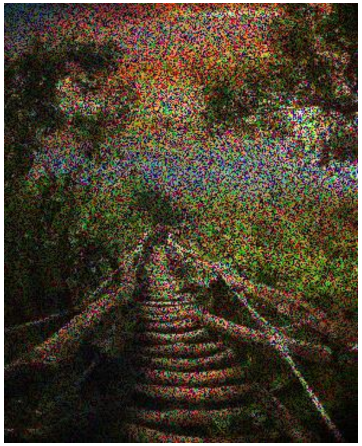
# Image Restoration

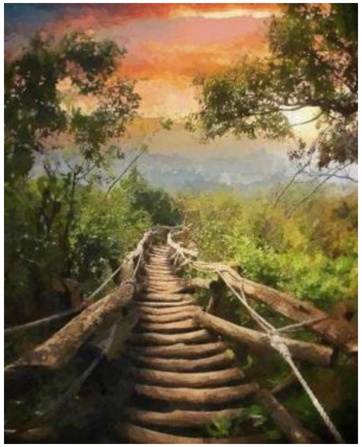
## Example-based Method

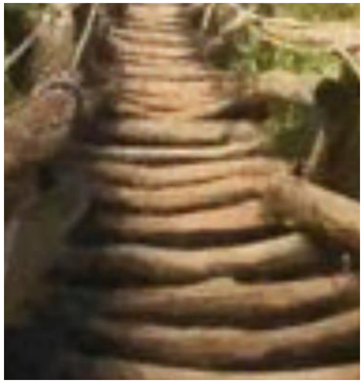
Talker: Xinglu Wang

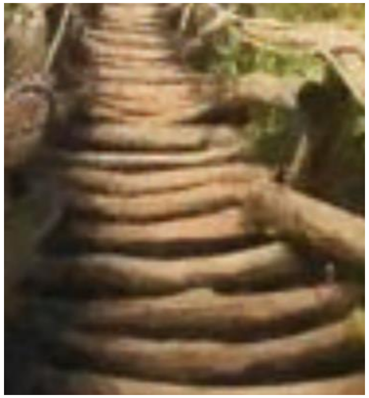
Mentor: Xi Li

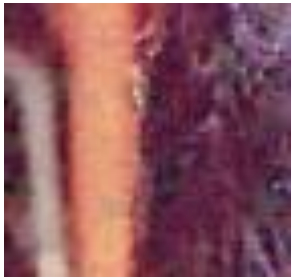
# Results

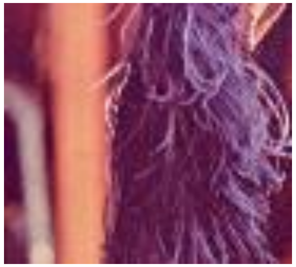


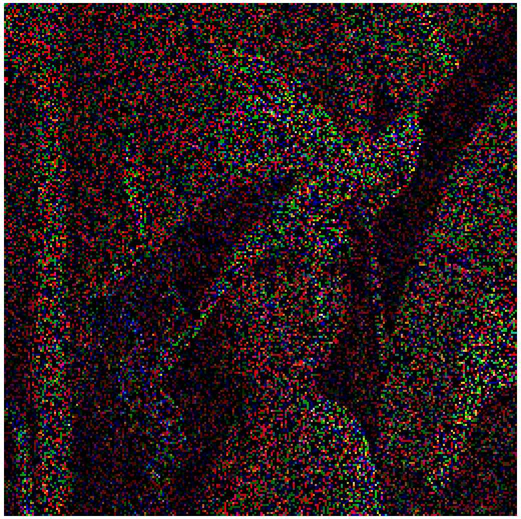


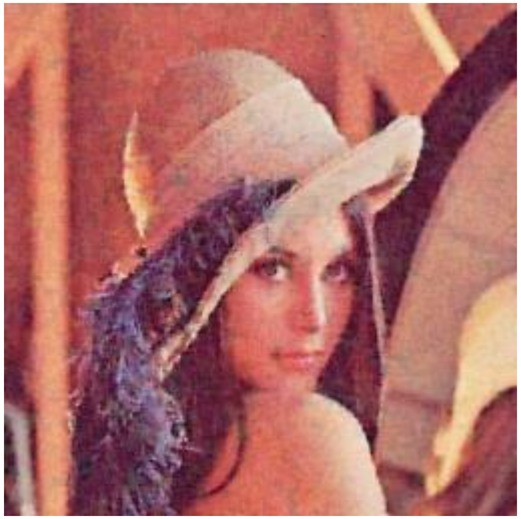




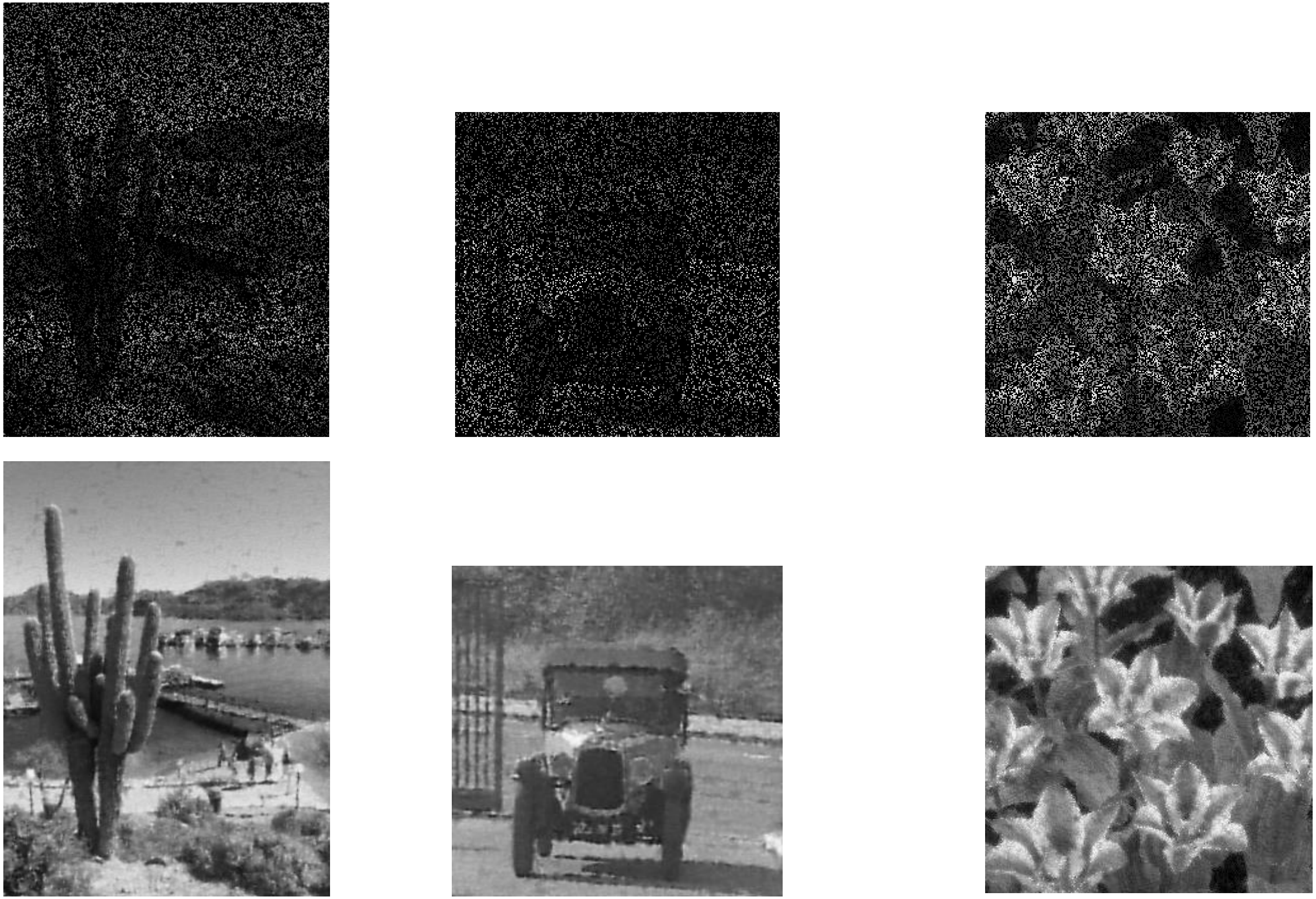




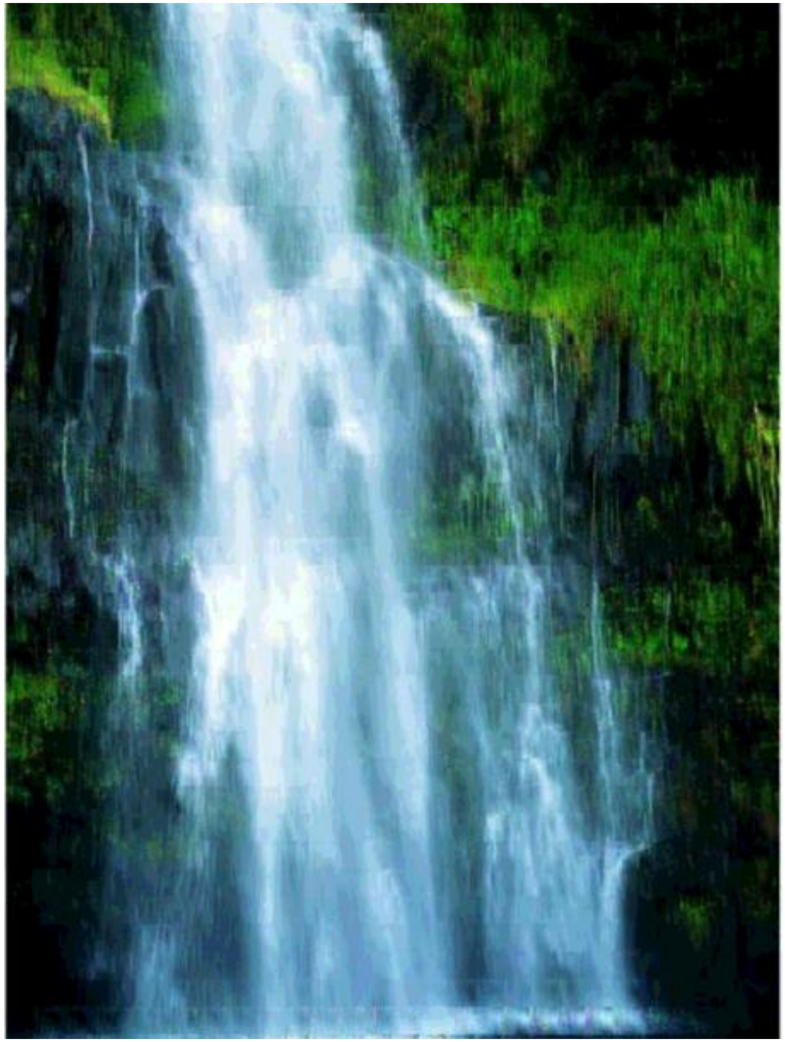




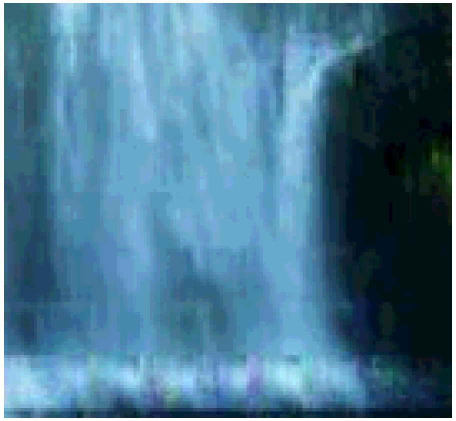
