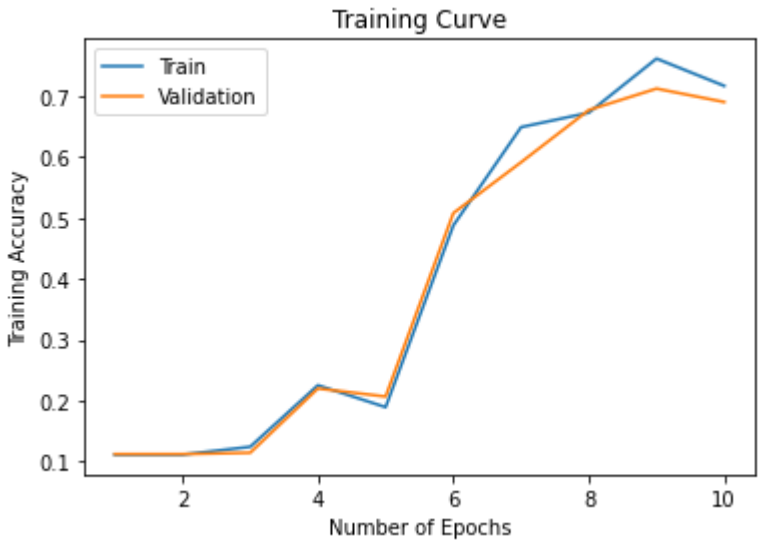
```
end_time= time.time()

# plotting


plt.title("Training Curve")
plt.plot(n_epochs, train_acc, label="Train")
plt.plot(n_epochs, val_acc, label="Validation")
plt.xlabel("Number of Epochs")
plt.ylabel("Training Accuracy")
plt.legend(loc='best')
plt.show()

#plt.title("Training Curve")
#plt.plot(iteration, train_acc, label="Train")
#plt.plot(iteration, val_acc, label="Validation")
#plt.xlabel("Number of Iterations")
#plt.ylabel("Training Accuracy")
#plt.legend(loc='best')
#plt.show()

print("Final Training Accuracy: {}".format(train_acc[-1]))
print("Final Validation Accuracy: {}".format(val_acc[-1]))
```

In [ ]:
```
!nvidia-smi
```

```
Sat Feb 20 16:10:11 2021
+-----------------------------------------------------------------------------
-+
| NVIDIA-SMI 460.39       Driver Version: 460.32.03    CUDA Version: 11.2
|
|-------------------------------+----------------------+----------------------
-+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC
|
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M.
|
|                               |                      |               MIG M.
|
|===============================+======================+======================
=|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0
|
| N/A   67C    P0    31W /  70W |   1292MiB / 15109MiB |      0%      Default
|
|                               |                      |                  N/A
|
+-------------------------------+----------------------+----------------------
-+

+-----------------------------------------------------------------------------
-+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory
|
|        ID   ID                                                   Usage
|
|=============================================================================
=|
+-----------------------------------------------------------------------------
-+
```

In [ ]:
```
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=32,num_epochs=10,lr=0.01 )
```

```
CUDA is available!  Training on GPU ...
Iteration:  47 Progress:   0.00  % Time Elapsed:   6.63 s
Epoch 0 Finished.  Time per Epoch:   6.63 s
Iteration:  94 Progress:  10.00  % Time Elapsed:  13.34 s
Epoch 1 Finished.  Time per Epoch:   6.67 s
Iteration:  141 Progress:  20.00  % Time Elapsed:  19.98 s
Epoch 2 Finished.  Time per Epoch:   6.66 s
Iteration:  188 Progress:  30.00  % Time Elapsed:  26.59 s
Epoch 3 Finished.  Time per Epoch:   6.65 s
Iteration:  235 Progress:  40.00  % Time Elapsed:  33.24 s
Epoch 4 Finished.  Time per Epoch:   6.65 s
Iteration:  282 Progress:  50.00  % Time Elapsed:  39.94 s
Epoch 5 Finished.  Time per Epoch:   6.66 s
Iteration:  329 Progress:  60.00  % Time Elapsed:  46.55 s
Epoch 6 Finished.  Time per Epoch:   6.65 s
Iteration:  376 Progress:  70.00  % Time Elapsed:  53.24 s
Epoch 7 Finished.  Time per Epoch:   6.65 s
Iteration:  423 Progress:  80.00  % Time Elapsed:  59.99 s
Epoch 8 Finished.  Time per Epoch:   6.67 s
Iteration:  470 Progress:  90.00  % Time Elapsed:  66.62 s
Epoch 9 Finished.  Time per Epoch:   6.66 s
```




```
Final Training Accuracy: 0.7166101694915255
Final Validation Accuracy: 0.6901098901098901
```

Part (c) “Overfit” to a Small Dataset - 5 pt

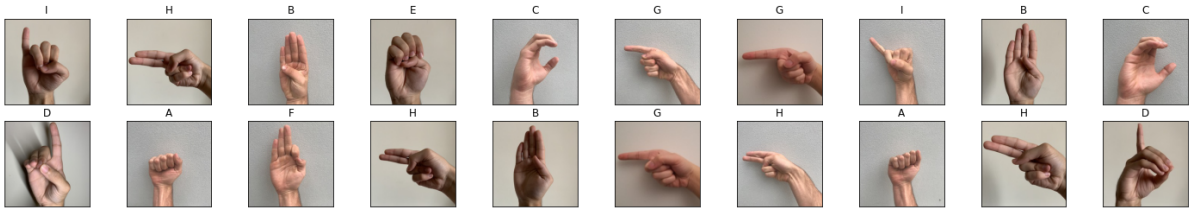
One way to sanity check our neural network model and training code is to check whether the model is capable of “overfitting” or “memorizing” a small dataset. A properly constructed CNN with correct training code should be able to memorize the answers to a small number of images quickly.

Construct a small dataset (e.g. just the images that you have collected). Then show that your model and training code is capable of memorizing the labels of this small data set.

With a large batch size (e.g. the entire small dataset) and learning rate that is not too high, You should be able to obtain a 100% training accuracy on that small dataset relatively quickly (within 200 iterations).

In [ ]:
```
# define training and test data directories
data_dir_small = '/root/datasets/'
train_dir_small = os.path.join(data_dir_small, 'Small/') # Only 54 images are
 in the small dataset

# resize and ensure all images to 224 x 224 pixels, random crop is not because
the images are created consistently
data_transform = transforms.Compose([transforms.Resize((224,224)),
                                     transforms.ToTensor()])

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

train_data_small = datasets.ImageFolder(train_dir_small, transform=data_transf
orm)
print(len(train_data_small))

# define dataloader parameters
batch_size  = 27 # process 27 images at a time
num_workers = 1 # we only need 1 worker here

# prepare data loaders
train_loader_small = torch.utils.data.DataLoader(train_data_small, batch_size=
batch_size,
                                                 num_workers=num_workers, shuffle=Tr
ue)
val_loader_small = train_loader_small

# obtain one batch of training images
dataiter = iter(train_loader_small)
images, labels = dataiter.next()
images = images.numpy() # convert images to numpy for display

# plot the images in the batch, along with the corresponding labels
fig = plt.figure(figsize=(25, 4))
for idx in np.arange(20):
    ax = fig.add_subplot(2, 20/2, idx+1, xticks=[], yticks=[])
    plt.imshow(np.transpose(images[idx], (1, 2, 0)))
    ax.set_title(classes[labels[idx]])
```

