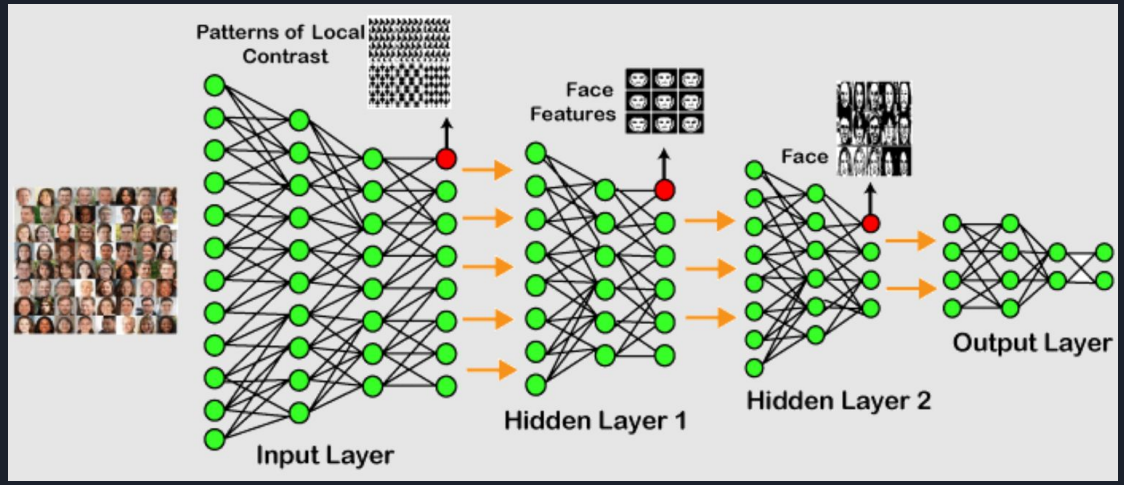
Pixelmind AI

Team DS&Chill

The General Idea

The basic idea was to implement one of the existing Deep Learning models, potentially train it on the given pictures dataset, inspect the results across different models, and select the best one. We also had the idea of piping the output to another model to compensate for the former models “weaknesses”, such as some randomly generated noise in the process.



Hurdles we faced!

Although this did not quite work out we did get valuable insight into the nature of the models and the task on the whole.

We then proceeded to find a model that was efficient enough to run on colab gpus and not run out of gpu ram in the process, while also being able to finish the task in a reasonable amount of time.

This turned out to be a more difficult task than initially anticipated mostly due to the heavy nature of the Deep Learning model at the forefront of this particular task.

We went through the following models and did not find them suitable due to various reasons such as out of memory errors, lack of reproducibility, difficult to recreate dependencies etc.

aiff22 / DPED

Star 1.6k

Code Issues Pull requests

Software and pre-trained models for automatic photo quality enhancement using Deep Convolutional Networks

computer-vision deep-learning image-processing gan convolutional-neural-networks generative-adversarial-networks image-enhancement dped

Updated on Dec 17, 2021 Python

google-research / maxim

Star 707

Code Issues Pull requests

[CVPR 2022 Oral] Official repository for "MAXIM: Multi-Axis MLP for Image Processing". SOTA for denoising, deblurring, deraining, dehazing, and enhancement.

image computer-vision architecture image-processing transformer mlp enhancement image-restoration restoration deblurring denoising dehazing image-enhancement low-level-vision deraining retouching

Updated on Jan 9 Python

HuiZeng / Image-Adaptive-3DLUT

Star 614

Code Issues Pull requests

Learning Image-adaptive 3D Lookup Tables for High Performance Photo Enhancement in Real-time

image-processing color-manipulation computational-photography image-enhancement 3d-luts color-enhancement photo-retouching

Updated on Nov 26, 2022 Python

swz30 / MIRNet

Star 545

Code Issues Pull requests

[ECCV 2020] Learning Enriched Features for Real Image Restoration and Enhancement. SOTA results for image denoising, super-resolution, and image enhancement.