# Results



# Motivations

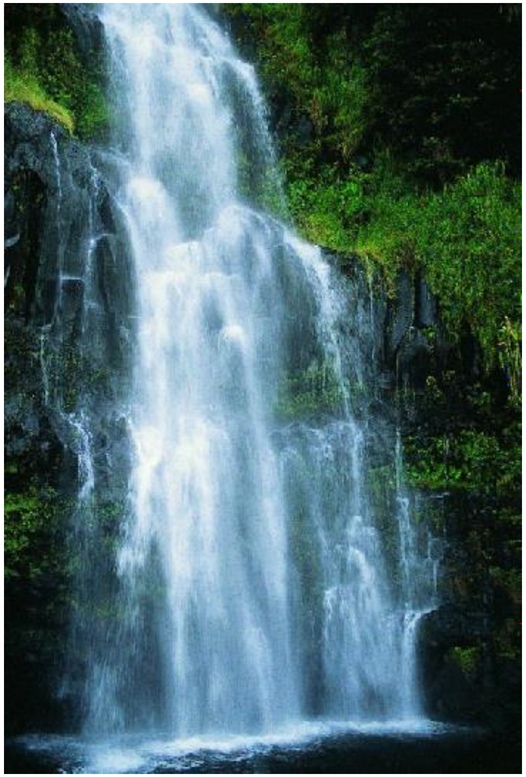


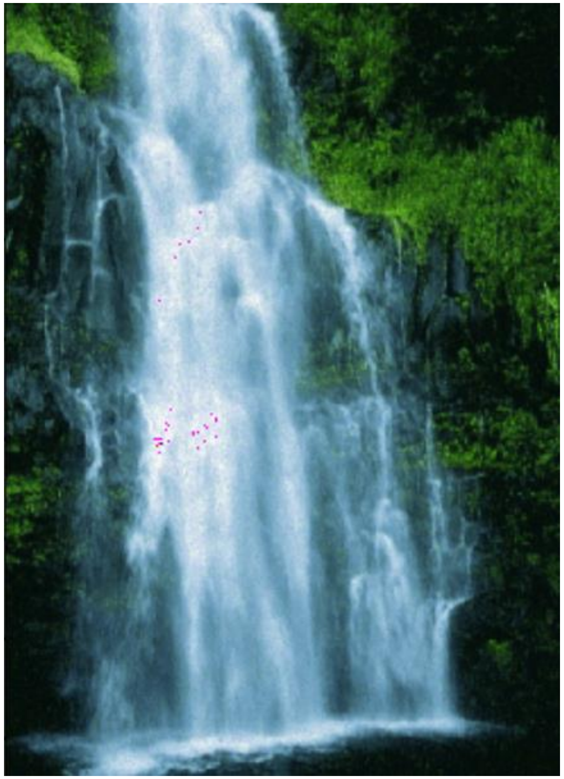
- Region-Based Regression
  - Block-liked Artifacts
  - F(Pixel Index)=Pixel Value
    - Cannot Generalize to Pixels Outside Windows or Imgs



# Internel Example-based

- Make Patches overlapped with each others
- F(Pixel Value)=Pixel Value



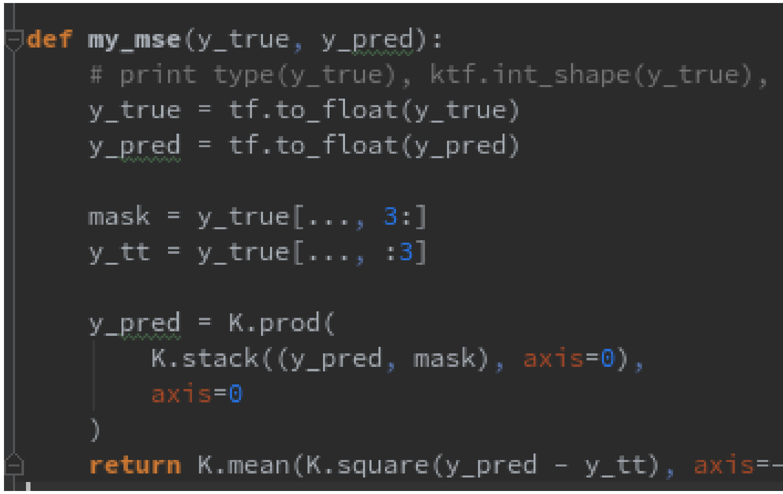


# Internel Example-based

- Holdoff --> Train+Validation
  - Make Many 8*8 Patches
  - Train:Validation=9:1
  - Never Using Ground Truth img
- Customized Loss
  - Omit Unknown Pixels
- Early Stop
  - Prevent overfitting

