Improve Mobile Phone's Image Quality by Deep Learning

· Introduction & Review ·

· Augmented Dataset ·

· MPSE(Multi-Phase Semantic Enhancement) ·

· MDIE(Multi-Domain Intrinsic Enhancement) ·

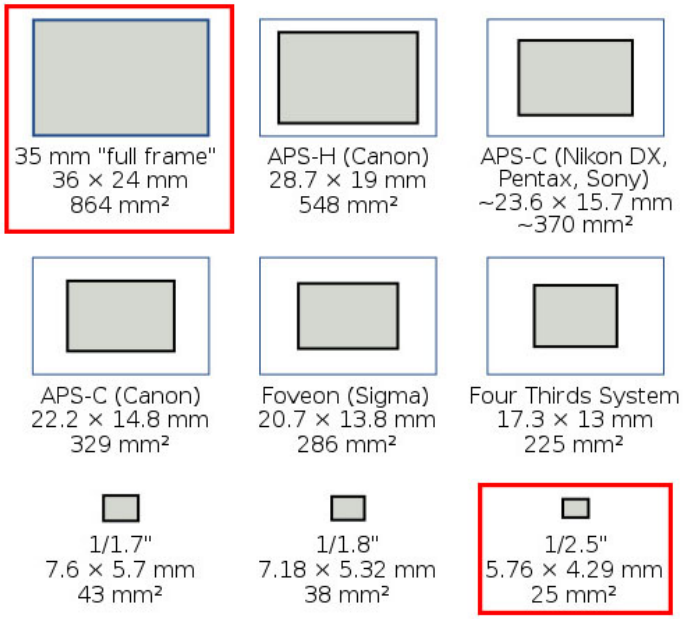
· Future Work·

· Q&A·

HUANG Cen | LIU Yicun

01 Introduction & Review

Problem Introduction

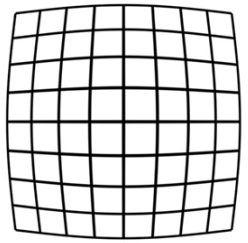


Mobile Phone vs. DSLR

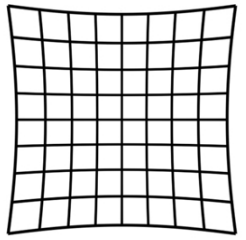
- Small sensors, low-aperture optics
- Low-resolution, lack of details and contrast
- Poor dynamic range, high noise level
- Fall behind their DSLR counterparts

Problem 1: Misalignment in PR Images

- Real-world Paired Data: Distortion and Misalignment
- End-to-End Training: Strait Registration is Essential
- Method Covered: SIFT-Feature Detection, Chessboard Registration, ECC Alignment, Optical Flow, Patch-Match and NNF



Barrel Distortion



Pincushion Distortion

Problem 2: Lack of Semantic Supervision in PR Training

Pairwise Enhancement Scheme:

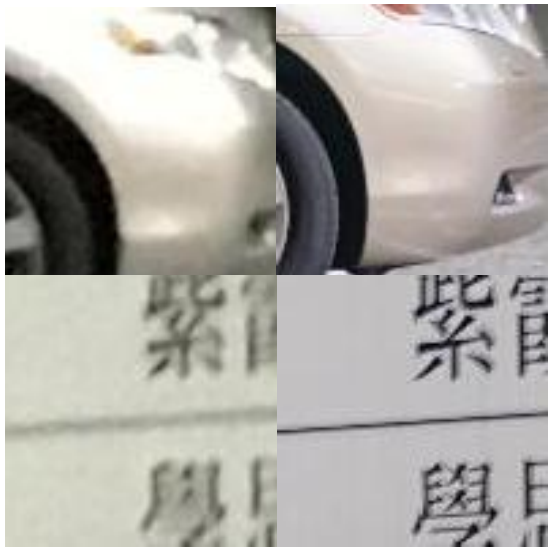
- Random sampling, fixed-size cropping of small patches

Semantic Information Loss:

