Image Processing in Python

This is a tutorial

Make sure you have your laptop at hand!

Logistics for Cloud Computing

- We will use Google Cloud Platform

- Everyone will get $50 credits (see Canvas)

- Use your credits wisely!

(They should be reserved for GPUs)

Python Tutorial (Not Covered)

- We will support both Python 2.X & Python 3.X

- We recommend Python 3

- Python tutorial from Google (a good starting point)
https://developers.google.com/edu/python/

- The official tutorial (TL;DR)
https://docs.python.org/3/tutorial/index.html

NumPy: Scientific Computing in Python

- Documentation

https://docs.scipy.org/doc/numpy/reference/index.html

- A quick tutorial

https://docs.scipy.org/doc/numpy/user/quickstart.html

- If you are switching from Matlab to Numpy

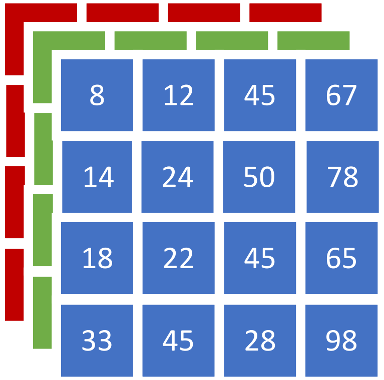
http://mathesaurus.sourceforge.net/matlab-numpy.html

Image Processing Packages in Python

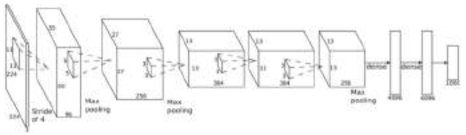
- Python Image Library – PIL (latest dev **Pillow**)
  - Light weighted library for basic image processing
  - Well polished functions
  - Limited functionality, not very efficient

- Open Source Computer Vision (**OpenCV**)
  - Advanced library for image processing and computer vision
  - Good coverage of functionality, highly optimized
  - Sometimes can be bugy ☹

Putting them together …



Learning Based Methods



Data IO

- Load images / videos
- Decode data
- Pillow/OpenCV

Image Transforms

- Pre-processing
- Data augmentation
- NumPy + Pillow/OpenCV

And everything is done on the cloud …

* Thanks for the support from Google Cloud Platform

Goals

20 min: Set up the cloud computing environment

When you need to use the GPUs

10 min: Basic image IO and image manipulations

How to use OpenCV and Pillow

20 min: Image Processing in Python

Resize, color transforms, filtering, rotation, …

Setup the Cloud

- Sign up for Google Cloud (use your wisc edu account)
- Cloud Console https://console.cloud.google.com/
- Create projects to better manage your computing resources
- Use the navigation bar on the left to
  - Create new instance – a virtual remote server for computing
  - Check your bill, monitor your instances, etc

Create a Cloud Instance

- Go to “Compute Engine” -> “VM Instances” -> Create

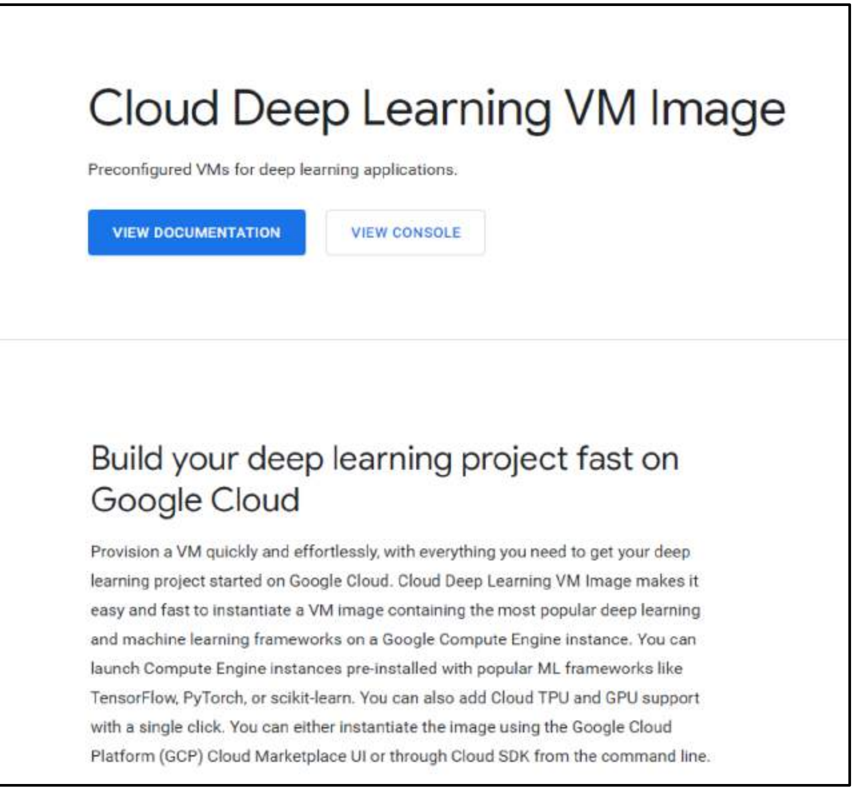
- Recommended: Use this link to launch an pre-configured VM
https://cloud.google.com/deep-learning-vm/