54



In [ ]:
```
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

train(model, train_loader_small,val_loader_small, batch_size=27,num_epochs=200
,lr=0.01 )
```

```
CUDA is available!  Training on GPU ...
Iteration:  2 Progress:   0.00  % Time Elapsed:   0.65 s
Epoch 0 Finished.  Time per Epoch:   0.65 s
Iteration:  4 Progress:   0.50  % Time Elapsed:   1.30 s
Epoch 1 Finished.  Time per Epoch:   0.65 s
Iteration:  6 Progress:   1.00  % Time Elapsed:   1.90 s
Epoch 2 Finished.  Time per Epoch:   0.63 s
Iteration:  8 Progress:   1.50  % Time Elapsed:   2.52 s
Epoch 3 Finished.  Time per Epoch:   0.63 s
Iteration:  10 Progress:   2.00  % Time Elapsed:   3.13 s
Epoch 4 Finished.  Time per Epoch:   0.63 s
Iteration:  12 Progress:   2.50  % Time Elapsed:   3.75 s
Epoch 5 Finished.  Time per Epoch:   0.62 s
Iteration:  14 Progress:   3.00  % Time Elapsed:   4.36 s
Epoch 6 Finished.  Time per Epoch:   0.62 s
Iteration:  16 Progress:   3.50  % Time Elapsed:   4.97 s
Epoch 7 Finished.  Time per Epoch:   0.62 s
Iteration:  18 Progress:   4.00  % Time Elapsed:   5.58 s
Epoch 8 Finished.  Time per Epoch:   0.62 s
Iteration:  20 Progress:   4.50  % Time Elapsed:   6.18 s
Epoch 9 Finished.  Time per Epoch:   0.62 s
Iteration:  22 Progress:   5.00  % Time Elapsed:   6.81 s
Epoch 10 Finished.  Time per Epoch:   0.62 s
Iteration:  24 Progress:   5.50  % Time Elapsed:   7.42 s
Epoch 11 Finished.  Time per Epoch:   0.62 s
Iteration:  26 Progress:   6.00  % Time Elapsed:   8.02 s
Epoch 12 Finished.  Time per Epoch:   0.62 s
Iteration:  28 Progress:   6.50  % Time Elapsed:   8.63 s
Epoch 13 Finished.  Time per Epoch:   0.62 s
Iteration:  30 Progress:   7.00  % Time Elapsed:   9.25 s
Epoch 14 Finished.  Time per Epoch:   0.62 s
Iteration:  32 Progress:   7.50  % Time Elapsed:   9.89 s
Epoch 15 Finished.  Time per Epoch:   0.62 s
Iteration:  34 Progress:   8.00  % Time Elapsed:  10.51 s
Epoch 16 Finished.  Time per Epoch:   0.62 s
Iteration:  36 Progress:   8.50  % Time Elapsed:  11.11 s
Epoch 17 Finished.  Time per Epoch:   0.62 s
Iteration:  38 Progress:   9.00  % Time Elapsed:  11.73 s
Epoch 18 Finished.  Time per Epoch:   0.62 s
Iteration:  40 Progress:   9.50  % Time Elapsed:  12.34 s
Epoch 19 Finished.  Time per Epoch:   0.62 s
Iteration:  42 Progress:  10.00  % Time Elapsed:  12.95 s
Epoch 20 Finished.  Time per Epoch:   0.62 s
Iteration:  44 Progress:  10.50  % Time Elapsed:  13.56 s
Epoch 21 Finished.  Time per Epoch:   0.62 s
Iteration:  46 Progress:  11.00  % Time Elapsed:  14.17 s
Epoch 22 Finished.  Time per Epoch:   0.62 s
Iteration:  48 Progress:  11.50  % Time Elapsed:  14.79 s
Epoch 23 Finished.  Time per Epoch:   0.62 s
Iteration:  50 Progress:  12.00  % Time Elapsed:  15.41 s
Epoch 24 Finished.  Time per Epoch:   0.62 s
Iteration:  52 Progress:  12.50  % Time Elapsed:  16.01 s
Epoch 25 Finished.  Time per Epoch:   0.62 s
Iteration:  54 Progress:  13.00  % Time Elapsed:  16.62 s
Epoch 26 Finished.  Time per Epoch:   0.62 s
Iteration:  56 Progress:  13.50  % Time Elapsed:  17.23 s
Epoch 27 Finished.  Time per Epoch:   0.62 s
```

```
Iteration:  58 Progress:  14.00  % Time Elapsed:  17.84 s
Epoch 28 Finished.  Time per Epoch:   0.62 s
Iteration:  60 Progress:  14.50  % Time Elapsed:  18.45 s
Epoch 29 Finished.  Time per Epoch:   0.61 s
Iteration:  62 Progress:  15.00  % Time Elapsed:  19.06 s
Epoch 30 Finished.  Time per Epoch:   0.61 s
Iteration:  64 Progress:  15.50  % Time Elapsed:  19.67 s
Epoch 31 Finished.  Time per Epoch:   0.61 s
Iteration:  66 Progress:  16.00  % Time Elapsed:  20.27 s
Epoch 32 Finished.  Time per Epoch:   0.61 s
Iteration:  68 Progress:  16.50  % Time Elapsed:  20.87 s
Epoch 33 Finished.  Time per Epoch:   0.61 s
Iteration:  70 Progress:  17.00  % Time Elapsed:  21.48 s
Epoch 34 Finished.  Time per Epoch:   0.61 s
Iteration:  72 Progress:  17.50  % Time Elapsed:  22.08 s
Epoch 35 Finished.  Time per Epoch:   0.61 s
Iteration:  74 Progress:  18.00  % Time Elapsed:  22.69 s
Epoch 36 Finished.  Time per Epoch:   0.61 s
Iteration:  76 Progress:  18.50  % Time Elapsed:  23.29 s
Epoch 37 Finished.  Time per Epoch:   0.61 s
Iteration:  78 Progress:  19.00  % Time Elapsed:  23.89 s
Epoch 38 Finished.  Time per Epoch:   0.61 s
Iteration:  80 Progress:  19.50  % Time Elapsed:  24.51 s
Epoch 39 Finished.  Time per Epoch:   0.61 s
Iteration:  82 Progress:  20.00  % Time Elapsed:  25.13 s
Epoch 40 Finished.  Time per Epoch:   0.61 s
Iteration:  84 Progress:  20.50  % Time Elapsed:  25.76 s
Epoch 41 Finished.  Time per Epoch:   0.61 s
Iteration:  86 Progress:  21.00  % Time Elapsed:  26.36 s
Epoch 42 Finished.  Time per Epoch:   0.61 s
Iteration:  88 Progress:  21.50  % Time Elapsed:  26.96 s
Epoch 43 Finished.  Time per Epoch:   0.61 s
Iteration:  90 Progress:  22.00  % Time Elapsed:  27.59 s
Epoch 44 Finished.  Time per Epoch:   0.61 s
Iteration:  92 Progress:  22.50  % Time Elapsed:  28.22 s
Epoch 45 Finished.  Time per Epoch:   0.61 s
Iteration:  94 Progress:  23.00  % Time Elapsed:  28.86 s
Epoch 46 Finished.  Time per Epoch:   0.61 s
Iteration:  96 Progress:  23.50  % Time Elapsed:  29.47 s
Epoch 47 Finished.  Time per Epoch:   0.61 s
Iteration:  98 Progress:  24.00  % Time Elapsed:  30.08 s
Epoch 48 Finished.  Time per Epoch:   0.61 s
Iteration:  100 Progress:  24.50  % Time Elapsed:  30.70 s
Epoch 49 Finished.  Time per Epoch:   0.61 s
Iteration:  102 Progress:  25.00  % Time Elapsed:  31.32 s
Epoch 50 Finished.  Time per Epoch:   0.61 s
Iteration:  104 Progress:  25.50  % Time Elapsed:  31.94 s
Epoch 51 Finished.  Time per Epoch:   0.61 s
Iteration:  106 Progress:  26.00  % Time Elapsed:  32.56 s
Epoch 52 Finished.  Time per Epoch:   0.61 s
Iteration:  108 Progress:  26.50  % Time Elapsed:  33.17 s
Epoch 53 Finished.  Time per Epoch:   0.61 s
Iteration:  110 Progress:  27.00  % Time Elapsed:  33.78 s
Epoch 54 Finished.  Time per Epoch:   0.61 s
Iteration:  112 Progress:  27.50  % Time Elapsed:  34.39 s
Epoch 55 Finished.  Time per Epoch:   0.61 s
Iteration:  114 Progress:  28.00  % Time Elapsed:  35.00 s
```

```
Epoch 56 Finished.  Time per Epoch:   0.61 s
Iteration:  116 Progress:  28.50  % Time Elapsed:  35.61 s
Epoch 57 Finished.  Time per Epoch:   0.61 s
Iteration:  118 Progress:  29.00  % Time Elapsed:  36.22 s
Epoch 58 Finished.  Time per Epoch:   0.61 s
Iteration:  120 Progress:  29.50  % Time Elapsed:  36.85 s
Epoch 59 Finished.  Time per Epoch:   0.61 s
Iteration:  122 Progress:  30.00  % Time Elapsed:  37.47 s
Epoch 60 Finished.  Time per Epoch:   0.61 s
Iteration:  124 Progress:  30.50  % Time Elapsed:  38.09 s
Epoch 61 Finished.  Time per Epoch:   0.61 s
Iteration:  126 Progress:  31.00  % Time Elapsed:  38.71 s
Epoch 62 Finished.  Time per Epoch:   0.61 s
Iteration:  128 Progress:  31.50  % Time Elapsed:  39.32 s
Epoch 63 Finished.  Time per Epoch:   0.61 s
Iteration:  130 Progress:  32.00  % Time Elapsed:  39.93 s
Epoch 64 Finished.  Time per Epoch:   0.61 s
Iteration:  132 Progress:  32.50  % Time Elapsed:  40.54 s
Epoch 65 Finished.  Time per Epoch:   0.61 s
Iteration:  134 Progress:  33.00  % Time Elapsed:  41.15 s
Epoch 66 Finished.  Time per Epoch:   0.61 s
Iteration:  136 Progress:  33.50  % Time Elapsed:  41.77 s
Epoch 67 Finished.  Time per Epoch:   0.61 s
Iteration:  138 Progress:  34.00  % Time Elapsed:  42.39 s
Epoch 68 Finished.  Time per Epoch:   0.61 s
Iteration:  140 Progress:  34.50  % Time Elapsed:  43.01 s
Epoch 69 Finished.  Time per Epoch:   0.61 s
Iteration:  142 Progress:  35.00  % Time Elapsed:  43.63 s
Epoch 70 Finished.  Time per Epoch:   0.61 s
Iteration:  144 Progress:  35.50  % Time Elapsed:  44.24 s
Epoch 71 Finished.  Time per Epoch:   0.61 s
Iteration:  146 Progress:  36.00  % Time Elapsed:  44.83 s
Epoch 72 Finished.  Time per Epoch:   0.61 s
Iteration:  148 Progress:  36.50  % Time Elapsed:  45.45 s
Epoch 73 Finished.  Time per Epoch:   0.61 s
Iteration:  150 Progress:  37.00  % Time Elapsed:  46.05 s
Epoch 74 Finished.  Time per Epoch:   0.61 s
Iteration:  152 Progress:  37.50  % Time Elapsed:  46.65 s
Epoch 75 Finished.  Time per Epoch:   0.61 s
Iteration:  154 Progress:  38.00  % Time Elapsed:  47.29 s
Epoch 76 Finished.  Time per Epoch:   0.61 s
Iteration:  156 Progress:  38.50  % Time Elapsed:  47.90 s
Epoch 77 Finished.  Time per Epoch:   0.61 s
Iteration:  158 Progress:  39.00  % Time Elapsed:  48.51 s
Epoch 78 Finished.  Time per Epoch:   0.61 s
Iteration:  160 Progress:  39.50  % Time Elapsed:  49.14 s
Epoch 79 Finished.  Time per Epoch:   0.61 s
Iteration:  162 Progress:  40.00  % Time Elapsed:  49.74 s
Epoch 80 Finished.  Time per Epoch:   0.61 s
Iteration:  164 Progress:  40.50  % Time Elapsed:  50.34 s
Epoch 81 Finished.  Time per Epoch:   0.61 s
Iteration:  166 Progress:  41.00  % Time Elapsed:  50.94 s
Epoch 82 Finished.  Time per Epoch:   0.61 s
Iteration:  168 Progress:  41.50  % Time Elapsed:  51.55 s
Epoch 83 Finished.  Time per Epoch:   0.61 s
Iteration:  170 Progress:  42.00  % Time Elapsed:  52.15 s
Epoch 84 Finished.  Time per Epoch:   0.61 s
```