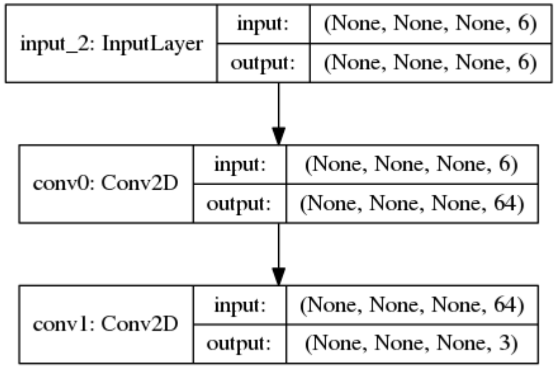
# Internel Example-based

- Holdoff --> Train+Validation
  - img: (512,512,3)
  - x: (62001,8,8,6)
  - y_predict=f(x): (62001,8,8,3)
  - y_true: (62001,8,8,3)
- Customized Loss
  - Omit Unknown Pixels



```
def my_mse(y_true, y_pred):
    # print type(y_true), ktf.int_shape(y_true),
    y_true = tf.to_float(y_true)
    y_pred = tf.to_float(y_pred)

    mask = y_true[..., 3:]
    y_tt = y_true[..., :3]

    y_pred = K.prod(
        K.stack((y_pred, mask), axis=0),
        axis=0
    )
    return K.mean(K.square(y_pred - y_tt), axis=-
```

