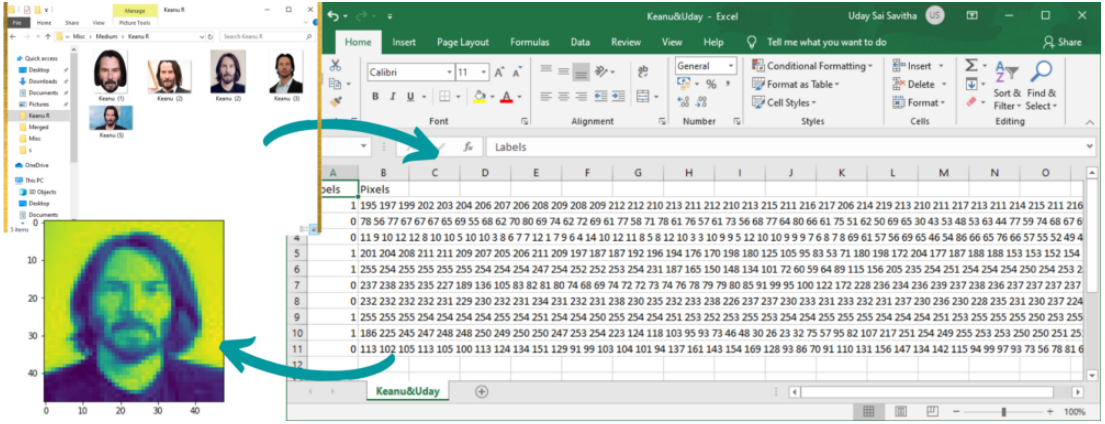


Image by the author

DATA SCIENCE, MACHINE LEARNING

How to Create a New Custom Dataset from Images

Uday Sai Follow
Aug 24 · 7 min read

If you're a person like me, trying to build your custom image dataset out of raw images, then ***this article is just for you!***

We all have learned how to build machine learning models on the classic MNIST/Fashion MNIST datasets. But, what if you want to train a model to recognize your friends' faces? A dataset for that purpose is not readily available on the internet.

After working on public datasets for months, I wanted to create a custom dataset of my face images and use them for face identification.

Real expertise is demonstrated by using machine learning to solve your own problems. Building *your own image dataset* is a non-trivial task by itself. Surprisingly, it is covered far less comprehensively in most online courses.

I searched for ways to do it and finally figured it out.

In this article, you will learn how to prepare your own dataset of raw images, which you can then use for your own image classification/computer vision projects.

Steps

1. Gather images for your dataset
2. Rename the pictures according to their classes
3. Merge them into one folder
4. Resize the pictures
5. Convert all images into the same file format
6. Convert images into a CSV file
7. A few tweaks to the CSV file
8. Load the CSV (BONUS)

Gather images for your dataset

As an example, let's say that I want to build a model that can differentiate between Keanu Reeves and me XD.

If you need to create a dataset of your own face or bulk download images from google, this article from *pyimagesearch* walks you through it.

After getting the images, sort the images into different folders according to their classes. For the sake of simplicity, I'm going to use just five images per class (You can use as many as you want. The more, the better).