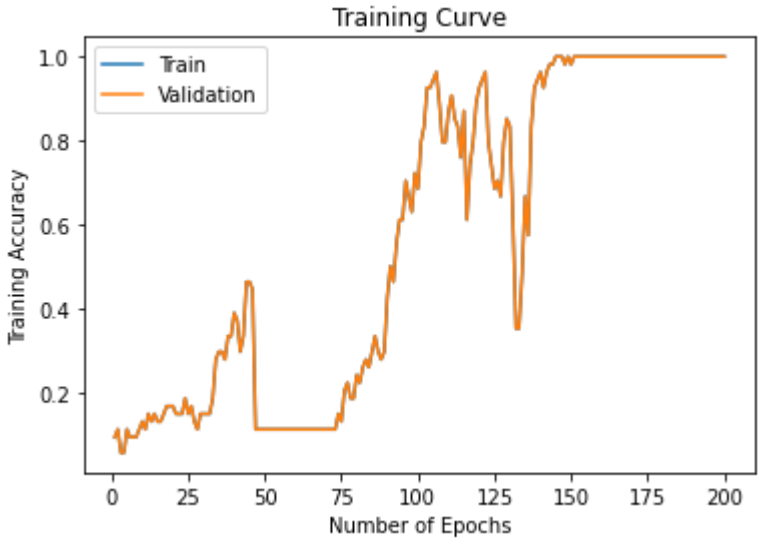
```
Iteration:  172 Progress:  42.50  % Time Elapsed:  52.75 s
Epoch 85 Finished.  Time per Epoch:   0.61 s
Iteration:  174 Progress:  43.00  % Time Elapsed:  53.37 s
Epoch 86 Finished.  Time per Epoch:   0.61 s
Iteration:  176 Progress:  43.50  % Time Elapsed:  53.99 s
Epoch 87 Finished.  Time per Epoch:   0.61 s
Iteration:  178 Progress:  44.00  % Time Elapsed:  54.61 s
Epoch 88 Finished.  Time per Epoch:   0.61 s
Iteration:  180 Progress:  44.50  % Time Elapsed:  55.23 s
Epoch 89 Finished.  Time per Epoch:   0.61 s
Iteration:  182 Progress:  45.00  % Time Elapsed:  55.83 s
Epoch 90 Finished.  Time per Epoch:   0.61 s
Iteration:  184 Progress:  45.50  % Time Elapsed:  56.43 s
Epoch 91 Finished.  Time per Epoch:   0.61 s
Iteration:  186 Progress:  46.00  % Time Elapsed:  57.03 s
Epoch 92 Finished.  Time per Epoch:   0.61 s
Iteration:  188 Progress:  46.50  % Time Elapsed:  57.67 s
Epoch 93 Finished.  Time per Epoch:   0.61 s
Iteration:  190 Progress:  47.00  % Time Elapsed:  58.27 s
Epoch 94 Finished.  Time per Epoch:   0.61 s
Iteration:  192 Progress:  47.50  % Time Elapsed:  58.86 s
Epoch 95 Finished.  Time per Epoch:   0.61 s
Iteration:  194 Progress:  48.00  % Time Elapsed:  59.49 s
Epoch 96 Finished.  Time per Epoch:   0.61 s
Iteration:  196 Progress:  48.50  % Time Elapsed:  60.09 s
Epoch 97 Finished.  Time per Epoch:   0.61 s
Iteration:  198 Progress:  49.00  % Time Elapsed:  60.70 s
Epoch 98 Finished.  Time per Epoch:   0.61 s
Iteration:  200 Progress:  49.50  % Time Elapsed:  61.30 s
Epoch 99 Finished.  Time per Epoch:   0.61 s
Iteration:  202 Progress:  50.00  % Time Elapsed:  61.90 s
Epoch 100 Finished.  Time per Epoch:   0.61 s
Iteration:  204 Progress:  50.50  % Time Elapsed:  62.51 s
Epoch 101 Finished.  Time per Epoch:   0.61 s
Iteration:  206 Progress:  51.00  % Time Elapsed:  63.10 s
Epoch 102 Finished.  Time per Epoch:   0.61 s
Iteration:  208 Progress:  51.50  % Time Elapsed:  63.71 s
Epoch 103 Finished.  Time per Epoch:   0.61 s
Iteration:  210 Progress:  52.00  % Time Elapsed:  64.32 s
Epoch 104 Finished.  Time per Epoch:   0.61 s
Iteration:  212 Progress:  52.50  % Time Elapsed:  64.92 s
Epoch 105 Finished.  Time per Epoch:   0.61 s
Iteration:  214 Progress:  53.00  % Time Elapsed:  65.53 s
Epoch 106 Finished.  Time per Epoch:   0.61 s
Iteration:  216 Progress:  53.50  % Time Elapsed:  66.14 s
Epoch 107 Finished.  Time per Epoch:   0.61 s
Iteration:  218 Progress:  54.00  % Time Elapsed:  66.74 s
Epoch 108 Finished.  Time per Epoch:   0.61 s
Iteration:  220 Progress:  54.50  % Time Elapsed:  67.36 s
Epoch 109 Finished.  Time per Epoch:   0.61 s
Iteration:  222 Progress:  55.00  % Time Elapsed:  67.99 s
Epoch 110 Finished.  Time per Epoch:   0.61 s
Iteration:  224 Progress:  55.50  % Time Elapsed:  68.60 s
Epoch 111 Finished.  Time per Epoch:   0.61 s
Iteration:  226 Progress:  56.00  % Time Elapsed:  69.21 s
Epoch 112 Finished.  Time per Epoch:   0.61 s
Iteration:  228 Progress:  56.50  % Time Elapsed:  69.82 s
```

```
Epoch 113 Finished.  Time per Epoch:   0.61 s
Iteration:  230 Progress:  57.00  % Time Elapsed:  70.43 s
Epoch 114 Finished.  Time per Epoch:   0.61 s
Iteration:  232 Progress:  57.50  % Time Elapsed:  71.04 s
Epoch 115 Finished.  Time per Epoch:   0.61 s
Iteration:  234 Progress:  58.00  % Time Elapsed:  71.65 s
Epoch 116 Finished.  Time per Epoch:   0.61 s
Iteration:  236 Progress:  58.50  % Time Elapsed:  72.25 s
Epoch 117 Finished.  Time per Epoch:   0.61 s
Iteration:  238 Progress:  59.00  % Time Elapsed:  72.86 s
Epoch 118 Finished.  Time per Epoch:   0.61 s
Iteration:  240 Progress:  59.50  % Time Elapsed:  73.49 s
Epoch 119 Finished.  Time per Epoch:   0.61 s
Iteration:  242 Progress:  60.00  % Time Elapsed:  74.09 s
Epoch 120 Finished.  Time per Epoch:   0.61 s
Iteration:  244 Progress:  60.50  % Time Elapsed:  74.72 s
Epoch 121 Finished.  Time per Epoch:   0.61 s
Iteration:  246 Progress:  61.00  % Time Elapsed:  75.35 s
Epoch 122 Finished.  Time per Epoch:   0.61 s
Iteration:  248 Progress:  61.50  % Time Elapsed:  75.95 s
Epoch 123 Finished.  Time per Epoch:   0.61 s
Iteration:  250 Progress:  62.00  % Time Elapsed:  76.59 s
Epoch 124 Finished.  Time per Epoch:   0.61 s
Iteration:  252 Progress:  62.50  % Time Elapsed:  77.22 s
Epoch 125 Finished.  Time per Epoch:   0.61 s
Iteration:  254 Progress:  63.00  % Time Elapsed:  77.85 s
Epoch 126 Finished.  Time per Epoch:   0.61 s
Iteration:  256 Progress:  63.50  % Time Elapsed:  78.47 s
Epoch 127 Finished.  Time per Epoch:   0.61 s
Iteration:  258 Progress:  64.00  % Time Elapsed:  79.09 s
Epoch 128 Finished.  Time per Epoch:   0.61 s
Iteration:  260 Progress:  64.50  % Time Elapsed:  79.70 s
Epoch 129 Finished.  Time per Epoch:   0.61 s
Iteration:  262 Progress:  65.00  % Time Elapsed:  80.32 s
Epoch 130 Finished.  Time per Epoch:   0.61 s
Iteration:  264 Progress:  65.50  % Time Elapsed:  80.93 s
Epoch 131 Finished.  Time per Epoch:   0.61 s
Iteration:  266 Progress:  66.00  % Time Elapsed:  81.55 s
Epoch 132 Finished.  Time per Epoch:   0.61 s
Iteration:  268 Progress:  66.50  % Time Elapsed:  82.15 s
Epoch 133 Finished.  Time per Epoch:   0.61 s
Iteration:  270 Progress:  67.00  % Time Elapsed:  82.78 s
Epoch 134 Finished.  Time per Epoch:   0.61 s
Iteration:  272 Progress:  67.50  % Time Elapsed:  83.39 s
Epoch 135 Finished.  Time per Epoch:   0.61 s
Iteration:  274 Progress:  68.00  % Time Elapsed:  84.00 s
Epoch 136 Finished.  Time per Epoch:   0.61 s
Iteration:  276 Progress:  68.50  % Time Elapsed:  84.62 s
Epoch 137 Finished.  Time per Epoch:   0.61 s
Iteration:  278 Progress:  69.00  % Time Elapsed:  85.23 s
Epoch 138 Finished.  Time per Epoch:   0.61 s
Iteration:  280 Progress:  69.50  % Time Elapsed:  85.85 s
Epoch 139 Finished.  Time per Epoch:   0.61 s
Iteration:  282 Progress:  70.00  % Time Elapsed:  86.46 s
Epoch 140 Finished.  Time per Epoch:   0.61 s
Iteration:  284 Progress:  70.50  % Time Elapsed:  87.07 s
Epoch 141 Finished.  Time per Epoch:   0.61 s
```

```
Iteration:  286 Progress:  71.00  % Time Elapsed:  87.71 s
Epoch 142 Finished.  Time per Epoch:   0.61 s
Iteration:  288 Progress:  71.50  % Time Elapsed:  88.33 s
Epoch 143 Finished.  Time per Epoch:   0.61 s
Iteration:  290 Progress:  72.00  % Time Elapsed:  88.95 s
Epoch 144 Finished.  Time per Epoch:   0.61 s
Iteration:  292 Progress:  72.50  % Time Elapsed:  89.57 s
Epoch 145 Finished.  Time per Epoch:   0.61 s
Iteration:  294 Progress:  73.00  % Time Elapsed:  90.18 s
Epoch 146 Finished.  Time per Epoch:   0.61 s
Iteration:  296 Progress:  73.50  % Time Elapsed:  90.79 s
Epoch 147 Finished.  Time per Epoch:   0.61 s
Iteration:  298 Progress:  74.00  % Time Elapsed:  91.42 s
Epoch 148 Finished.  Time per Epoch:   0.61 s
Iteration:  300 Progress:  74.50  % Time Elapsed:  92.05 s
Epoch 149 Finished.  Time per Epoch:   0.61 s
Iteration:  302 Progress:  75.00  % Time Elapsed:  92.67 s
Epoch 150 Finished.  Time per Epoch:   0.61 s
Iteration:  304 Progress:  75.50  % Time Elapsed:  93.27 s
Epoch 151 Finished.  Time per Epoch:   0.61 s
Iteration:  306 Progress:  76.00  % Time Elapsed:  93.89 s
Epoch 152 Finished.  Time per Epoch:   0.61 s
Iteration:  308 Progress:  76.50  % Time Elapsed:  94.50 s
Epoch 153 Finished.  Time per Epoch:   0.61 s
Iteration:  310 Progress:  77.00  % Time Elapsed:  95.12 s
Epoch 154 Finished.  Time per Epoch:   0.61 s
Iteration:  312 Progress:  77.50  % Time Elapsed:  95.73 s
Epoch 155 Finished.  Time per Epoch:   0.61 s
Iteration:  314 Progress:  78.00  % Time Elapsed:  96.33 s
Epoch 156 Finished.  Time per Epoch:   0.61 s
Iteration:  316 Progress:  78.50  % Time Elapsed:  96.94 s
Epoch 157 Finished.  Time per Epoch:   0.61 s
Iteration:  318 Progress:  79.00  % Time Elapsed:  97.55 s
Epoch 158 Finished.  Time per Epoch:   0.61 s
Iteration:  320 Progress:  79.50  % Time Elapsed:  98.17 s
Epoch 159 Finished.  Time per Epoch:   0.61 s
Iteration:  322 Progress:  80.00  % Time Elapsed:  98.78 s
Epoch 160 Finished.  Time per Epoch:   0.61 s
Iteration:  324 Progress:  80.50  % Time Elapsed:  99.39 s
Epoch 161 Finished.  Time per Epoch:   0.61 s
Iteration:  326 Progress:  81.00  % Time Elapsed:  100.01 s
Epoch 162 Finished.  Time per Epoch:   0.61 s
Iteration:  328 Progress:  81.50  % Time Elapsed:  100.61 s
Epoch 163 Finished.  Time per Epoch:   0.61 s
Iteration:  330 Progress:  82.00  % Time Elapsed:  101.22 s
Epoch 164 Finished.  Time per Epoch:   0.61 s
Iteration:  332 Progress:  82.50  % Time Elapsed:  101.83 s
Epoch 165 Finished.  Time per Epoch:   0.61 s
Iteration:  334 Progress:  83.00  % Time Elapsed:  102.44 s
Epoch 166 Finished.  Time per Epoch:   0.61 s
Iteration:  336 Progress:  83.50  % Time Elapsed:  103.06 s
Epoch 167 Finished.  Time per Epoch:   0.61 s
Iteration:  338 Progress:  84.00  % Time Elapsed:  103.66 s
Epoch 168 Finished.  Time per Epoch:   0.61 s
Iteration:  340 Progress:  84.50  % Time Elapsed:  104.27 s
Epoch 169 Finished.  Time per Epoch:   0.61 s
Iteration:  342 Progress:  85.00  % Time Elapsed:  104.87 s
```

```
Epoch 170 Finished.  Time per Epoch:   0.61 s
Iteration:  344 Progress:  85.50  % Time Elapsed:  105.48 s
Epoch 171 Finished.  Time per Epoch:   0.61 s
Iteration:  346 Progress:  86.00  % Time Elapsed:  106.09 s
Epoch 172 Finished.  Time per Epoch:   0.61 s
Iteration:  348 Progress:  86.50  % Time Elapsed:  106.70 s
Epoch 173 Finished.  Time per Epoch:   0.61 s
Iteration:  350 Progress:  87.00  % Time Elapsed:  107.30 s
Epoch 174 Finished.  Time per Epoch:   0.61 s
Iteration:  352 Progress:  87.50  % Time Elapsed:  107.91 s
Epoch 175 Finished.  Time per Epoch:   0.61 s
Iteration:  354 Progress:  88.00  % Time Elapsed:  108.53 s
Epoch 176 Finished.  Time per Epoch:   0.61 s
Iteration:  356 Progress:  88.50  % Time Elapsed:  109.15 s
Epoch 177 Finished.  Time per Epoch:   0.61 s
Iteration:  358 Progress:  89.00  % Time Elapsed:  109.75 s
Epoch 178 Finished.  Time per Epoch:   0.61 s
Iteration:  360 Progress:  89.50  % Time Elapsed:  110.38 s
Epoch 179 Finished.  Time per Epoch:   0.61 s
Iteration:  362 Progress:  90.00  % Time Elapsed:  110.98 s
Epoch 180 Finished.  Time per Epoch:   0.61 s
Iteration:  364 Progress:  90.50  % Time Elapsed:  111.60 s
Epoch 181 Finished.  Time per Epoch:   0.61 s
Iteration:  366 Progress:  91.00  % Time Elapsed:  112.21 s
Epoch 182 Finished.  Time per Epoch:   0.61 s
Iteration:  368 Progress:  91.50  % Time Elapsed:  112.81 s
Epoch 183 Finished.  Time per Epoch:   0.61 s
Iteration:  370 Progress:  92.00  % Time Elapsed:  113.43 s
Epoch 184 Finished.  Time per Epoch:   0.61 s
Iteration:  372 Progress:  92.50  % Time Elapsed:  114.05 s
Epoch 185 Finished.  Time per Epoch:   0.61 s
Iteration:  374 Progress:  93.00  % Time Elapsed:  114.68 s
Epoch 186 Finished.  Time per Epoch:   0.61 s
Iteration:  376 Progress:  93.50  % Time Elapsed:  115.31 s
Epoch 187 Finished.  Time per Epoch:   0.61 s
Iteration:  378 Progress:  94.00  % Time Elapsed:  115.92 s
Epoch 188 Finished.  Time per Epoch:   0.61 s
Iteration:  380 Progress:  94.50  % Time Elapsed:  116.54 s
Epoch 189 Finished.  Time per Epoch:   0.61 s
Iteration:  382 Progress:  95.00  % Time Elapsed:  117.16 s
Epoch 190 Finished.  Time per Epoch:   0.61 s
Iteration:  384 Progress:  95.50  % Time Elapsed:  117.77 s
Epoch 191 Finished.  Time per Epoch:   0.61 s
Iteration:  386 Progress:  96.00  % Time Elapsed:  118.40 s
Epoch 192 Finished.  Time per Epoch:   0.61 s
Iteration:  388 Progress:  96.50  % Time Elapsed:  119.01 s
Epoch 193 Finished.  Time per Epoch:   0.61 s
Iteration:  390 Progress:  97.00  % Time Elapsed:  119.63 s
Epoch 194 Finished.  Time per Epoch:   0.61 s
Iteration:  392 Progress:  97.50  % Time Elapsed:  120.25 s
Epoch 195 Finished.  Time per Epoch:   0.61 s
Iteration:  394 Progress:  98.00  % Time Elapsed:  120.86 s
Epoch 196 Finished.  Time per Epoch:   0.61 s
Iteration:  396 Progress:  98.50  % Time Elapsed:  121.49 s
Epoch 197 Finished.  Time per Epoch:   0.61 s
Iteration:  398 Progress:  99.00  % Time Elapsed:  122.11 s
Epoch 198 Finished.  Time per Epoch:   0.61 s
```

```
Iteration:  400 Progress:  99.50  % Time Elapsed:  122.71 s
Epoch 199 Finished.  Time per Epoch:   0.61 s
```




```
Final Training Accuracy: 1.0
Final Validation Accuracy: 1.0
```