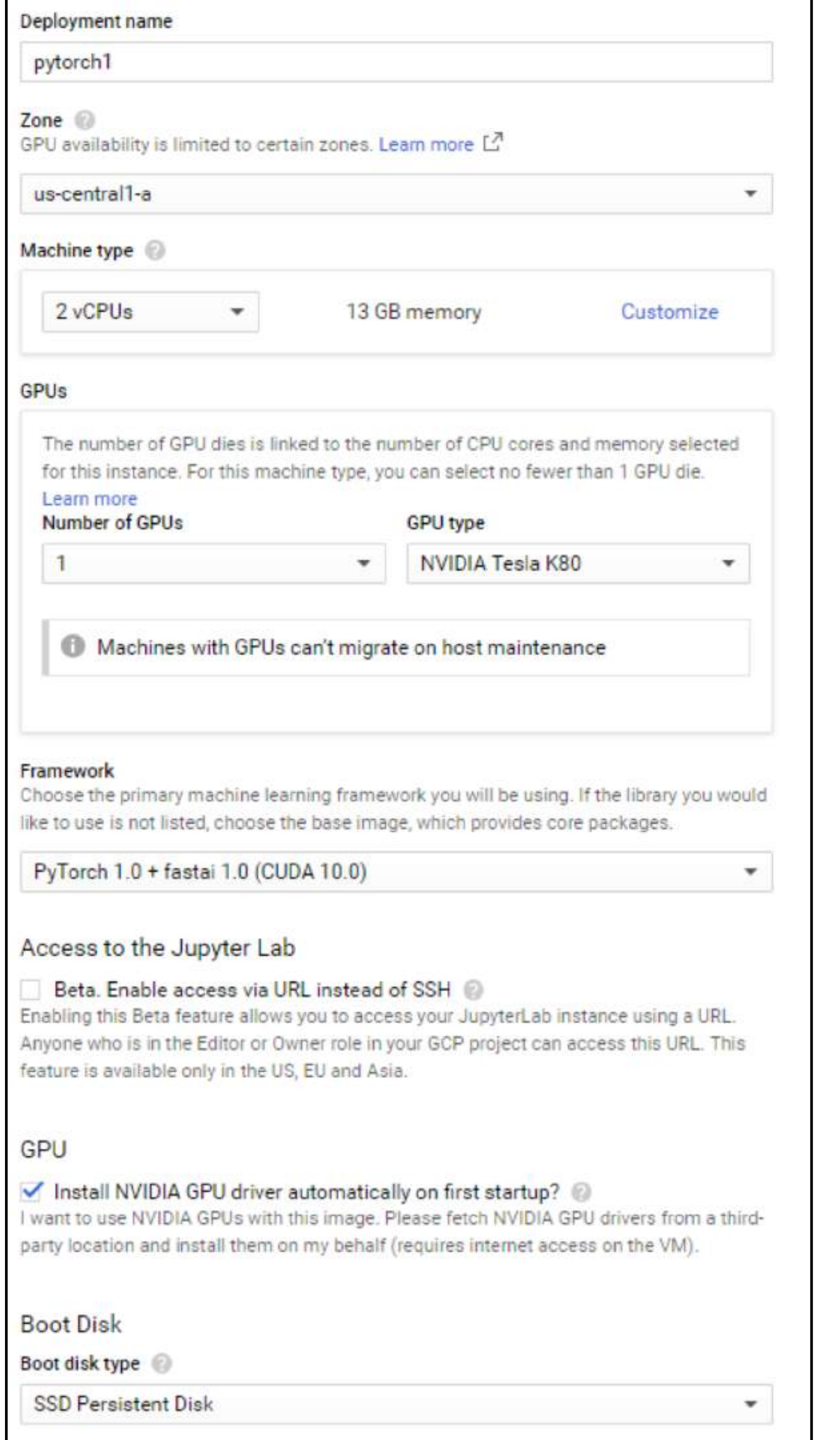
- You can choose CPUs / memory / GPUs

Create a Cloud Instance (cont)

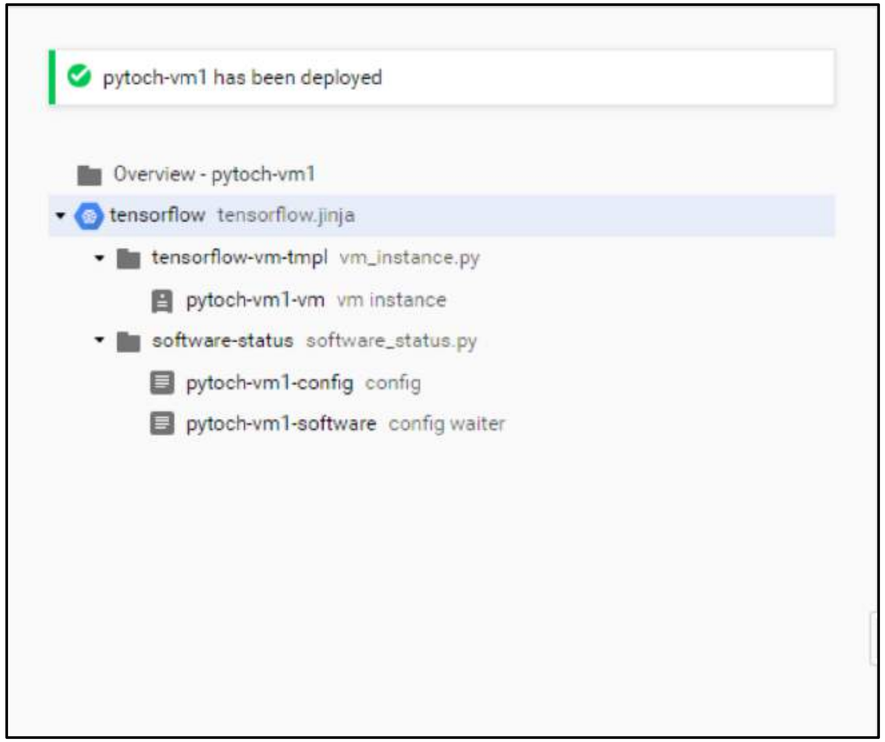
- You can specify CPUs and memory ("customize" in Machine type)
- **Make sure you choose your Deep Learning package**
- You will get an estimated monthly cost (**which is NOT accurate**)
- **When using GPUs, make sure you choose to install the driver**
- We recommend the Nvidia K80 to get started