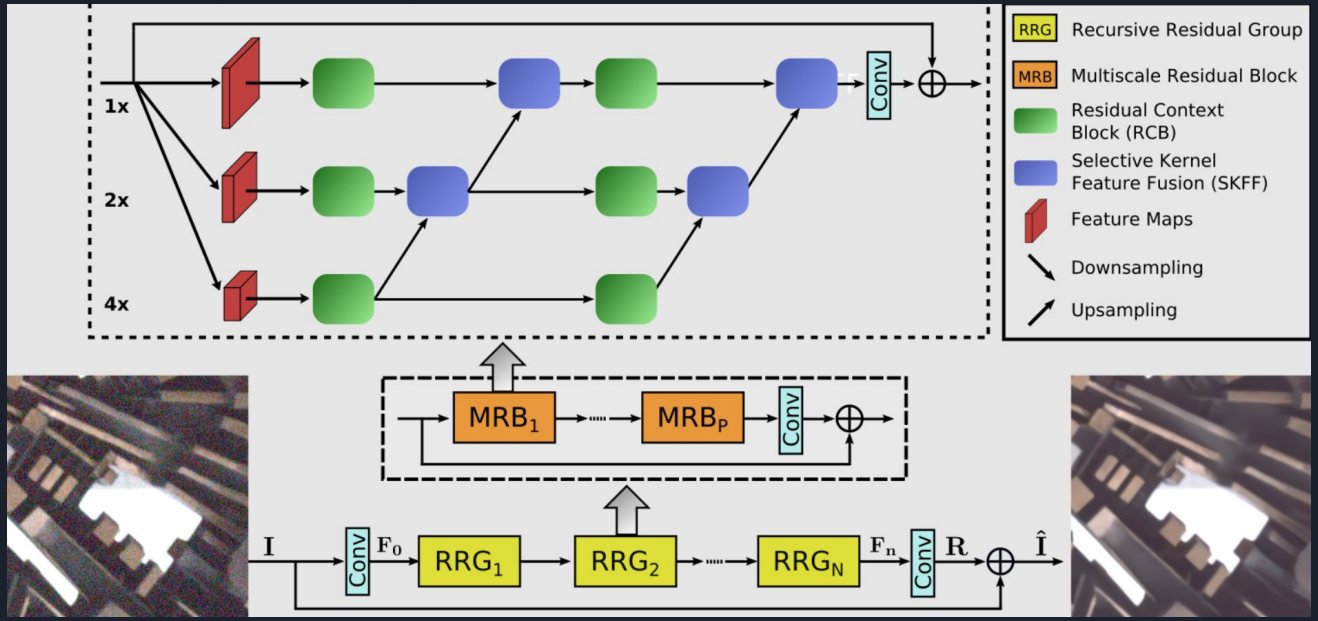
computer-vision pytorch attention-mechanism super-resolution image-denoising image-restoration image-enhancement low-level-vision eccv2020 multi-resolution-streams

Updated on Aug 23, 2022 Python

Finalizing

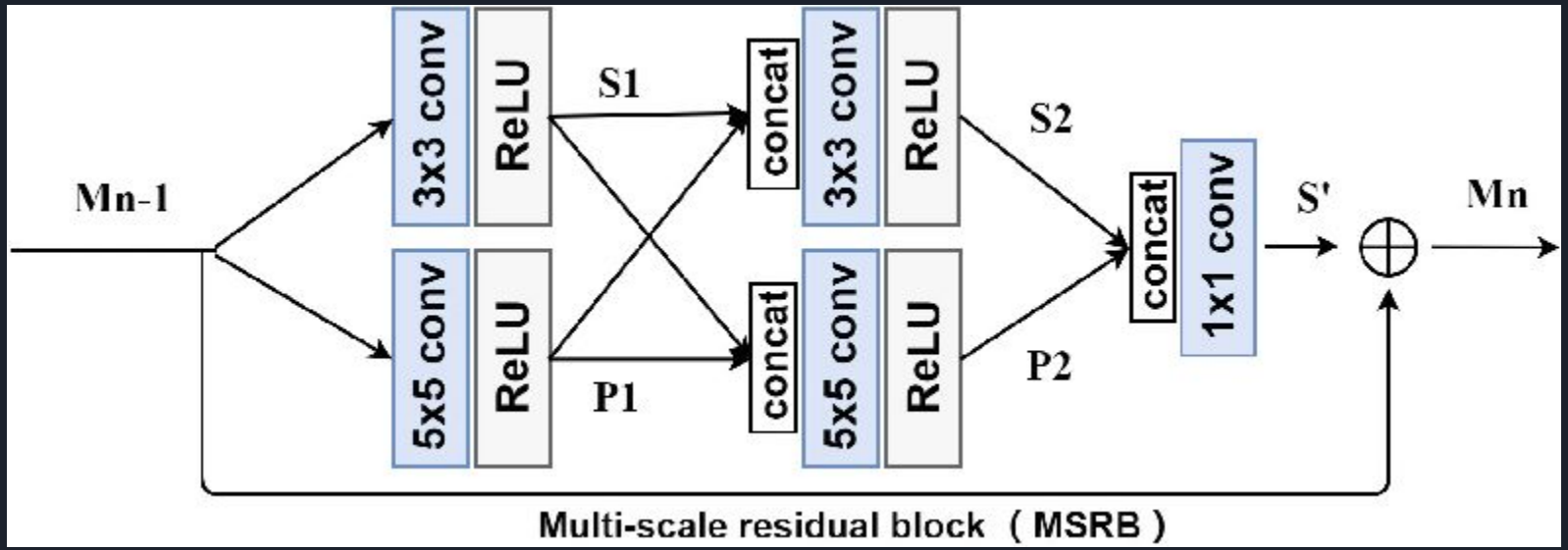
Finally the model we shortlisted was the MIRNetv2, which has the ability to learn contextual information while preserving the high-resolution spatial details, effectively losing lesser information in the process.