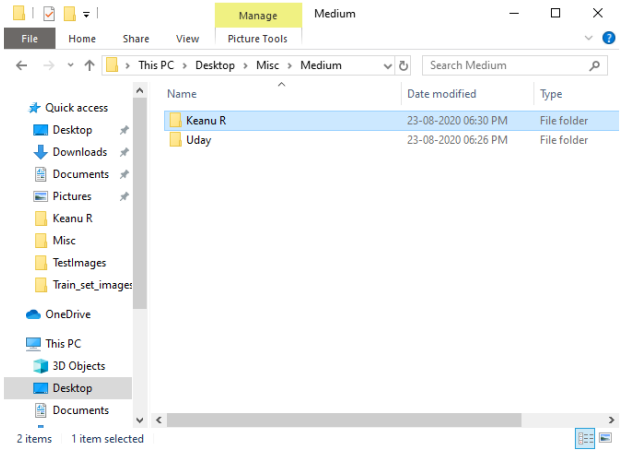
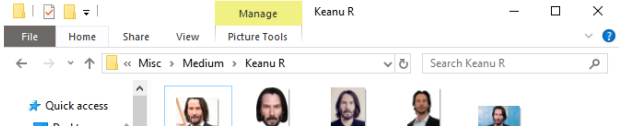
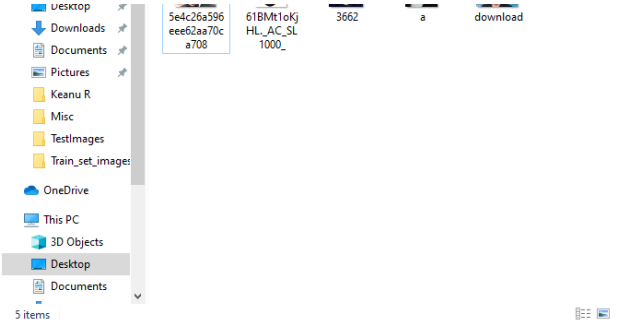
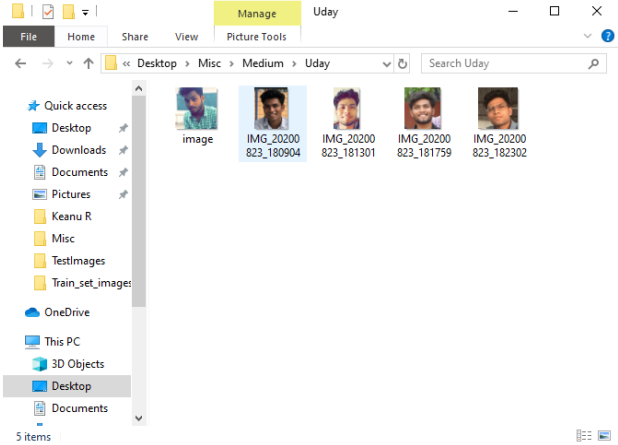


Image by the author





Folder containing images of Keanu Reeves | image by the author



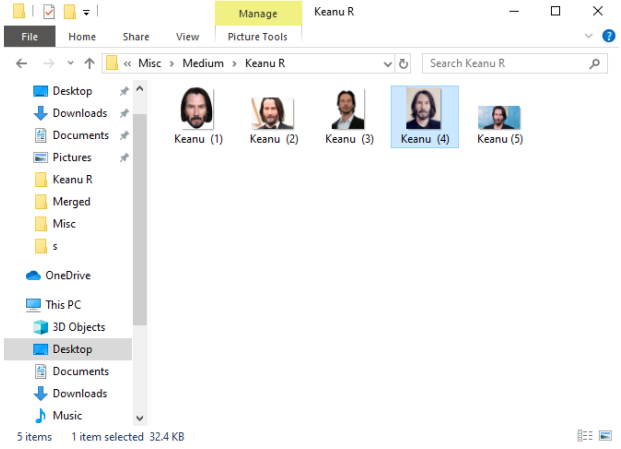
Folder containing my images | image by the author

> A machine learning model is only as good as the data we put into it.

Clean the data. Remove duplicates. Crop the images around your point of interest (in this example, faces of Keanu and me) to make the most of your data.

Rename the pictures according to their classes

1. Open the folder and select all images.
2. Right-click on them.
3. Rename all of them by their class.



After renaming the files | image by the author

4. Repeat it for all the remaining classes. Name the classes with at least one different alphabet (this is needed in the latter part of the process).

Merge all the images into a single folder.

Resize the pictures

The tool we use for this is *Image Resizer for Windows*. It's free, small, and completely malware-free.

Image by the author

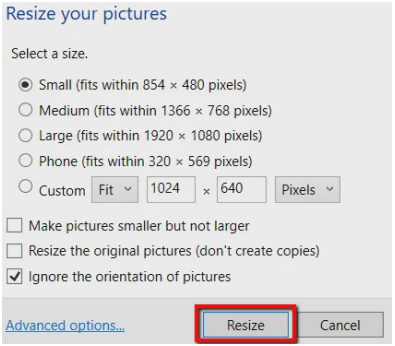
Once it's downloaded, click **Install**.

Once the program is installed on your computer, you're good to go. Now, go to the folder containing the photos that you want to resize.

Select your photos. Then right-click on them and choose to **Resize pictures** from the options.

A window will then pop up. Here, you can modify the basic settings for the pictures that will be processed.

You can select the size for the pictures. In this case, I resize the images to 48 x 48 pixels.



Don't forget to change from "Fit" to "Stretch." | image by the author

Note: Sometimes, smaller pictures get ignored by the resizer. After Resizing, select all the images and verify if all the images are of the same size.

Convert all images into the same file format

Here is a neat trick to do this easily and efficiently. You could either choose .png or .jpg format.

Step 1 — Type **cmd** on the taskbar search field and jointly press **Ctrl + Shift + Enter** keys. If you come across **UAC** prompt, click **Yes**.