Based on the results, the model can memorize the small dataset and achieve 100 accuracy in these images' recognition. Therefore, the model can casue overfitting and thus is valid. Validation accuracy is not important but just for running the code. Validation accuracy is 1 because val_data has the same data as the train_data.

3. Hyperparameter Search [10 pt]

Part (a) - 1 pt

List 3 hyperparameters that you think are most worth tuning. Choose at least one hyperparameter related to the model architecture.

In [ ]:
```
# I choose the number of epochs, batch size and the number of convolutional/hi
dden layers.
```

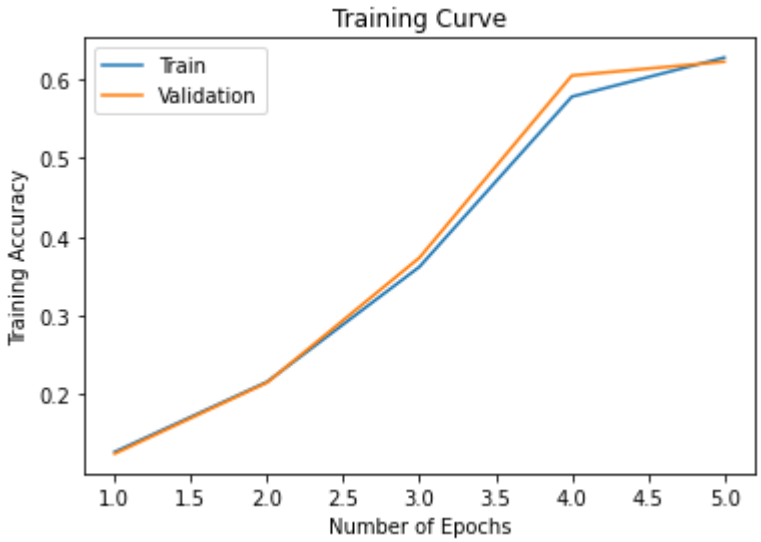
Part (b) - 5 pt

Tune the hyperparameters you listed in Part (a), trying as many values as you need to until you feel satisfied that you are getting a good model. Plot the training curve of at least 4 different hyperparameter settings.

In [ ]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F

import matplotlib.pyplot as plt # for plotting
import torch.optim as optim #for gradient descent

torch.manual_seed(1) # set the random seed

class HandGestureClassifier(nn.Module):
    def __init__(self):
        super(HandGestureClassifier, self).__init__()
        self.conv1 = nn.Conv2d(3, 5, 5) #3 in_channels, 5 out_chanels, 5 kerne
l_size 224*224*3 -> 220*220*5
        #self.pool2 = nn.MaxPool2d(4, 4) #kernel_size, stride 220*220*5 -> 55*
55*5
        self.conv2 = nn.Conv2d(5, 10, 6) #5 in_channels, 10 out_chanels, 6 ker
nel_size 55*55*5 -> 50*50*10
        self.pool = nn.MaxPool2d(2, 2) #2 kernel_size, 2 stride 50*50*10 -> 25
*25*10
        self.conv3 = nn.Conv2d(10, 10, 10) #10 in_channels, 10 out_chanels, 10
kernel_size 25*25*10 -> 16*16*10
        self.pool2 = nn.MaxPool2d(4, 4) #4 kernel_size, 4 stride 16*16*10 -> 4
*4*10
        self.fc1 = nn.Linear(160, 32) #Fully connected layer from now on, hidd
en units = 32 neurons with 1 hidden layer
        #self.fc1add = nn.Linear(64, 32) #Fully connected layer from now on, h
idden units = 32 neurons with 1 hidden layer
        self.fc2 = nn.Linear(32, 9)  # Softmax activation functions will be us
ed with 9 ouputs (9 Probabilities)

    def forward(self, x):
        x = self.pool2(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = self.pool2(F.relu(self.conv3(x)))
        x = x.view(-1, 160) # reshape the results, feed them into an fully con
nected network, and apply relu activation functions
        x = F.relu(self.fc1(x)) # relu activation function
        #x = F.relu(self.fc1add(x))
        x = self.fc2(x) # Softmax will be applied at crossentropyloss at the o
utput to get the 9 Probabilities

        return x
```

In [ ]:
```
# 1 number of epoches = 5
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=32,num_epochs=5,lr=0.01 )
```

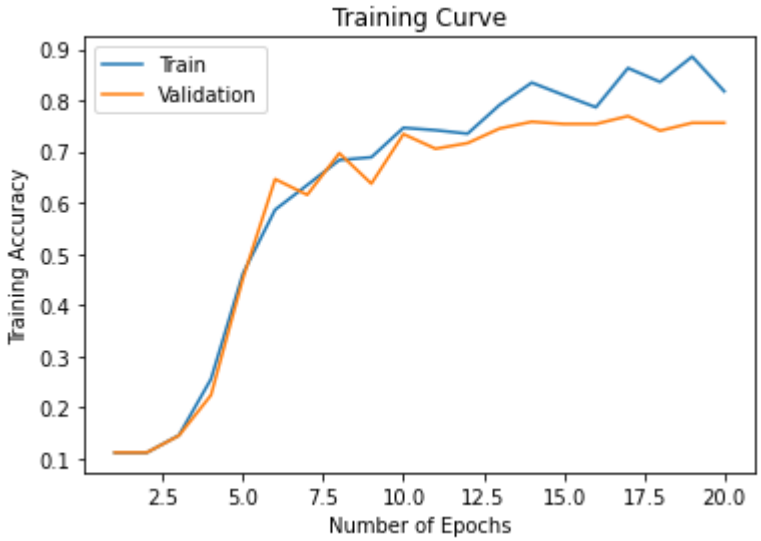
```
CUDA is available!  Training on GPU ...
Iteration:  47 Progress:   0.00  % Time Elapsed:   6.80 s
Epoch 0 Finished.  Time per Epoch:   6.80 s
Iteration:  94 Progress:  20.00  % Time Elapsed:  13.50 s
Epoch 1 Finished.  Time per Epoch:   6.75 s
Iteration:  141 Progress:  40.00  % Time Elapsed:  20.26 s
Epoch 2 Finished.  Time per Epoch:   6.76 s
Iteration:  188 Progress:  60.00  % Time Elapsed:  27.10 s
Epoch 3 Finished.  Time per Epoch:   6.78 s
Iteration:  235 Progress:  80.00  % Time Elapsed:  33.85 s
Epoch 4 Finished.  Time per Epoch:   6.77 s
```




```
Final Training Accuracy: 0.6271186440677966
Final Validation Accuracy: 0.621978021978022
```

In [ ]:
```
# 2 number of epoches = 20
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=32,num_epochs=20,lr=0.01 )
```

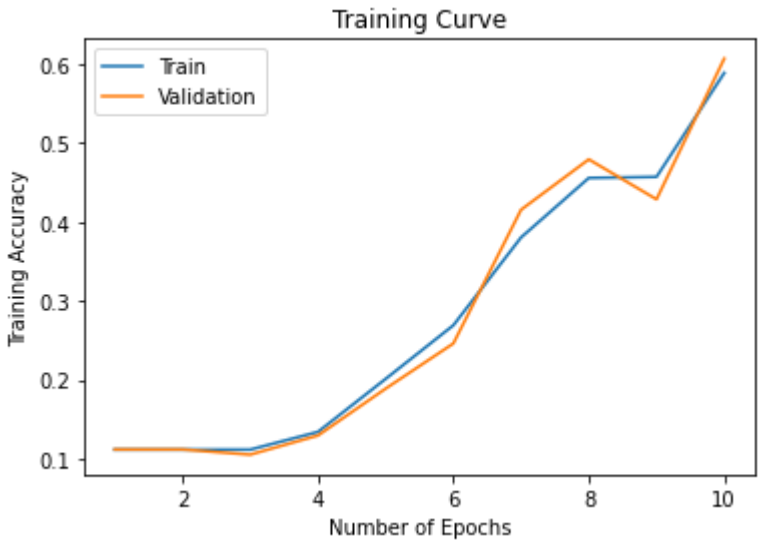
```
CUDA is available!  Training on GPU ...
Iteration:  47 Progress:   0.00  % Time Elapsed:   6.82 s
Epoch 0 Finished.  Time per Epoch:   6.82 s
Iteration:  94 Progress:   5.00  % Time Elapsed:  13.63 s
Epoch 1 Finished.  Time per Epoch:   6.81 s
Iteration:  141 Progress:  10.00  % Time Elapsed:  20.44 s
Epoch 2 Finished.  Time per Epoch:   6.82 s
Iteration:  188 Progress:  15.00  % Time Elapsed:  27.32 s
Epoch 3 Finished.  Time per Epoch:   6.83 s
Iteration:  235 Progress:  20.00  % Time Elapsed:  34.14 s
Epoch 4 Finished.  Time per Epoch:   6.83 s
Iteration:  282 Progress:  25.00  % Time Elapsed:  40.96 s
Epoch 5 Finished.  Time per Epoch:   6.83 s
Iteration:  329 Progress:  30.00  % Time Elapsed:  47.92 s
Epoch 6 Finished.  Time per Epoch:   6.85 s
Iteration:  376 Progress:  35.00  % Time Elapsed:  54.74 s
Epoch 7 Finished.  Time per Epoch:   6.84 s
Iteration:  423 Progress:  40.00  % Time Elapsed:  61.57 s
Epoch 8 Finished.  Time per Epoch:   6.84 s
Iteration:  470 Progress:  45.00  % Time Elapsed:  68.34 s
Epoch 9 Finished.  Time per Epoch:   6.83 s
Iteration:  517 Progress:  50.00  % Time Elapsed:  75.22 s
Epoch 10 Finished.  Time per Epoch:   6.84 s
Iteration:  564 Progress:  55.00  % Time Elapsed:  82.09 s
Epoch 11 Finished.  Time per Epoch:   6.84 s
Iteration:  611 Progress:  60.00  % Time Elapsed:  88.85 s
Epoch 12 Finished.  Time per Epoch:   6.83 s
Iteration:  658 Progress:  65.00  % Time Elapsed:  95.71 s
Epoch 13 Finished.  Time per Epoch:   6.84 s
Iteration:  705 Progress:  70.00  % Time Elapsed:  102.51 s
Epoch 14 Finished.  Time per Epoch:   6.83 s
Iteration:  752 Progress:  75.00  % Time Elapsed:  109.49 s
Epoch 15 Finished.  Time per Epoch:   6.84 s
Iteration:  799 Progress:  80.00  % Time Elapsed:  116.28 s
Epoch 16 Finished.  Time per Epoch:   6.84 s
Iteration:  846 Progress:  85.00  % Time Elapsed:  123.13 s
Epoch 17 Finished.  Time per Epoch:   6.84 s
Iteration:  893 Progress:  90.00  % Time Elapsed:  129.92 s
Epoch 18 Finished.  Time per Epoch:   6.84 s
Iteration:  940 Progress:  95.00  % Time Elapsed:  136.72 s
Epoch 19 Finished.  Time per Epoch:   6.84 s
```