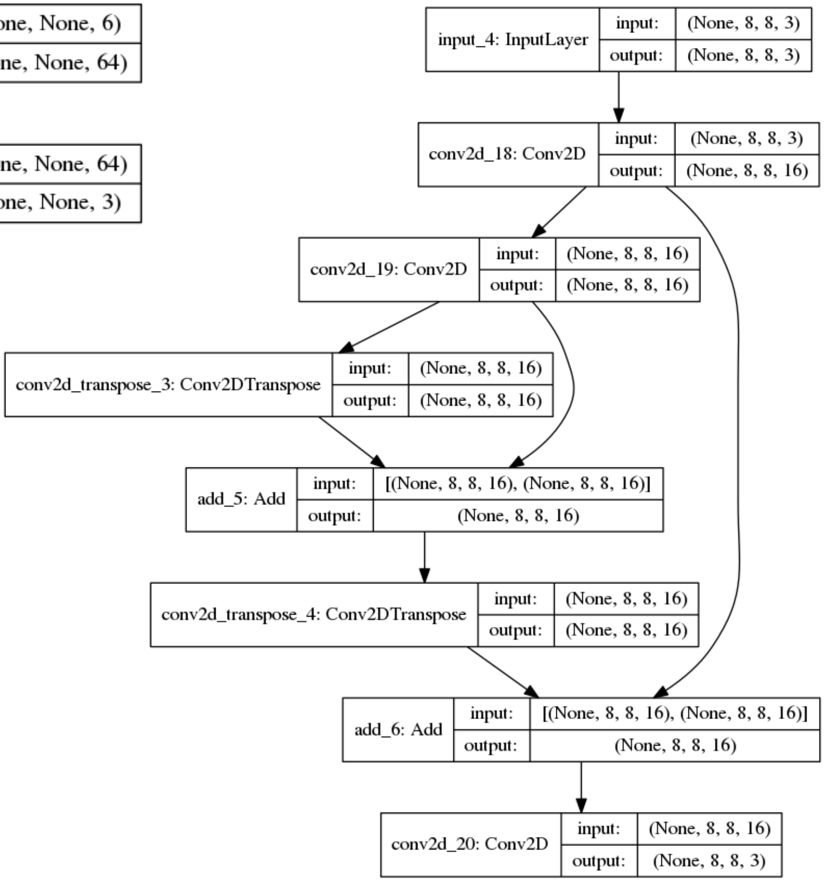
# Choose F( )



input_2: InputLayer input: (None, None, None, 6) output: (None, None, None, 6)

conv0: Conv2D input: (None, None, None, 6) output: (None, None, None, 64)

conv1: Conv2D input: (None, None, None, 64) output: (None, None, None, 3)



input_4: InputLayer input: (None, 8, 8, 3) output: (None, 8, 8, 3)

conv2d_18: Conv2D input: (None, 8, 8, 3) output: (None, 8, 8, 16)

conv2d_19: Conv2D input: (None, 8, 8, 16) output: (None, 8, 8, 16)

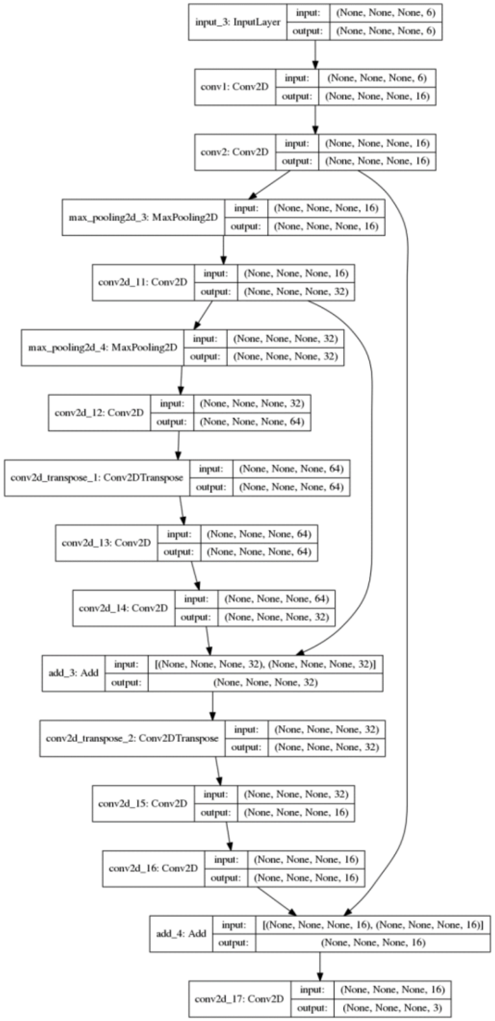
conv2d_transpose_3: Conv2DTranspose input: (None, 8, 8, 16) output: (None, 8, 8, 16)

add_5: Add input: [(None, 8, 8, 16), (None, 8, 8, 16)] output: (None, 8, 8, 16)

conv2d_transpose_4: Conv2DTranspose input: (None, 8, 8, 16) output: (None, 8, 8, 16)

add_6: Add input: [(None, 8, 8, 16), (None, 8, 8, 16)] output: (None, 8, 8, 16)

conv2d_20: Conv2D input: (None, 8, 8, 16) output: (None, 8, 8, 3)



input_3: InputLayer input: (None, None, None, 6) output: (None, None, None, 6)

conv1: Conv2D input: (None, None, None, 6) output: (None, None, None, 16)

conv2: Conv2D input: (None, None, None, 16) output: (None, None, None, 16)

max_pooling2d_3: MaxPooling2D input: (None, None, None, 16) output: (None, None, None, 16)

conv2d_11: Conv2D input: (None, None, None, 16) output: (None, None, None, 32)

max_pooling2d_4: MaxPooling2D input: (None, None, None, 32) output: (None, None, None, 32)

conv2d_12: Conv2D input: (None, None, None, 32) output: (None, None, None, 64)

conv2d_transpose_1: Conv2DTranspose input: (None, None, None, 64) output: (None, None, None, 64)

conv2d_13: Conv2D input: (None, None, None, 64) output: (None, None, None, 64)

conv2d_14: Conv2D input: (None, None, None, 64) output: (None, None, None, 32)

add_3: Add input: [(None, None, None, 32), (None, None, None, 32)] output: (None, None, None, 32)

conv2d_transpose_2: Conv2DTranspose input: (None, None, None, 32) output: (None, None, None, 32)