Examples





K80



Access the Cloud Instance

- You can check your launched instance under “Compute Engine” -> “VM Instances”

- You can ssh into the instance using Google Console

- But we recommend using SSH keys to get access

Access the Cloud Instance (cont)

- You can check your launched instance under “Compute Engine” -> “VM Instances”

- You can ssh into the instance using Google Console

- We recommend using SSH keys to get access the instance
  - Allow multiple users (your team members) to access the instance

Access the Cloud Instance (recommended)

- You will need the following tools
  - ssh-keygen: create your public and private key pairs
  - ssh client: to access the server
  - They are easy to setup in Linux/macOS
  - We recommend Putty for Windows users (http://www.putty.org/)
- Step 1: Create the SSH key pairs
  - *ssh-keygen -t rsa*
- Step 2: Copy your public key to Google Cloud
  - “Compute Engine” -> “VM Instances” -> Your instance -> edit -> SSH keys -> Add item -> Save
- Step 3: *ssh -i your_private_key your_account@server_ip*

Examples (cont)

```
overflocat@XPS-WYF:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/overflocat/.ssh/id_rsa):
/home/overflocat/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/overflocat/.ssh/id_rsa.
Your public key has been saved in /home/overflocat/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:mbSXzyhsqOCMkYKUvBiRWY/e79+jlfX1YXfKJPyqw64 overflocat@XPS-WYF
The key's randomart image is:
+---[RSA 2048]----+
|  .              |
| + o             |
|+ . .   .        |
|.o..   . + o     |
|.+. .   S o + .o+|
|+o.  . o . * *.o=|
|*..   o +.+ o = .|
|.= . o . ++  .   |
|. o . ..E+++.    |
+----[SHA256]-----+
overflocat@XPS-WYF:~$ ls
chinadns.txt     google-cloud-sdk-203.0.0-linux-x86_64.tar.gz  movie
download         hw3_grader                                    stop.sh
google-cloud-sdk img                                           styletransfer
overflocat@XPS-WYF:~$ cd .ssh
overflocat@XPS-WYF:~/.ssh$ ls
config                google_compute_engine.pub  id_rsa      known_hosts
google_compute_engine google_compute_known_hosts id_rsa.pub
overflocat@XPS-WYF:~/.ssh$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDJLR3rHIYQ2o0p5bOCmokWLdjsx8XzMk3IWppTyi9brWaQA8CnH
yxBtzCzL12s+x36gSgd8zKewSmrfof45cRQB9Azxah8hsOzSZhzDwLJfOPyPiVcnnBVBJAUAimcwg9GMdFcNfD/2Z
0Xs61VfVp9xvsJ1wmOG31Q05AjOOHCJs6IdrzqBR7tSSd4cdoksZo/cxZb5xeA0OCbGSWeiHii4h5yzjUnGx/adfU
yuone3kjry8BV5SFenFXsThro3pQRK/FXdG3QYskcxv/5t7W8xF5El0rUPqJuDonTZcybuQGsAiJKWcZPKOAV+R5J
GErAjk2Za1wQP+ugJKBcCMJx overflocat@XPS-WYF
overflocat@XPS-WYF:~/.ssh$
```

SSH Keys

Block project-wide SSH keys

When checked, project-wide SSH keys cannot access this instance Learn more

You have 0 SSH keys

overflocat

```
Azxah8hsOzSZhzDwLJfOPyPiVcnnBVBJAUAimcwg9G
MdFcNfD/2Z0Xs61VfVp9xvsJ1wmOG31Q05AjOOHCJs
6IdrzqBR7tSSd4cdoksZo/cxZb5xeA0OCbGSWeiHii
4h5yzjUnGx/adfUyuone3kjry8BV5SFenFXsThro3p
QRK/FXdG3QYskcxv/5t7W8xF5El0rUPqJuDonTZcyb
uQGsAiJKWcZPKOAV+R5JGErAjk2Za1wQP+ugJKBcCM
Jx overflocat@XPS-WYF
```

+ Add item

```
overflocat@XPS-WYF:~/.ssh$ ssh -i id_rsa overflocat@35.225.247.205
The authenticity of host '35.225.247.205 (35.225.247.205)' can't be established.
ECDSA key fingerprint is SHA256:Gk+odxhY9Oyln2+SE/s8/IUlJmfisL49JqqPeGWp1Uc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '35.225.247.205' (ECDSA) to the list of known hosts.
========================================
Welcome to the Google Deep Learning VM
========================================

Version: m19
Based on: Debian GNU/Linux 9.7 (stretch) (GNU/Linux 4.9.0-8-amd64 x86_64\n)

Resources:
 * Google Deep Learning Platform StackOverflow: https://stackoverflow.com/questions/tagge
d/google-dl-platform
 * Google Cloud Documentation: https://cloud.google.com/deep-learning-vm
 * Google Group: https://groups.google.com/forum/#!forum/google-dl-platform

To reinstall Nvidia driver (if needed) run:
sudo /opt/deeplearning/install-driver.sh
This image uses python 3.7 from the Anaconda. Anaconda is installed to:
/opt/anaconda3/

Linux pytoch-vm1-vm 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
overflocat@pytoch-vm1-vm:~$ python
Python 3.7.1 (default, Dec 14 2018, 19:28:38)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>>
```

Now that you have an instance …

- **The server has to be on Linux**, but you can still ssh from a window box
- Make sure you are using the right version of Python!
- PIL and OpenCV are pre-installed if you are using the pre-configured instance.
- Otherwise, you need to pip install them

Do remember to terminate your instance!

"Compute Engine" -> "VM Instances" -> "Delete"

Image IO

- Load / Save an image using PIL / OpenCV

- Convert the loaded image into NumPy array

- PIL and OpenCV loads a DIFFERENT channel ordering!