```
Final Training Accuracy: 0.8176271186440678
Final Validation Accuracy: 0.756043956043956
```

In [ ]:
```
# 3 batch size = 64
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=64,num_epochs=10,lr=0.01 )
```

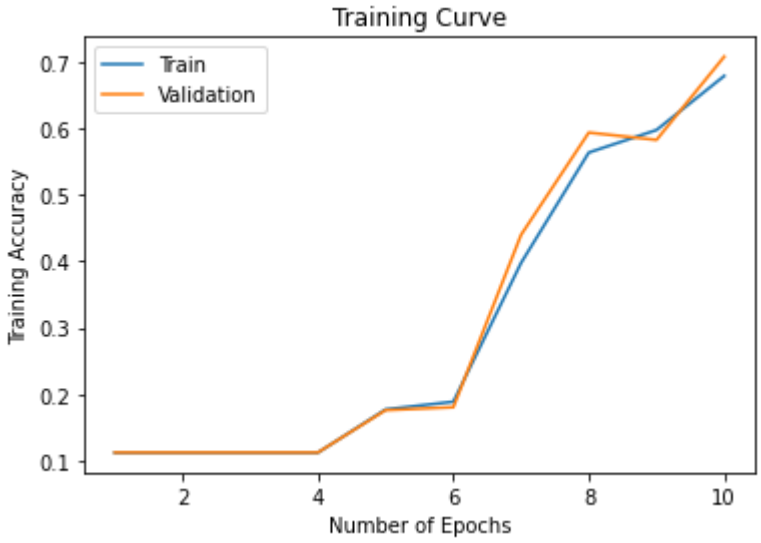
```
CUDA is available!  Training on GPU ...
Iteration:  24 Progress:   0.00  % Time Elapsed:   6.71 s
Epoch 0 Finished.  Time per Epoch:   6.71 s
Iteration:  48 Progress:  10.00  % Time Elapsed:  13.30 s
Epoch 1 Finished.  Time per Epoch:   6.65 s
Iteration:  72 Progress:  20.00  % Time Elapsed:  19.96 s
Epoch 2 Finished.  Time per Epoch:   6.65 s
Iteration:  96 Progress:  30.00  % Time Elapsed:  26.57 s
Epoch 3 Finished.  Time per Epoch:   6.64 s
Iteration:  120 Progress:  40.00  % Time Elapsed:  33.28 s
Epoch 4 Finished.  Time per Epoch:   6.66 s
Iteration:  144 Progress:  50.00  % Time Elapsed:  39.98 s
Epoch 5 Finished.  Time per Epoch:   6.66 s
Iteration:  168 Progress:  60.00  % Time Elapsed:  46.64 s
Epoch 6 Finished.  Time per Epoch:   6.66 s
Iteration:  192 Progress:  70.00  % Time Elapsed:  53.33 s
Epoch 7 Finished.  Time per Epoch:   6.67 s
Iteration:  216 Progress:  80.00  % Time Elapsed:  59.96 s
Epoch 8 Finished.  Time per Epoch:   6.66 s
Iteration:  240 Progress:  90.00  % Time Elapsed:  66.62 s
Epoch 9 Finished.  Time per Epoch:   6.66 s
```




```
Final Training Accuracy: 0.5884745762711865
Final Validation Accuracy: 0.6065934065934065
```

In [ ]:
```
# 4 Add one more Hidden Layer
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=32,num_epochs=10,lr=0.01 )
```

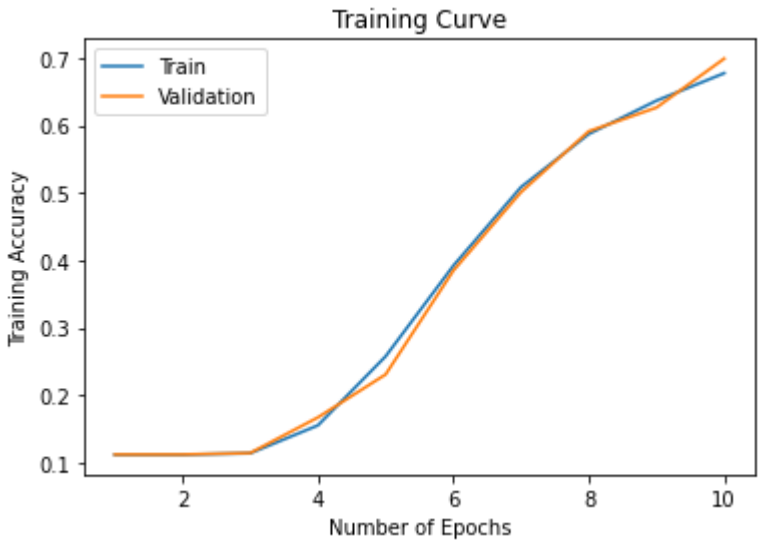
```
CUDA is available!  Training on GPU ...
Iteration:  47 Progress:   0.00  % Time Elapsed:   6.99 s
Epoch 0 Finished.  Time per Epoch:   6.99 s
Iteration:  94 Progress:  10.00  % Time Elapsed:  13.89 s
Epoch 1 Finished.  Time per Epoch:   6.95 s
Iteration:  141 Progress:  20.00  % Time Elapsed:  20.78 s
Epoch 2 Finished.  Time per Epoch:   6.93 s
Iteration:  188 Progress:  30.00  % Time Elapsed:  27.68 s
Epoch 3 Finished.  Time per Epoch:   6.92 s
Iteration:  235 Progress:  40.00  % Time Elapsed:  34.48 s
Epoch 4 Finished.  Time per Epoch:   6.90 s
Iteration:  282 Progress:  50.00  % Time Elapsed:  41.38 s
Epoch 5 Finished.  Time per Epoch:   6.90 s
Iteration:  329 Progress:  60.00  % Time Elapsed:  48.25 s
Epoch 6 Finished.  Time per Epoch:   6.89 s
Iteration:  376 Progress:  70.00  % Time Elapsed:  55.14 s
Epoch 7 Finished.  Time per Epoch:   6.89 s
Iteration:  423 Progress:  80.00  % Time Elapsed:  62.00 s
Epoch 8 Finished.  Time per Epoch:   6.89 s
Iteration:  470 Progress:  90.00  % Time Elapsed:  68.85 s
Epoch 9 Finished.  Time per Epoch:   6.89 s
```




```
Final Training Accuracy: 0.6786440677966101
Final Validation Accuracy: 0.7076923076923077
```

In [ ]:
```
# 5 batch size = 80
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader, batch_size=80,num_epochs=10,lr=0.01 )
```

```
CUDA is available!  Training on GPU ...
Iteration:  19 Progress:   0.00  % Time Elapsed:   6.59 s
Epoch 0 Finished.  Time per Epoch:   6.59 s
Iteration:  38 Progress:  10.00  % Time Elapsed:  13.20 s
Epoch 1 Finished.  Time per Epoch:   6.60 s
Iteration:  57 Progress:  20.00  % Time Elapsed:  19.82 s
Epoch 2 Finished.  Time per Epoch:   6.61 s
Iteration:  76 Progress:  30.00  % Time Elapsed:  26.37 s
Epoch 3 Finished.  Time per Epoch:   6.59 s
Iteration:  95 Progress:  40.00  % Time Elapsed:  32.89 s
Epoch 4 Finished.  Time per Epoch:   6.58 s
Iteration:  114 Progress:  50.00  % Time Elapsed:  39.44 s
Epoch 5 Finished.  Time per Epoch:   6.57 s
Iteration:  133 Progress:  60.00  % Time Elapsed:  46.04 s
Epoch 6 Finished.  Time per Epoch:   6.58 s
Iteration:  152 Progress:  70.00  % Time Elapsed:  52.57 s
Epoch 7 Finished.  Time per Epoch:   6.57 s
Iteration:  171 Progress:  80.00  % Time Elapsed:  59.08 s
Epoch 8 Finished.  Time per Epoch:   6.56 s
Iteration:  190 Progress:  90.00  % Time Elapsed:  65.63 s
Epoch 9 Finished.  Time per Epoch:   6.56 s
```




```
Final Training Accuracy: 0.6772881355932203
Final Validation Accuracy: 0.6989010989010989
```

Part (c) - 2 pt

Choose the best model out of all the ones that you have trained. Justify your choice.

In [ ]:
```
#The best one is the second one that only increases the number of epochs to 2
0. This model is the best because it has the highest Final Training Accuracy
 (0.8176271186440678)
#and Final Validation Accuracy (0.756043956043956). Moreover, based on the fiv
e models that I have tried above, model 2 also has the highest accuracy and th
e greatest improvement in accuracies
#than the first trial. Therefore, the second model is the best in my opinion.
```

Part (d) - 2 pt

Report the test accuracy of your best model. You should only do this step once and prior to this step you should have only used the training and validation data.

In [ ]:
```
import time
import os
import numpy as np
import torch

import torchvision
from torchvision import datasets, models, transforms
import matplotlib.pyplot as plt


# define training and test data directories
data_dir = '/root/datasets/'
train_dir = os.path.join(data_dir, 'train/')
val_dir = os.path.join(data_dir, 'val/')
test_dir = os.path.join(data_dir, 'test/')


# load and transform data using ImageFolder
# Data was already split randomly into train, validation, test data sets with
 approximately 60%, 20% and 20% of the total samples

# resize and ensure all images to 224 x 224 pixels, random crop is not because
the images are created consistently
data_transform = transforms.Compose([transforms.Resize((224,224)),
                                     transforms.ToTensor()])

train_data = datasets.ImageFolder(train_dir, transform=data_transform)
val_data = datasets.ImageFolder(val_dir, transform=data_transform)
test_data = datasets.ImageFolder(test_dir, transform=data_transform)