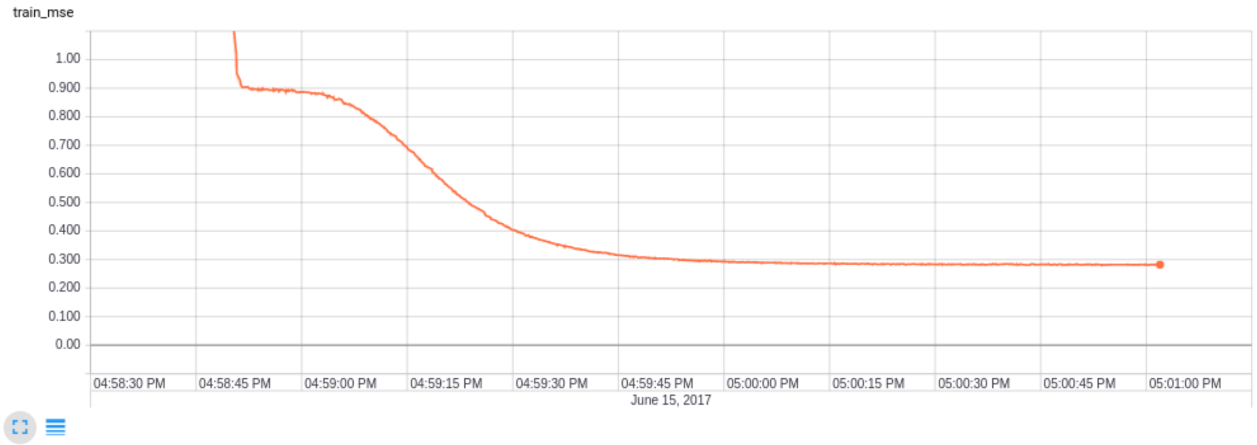
conv2d_15: Conv2D input: (None, None, None, 32) output: (None, None, None, 16)

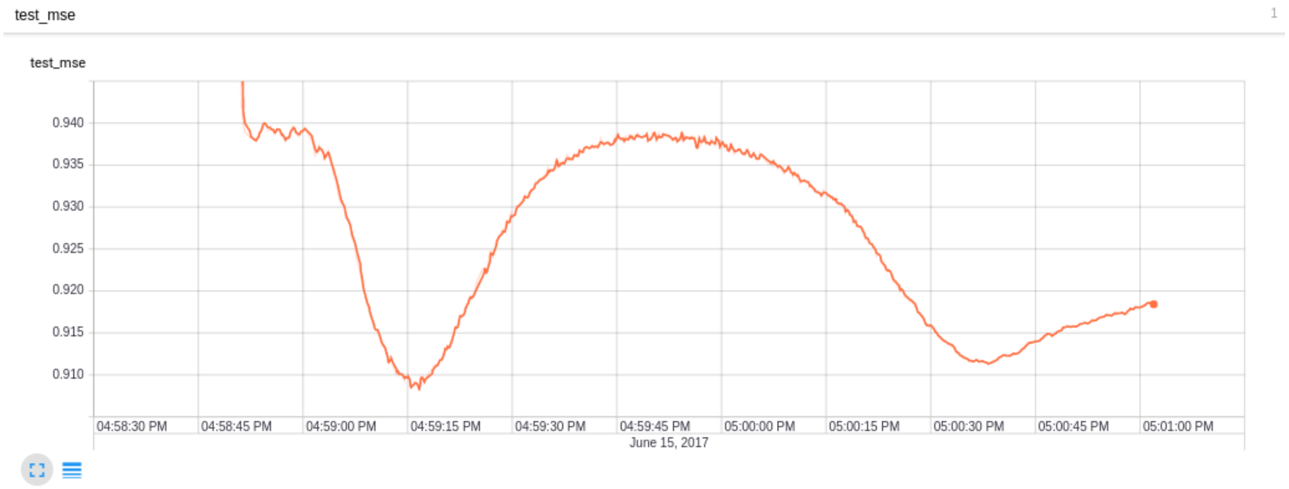
conv2d_16: Conv2D input: (None, None, None, 16) output: (None, None, None, 16)

add_4: Add input: [(None, None, None, 16), (None, None, None, 16)] output: (None, None, None, 16)

conv2d_17: Conv2D input: (None, None, None, 16) output: (None, None, None, 3)

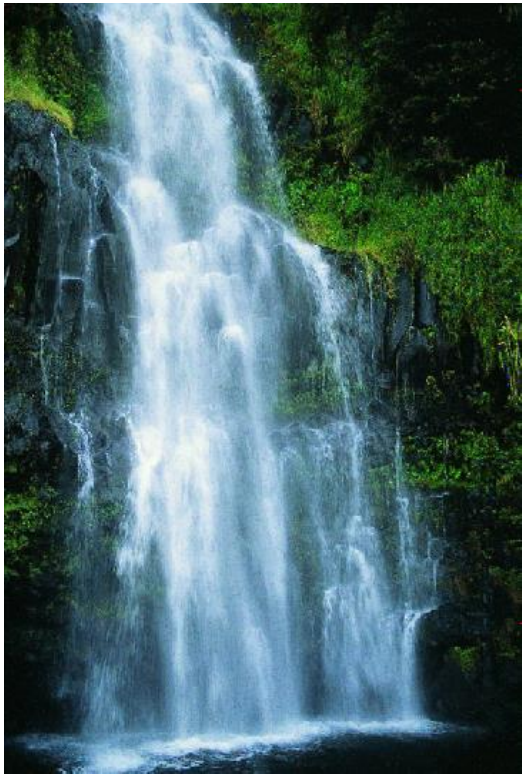
# Early Stop

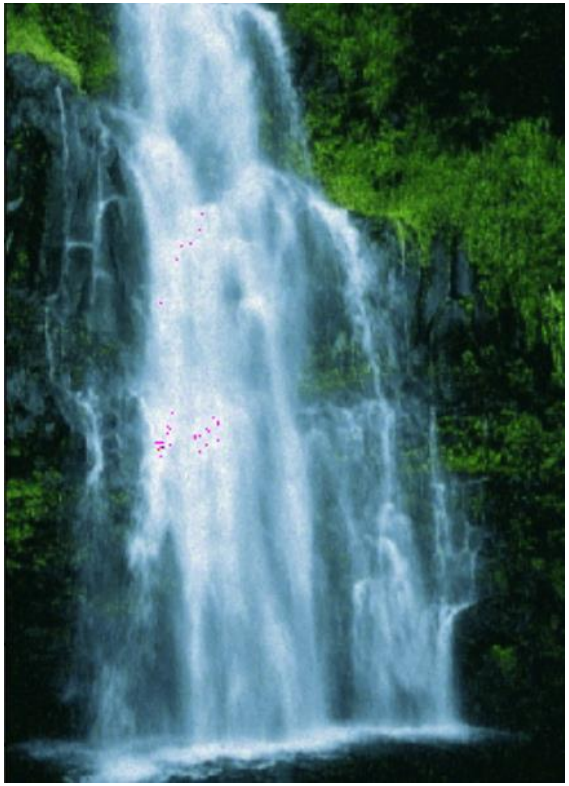

train_mse
1.00
0.900
0.800
0.700
0.600
0.500
0.400
0.300
0.200
0.100
0.00
04:58:30 PM
04:58:45 PM
04:59:00 PM
04:59:15 PM
04:59:30 PM
04:59:45 PM
05:00:00 PM
05:00:15 PM
05:00:30 PM
05:00:45 PM
05:01:00 PM
June 15, 2017


test_mse
1
test_mse
0.940
0.935
0.930
0.925
0.920
0.915
0.910
04:58:30 PM
04:58:45 PM
04:59:00 PM
04:59:15 PM
04:59:30 PM
04:59:45 PM
05:00:00 PM
05:00:15 PM
05:00:30 PM
05:00:45 PM
05:01:00 PM
June 15, 2017