- Image resolution (W, H) & Data Type (uint8)

Images in Python

- Suppose we have a NxM RGB image called "im"
  - im(0,0,0) = top-left pixel value in 1st channel
  - im(y, x, b) = y pixels down, x pixels to right in the 2nd channel
  - im(N-1, M-1, 2) = bottom-right pixel in the 3rd channel

column

row

R or B

0.92	0.93	0.94	0.97	0.62	0.37	0.85	0.97	0.93	0.92	0.99
0.95	0.89	0.82	0.89	0.56	0.31	0.75	0.92	0.81	0.95	0.91
0.89	0.72	0.51	0.55	0.51	0.42	0.57	0.41	0.49	0.91	0.92
0.96	0.95	0.88	0.94	0.56	0.46	0.91	0.87	0.90	0.97	0.95
0.71	0.81	0.81	0.87	0.57	0.37	0.80	0.88	0.89	0.79	0.85
0.49	0.62	0.60	0.58	0.50	0.60	0.58	0.50	0.61	0.45	0.33
0.86	0.84	0.74	0.58	0.51	0.39	0.73	0.92	0.91	0.49	0.74
0.96	0.67	0.54	0.85	0.48	0.37	0.88	0.90	0.94	0.82	0.93
0.69	0.49	0.56	0.66	0.43	0.42	0.77	0.73	0.71	0.90	0.99
0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

G

0.92	0.99
0.95	0.91
0.91	0.92
0.97	0.95
0.79	0.85
0.45	0.33
0.49	0.74
0.82	0.93
0.90	0.99
0.93	0.97
0.99	0.93

0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

B or R

0.92	0.99
0.95	0.91
0.91	0.92
0.97	0.95
0.79	0.85
0.45	0.33
0.49	0.74
0.82	0.93
0.90	0.99
0.93	0.97
0.99	0.93

0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

Slide Credit: James Hays

Image Manipulations

- Read a pixel value
- Modify pixel values within an image (careful: data type!)
- Swap the color channels
- Flip the image (vertical / horizontal)
- Crop a region from the image

Basic IO & Manipulation (PIL)

```python
from PIL import Image

im = Image.open("dog.jpg")
w, h = im.size
print('Original image size: %sx%s' % (w, h))
print('Image mode is: %s' % im.mode)
```

```
Original image size: 313x161
Image mode is: RGB
```

```python
out_hf = im.transpose(Image.FLIP_LEFT_RIGHT)
out_vf = im.transpose(Image.FLIP_TOP_BOTTOM)
out_hf.save("dog_hf.jpg")
out_vf.save("dog_vf.jpg")
box = (0, 0, 100, 100)
region = im.crop(box)
region.save("dog_crop.jpg")
```

```python
import numpy as np

region_np = np.array(region)
print('Data type is:', region_np.dtype)
w, h, c = region_np.shape
print('Original image size: %sx%sx%s' % (w, h, c))
region_np[0:20, 0:20] = np.array([0, 0, 255])
region_new = Image.fromarray(np.uint8(region_np))
region_new.save("dog_crop_new.jpg")
```

```
Data type is: uint8
Original image size: 100x100x3
```

The interactive OpenCV version is also uploaded on Canvas.

Image Resizing (Zoom in / out)

- Increase / reduce the resolution of the image
- Different interpolation schemes (how to fill in the pixels)
- Sometimes can be tricky