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# print out some data stats
print('Num training images: ', len(train_data))
print('Num validation images: ', len(val_data))
print('Num test images: ', len(test_data))

# define dataloader parameters
batch_size  = 32 # process 32 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders
train_loader = torch.utils.data.DataLoader(train_data, batch_size=batch_size,
                                           num_workers=num_workers, shuffle=Tr
ue)
val_loader = torch.utils.data.DataLoader(val_data, batch_size=batch_size,
                                         num_workers=num_workers, shuffle=Tru
e)
test_loader = torch.utils.data.DataLoader(test_data, batch_size=batch_size,
                                          num_workers=num_workers, shuffle=Tru
e)
```

```
Num training images:  1475
Num validation images:  455
Num test images:  501
```

In [ ]:
```python
def get_accuracy(model, data_loader):

    correct = 0
    total = 0
    for imgs, labels in data_loader:


        ################################################
        #To Enable GPU Usage
        if use_cuda and torch.cuda.is_available():
          imgs = imgs.cuda()
          labels = labels.cuda()
        ################################################

        output = model(imgs)

        #select index with maximum prediction score
        pred = output.max(1, keepdim=True)[1]
        correct += pred.eq(labels.view_as(pred)).sum().item()
        total += imgs.shape[0]
    return correct / total
```

In [ ]:
```
def train(model, train_load,val_load,test_load, batch_size=32, num_epochs=1,lr
=0.01 ):
    torch.manual_seed(1000)
    criterion = nn.CrossEntropyLoss() # softmax activation function is applied
and cross entropy loss is used for classification problem
    optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9) # stochasti
c gradient descent is used with momentum to improve efficiency and a learning
 rate of 0.01

    iteration, n_epochs, train_acc, val_acc,test_acc = [], [], [], [],[]

    # training
    n = 0 # the number of iterations
    start_time=time.time()
    for epoch in range(num_epochs):
        nepoch = epoch + 1
        n_epochs.append(nepoch)

        for imgs, labels in iter(train_load):


            ##############################################
            #To Enable GPU Usage
            if use_cuda and torch.cuda.is_available():
              imgs = imgs.cuda()
              labels = labels.cuda()
            ##############################################


            out = model(imgs)             # forward pass
            loss = criterion(out, labels) # compute the total loss
            loss.backward()               # backward pass (compute parameter u
pdates)
            optimizer.step()              # make the updates for each paramete
r
            optimizer.zero_grad()         # a clean up step for PyTorch
            n += 1
            iteration.append(n)


            # save the current training information
        val_acc.append(get_accuracy(model, val_load))  # compute validation ac
curacy
        train_acc.append(get_accuracy(model, train_load))  # compute validatio
n accuracy
        test_acc.append(get_accuracy(model, test_load))

        print("Iteration: ",n,'Progress: % 6.2f ' % ((epoch * len(train_load))
/ (num_epochs * len(train_load))*100),'%', "Time Elapsed: % 6.2f s " % (time.t
ime()-start_time))

        print ("Epoch %d Finished. " % epoch ,"Time per Epoch: % 6.2f s "% ((t
ime.time()-start_time) / (epoch +1)))


    end_time= time.time()
```

```
#plt.title("Training Curve")
#plt.plot(iteration, train_acc, label="Train")
#plt.plot(iteration, val_acc, label="Validation")
#plt.xlabel("Number of Iterations")
#plt.ylabel("Training Accuracy")
#plt.legend(loc='best')
#plt.show()

print("Final Training Accuracy: {}".format(train_acc[-1]))
print("Final Validation Accuracy: {}".format(val_acc[-1]))
print("Final Test Accuracy: {}".format(test_acc[-1]))
```

In [ ]:
```
# 2 number of epoches = 20
use_cuda = True

model = HandGestureClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model

train(model, train_loader,val_loader,test_loader, batch_size=32,num_epochs=20,
lr=0.01 )
```

```
CUDA is available!  Training on GPU ...
Iteration:  47 Progress:   0.00  % Time Elapsed:   7.84 s
Epoch 0 Finished.  Time per Epoch:   7.84 s
Iteration:  94 Progress:   5.00  % Time Elapsed:  15.66 s
Epoch 1 Finished.  Time per Epoch:   7.83 s
Iteration:  141 Progress:  10.00  % Time Elapsed:  23.50 s
Epoch 2 Finished.  Time per Epoch:   7.83 s
Iteration:  188 Progress:  15.00  % Time Elapsed:  31.42 s
Epoch 3 Finished.  Time per Epoch:   7.85 s
Iteration:  235 Progress:  20.00  % Time Elapsed:  39.23 s
Epoch 4 Finished.  Time per Epoch:   7.85 s
Iteration:  282 Progress:  25.00  % Time Elapsed:  47.01 s
Epoch 5 Finished.  Time per Epoch:   7.84 s
Iteration:  329 Progress:  30.00  % Time Elapsed:  54.86 s
Epoch 6 Finished.  Time per Epoch:   7.84 s
Iteration:  376 Progress:  35.00  % Time Elapsed:  62.80 s
Epoch 7 Finished.  Time per Epoch:   7.85 s
Iteration:  423 Progress:  40.00  % Time Elapsed:  70.60 s
Epoch 8 Finished.  Time per Epoch:   7.84 s
Iteration:  470 Progress:  45.00  % Time Elapsed:  78.44 s
Epoch 9 Finished.  Time per Epoch:   7.84 s
Iteration:  517 Progress:  50.00  % Time Elapsed:  86.26 s
Epoch 10 Finished.  Time per Epoch:   7.84 s
Iteration:  564 Progress:  55.00  % Time Elapsed:  94.13 s
Epoch 11 Finished.  Time per Epoch:   7.84 s
Iteration:  611 Progress:  60.00  % Time Elapsed:  102.01 s
Epoch 12 Finished.  Time per Epoch:   7.85 s
Iteration:  658 Progress:  65.00  % Time Elapsed:  109.94 s
Epoch 13 Finished.  Time per Epoch:   7.85 s
Iteration:  705 Progress:  70.00  % Time Elapsed:  117.77 s
Epoch 14 Finished.  Time per Epoch:   7.85 s
Iteration:  752 Progress:  75.00  % Time Elapsed:  125.62 s
Epoch 15 Finished.  Time per Epoch:   7.85 s
Iteration:  799 Progress:  80.00  % Time Elapsed:  133.69 s
Epoch 16 Finished.  Time per Epoch:   7.86 s
Iteration:  846 Progress:  85.00  % Time Elapsed:  141.54 s
Epoch 17 Finished.  Time per Epoch:   7.86 s
Iteration:  893 Progress:  90.00  % Time Elapsed:  149.40 s
Epoch 18 Finished.  Time per Epoch:   7.86 s
Iteration:  940 Progress:  95.00  % Time Elapsed:  157.34 s
Epoch 19 Finished.  Time per Epoch:   7.87 s
Final Training Accuracy: 0.8386440677966102
Final Validation Accuracy: 0.7802197802197802
Final Test Accuracy: 0.8143712574850299
```

4. Transfer Learning [15 pt]

For many image classification tasks, it is generally not a good idea to train a very large deep neural network model from scratch due to the enormous compute requirements and lack of sufficient amounts of training data.

One of the better options is to try using an existing model that performs a similar task to the one you need to solve. This method of utilizing a pre-trained network for other similar tasks is broadly termed **Transfer Learning**. In this assignment, we will use Transfer Learning to extract features from the hand gesture images. Then, train a smaller network to use these features as input and classify the hand gestures.

As you have learned from the CNN lecture, convolution layers extract various features from the images which get utilized by the fully connected layers for correct classification. AlexNet architecture played a pivotal role in establishing Deep Neural Nets as a go-to tool for image classification problems and we will use an ImageNet pre-trained AlexNet model to extract features in this assignment.

Part (a) - 5 pt

Here is the code to load the AlexNet network, with pretrained weights. When you first run the code, PyTorch will download the pretrained weights from the internet.

In [ ]:
```
import torchvision.models
alexnet = torchvision.models.alexnet(pretrained=True)
```

```
Downloading: "https://download.pytorch.org/models/alexnet-owt-4df8aa71.pth" t
o /root/.cache/torch/hub/checkpoints/alexnet-owt-4df8aa71.pth
```

The alexnet model is split up into two components: *alexnet.features* and *alexnet.classifier*. The first neural network component, *alexnet.features*, is used to compute convolutional features, which are taken as input in *alexnet.classifier*.

The neural network alexnet.features expects an image tensor of shape Nx3x224x224 as input and it will output a tensor of shape Nx256x6x6 . (N = batch size).

Compute the AlexNet features for each of your training, validation, and test data. Here is an example code snippet showing how you can compute the AlexNet features for some images (your actual code might be different):

In [ ]:
```
import time
import os
import numpy as np
import torch

import torchvision
from torchvision import datasets, models, transforms
import matplotlib.pyplot as plt

import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim #for gradient descent

# alexnet
import torchvision.models

torch.manual_seed(1) # set the random seed

# define training and test data directories
data_dir = '/root/datasets/'
train_dir = os.path.join(data_dir, 'train/')
val_dir = os.path.join(data_dir, 'val/')
test_dir = os.path.join(data_dir, 'test/')


# load and transform data using ImageFolder
# Data was already split randomly into train, validation, test data sets with
 approximately 60%, 20% and 20% of the total samples

# resize and ensure all images to 224 x 224 pixels, random crop is not because
the images are created consistently
data_transform = transforms.Compose([transforms.Resize((224,224)),
                                     transforms.ToTensor()])

train_data = datasets.ImageFolder(train_dir, transform=data_transform)
val_data = datasets.ImageFolder(val_dir, transform=data_transform)
test_data = datasets.ImageFolder(test_dir, transform=data_transform)

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# print out some data stats
print('Num training images: ', len(train_data))
print('Num validation images: ', len(val_data))
print('Num test images: ', len(test_data))

# define dataloader parameters
batch_size  = 1 # process 1 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders
train_loader = torch.utils.data.DataLoader(train_data, batch_size=batch_size,
                                           num_workers=num_workers, shuffle=Tr
ue)
val_loader = torch.utils.data.DataLoader(val_data, batch_size=batch_size,
                                         num_workers=num_workers, shuffle=Tru
```

```
e)
test_loader = torch.utils.data.DataLoader(test_data, batch_size=batch_size,
                                          num_workers=num_workers, shuffle=Tru
e)



# img = ... a PyTorch tensor with shape [N,3,224,224] containing hand images
 ...
#features = alexnet.features(img)

alexnet = torchvision.models.alexnet(pretrained=True)
# location on Google Drive
train_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/
train/'
valid_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/
val/'
test_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/t
est/'

#save the features of the trainning data
n = 1
for images, labels in train_loader:
    features = alexnet.features(images)
    features_train = torch.from_numpy(features.detach().numpy())

    folder_train = train_data_path + str(classes[labels])
    if not os.path.isdir(folder_train):
      os.mkdir(folder_train)
    torch.save(features_train.squeeze(0), folder_train + '/' + str(n) + '.tens
or') # get rid of the batch size
    n += 1


#save the features of the validation data
n = 1
for images, labels in val_loader:
    features = alexnet.features(images)
    features_val = torch.from_numpy(features.detach().numpy())

    folder_val = valid_data_path + str(classes[labels])
    if not os.path.isdir(folder_val):
      os.mkdir(folder_val)
    torch.save(features_val.squeeze(0), folder_val + '/' + str(n) + '.tensor')
# get rid of the batch size
    n += 1


#save the features of the test data
n = 1
for images, labels in test_loader:
    features = alexnet.features(images)
    features_test = torch.from_numpy(features.detach().numpy())

    folder_test = test_data_path + str(classes[labels])
    if not os.path.isdir(folder_test):
      os.mkdir(folder_test)
```