The network architecture of the model is illustrated below:

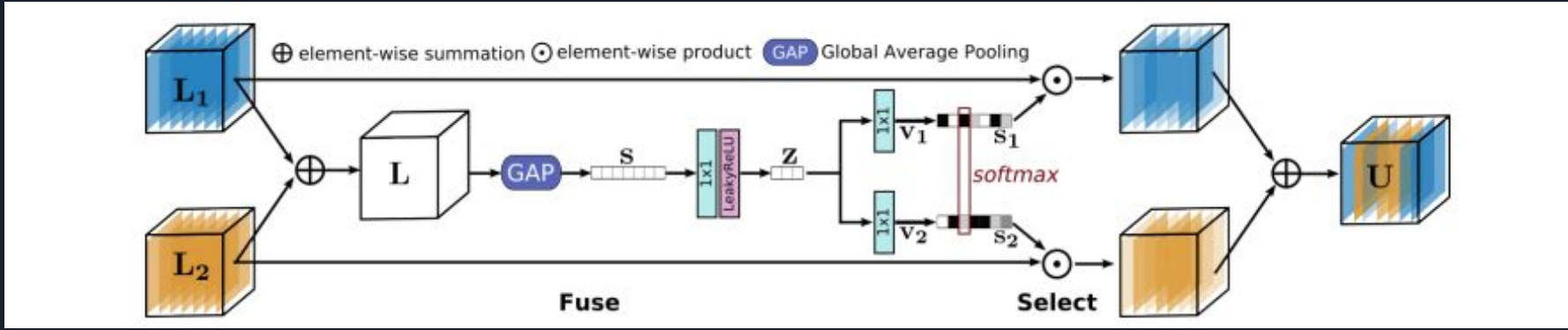


About the network architecture

The fundamental building block of the model is the multi-scale residual block, illustrated below:



Another building block was the selective kernel feature fusion:



Overall Pipeline. Given an image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, the proposed model first applies a convolutional layer to extract low-level features $\mathbf{F_0} \in \mathbb{R}^{H \times W \times C}$. Next, the feature maps $\mathbf{F_0}$ pass through N number of recursive residual groups (RRGs), yielding deep features $\mathbf{F_n} \in \mathbb{R}^{H \times W \times C}$. We note that each RRG contains several multi-scale residual blocks, which is described in Section 3.1. Next, we apply a convolution layer to deep features $\mathbf{F_n}$ and obtain a residual image $\mathbf{R} \in \mathbb{R}^{H \times W \times 3}$. Finally, the restored image is obtained as $\hat{\mathbf{I}} = \mathbf{I} + \mathbf{R}$. We optimize the proposed network using the Charbonnier loss [97]:

$$\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}^*) = \sqrt{\|\hat{\mathbf{I}} - \mathbf{I}^*\|^2 + \varepsilon^2}, \tag{1}$$