Nearest Neighbor Bicubic

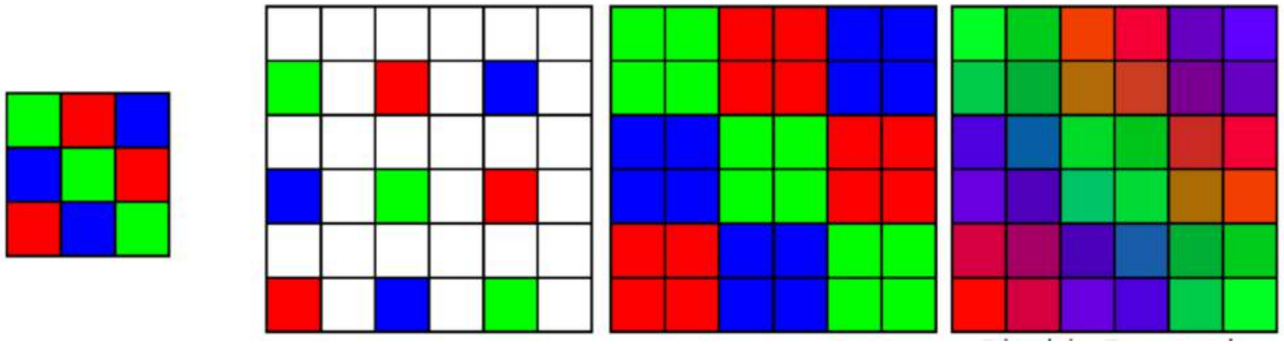
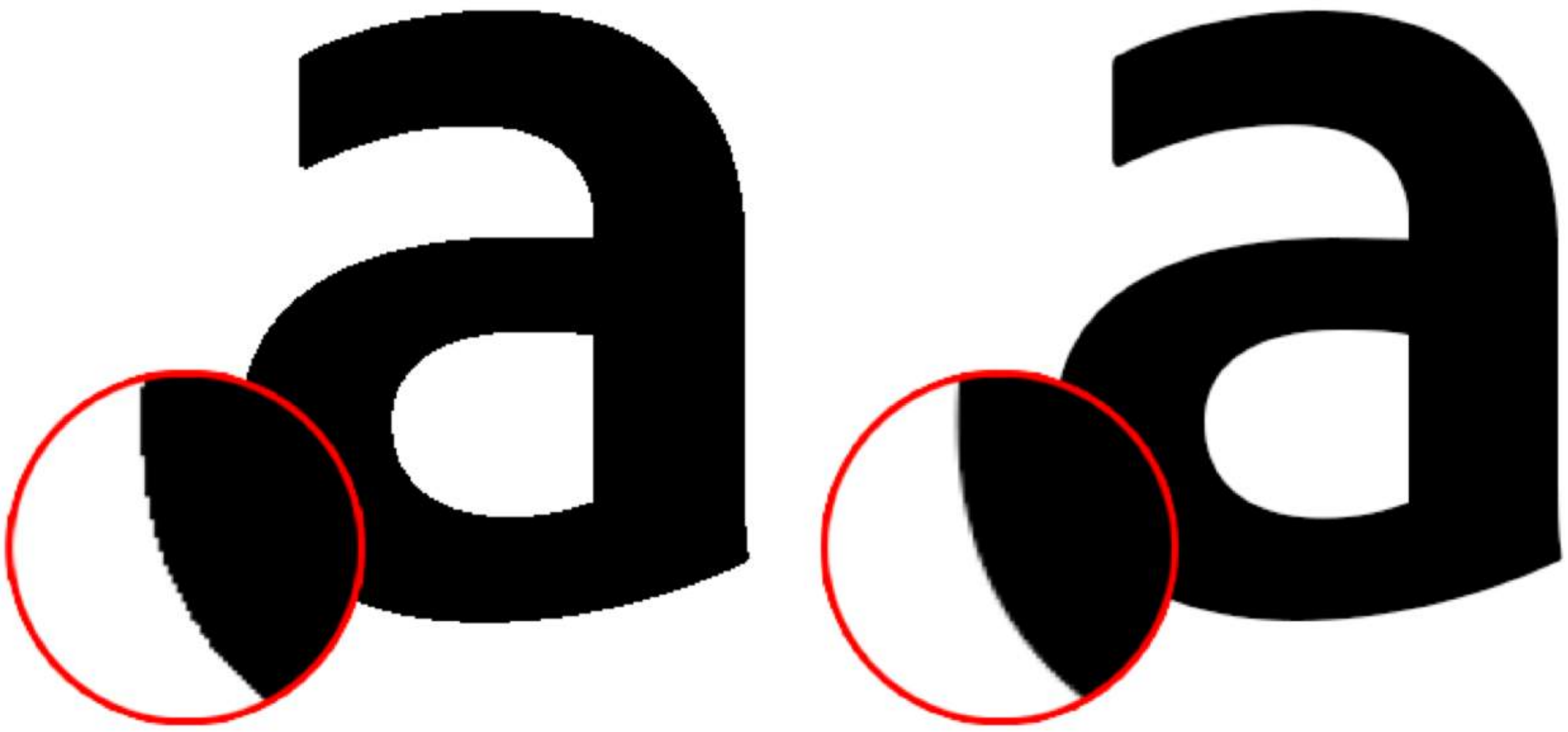


Image Resizing – The Tricks

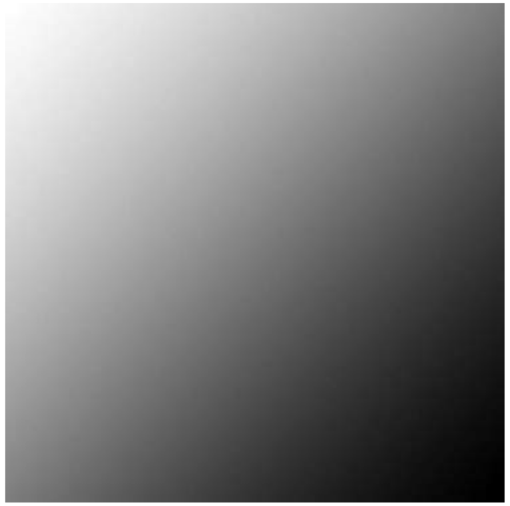
- Anti-aliasing (blur a bit before resizing)



What if you want to resize a mask?

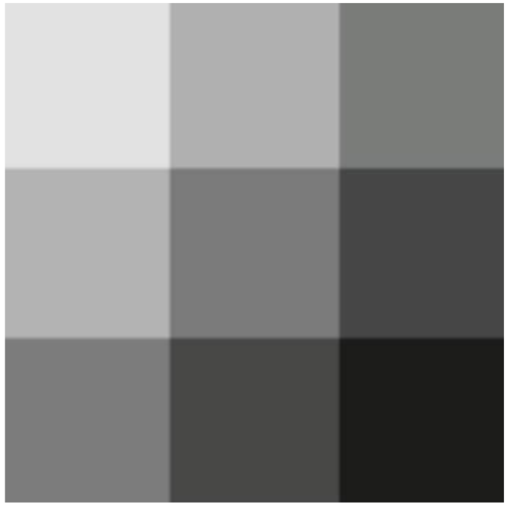
Even nearest neighbor interpolation can be bugy!

Input Image



OpenCV

Pillow



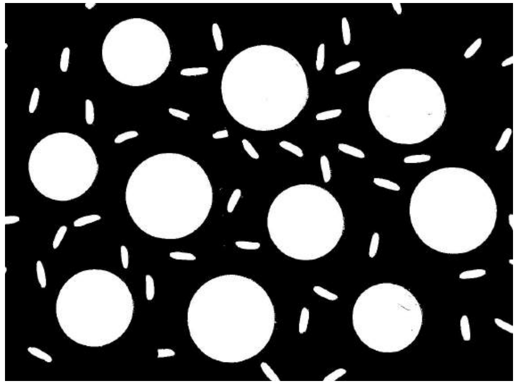
https://github.com/opencv/opencv/issues/9096

Image Upsampling (PIL)

```
im = Image.open('lenna.jpg')
w, h = im.size
print('Original image size: %sx%s' % (w, h))
im_up_nn = im.resize((w*10, h*10), resample=Image.NEAREST)
im_up_bl = im.resize((w*10, h*10), resample=Image.BILINEAR)
im_up_bc = im.resize((w*10, h*10), resample=Image.BICUBIC)
w, h = im_up_nn.size
print('Upsampled image size: %sx%s' % (w, h))
im_up_nn.save('lenna_up_nn.jpg')
im_up_bl.save('lenna_up_bl.jpg')
im_up_bc.save('lenna_up_bc.jpg')
```

```
Original image size: 326x326
Upsampled image size: 3260x3260
```

Original Image 1



Original Image 2
(From CS534)

The interactive OpenCV version is also uploaded on Canvas.