# Internel Example-based

- Make Patches overlapped with each others
- F(Pixel Value)=Pixel Value



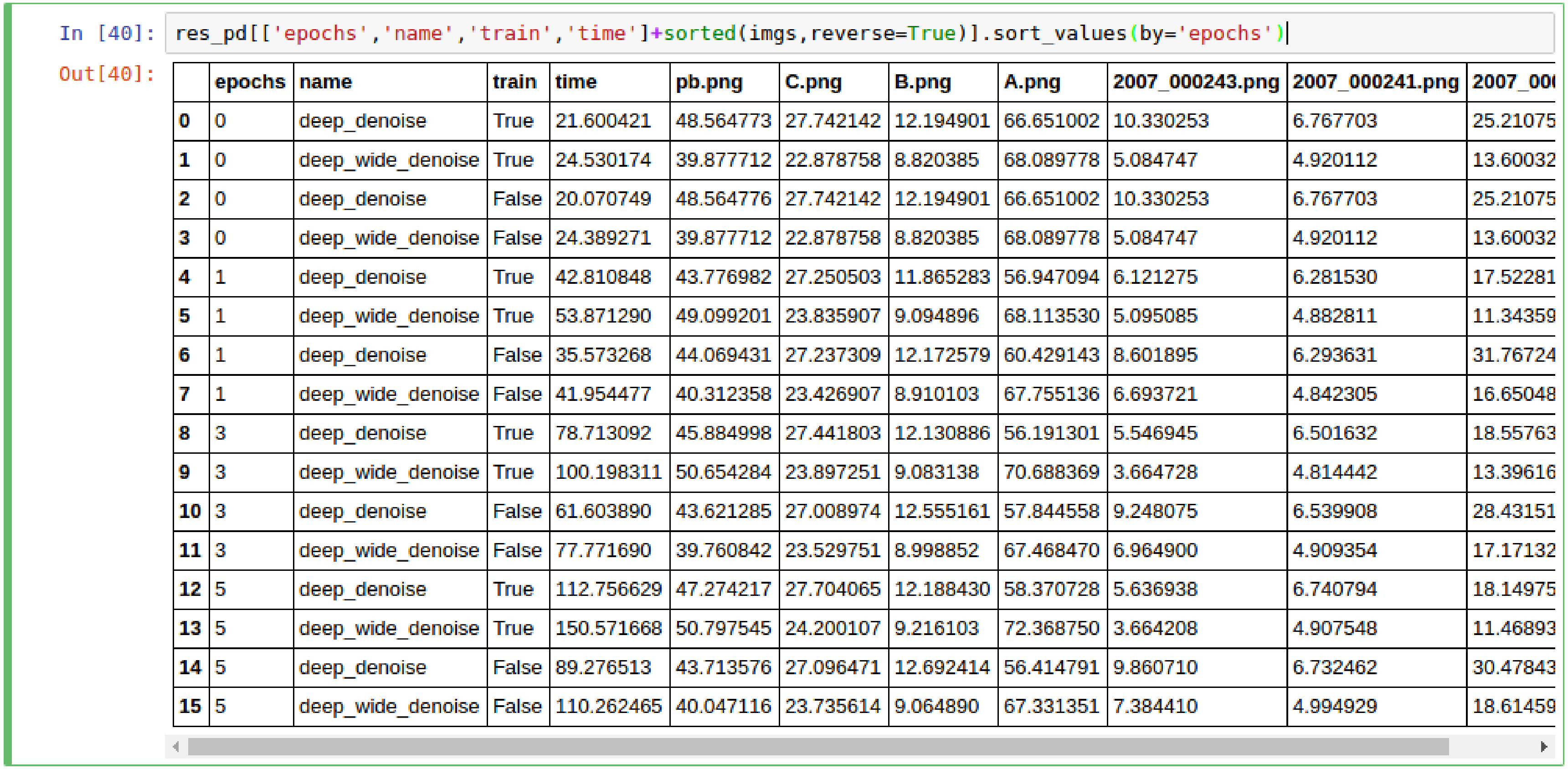


# Learning From Img Database

- Train model offline from VOC2007
- Fully Use Supervision Information
- Grid Search Find Best Hyperparameter

# Externel Example-based

- Will Online Learning Improve the Model Trained From Big Database?