The relevant collab notebook:

```
import os
import shutil
from google.colab import files
import torch
import torch.nn.functional as F
import torchvision.transforms.functional as TF
from runpy import run_path
from skimage import img_as_ubyte
from natsort import natsorted
from glob import glob
import cv2
from tqdm import tqdm
import argparse
import numpy as np
import matplotlib.pyplot as plt
```

```
if os.path.isdir('MIRNetv2'):
  !rm -r MIRNetv2

!git clone https://github.com/swz30/MIRNetv2.git
%cd MIRNetv2
```

```
Cloning into 'MIRNetv2'...
remote: Enumerating objects: 207, done.
remote: Counting objects: 100% (207/207), done.
remote: Compressing objects: 100% (172/172), done.
remote: Total 207 (delta 44), reused 171 (delta 22), pack-reused 0
Receiving objects: 100% (207/207), 4.47 MiB | 22.64 MiB/s, done.
Resolving deltas: 100% (44/44), done.
/content/MIRNetv2/MIRNetv2
```

```
task = 'lowlight_enhancement'
!wget https://github.com/swz30/MIRNetv2/releases/download/v1.0.0/enhancement_lol.pth -P Enhancement/pretrained_models
```

```
--2023-05-26 07:48:26--  https://github.com/swz30/MIRNetv2/releases/download/v1.0.0/enhancement_lol.pth
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/392662568/6b7cbd6f-d174-4327-8086-671c1fe1109a?X-Amz-Alg
--2023-05-26 07:48:26--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/392662568/6b7cbd6f-d174-4327-8086-671c1fe
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 23560589 (22M) [application/octet-stream]
Saving to: ‘Enhancement/pretrained_models/enhancement_lol.pth’

enhancement_lol.pth 100%[===================>]  22.47M  --.-KB/s    in 0.1s

2023-05-26 07:48:26 (163 MB/s) - ‘Enhancement/pretrained_models/enhancement_lol.pth’ saved [23560589/23560589]
```

```
!rm -r demo/*
input_dir = 'demo/sample_images/'+task+'/degraded'
os.makedirs(input_dir, exist_ok=True)
uploaded = files.upload()
for filename in uploaded.keys():
  input_path = os.path.join(input_dir, filename)
  shutil.move(filename, input_path)
```

```python
def get_weights_and_parameters(task, parameters):
  if task == 'lowlight_enhancement':
    weights = os.path.join('Enhancement', 'pretrained_models', 'enhancement_lol.pth')
  return weights, parameters

parameters = {
    'inp_channels':3,
    'out_channels':3,
    'n_feat':80,
    'chan_factor':1.5,
    'n_RRG':4,
    'n_MRB':2,
    'height':3,
    'width':2,
    'bias':False,
    'scale':1,
    'task': task
    }

weights, parameters = get_weights_and_parameters(task, parameters)

load_arch = run_path(os.path.join('basicsr', 'models', 'archs', 'mirnet_v2_arch.py'))
model = load_arch['MIRNet_v2'](**parameters)
model.cuda()

checkpoint = torch.load(weights)
model.load_state_dict(checkpoint['params'])
model.eval()
```

```
input_dir = 'demo/sample_images/'+task+'/degraded'
out_dir = 'demo/sample_images/'+task+'/restored'
os.makedirs(out_dir, exist_ok=True)
extensions = ['jpg', 'JPG', 'png', 'PNG', 'jpeg', 'JPEG', 'bmp', 'BMP']
files = natsorted(glob(os.path.join(input_dir, '*')))

img_multiple_of = 4

print(f"\n ==> Running {task} with weights {weights}\n ")
with torch.no_grad():
  for filepath in tqdm(files):
      # print(file_)
      torch.cuda.ipc_collect()
      torch.cuda.empty_cache()
      img = cv2.resize(cv2.cvtColor(cv2.imread(filepath), cv2.COLOR_BGR2RGB),(0,0),fx=0.5,fy=0.5)
      #img = cv2.resize(cv2.cvtColor(cv2.imread(filepath), cv2.COLOR_BGR2RGB),(0,0),fx=1,fy=1)
      img=cv2.convertScaleAbs(img, alpha=0.85)
      input_ = torch.from_numpy(img).float().div(255.).permute(2,0,1).unsqueeze(0).cuda()

      # Pad the input if not_multiple_of 4
      h,w = input_.shape[2], input_.shape[3]
      H,W = ((h+img_multiple_of)//img_multiple_of)*img_multiple_of, ((w+img_multiple_of)//img_multiple_of)*img_multiple_of
      padh = H-h if h%img_multiple_of!=0 else 0
      padw = W-w if w%img_multiple_of!=0 else 0
      input_ = F.pad(input_, (0,padw,0,padh), 'reflect')

      restored = model(input_)
      restored = torch.clamp(restored, 0, 1)

      # Unpad the output
      restored = restored[:,:,:h,:w]

      restored = restored.permute(0, 2, 3, 1).cpu().detach().numpy()
      restored = img_as_ubyte(restored[0])

      filename = os.path.split(filepath)[-1]
      cv2.imwrite(os.path.join("/content/final", filename),cv2.cvtColor(restored, cv2.COLOR_RGB2BGR))
```

Processing a random sample image.

Before:

After:


IC & SR
OFFICE OF
INDUSTRIAL CONSULTANC
& SPONSORED RESEARC
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

Citations:

Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M-H., & Shao, L. (2022). Learning Enriched Features for Fast Image Restoration and Enhancement. IEEE Transactions on Pattern Analysis and Machine Intelligence.

Thank you!