```
    torch.save(features_test.squeeze(0), folder_test + '/' + str(n) + '.tenso
r') # get rid of the batch size
    n += 1
```

```
Num training images:  1475
Num validation images:  455
Num test images:  501
```

```
Downloading: "https://download.pytorch.org/models/alexnet-owt-4df8aa71.pth" t
o /root/.cache/torch/hub/checkpoints/alexnet-owt-4df8aa71.pth
```

Save the computed features. You will be using these features as input to your neural network in Part (b), and you do not want to re-compute the features every time. Instead, run *alexnet.features* once for each image, and save the result.

Part (b) - 3 pt

Build a convolutional neural network model that takes as input these AlexNet features, and makes a prediction. Your model should be a subclass of nn.Module.

Explain your choice of neural network architecture: how many layers did you choose? What types of layers did you use: fully-connected or convolutional? What about other decisions like pooling layers, activation functions, number of channels / hidden units in each layer?

Here is an example of how your model may be called:

In [ ]:
```
# features = ... load precomputed alexnet.features(img) ...
#output = model(features)
#prob = F.softmax(output)
print(features_train.shape)
```

```
torch.Size([1, 256, 6, 6])
```

In [ ]:
```
#Artifical Neural Network Architecture
class ANNClassifier(nn.Module):
    def __init__(self):
        super(ANNClassifier, self).__init__()  # Fully connector layers with 3
hidden layers
        self.name = 'ANNClassifier'
        self.fc1 = nn.Linear(256 * 6 * 6, 256) # the ouput image size is 256*6
*6, batch size was eliminated as shown before
        self.fc2 = nn.Linear(256, 128) # Hidden units = 128
        self.fc3 = nn.Linear(128, 32) # Hidden units = 32
        self.fc4 = nn.Linear(32, 9)

    def forward(self, x):
        x = x.view(-1, 256 * 6 * 6) #flatten feature data
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = F.relu(self.fc3(x))
        x = self.fc4(x)
        return x
```

I decided to use ANN architecture. This is because that the ANN acts as a classifier and should accept the output data from a pretrained CNN network(AlexNet). Thus, I do not feel the need to add convolutional layers again in my model, but I may add them in the future. I used four layers in total with three hidden layers, all fully connected. Relu activation functions are applied after each layer. The hidden units are designed such that the model accepts the AlexNet features as inputs and pass them down different layers of neurons to reduce the amount of neurons down the layers. Hidden units in some layers are described in the comments.

Part (c) - 5 pt

Train your new network, including any hyperparameter tuning. Plot and submit the training curve of your best model only.

Note: Depending on how you are caching (saving) your AlexNet features, PyTorch might still be tracking updates to the **AlexNet weights**, which we are not tuning. One workaround is to convert your AlexNet feature tensor into a numpy array, and then back into a PyTorch tensor.

In [ ]:
```
#tensor = torch.from_numpy(tensor.detach().numpy()) # already done before
```

In [ ]:
```
def get_accuracy(model, train_loader,val_loader, train=False):
    if train:
        data_loader = train_loader
    else:
        data_loader = val_loader

    correct = 0
    total = 0
    for imgs, labels in data_loader:


        ###############################################
        #To Enable GPU Usage
        if use_cuda and torch.cuda.is_available():
          imgs = imgs.cuda()
          labels = labels.cuda()
        ###############################################

        output = model(imgs)

        #select index with maximum prediction score
        pred = output.max(1, keepdim=True)[1]
        correct += pred.eq(labels.view_as(pred)).sum().item()
        total += imgs.shape[0]
    return correct / total
```

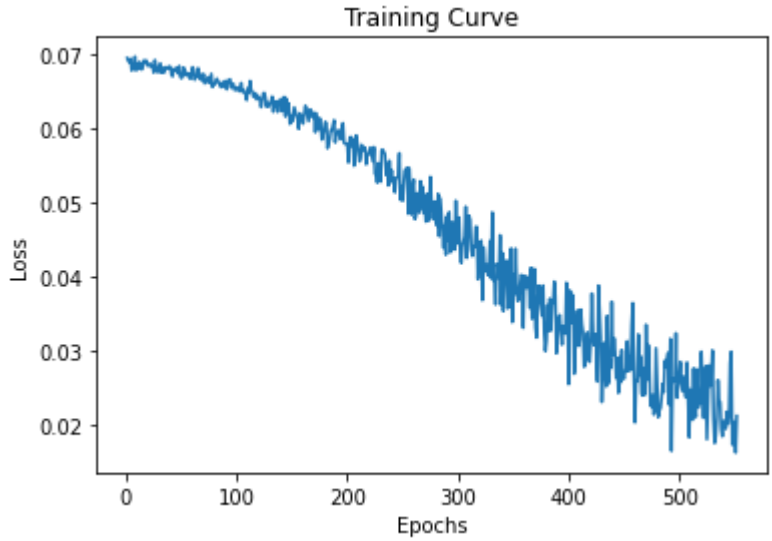
In [ ]:
```
def train(model, train_loader,val_loader, batch_size=32, num_epochs=1,lr = 0.0
01):
    #train_loader = torch.utils.data.DataLoader(data, batch_size=batch_size)
    # train_loader = torch.utils.data.DataLoader(train_data, batch_size=batch_
size,
                                           #num_workers=num_workers, shuffle=T
rue)
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9)

    n_epochs, iters, losses, train_acc, val_acc = [], [], [], [],[]

    # training
    n = 0 # the number of iterations
    start_time=time.time()
    nepochs = 0

    for epoch in range(num_epochs):
        for features, labels in iter(train_loader):

            ##############################################
            #To Enable GPU Usage
            if use_cuda and torch.cuda.is_available():
              features = features.cuda()
              labels = labels.cuda()
            ##############################################

          #### ALNC is alexNet.features (AlexNet without classifier) ####

            out = model(features)             # forward pass
            loss = criterion(out, labels) # compute the total loss
            loss.backward()               # backward pass (compute parameter u
pdates)
            optimizer.step()              # make the updates for each paramete
r
            optimizer.zero_grad()         # a clean up step for PyTorch
            n += 1


          # save the current training information
            iters.append(n)
            losses.append(float(loss)/batch_size)             # compute *avera
ge* loss
        nepochs+=1
        n_epochs.append(nepochs)
        val_acc.append(get_accuracy(model, train_loader,val_loader, train=Fals
e))  # compute validation accuracy
        train_acc.append(get_accuracy(model, train_loader,val_loader, train=Tr
ue))  # compute validation accuracy
        model_path = "model_{0}_bs{1}_lr{2}_epoch{3}".format(model.name, batch
_size, lr, num_epochs)
        torch.save(model.state_dict(), model_path)


        print("Iteration: ",n,'Progress: % 6.2f ' % ((epoch * len(train_loader
)) / (num_epochs * len(train_loader))*100),'%', "Time Elapsed: % 6.2f s " % (t
```

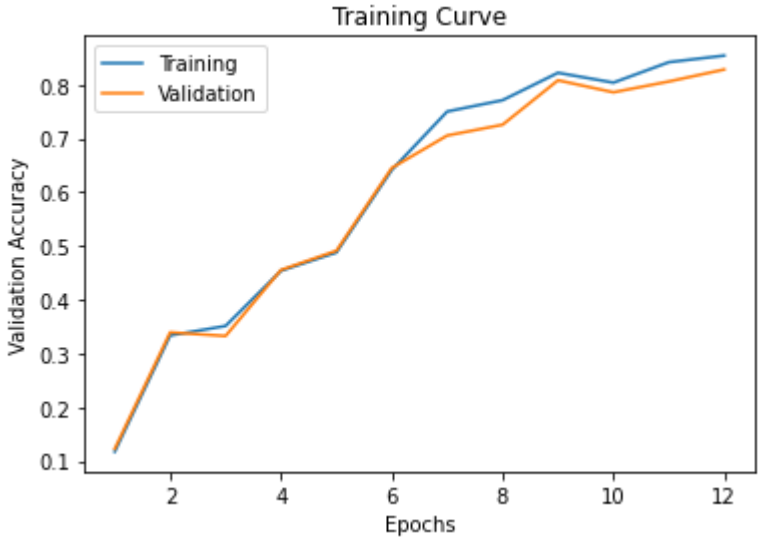
```
ime.time()-start_time))


    print ("Epoch %d Finished. " % epoch ,"Time per Epoch: % 6.2f s "% ((time.
time()-start_time) / (epoch +1)))


    end_time= time.time()
    # plotting
    plt.title("Training Curve")
    plt.plot(iters, losses, label="Train")
    plt.xlabel("Epochs")
    plt.ylabel("Loss")
    plt.show()

    plt.title("Training Curve")
    plt.plot(n_epochs, train_acc, label="Training")
    plt.plot(n_epochs, val_acc, label="Validation")
    plt.xlabel("Epochs")
    plt.ylabel("Validation Accuracy")
    plt.legend(loc='best')
    plt.show()

    print("Final Training Accuracy: {}".format(train_acc[-1]))
    print("Final Validation Accuracy: {}".format(val_acc[-1]))
    print ("Total time:  % 6.2f s  Time per Epoch: % 6.2f s " % ( (end_time-st
art_time), ((end_time-start_time) / num_epochs) ))
```

In [ ]:
```
# load features
# location on Google Drive

# define training and test data directories
data_dir = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/'


train_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/
train/'
val_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/va
l/'
test_data_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/t
est/'

# Load data from Google Drive using DatasetFolder
train_dataset_new = torchvision.datasets.DatasetFolder(train_data_path, loader
=torch.load, extensions=('.tensor'))
val_dataset_new = torchvision.datasets.DatasetFolder(val_data_path, loader=tor
ch.load, extensions=('.tensor'))
test_dataset_new = torchvision.datasets.DatasetFolder(test_data_path, loader=t
orch.load, extensions=('.tensor'))

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# define dataloader parameters
batch_size  = 32 # process 32 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders
train_data_loader = torch.utils.data.DataLoader(train_dataset_new, batch_size=
batch_size,
                                                num_workers=num_workers, shuffle=Tr
ue, drop_last = True)
val_data_loader = torch.utils.data.DataLoader(val_dataset_new, batch_size=batc
h_size,
                                              num_workers=num_workers, shuffle=Tru
e, drop_last = True)
test_data_loader = torch.utils.data.DataLoader(test_dataset_new, batch_size=ba
tch_size,
                                               num_workers=num_workers, shuffle=Tru
e, drop_last = True)
```

In [ ]:
```
use_cuda = True

model = ANNClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model
train(model, train_data_loader,val_data_loader, batch_size=32, num_epochs=12,l
r=0.0005)
```

```
CUDA is available!  Training on GPU ...
Iteration:  46 Progress:   0.00  % Time Elapsed:   3.58 s
Iteration:  92 Progress:   8.33  % Time Elapsed:   7.21 s
Iteration:  138 Progress:  16.67  % Time Elapsed:  10.82 s
Iteration:  184 Progress:  25.00  % Time Elapsed:  14.30 s
Iteration:  230 Progress:  33.33  % Time Elapsed:  17.84 s
Iteration:  276 Progress:  41.67  % Time Elapsed:  21.36 s
Iteration:  322 Progress:  50.00  % Time Elapsed:  24.78 s
Iteration:  368 Progress:  58.33  % Time Elapsed:  28.34 s
Iteration:  414 Progress:  66.67  % Time Elapsed:  31.83 s
Iteration:  460 Progress:  75.00  % Time Elapsed:  35.41 s
Iteration:  506 Progress:  83.33  % Time Elapsed:  38.94 s
Iteration:  552 Progress:  91.67  % Time Elapsed:  42.39 s
Epoch 11 Finished.  Time per Epoch:   3.53 s
```






