◢ Bringing Old Photos Back to Life

This is a reference implementation of our CVPR 2020 paper [1], which revives an old photo to modern style. Should you be making use of our work, please cite our paper [1].

◢ Verify Runtime Settings

IMPORTANT

In the "Runtime" menu for the notebook window, select "Change runtime type." Ensure that the following are selected:

- Runtime Type = Python 3
- Hardware Accelerator = GPU

◢ Git clone

```
!git clone https://github.com/microsoft/Bringing-Old-Photos-Back-to-Life.git photo_restora
```

```
Cloning into 'photo_restoration'...
remote: Enumerating objects: 68, done.
remote: Counting objects: 100% (68/68), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 258 (delta 22), reused 36 (delta 15), pack-reused 190
Receiving objects: 100% (258/258), 16.91 MiB | 47.32 MiB/s, done.
Resolving deltas: 100% (65/65), done.
```

◢ Set up the environment

```
# pull the syncBN repo
%cd photo_restoration/Face_Enhancement/models/networks
!git clone https://github.com/vacancy/Synchronized-BatchNorm-PyTorch
!cp -rf Synchronized-BatchNorm-PyTorch/sync_batchnorm .
%cd ../../../

%cd Global/detection_models
!git clone https://github.com/vacancy/Synchronized-BatchNorm-PyTorch
!cp -rf Synchronized-BatchNorm-PyTorch/sync_batchnorm .
%cd ../../

# download the landmark detection model
%cd Face_Detection/
```

```
!wget http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2
!bzip2 -d shape_predictor_68_face_landmarks.dat.bz2
%cd ../

# download the pretrained model
%cd Face_Enhancement/
!wget https://facevc.blob.core.windows.net/zhanbo/old_photo/pretrain/Face_Enhancement/chec
!unzip checkpoints.zip
%cd ../

%cd Global/
!wget https://facevc.blob.core.windows.net/zhanbo/old_photo/pretrain/Global/checkpoints.zi
!unzip checkpoints.zip
%cd ../
```

```
/content/photo_restoration/Face_Enhancement/models/networks
Cloning into 'Synchronized-BatchNorm-PyTorch'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 177 (delta 8), reused 9 (delta 4), pack-reused 161
Receiving objects: 100% (177/177), 41.14 KiB | 13.71 MiB/s, done.
Resolving deltas: 100% (104/104), done.
/content/photo_restoration
/content/photo_restoration/Global/detection_models
Cloning into 'Synchronized-BatchNorm-PyTorch'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 177 (delta 8), reused 9 (delta 4), pack-reused 161
Receiving objects: 100% (177/177), 41.14 KiB | 13.71 MiB/s, done.
Resolving deltas: 100% (104/104), done.
/content/photo_restoration
/content/photo_restoration/Face_Detection
--2020-11-30 06:23:23--  http://dlib.net/files/shape_predictor_68_face_landmarks.
Resolving dlib.net (dlib.net)... 107.180.26.78
Connecting to dlib.net (dlib.net)|107.180.26.78|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 64040097 (61M)
Saving to: ‘shape_predictor_68_face_landmarks.dat.bz2’

shape_predictor_68_ 100%[===================>]  61.07M  21.5MB/s    in 2.8s

2020-11-30 06:23:26 (21.5 MB/s) - ‘shape_predictor_68_face_landmarks.dat.bz2’ sav

/content/photo_restoration
/content/photo_restoration/Face_Enhancement
--2020-11-30 06:23:33--  https://facevc.blob.core.windows.net/zhanbo/old_photo/pr
Resolving facevc.blob.core.windows.net (facevc.blob.core.windows.net)... 20.150.7
Connecting to facevc.blob.core.windows.net (facevc.blob.core.windows.net)|20.150.
HTTP request sent, awaiting response... 200 OK
Length: 342496657 (327M) [application/x-zip-compressed]
Saving to: ‘checkpoints.zip’

checkpoints.zip     100%[===================>] 326.63M  20.1MB/s    in 21s

2020-11-30 06:23:54 (15.4 MB/s) - ‘checkpoints.zip’ saved [342496657/342496657]
```

```
Archive:  checkpoints.zip
   creating: checkpoints/
   creating: checkpoints/Setting_9_epoch_100/
  inflating: checkpoints/Setting_9_epoch_100/latest_net_G.pth
/content/photo_restoration
/content/photo_restoration/Global
--2020-11-30 06:23:58--  https://facevc.blob.core.windows.net/zhanbo/old_photo/pr
Resolving facevc.blob.core.windows.net (facevc.blob.core.windows.net)... 20.150.7
Connecting to facevc.blob.core.windows.net (facevc.blob.core.windows.net)|20.150.
HTTP request sent, awaiting response... 200 OK
Length: 1739076350 (1.6G) [application/x-zip-compressed]
Saving to: ‘checkpoints.zip’
```

```
! pip install -r requirements.txt
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: torchvision in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: dlib in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: scikit-image in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: easydict in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: PyYAML in /usr/local/lib/python3.6/dist-packages (fro
Collecting dominate>=2.3.1
  Downloading https://files.pythonhosted.org/packages/ef/a8/4354f8122c39e35516a27087
Requirement already satisfied: dill in /usr/local/lib/python3.6/dist-packages (from
Collecting tensorboardX
  Downloading https://files.pythonhosted.org/packages/af/0c/4f41bcd45db376e6fe5c619c
     |████████████████████████████████| 317kB 20.6MB/s
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: opencv-python in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: dataclasses in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: future in /usr/local/lib/python3.6/dist-packages (fro
Requirement already satisfied: pillow>=4.1.1 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: networkx>=2.0 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: PyWavelets>=0.4.0 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: imageio>=2.3.0 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.0 in /usr/local/lib/python3.6
Requirement already satisfied: protobuf>=3.8.0 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from t
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.6/dist
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/loca
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages
Installing collected packages: dominate, tensorboardX
Successfully installed dominate-2.6.0 tensorboardX-2.1
```