Nearest Neighbor

Nearest Neighbor

Bilinear

Bilinear

Bicubic

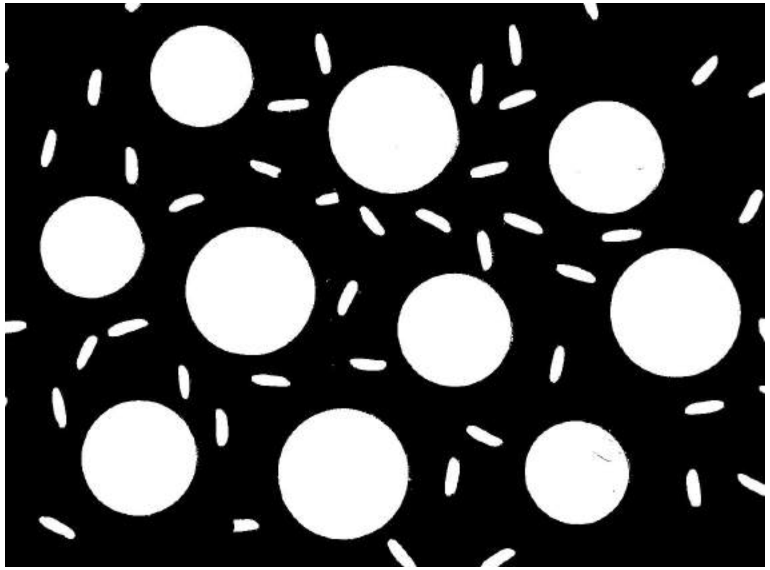
Bicubic

Image Downsampling (PIL)

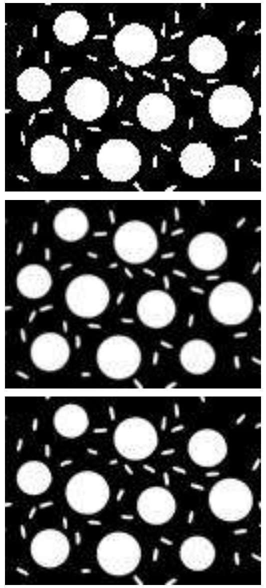
```
im = Image.open('lenna.jpg')
w, h = im.size
print('Original image size: %sx%s' % (w, h))
im_up_nn = im.resize((w//3, h//3), resample=Image.NEAREST)
im_up_bl = im.resize((w//3, h//3), resample=Image.BILINEAR)
im_up_bc = im.resize((w//3, h//3), resample=Image.BICUBIC)
w, h = im_up_nn.size
print('Downsampled image size: %sx%s' % (w, h))
im_up_nn.save('lenna_down_nn.jpg')
im_up_bl.save('lenna_down_bl.jpg')
im_up_bc.save('lenna_down_bc.jpg')
```

```
Original image size: 326x326
Downsampled image size: 108x108
```

Original Image 1



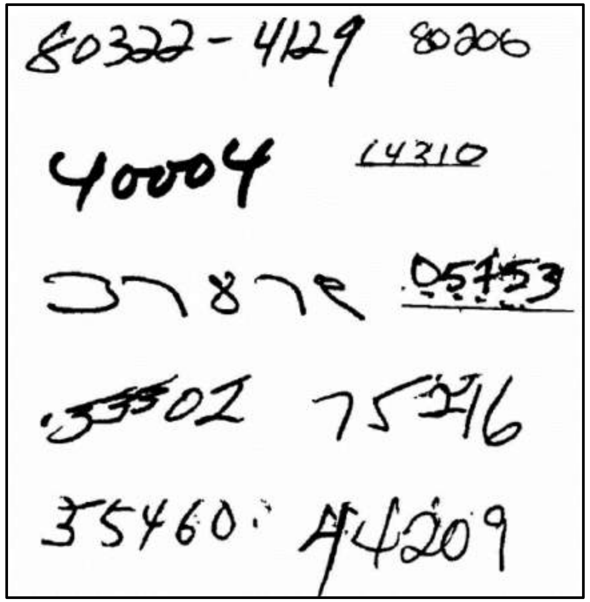
Original Image 2
(From CS534)



The interactive OpenCV version is also uploaded on Canvas.