```
Final Training Accuracy: 0.8539402173913043
Final Validation Accuracy: 0.828125
Total time:   42.39 s  Time per Epoch:   3.53 s
```

Part (d) - 2 pt

Report the test accuracy of your best model. How does the test accuracy compare to Part 3(d) without transfer learning?

In [ ]:
```
#The best model is shown above. Its test accuracy is calculated below.
def train(model, train_loader,val_loader,test_loader, batch_size=32, num_epoch
s=1,lr = 0.001):
    #train_loader = torch.utils.data.DataLoader(data, batch_size=batch_size)
    # train_loader = torch.utils.data.DataLoader(train_data, batch_size=batch_
size,
                                               #num_workers=num_workers, shuffle=T
rue)
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9)

    test_acc, n_epochs, iters, losses, train_acc, val_acc = [], [], [], [],[],
[]

    # training
    n = 0 # the number of iterations
    start_time=time.time()
    nepochs = 0

    for epoch in range(num_epochs):
        for features, labels in iter(train_loader):

            ##############################################
            #To Enable GPU Usage
            if use_cuda and torch.cuda.is_available():
              features = features.cuda()
              labels = labels.cuda()
            ##############################################

          #### ALNC is alexNet.features (AlexNet without classifier) ####

            out = model(features)             # forward pass
            loss = criterion(out, labels) # compute the total loss
            loss.backward()               # backward pass (compute parameter u
pdates)
            optimizer.step()              # make the updates for each paramete
r
            optimizer.zero_grad()         # a clean up step for PyTorch
            n += 1


          # save the current training information
            iters.append(n)
            losses.append(float(loss)/batch_size)             # compute *avera
ge* loss
        nepochs+=1
        n_epochs.append(nepochs)
        val_acc.append(get_accuracy(model, train_loader,val_loader, train=Fals
e))  # compute validation accuracy
        train_acc.append(get_accuracy(model, train_loader,val_loader, train=Tr
ue))  # compute validation accuracy
        test_acc.append(get_accuracy(model, train_loader,test_loader, train=Fa
lse))

        print("Iteration: ",n,'Progress: % 6.2f ' % ((epoch * len(train_loader
)) / (num_epochs * len(train_loader))*100),'%', "Time Elapsed: % 6.2f s " % (t
```

```
ime.time()-start_time))


    print ("Epoch %d Finished. " % epoch ,"Time per Epoch: % 6.2f s "% ((time.
time()-start_time) / (epoch +1)))


    end_time= time.time()

    print("Final Training Accuracy: {}".format(train_acc[-1]))
    print("Final Validation Accuracy: {}".format(val_acc[-1]))
    print("Final Test Accuracy: {}".format(test_acc[-1]))
    print ("Total time:  % 6.2f s  Time per Epoch: % 6.2f s " % ( (end_time-st
art_time), ((end_time-start_time) / num_epochs) ))
```

In [ ]:
```
use_cuda = True

model = ANNClassifier()

if use_cuda and torch.cuda.is_available():
  model.cuda()
  print('CUDA is available!  Training on GPU ...')
else:
  print('CUDA is not available.  Training on CPU ...')

#proper model
train(model, train_data_loader,val_data_loader,test_data_loader, batch_size=32
, num_epochs=12,lr=0.0005)
```

```
CUDA is available!  Training on GPU ...
Iteration:  46 Progress:   0.00  % Time Elapsed:   4.09 s
Iteration:  92 Progress:   8.33  % Time Elapsed:   8.12 s
Iteration:  138 Progress:  16.67  % Time Elapsed:  12.13 s
Iteration:  184 Progress:  25.00  % Time Elapsed:  16.20 s
Iteration:  230 Progress:  33.33  % Time Elapsed:  20.16 s
Iteration:  276 Progress:  41.67  % Time Elapsed:  24.03 s
Iteration:  322 Progress:  50.00  % Time Elapsed:  28.11 s
Iteration:  368 Progress:  58.33  % Time Elapsed:  32.03 s
Iteration:  414 Progress:  66.67  % Time Elapsed:  36.05 s
Iteration:  460 Progress:  75.00  % Time Elapsed:  40.14 s
Iteration:  506 Progress:  83.33  % Time Elapsed:  44.12 s
Iteration:  552 Progress:  91.67  % Time Elapsed:  48.24 s
Epoch 11 Finished.  Time per Epoch:   4.02 s
Final Training Accuracy: 0.8471467391304348
Final Validation Accuracy: 0.8147321428571429
Final Test Accuracy: 0.8604166666666667
Total time:   48.24 s  Time per Epoch:   4.02 s
```

The test accuracy in Part 3(d) is 0.8143712574850299, whereas the test accuracy using transfer learning of the Alexnet is 0.8604166666666667 that is comparatively higher. Thus, this has proven the effectiveness of transfer learning. This demonstrates that using an existing well-trained model on image recgonition tasks can be used as the feature extractor. In this way, one only needs to utilize the advantages of the tranined model and train an additional easier and smaller network to achieve high test accuracy. Transfer learning can dramatically improve the effiency in solving our own image recognition problem.

5. Additional Testing [5 pt]

As a final step in testing we will be revisiting the sample images that you had collected and submitted at the start of this lab. These sample images should be untouched and will be used to demonstrate how well your model works at identifying your hand guestures.

Using the best transfer learning model developed in Part 4. Report the test accuracy on your sample images and how it compares to the test accuracy obtained in Part 4(d)? How well did your model do for the different hand guestures? Provide an explanation for why you think your model performed the way it did?

In [ ]:
```
!unzip '/content/drive/My Drive/APS360/TUT and Lab03/PAN_1003948115.zip' -d '/root/datasets/'
```

```
Archive:  /content/drive/My Drive/APS360/TUT and Lab03/PAN_1003948115.zip
   creating: /root/datasets/test_self/A/
  inflating: /root/datasets/test_self/A/1003948115_A_1.jpg
  inflating: /root/datasets/test_self/A/1003948115_A_2.jpg
  inflating: /root/datasets/test_self/A/1003948115_A_3.jpg
   creating: /root/datasets/test_self/B/
  inflating: /root/datasets/test_self/B/1003948115_B_1.jpg
  inflating: /root/datasets/test_self/B/1003948115_B_2.jpg
  inflating: /root/datasets/test_self/B/1003948115_B_3.jpg
   creating: /root/datasets/test_self/C/
  inflating: /root/datasets/test_self/C/1003948115_C_1.jpg
  inflating: /root/datasets/test_self/C/1003948115_C_2.jpg
  inflating: /root/datasets/test_self/C/1003948115_C_3.jpg
   creating: /root/datasets/test_self/D/
  inflating: /root/datasets/test_self/D/1003948115_D_1.jpg
  inflating: /root/datasets/test_self/D/1003948115_D_2.jpg
  inflating: /root/datasets/test_self/D/1003948115_D_3.jpg
   creating: /root/datasets/test_self/E/
  inflating: /root/datasets/test_self/E/1003948115_E_1.jpg
  inflating: /root/datasets/test_self/E/1003948115_E_2.jpg
  inflating: /root/datasets/test_self/E/1003948115_E_3.jpg
   creating: /root/datasets/test_self/F/
  inflating: /root/datasets/test_self/F/1003948115_F_1.jpg
  inflating: /root/datasets/test_self/F/1003948115_F_2.jpg
  inflating: /root/datasets/test_self/F/1003948115_F_3.jpg
   creating: /root/datasets/test_self/G/
  inflating: /root/datasets/test_self/G/1003948115_G_1.jpg
  inflating: /root/datasets/test_self/G/1003948115_G_2.jpg
  inflating: /root/datasets/test_self/G/1003948115_G_3.jpg
   creating: /root/datasets/test_self/H/
  inflating: /root/datasets/test_self/H/1003948115_H_1.jpg
  inflating: /root/datasets/test_self/H/1003948115_H_2.jpg
  inflating: /root/datasets/test_self/H/1003948115_H_3.jpg
   creating: /root/datasets/test_self/I/
  inflating: /root/datasets/test_self/I/1003948115_I_1.jpg
  inflating: /root/datasets/test_self/I/1003948115_I_2.jpg
  inflating: /root/datasets/test_self/I/1003948115_I_3.jpg
```

In [ ]:
```
def get_accuracy(model, data_loader):

    correct = 0
    total = 0
    for imgs, labels in data_loader:
        ###############################################
        #To Enable GPU Usage
        if use_cuda and torch.cuda.is_available():
          imgs = imgs.cuda()
          labels = labels.cuda()
        ###############################################

        output = model(imgs)

        #select index with maximum prediction score
        pred = output.max(1, keepdim=True)[1]
        correct += pred.eq(labels.view_as(pred)).sum().item()
        total += imgs.shape[0]
    return correct / total
```

In [ ]:
```
# best parameters: batch_size = 32, learning_rate = 0.0005, num_epochs = 12
best_mod = ANNClassifier()
best_mod_path = "model_{0}_bs{1}_lr{2}_epoch{3}".format("ANNClassifier", 32,
0.0005, 12)
state = torch.load(best_mod_path)
best_mod.load_state_dict(state)
if use_cuda and torch.cuda.is_available():
    best_mod.cuda()
```

In [ ]:
```
# define training and test data directories
data_dir = '/root/datasets/'
test_dir = os.path.join(data_dir, 'test_self/')

# load and transform data using ImageFolder
# Data was already split randomly into train, validation, test data sets with
 approximately 60%, 20% and 20% of the total samples

# resize and ensure all images to 224 x 224 pixels, random crop is not because
the images are created consistently
data_transform = transforms.Compose([transforms.Resize((224,224)),
                                     transforms.ToTensor()])

test_data = datasets.ImageFolder(test_dir, transform=data_transform)

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# define dataloader parameters
batch_size  = 1 # process 1 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders
test_loader_self = torch.utils.data.DataLoader(test_data, batch_size=batch_siz
e,
                                               num_workers=num_workers, shuffle=Tru
e)

# Visualize some sample data
```

In [ ]:
```
#save the features of the test data

test_data_self_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_
New/test_self/'
n = 1
for images, labels in test_loader_self:
    features = alexnet.features(images)
    features_test_self = torch.from_numpy(features.detach().numpy())

    folder_test_self = test_data_self_path + str(classes[labels])
    if not os.path.isdir(folder_test_self):
      os.mkdir(folder_test_self)
    torch.save(features_test_self.squeeze(0), folder_test_self + '/' + str(n)
+ '.tensor') # get rid of the batch size
    n += 1
```

In [ ]:
```
# define training and test data directories
data_dir = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_New/'

test_data_self_path = '/content/drive/My Drive/Colab Notebooks/360_Lab3_partB_
New/test/'

# Load data from Google Drive using DatasetFolder
test_dataset_self_new = torchvision.datasets.DatasetFolder(test_data_self_path
, loader=torch.load, extensions=('.tensor'))

# classes are folders in each directory with these names
classes = ['A', 'B', 'C', 'D', 'E','F','G','H','I']

# define dataloader parameters
batch_size  = 32 # process 32 images at a time
num_workers = 0 # we only need 1 worker here

# prepare data loaders

test_data_self_loader = torch.utils.data.DataLoader(test_dataset_self_new, bat
ch_size=batch_size,
                                                    num_workers=num_workers, shuffle=Tru
e, drop_last = True)
```

In [ ]:
```
test_accuracy = get_accuracy(best_mod, test_data_self_loader)
print("test accuracy:", test_accuracy)
```

```
test accuracy: 0.8676470588235294
```

The test accuracy on my sample images calculated by the best model is found to be 0.8676470588235294 that is slightly higher than the one obtained in Part 4 d) that is 0.8604166666666667. Overall, my model has a quite stable test accuracy for different gestures and perform almost equally well on my own gestures and the provided sample gestures. The results have shown that my model can be generalized quite well in hand gesture identifications. The slight improvement in my own data may be because that I produced my data with high quality, decent light exposure and minimal shadows, which thus increase the test accuracy of the model on my own gestures.