◢ Run the code

Restore photos (normal mode)

```
%cd /content/photo_restoration/
input_folder = "test_images/old"
output_folder = "output"

import os
basepath = os.getcwd()
input_path = os.path.join(basepath, input_folder)
output_path = os.path.join(basepath, output_folder)
os.mkdir(output_path)

!python run.py --input_folder /content/photo_restoration/test_images/old --output_folder /
```

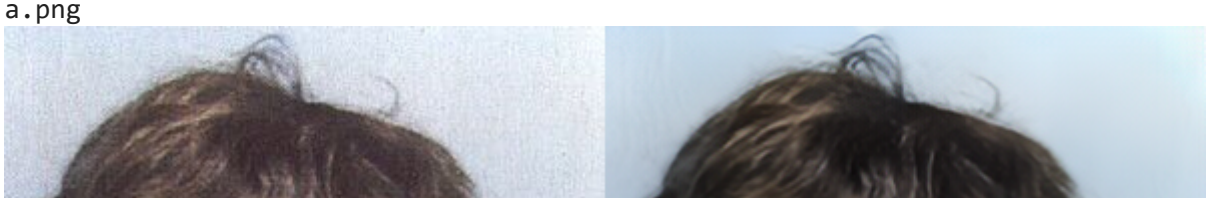
```
/content/photo_restoration
Running Stage 1: Overall restoration
Now you are processing a.png
Now you are processing b.png
Now you are processing c.png
Now you are processing d.png
Now you are processing e.png
Now you are processing f.png
Now you are processing g.png
Now you are processing h.png
Finish Stage 1 ...


Running Stage 2: Face Detection
Warning: There is no face in d.png
1
1
Warning: There is no face in b.png
1
Warning: There is no face in f.png
1
Warning: There is no face in e.png
Finish Stage 2 ...


Running Stage 3: Face Enhancement
The main GPU is
0
dataset [FaceTestDataset] of size 4 was created
The size of the latent vector size is [8,8]
Network [SPADEGenerator] was created. Total number of parameters: 92.1 million. To s
hi :)
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:3063: UserWarning: Def
  "See the documentation of nn.Upsample for details.".format(mode))
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:1628: UserWarning: nn.
  warnings.warn("nn.functional.tanh is deprecated. Use torch.tanh instead.")
Finish Stage 3 ...


Running Stage 4: Blending
Warning: There is no face in d.png
Warning: There is no face in b.png
Warning: There is no face in f.png
Warning: There is no face in e.png
```

```
Finish Stage 4 ...


All the processing is done. Please check the results.
```

```
import io
import IPython.display
import numpy as np
import PIL.Image

def imshow(a, format='png', jpeg_fallback=True):
    a = np.asarray(a, dtype=np.uint8)
    data = io.BytesIO()
    PIL.Image.fromarray(a).save(data, format)
    im_data = data.getvalue()
    try:
      disp = IPython.display.display(IPython.display.Image(im_data))
    except IOError:
      if jpeg_fallback and format != 'jpeg':
        print(('Warning: image was too large to display in format "{}"; '
              'trying jpeg instead.').format(format))
        return imshow(a, format='jpeg')
      else:
        raise
    return disp

def make_grid(I1, I2, resize=True):
    I1 = np.asarray(I1)
    H, W = I1.shape[0], I1.shape[1]

    if I1.ndim >= 3:
        I2 = np.asarray(I2.resize((W,H)))
        I_combine = np.zeros((H,W*2,3))
        I_combine[:,:W,:] = I1[:,:,:3]
        I_combine[:,W:,:] = I2[:,:,:3]
    else:
        I2 = np.asarray(I2.resize((W,H)).convert('L'))
        I_combine = np.zeros((H,W*2))
        I_combine[:,:W] = I1[:,:]
        I_combine[:,W:] = I2[:,:]
    I_combine = PIL.Image.fromarray(np.uint8(I_combine))

    W_base = 600
    if resize:
      ratio = W_base / (W*2)
      H_new = int(H * ratio)
      I_combine = I_combine.resize((W_base, H_new), PIL.Image.LANCZOS)

    return I_combine


filenames = os.listdir(os.path.join(input_path))
filenames.sort()
```

```
for filename in filenames:
    print(filename)
    image_original = PIL.Image.open(os.path.join(input_path, filename))
    image_restore = PIL.Image.open(os.path.join(output_path, 'final_output', filename))

    display(make_grid(image_original, image_restore))
```

a.png



Restore the photos with scratches

```
!rm -rf /content/photo_restoration/output/*
!python run.py --input_folder /content/photo_restoration/test_images/old_w_scratch/ --outp
```

```
Running Stage 1: Overall restoration
initializing the dataloader
model weights loaded
directory of testing image: /content/photo_restoration/test_images/old_w_scratch
processing a.png
processing b.png
processing c.png
processing d.png
You are using NL + Res
Now you are processing a.png
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:3063: UserWarning: Def
  "See the documentation of nn.Upsample for details.".format(mode))
Now you are processing b.png
Now you are processing c.png
Now you are processing d.png
Finish Stage 1 ...


Running Stage 2: Face Detection
2
1
1
1
Finish Stage 2 ...


Running Stage 3: Face Enhancement
The main GPU is
0
dataset [FaceTestDataset] of size 5 was created
The size of the latent vector size is [8,8]
Network [SPADEGenerator] was created. Total number of parameters: 92.1 million. To s
hi :)
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:3063: UserWarning: Def
  "See the documentation of nn.Upsample for details.".format(mode))
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:1628: UserWarning: nn.
  warnings.warn("nn.functional.tanh is deprecated. Use torch.tanh instead.")
Finish Stage 3 ...


Running Stage 4: Blending
Finish Stage 4 ...


All the processing is done. Please check the results.
```

```
input_folder = "test_images/old_w_scratch"
output_folder = "output"
input_path = os.path.join(basepath, input_folder)
output_path = os.path.join(basepath, output_folder)

filenames = os.listdir(os.path.join(input_path))
filenames.sort()

for filename in filenames:
    print(filename)
    image_original = PIL.Image.open(os.path.join(input_path, filename))
    image_restore = PIL.Image.open(os.path.join(output_path, 'final_output', filename))

    display(make_grid(image_original, image_restore))
```