Image Filtering

Input

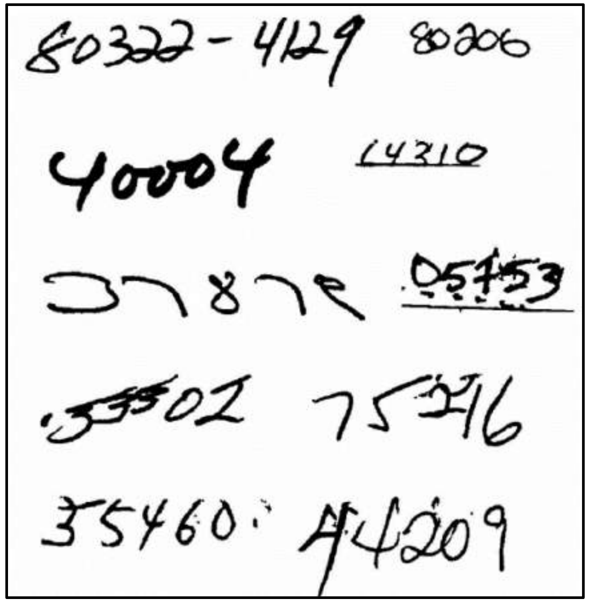


-1	0	+1
-2	0	+2
-1	0	+1

x filter

Image Filtering

Input



$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

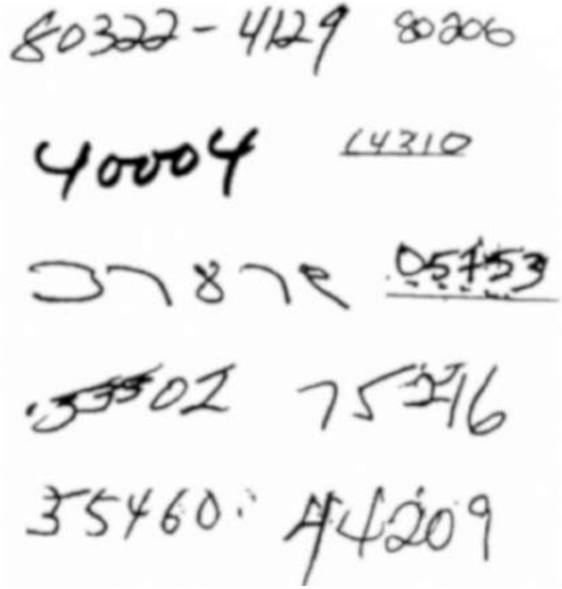
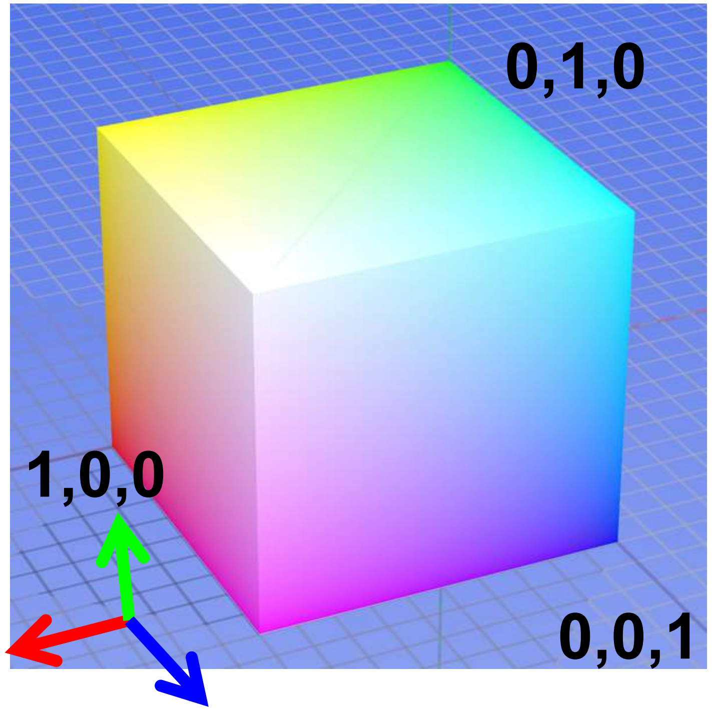
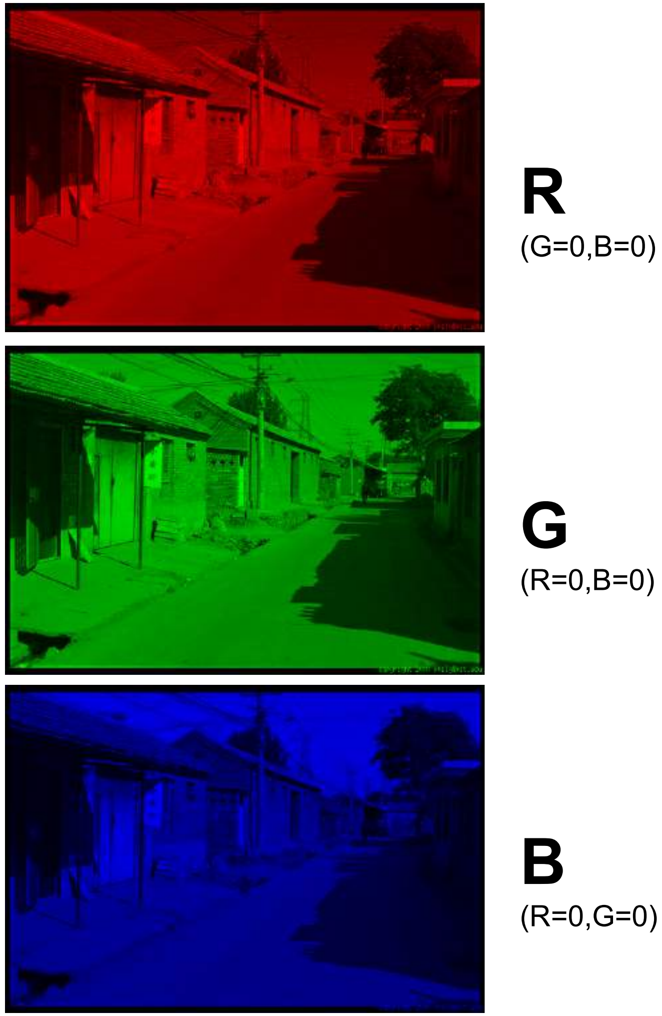


Image Color Space: RGB



Some drawbacks
- Strongly correlated channels
- Non-perceptual



Slide Credit: James Hays

Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Image Color Perturbation

- Rescale the color channels a bit (color jittering)
- Look quite different yet both can be realistic