Step 2 — In the Command Prompt, first input the path of the new folder where you stored the files (images of Spotlight). To do so, type in –

```
cd path of the folder
```

Note — Please replace the **pathofthefolder** with the actual path.

```
cd C:\Users\Uday\Desktop\Misc\Medium\Merged
```

Step 3 — To Change the images to **JPG** format, type in the given batch command, and press **Enter**.

```
Ren *.* *.jpg
```

Step 4 — To convert the images to **PNG** format, use the following batch command –

```
Ren *.* *.png
```

Convert the images into a CSV

Run the following code to convert all the images into a CSV and label them accordingly.

```
from PIL import Image
import numpy as np
import sys
import os
import csv


# default format can be changed as needed
def createFileList(myDir, format='.jpg'):
    fileList = []
    print(myDir)
    labels = []
    names = []
    keywords = {"K" : "1","U": "0",} # keys and values to be changed
as needed


for root, dirs, files in os.walk(myDir, topdown=True):
        for name in files:
            if name.endswith(format):
                fullName = os.path.join(root, name)
```

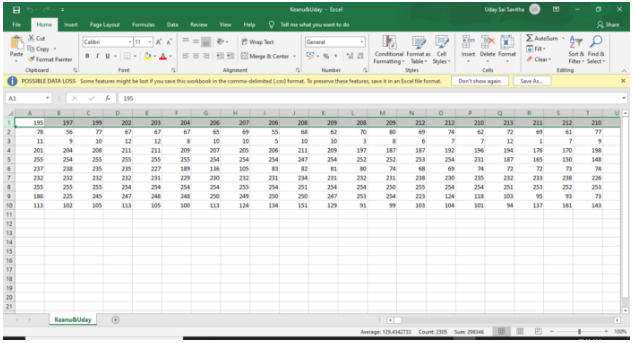
```
                fileList.append(fullName)
            for keyword in keywords:
                if keyword in name:
                    labels.append(keywords[keyword])
                else:
                    continue
            names.append(name)
    return fileList, labels, names


# load the original image
myFileList, labels, names  = createFileList('/content/')
i = 0
for file in myFileList:
    print(file)
    img_file = Image.open(file)
    # img_file.show()


# get original image parameters...
    width, height = img_file.size
    format = img_file.format
    mode = img_file.mode


# Make image Greyscale
    img_grey = img_file.convert('L')
    #img_grey.save('result.png')
    #img_grey.show()


# Save Greyscale values
    value = np.asarray(img_grey.getdata(),
dtype=np.int).reshape((width, height))
    value = value.flatten()

    value = np.append(value,labels[i])
    i +=1

    print(value)
    with open("name_you_want.csv", 'a') as f:
        writer = csv.writer(f)
        writer.writerow(value)
```

1. I've used K and U alphabets as keys to recognize the classes from the file names (Keanu has K in it and Uday has U in it). Change it as per your needs.
2. To keep images in color instead of greyscale images replace 'L' with 'RGB.' Also, add depth value before saving the image. Depth = 3 representing the number of color channels (Red, Green, Blue).

```
img_grey = img_file.convert('L')# replace L with RGB
value = np.asarray(img_grey.getdata(), dtype=np.int).reshape((width,
height, 3))
```

3. *name_you_want* will be the name of the CSV file created. Feel free to change it.

You have your dataset ready. Well, almost ready.



And the last column in the sheet are the labels | image by the author

A few tweaks to the CSV file

1. Scroll to the end, click on an empty cell and use the following Excel formula to concatenate the pixel values.

```
=TEXTJOIN(" ", TRUE, (A1:B1))
#replace B1 with last but one column name
```