- Same structural details are enhanced differently

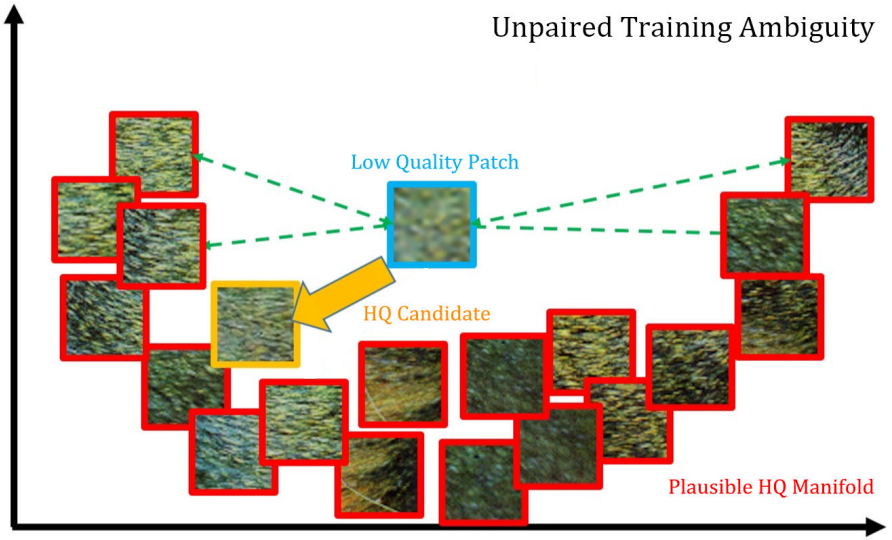
Major Existing Problems:

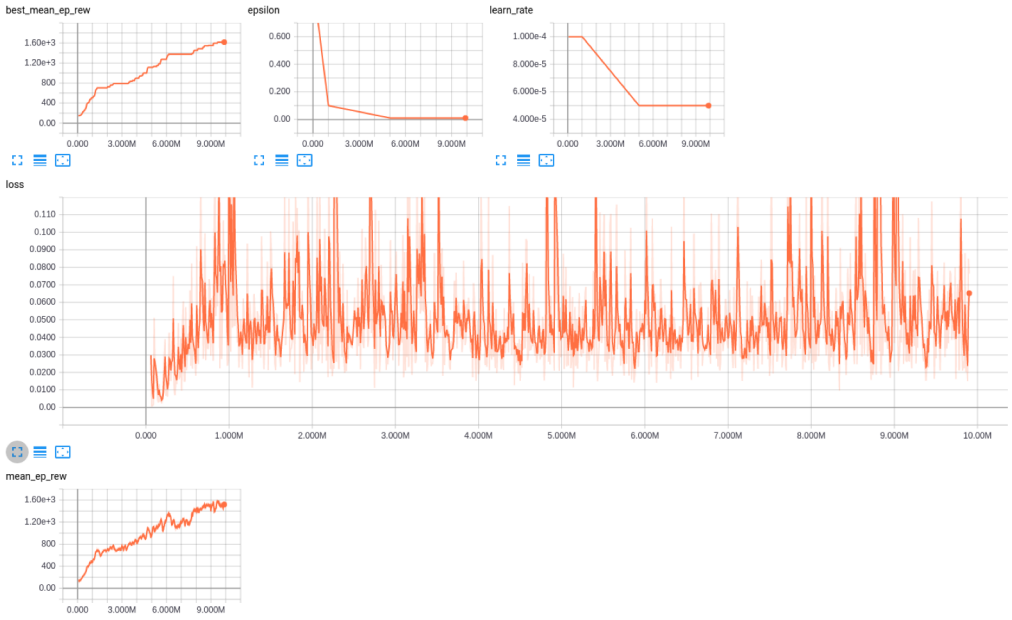
- Block artifacts, structure distortion, unbalanced enhancement



Problem 3: Ambiguity in UPR Training

- Unpaired Methods: Ambiguity in training directions
- Lack of high-level structural supervision: Mismatching texture
- Divergence in weak supervision: weak constraints need to be modified





Sub-Domain Problems

- Image Super-Resolution: SRCNN, FSRCNN, ESPCN, SR-GAN, ResNet, EDSR
- Image Denoising and Restoration: MLP, Auto Encoder, RedNet, BM3D, DND Benchmark Dataset
- Contrast Enhancement and Deblurring: HE, Dark Channel Prior, LIME , HDRNet, Gamma Correction
- Intrinsic Image Decomposition: Intrinsic in the Wild, Self-Supervised Intrinsic Decomposition
- Image-to-Image Transformation: Dual-Domain (Cycle-GAN, Conditional-GAN), Multi-Domain(Star-GAN)
- Photorealistic Style Transfer: Deep Image Analog, Deep Semantic Style, and MPSE
- Data-Driven Mixed Enhancement: DPED, DPED+, WESPE