a.png

Try it on your own photos!

```
from google.colab import files
import shutil

upload_path = os.path.join(basepath, "test_images", "upload")
upload_output_path = os.path.join(basepath, "upload_output")

if os.path.isdir(upload_output_path):
    shutil.rmtree(upload_output_path)

if os.path.isdir(upload_path):
    shutil.rmtree(upload_path)

os.mkdir(upload_output_path)
os.mkdir(upload_path)

uploaded = files.upload()
for filename in uploaded.keys():
    shutil.move(os.path.join(basepath, filename), os.path.join(upload_path, filename))
```

Choose Files No file chosen Upload widget is only available when the cell has been execute
this cell to enable.

```
Saving (2).jpg_a.png to (2).jpg_a.png
Saving (29).jpg_a.png to (29).jpg_a.png
Saving (31).jpg_a.png to (31).jpg_a.png
Saving (33).jpg_a.png to (33).jpg_a.png
Saving (62).jpg_a.png to (62).jpg_a.png
Saving (94).jpg_a.png to (94).jpg_a.png
Saving (101).jpg_a.png to (101).jpg_a.png
Saving (107).jpg_a.png to (107).jpg_a.png
Saving (115).jpg_a.png to (115).jpg_a.png
```

Run the processing with:

```
d.png
```

```
!python run.py --input_folder /content/photo_restoration/test_images/upload --output_folde
```

```
Running Stage 1: Overall restoration
Now you are processing (101).jpg_a.png
Now you are processing (107).jpg_a.png
Now you are processing (115).jpg_a.png
Now you are processing (2).jpg_a.png
Now you are processing (29).jpg_a.png
Now you are processing (31).jpg_a.png
Now you are processing (33).jpg_a.png
Now you are processing (62).jpg_a.png
Now you are processing (94).jpg_a.png
Finish Stage 1 ...


Running Stage 2: Face Detection
1
1
1
1
1
1
1
1
1
Finish Stage 2 ...


Running Stage 3: Face Enhancement
The main GPU is
0
dataset [FaceTestDataset] of size 9 was created
The size of the latent vector size is [8,8]
Network [SPADEGenerator] was created. Total number of parameters: 92.1 million. To s
hi :)
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:3063: UserWarning: Def
  "See the documentation of nn.Upsample for details.".format(mode))
/usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:1628: UserWarning: nn.
  warnings.warn("nn.functional.tanh is deprecated. Use torch.tanh instead.")
Finish Stage 3 ...


Running Stage 4: Blending
Finish Stage 4 ...


All the processing is done. Please check the results.
```

Visualize

Now you have all your results under the folder `upload_output` and you can *manually* right click and download them.

Here we use the child photos of celebrities from https://www.boredpanda.com/childhood-celebrities-when-they-were-young-kids/?utm_source=google&utm_medium=organic&utm_campaign=organic

```
filenames_upload = os.listdir(os.path.join(upload_path))
filenames_upload.sort()

filenames_upload_output = os.listdir(os.path.join(upload_output_path, "final_output"))
filenames_upload_output.sort()

for filename, filename_output in zip(filenames_upload, filenames_upload_output):
    image_original = PIL.Image.open(os.path.join(upload_path, filename))
    image_restore = PIL.Image.open(os.path.join(upload_output_path, "final_output", filena

    display(make_grid(image_original, image_restore))
    print("")
```

Download your results

```
output_folder = os.path.join(upload_output_path, "final_output")
print(output_folder)
os.system(f"zip -r -j download.zip {output_folder}/*")
files.download("download.zip")
```

```
/content/photo_restoration/upload_output/final_output
```