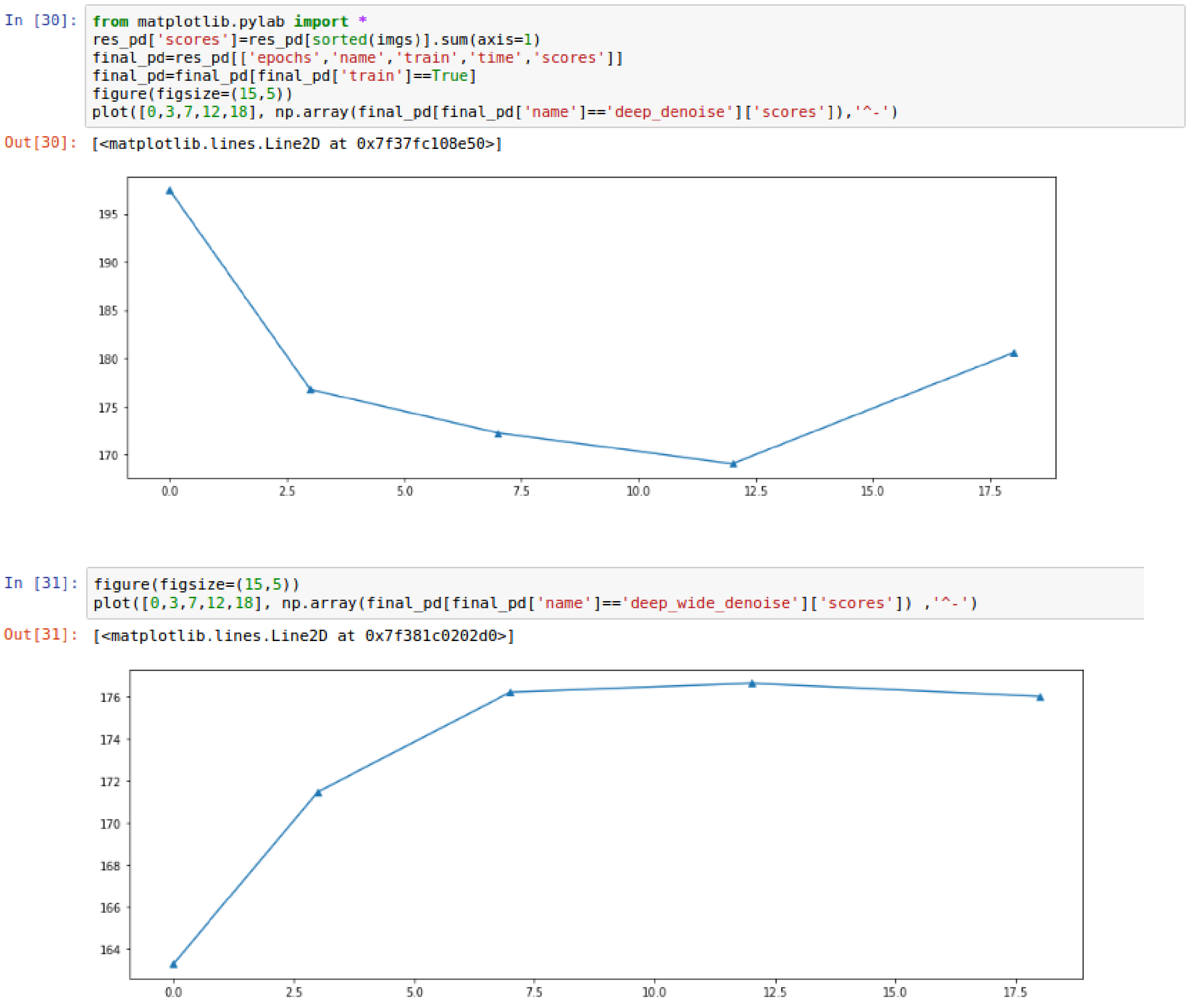


```
In [40]: res_pd[['epochs','name','train','time']+sorted(imgs,reverse=True)].sort_values(by='epochs')
```

Out[40]:

| | epochs | name | train | time | pb.png | C.png | B.png | A.png | 2007_000243.png | 2007_000241.png | 2007_00 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | deep_denoise | True | 21.600421 | 48.564773 | 27.742142 | 12.194901 | 66.651002 | 10.330253 | 6.767703 | 25.21075 |
| **1** | 0 | deep_wide_denoise | True | 24.530174 | 39.877712 | 22.878758 | 8.820385 | 68.089778 | 5.084747 | 4.920112 | 13.60032 |
| **2** | 0 | deep_denoise | False | 20.070749 | 48.564776 | 27.742142 | 12.194901 | 66.651002 | 10.330253 | 6.767703 | 25.21075 |
| **3** | 0 | deep_wide_denoise | False | 24.389271 | 39.877712 | 22.878758 | 8.820385 | 68.089778 | 5.084747 | 4.920112 | 13.60032 |
| **4** | 1 | deep_denoise | True | 42.810848 | 43.776982 | 27.250503 | 11.865283 | 56.947094 | 6.121275 | 6.281530 | 17.52281 |
| **5** | 1 | deep_wide_denoise | True | 53.871290 | 49.099201 | 23.835907 | 9.094896 | 68.113530 | 5.095085 | 4.882811 | 11.34359 |
| **6** | 1 | deep_denoise | False | 35.573268 | 44.069431 | 27.237309 | 12.172579 | 60.429143 | 8.601895 | 6.293631 | 31.76724 |
| **7** | 1 | deep_wide_denoise | False | 41.954477 | 40.312358 | 23.426907 | 8.910103 | 67.755136 | 6.693721 | 4.842305 | 16.65048 |
| **8** | 3 | deep_denoise | True | 78.713092 | 45.884998 | 27.441803 | 12.130886 | 56.191301 | 5.546945 | 6.501632 | 18.55763 |
| **9** | 3 | deep_wide_denoise | True | 100.198311 | 50.654284 | 23.897251 | 9.083138 | 70.688369 | 3.664728 | 4.814442 | 13.39616 |
| **10** | 3 | deep_denoise | False | 61.603890 | 43.621285 | 27.008974 | 12.555161 | 57.844558 | 9.248075 | 6.539908 | 28.43151 |
| **11** | 3 | deep_wide_denoise | False | 77.771690 | 39.760842 | 23.529751 | 8.998852 | 67.468470 | 6.964900 | 4.909354 | 17.17132 |
| **12** | 5 | deep_denoise | True | 112.756629 | 47.274217 | 27.704065 | 12.188430 | 58.370728 | 5.636938 | 6.740794 | 18.14975 |
| **13** | 5 | deep_wide_denoise | True | 150.571668 | 50.797545 | 24.200107 | 9.216103 | 72.368750 | 3.664208 | 4.907548 | 11.46893 |
| **14** | 5 | deep_denoise | False | 89.276513 | 43.713576 | 27.096471 | 12.692414 | 56.414791 | 9.860710 | 6.732462 | 30.47843 |
| **15** | 5 | deep_wide_denoise | False | 110.262465 | 40.047116 | 23.735614 | 9.064890 | 67.331351 | 7.384410 | 4.994929 | 18.61459 |