02 Augmented Dataset

Synthesized Data vs. Real-world Data

- Image Super-Resolution: Deterministic Blur Kernel (Bicubic, Bilinear and etc.)
- Image Denoising and Restoration: Gaussian White Noise, Poisson Noise
- Model trained on synthesized data performs poorly in real-world blind denoising challenge
- Denoise Demo:
  (a) RedNet trained on synthesized data, generated Gaussian white noise
  (b) RedNet trained on real-world data, obtained by multi-frame denoise
- DND Denoising Benchmark: Approximate the Zero Noise ground-truth

(a) (b) (c)

Sampling Paired Data

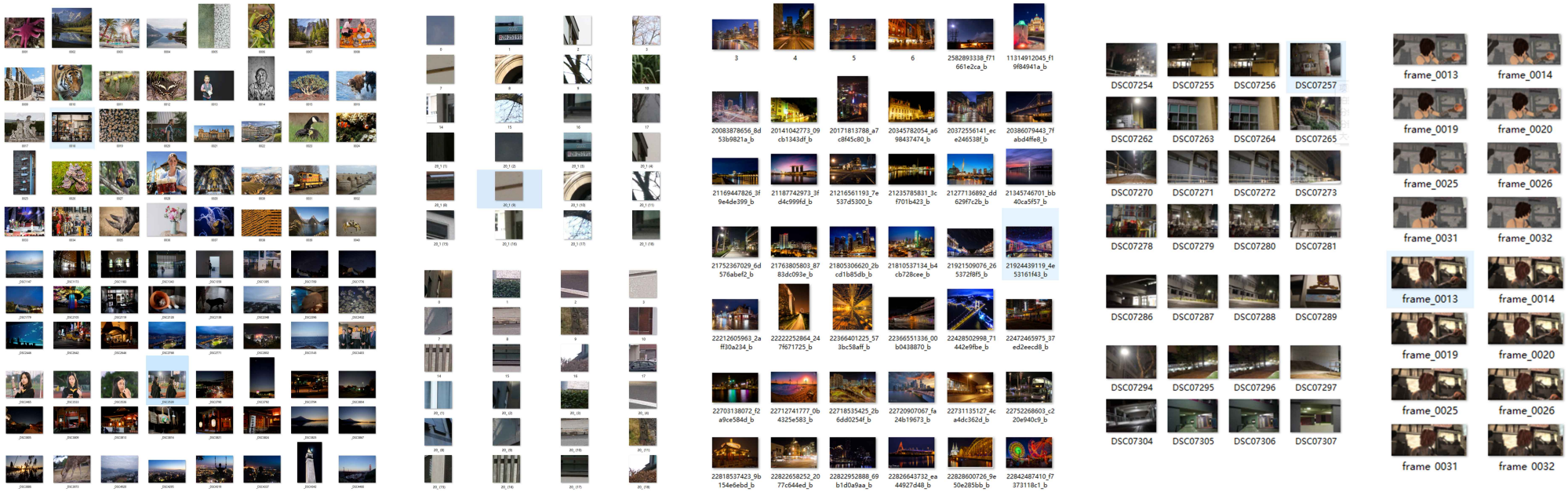
- Pro Camera
  - SONY A7,
  - Full-Frame Sensor with HDR
  - (6 stops of dynamic range)
  - 55 mm lens, 2x zooming relative to iPhone
  - (29 mm equivalent)
  - 24 mega pixels, nearly no noise

- Mobile Phone
  - iPhone 6s, normal capture mode, no HDR
  - 29 mm lens, limit to its sensor and lens
  - 12 mega pixel, noisy, dark, lack of detail

- Registration
  - Adopt SIFT in patch level
  - Imperfect Alignment

Our Dataset

- **DIV 2K Dataset** (with Night View Enrichment), 800 Images of 2K Resolution
- **PR Dataset**, DPED(7G) + Night View Enrichment(3G)
- **UPR Dataset**, 1500 HQ Pro Images + 300 LQ iPhone Images
- DND Denoising Benchmark Dataset, 50+ Real-Life Images with Ground-truth of Nearly Zero Noise
- MPI Sintel Dataset, Albedo and Original Image
- Adobe-5K Dataset, 5000 Raw Images with Human Retouch of 6 Experts (over 30 categories)