Image Rotation & Warping

translation

rotation

aspect

affine

perspective

cylindrical

Image Processing Examples (PIL)

```python
import numpy as np
from PIL import Image
from PIL import ImageFilter
im = Image.open('lenna.jpg')
w, h = im.size
print('Original image size: %sx%s' % (w, h))
im_gaussian = im.filter(ImageFilter.GaussianBlur(radius=5))
im_rotated_0 = im.rotate(30, expand=0)
im_rotated_1 = im.rotate(30, expand=1)
im_np = np.array(im).astype(float)
im_np[:, :, 0] = im_np[:, :, 0] / 2
im_jittered = Image.fromarray(np.uint8(im_np))
im_gaussian.save('lenna_gaussian.jpg')
im_rotated_0.save('lenna_rotated_0.jpg')
im_rotated_1.save('lenna_rotated_1.jpg')
im_jittered.save('lenna_jittered.jpg')
w, h = im_rotated_0.size
print('Size without expansion: %sx%s' % (w, h))
w, h = im_rotated_1.size
print('Size with expansion: %sx%s' % (w, h))
```

```
Original image size: 326x326
Size without expansion: 326x326
Size with expansion: 446x446
```

Gaussian Filtering

Color Perturbation

Rotation w/ expansion

Rotation w/o expansion

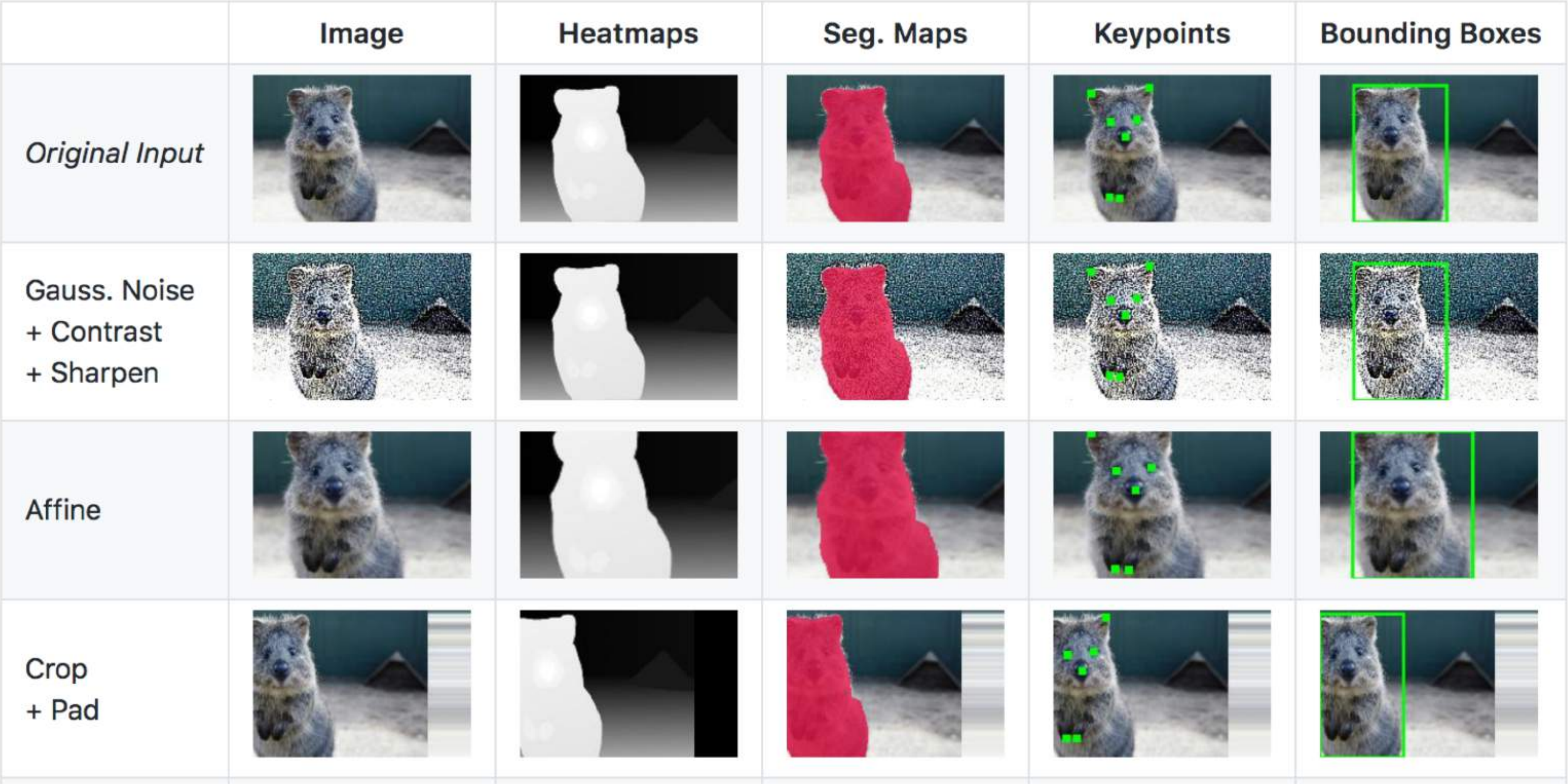
The interactive OpenCV version is also uploaded on Canvas.

Multiple (Random) Image Transforms

- Flip horizontally
- Color jittering
- Rotation

https://colab.research.google.com/drive/109vu3F1LTzD1gdVV6cho9fKGx7lzbFll#scrollTo=BM7-0e4pzLXN

Multiple Image Transforms + Label Transforms



https://github.com/aleju/imgaug

Multiple Image Transforms as Data Augmentation

- Image transforms can drastically change the pixel values

- But the semantics often remain the same

- Get multiple different versions of the same data sample

- Help the learning, e.g., via preventing overfitting

- Create and access an instance on the Cloud
- Basic image IO and manipulations
- Simple image transforms

The end