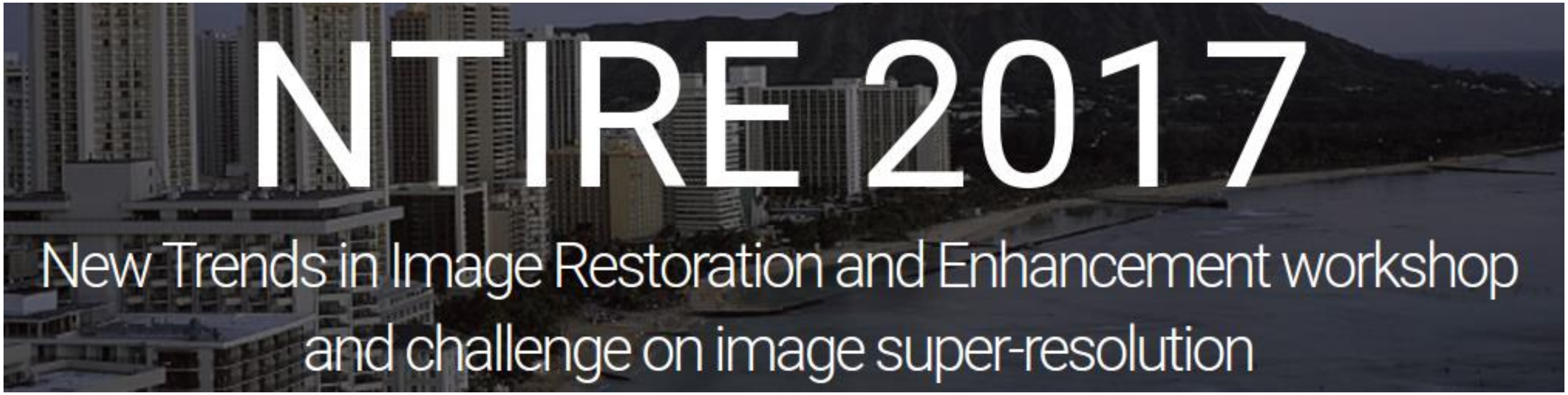
# Externel Example-based

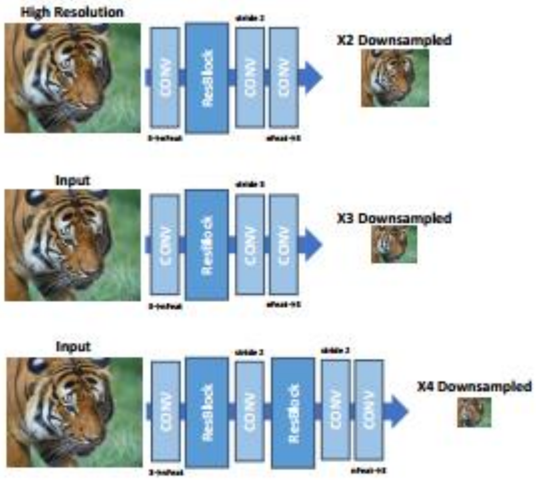


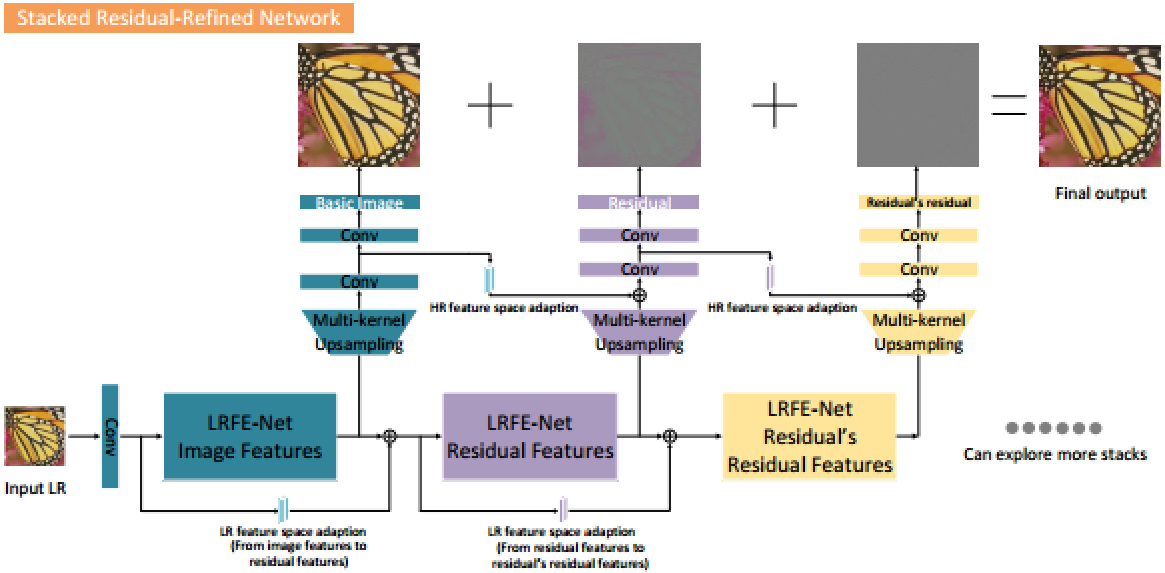
```
In [30]: from matplotlib.pylab import *
         res_pd['scores']=res_pd[sorted(imgs)].sum(axis=1)
         final_pd=res_pd[['epochs','name','train','time','scores']]
         final_pd=final_pd[final_pd['train']==True]
         figure(figsize=(15,5))
         plot([0,3,7,12,18], np.array(final_pd[final_pd['name']=='deep_denoise']['scores']),'^-')

Out[30]: [<matplotlib.lines.Line2D at 0x7f37fc108e50>]
```

```
In [31]: figure(figsize=(15,5))
         plot([0,3,7,12,18], np.array(final_pd[final_pd['name']=='deep_wide_denoise']['scores']) ,'^-')

Out[31]: [<matplotlib.lines.Line2D at 0x7f381c0202d0>]
```

# Related Works


NTIRE 2017
New Trends in Image Restoration and Enhancement workshop
and challenge on image super-resolution




Stacked Residual-Refined Network
+
+
=
Final output
Basic Image
Conv
Conv
Multi-kernel Upsampling
Residual
Conv
Conv
Multi-kernel Upsampling
Residual's residual
Conv
Conv
Multi-kernel Upsampling
HR feature space adaption
HR feature space adaption
Input LR
Conv
LRFE-Net Image Features
LRFE-Net Residual Features
LRFE-Net Residual's Residual Features
Can explore more stacks
LR feature space adaption (From image features to residual features)
LR feature space adaption (From residual features to residual's residual features)

# Thank You!