03 Multi-Phase Semantic Enhancement (MPSE)

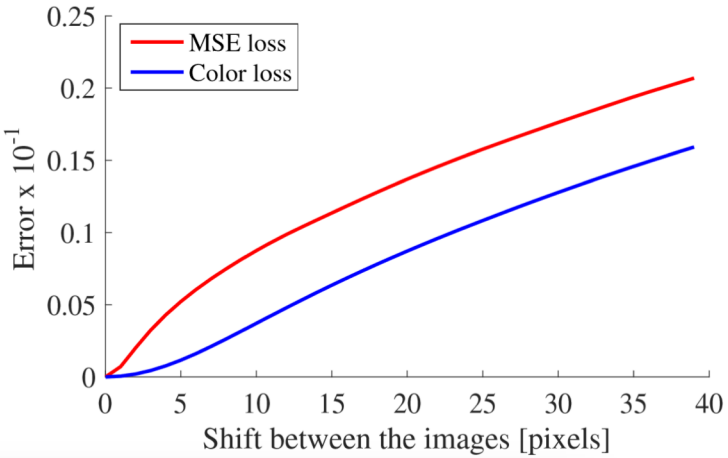
Phase 1 : Paired Supervision (DPED+)

- Baseline Method: DPED
- Enhanced Implementation: Original feature maps scanned by different Gaussian Kernels

$$\mathcal{L}_{\text{color}}(X, Y) = \|X_b - Y_b\|_2^2,$$

$$X_b(i, j) = \sum_{k,l} X(i + k, j + l) \cdot G(k, l),$$

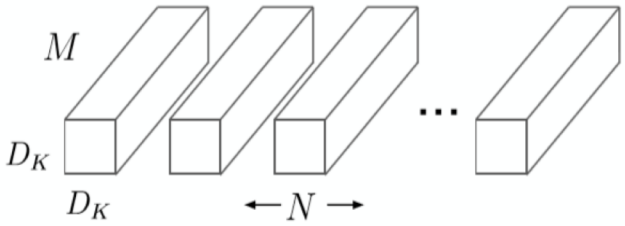
$$G(k, l) = A \exp\left(-\frac{(k - \mu_x)^2}{2\sigma_x} - \frac{(l - \mu_y)^2}{2\sigma_y}\right)$$



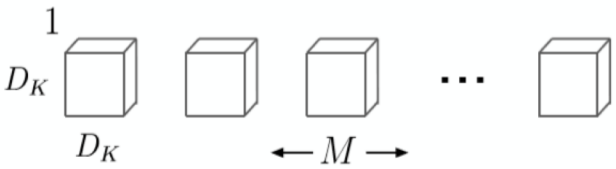

```python
def gauss_kernel(kernlen=21, nsig=3, channels=1):
    interval = (2*nsig+1.)/(kernlen)
    x = np.linspace(-nsig-interval/2., nsig+interval/2., kernlen+1)
    kern1d = np.diff(st.norm.cdf(x))
    kernel_raw = np.sqrt(np.outer(kern1d, kern1d))
    kernel = kernel_raw/kernel_raw.sum()
    out_filter = np.array(kernel, dtype = np.float32)
    out_filter = out_filter.reshape((kernlen, kernlen, 1, 1))
    out_filter = np.repeat(out_filter, channels, axis = 2)
    return out_filter

def blur(x):
    kernel_var = gauss_kernel(21, 3, 3)
    return tf.nn.depthwise_conv2d(x, kernel_var, [1, 1, 1, 1], padding='SAME')
```

Phase 1 : Paired Supervision (DPED+)

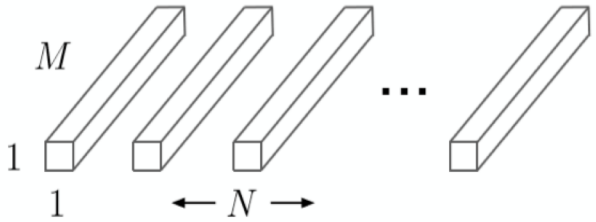
- Baseline Method: DPED
- Enhanced Implementation: Revised MobileNet Structure for better convergence