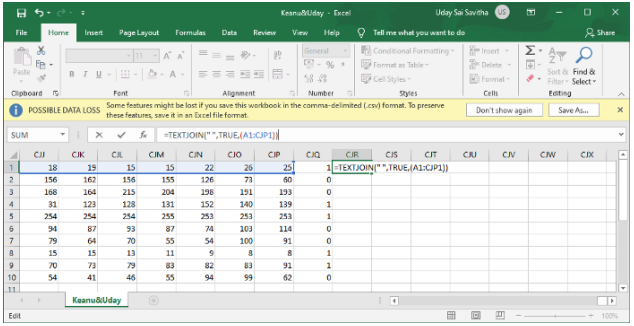


Image by the author

2. Drag the formula to the remaining rows.

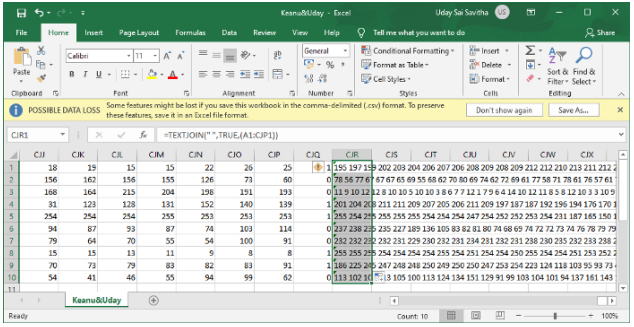


Image by the author

3. Copy that column values to the notepad. Re-copy them and paste them back. This way, you will retain the pixel values and not the formula.

4. Now select all cells except the labels and concatenated values and delete them.

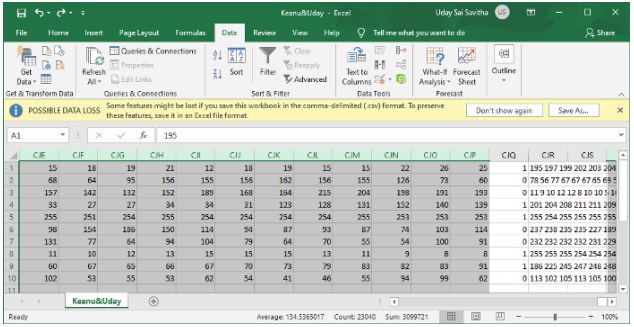


Image by the author

5. Cut the remaining columns and paste them at the beginning of the sheet.

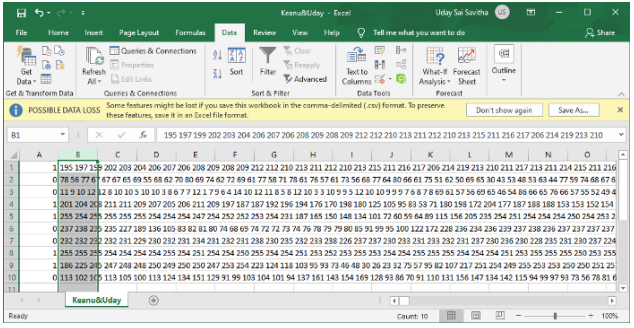


Image by the author

6. Name the columns accordingly.

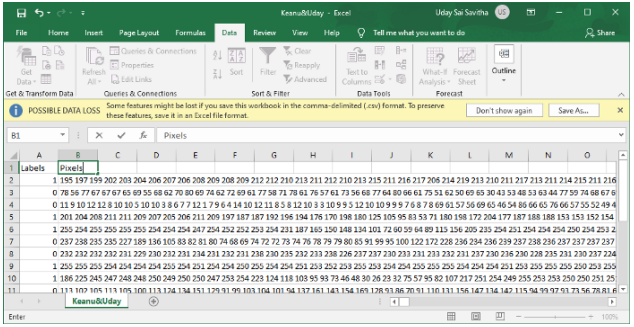


Image by the author

Aaaaanndddddd, we're DONE!

> *Congratulations! You've created a brand new custom image dataset from scratch.*

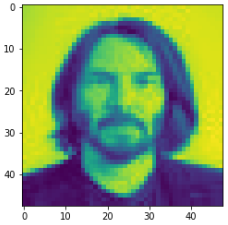
Bonus

Load the CSV

Load the CSV and run this following code snippet and you all good to good.

```
import pandas as pd
import cv2
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split

dataset_path = '/content/Keanu&Uday.csv'
image_size=(48,48) #add 3 if RGB image
```

```
def load():
    data = pd.read_csv(dataset_path)
    pixels = data['Pixels'].tolist()
    width, height= 48, 48 ,# add depth 3 if RGB image
    faces = []
    for pixel_sequence in pixels:
        face = [int(pixel) for pixel in pixel_sequence.split(' ')]
        face = np.asarray(face).reshape(width, height,) #add depth if
RGB image
        a = face
        face = np.resize(face.astype('uint8'),image_size)
        faces.append(face.astype('float32'))


faces = np.asarray(faces)
    A = faces
    faces = np.expand_dims(faces, -1)
    return faces, A

faces,A = load()
plt.imshow(A[0].astype("uint8"))
```



Output for the above code snippet | image by the author

Thanks for reading! I hope you found this article useful. Here's the link to the Colab notebook.

Resources: GitHub repository and Google Colab