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

load_dataset.py | train_model.py | test_model.py | models.py

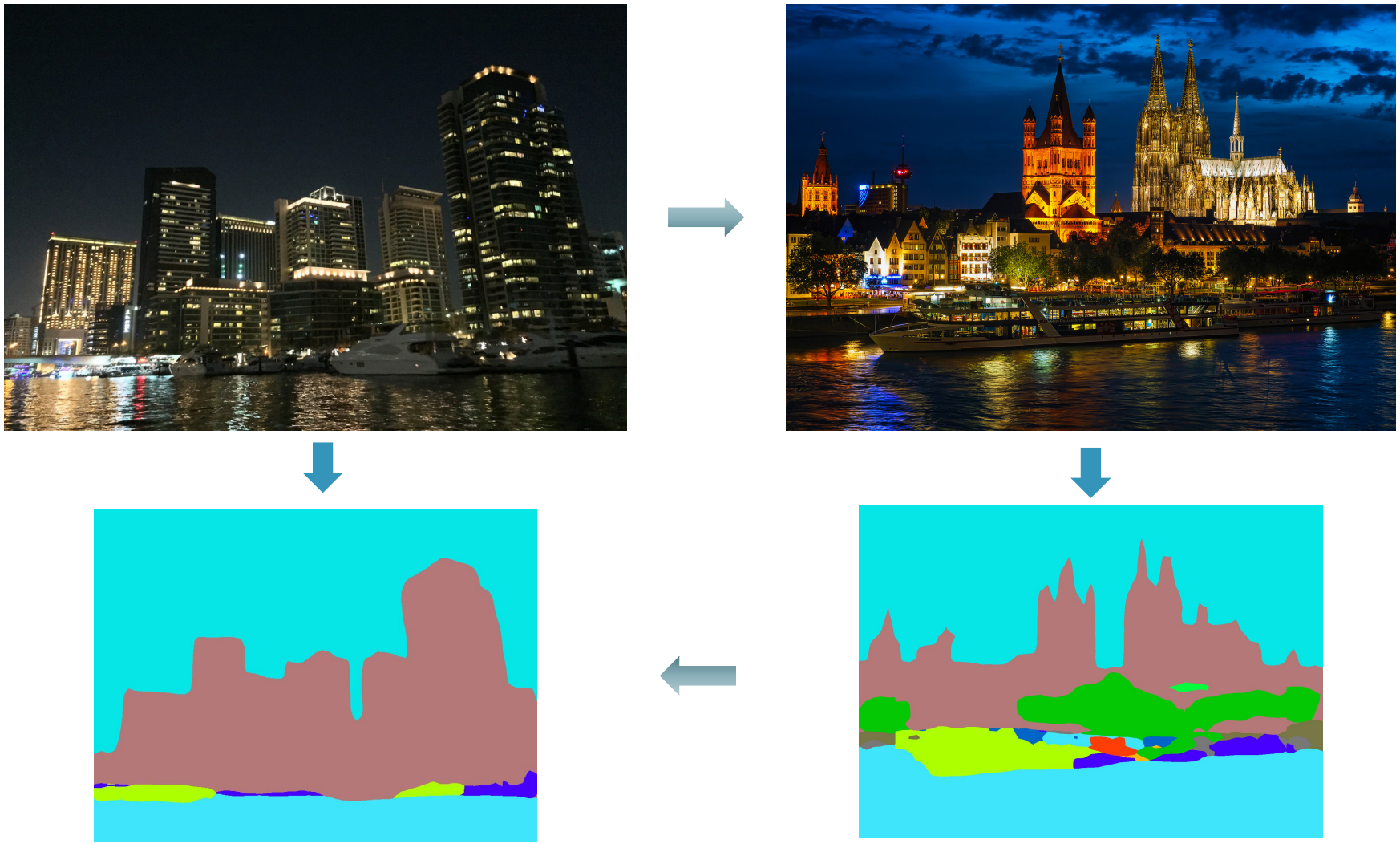
```python
import tensorflow as tf

def lrelu(x, alpha):
    alpha = 0.005
    return tf.nn.relu(x) - alpha * tf.nn.relu(-x)

def resnet(input_image):

    with tf.variable_scope("generator"):

        W1 = weight_variable([9, 9, 3, 64], name="W1"); b1 = bias_variable([64], name="b1");
        c1 = lrelu(conv2d(input_image, W1) + b1)

        # residual 1

        W2 = weight_variable([3, 3, 64, 64], name="W2"); b2 = bias_variable([64], name="b2");
        c2 = lrelu(_instance_norm(conv2d(c1, W2) + b2))

        W3 = weight_variable([3, 3, 64, 64], name="W3"); b3 = bias_variable([64], name="b3");
        c3 = lrelu(_instance_norm(conv2d(c2, W3) + b3)) + c1

        # residual 2

        W4 = weight_variable([3, 3, 64, 64], name="W4"); b4 = bias_variable([64], name="b4");
        c4 = lrelu(_instance_norm(conv2d(c3, W4) + b4))
```

- Experimental Observations:
  - 0.4dB loss (27.4132 vs 27.8025) of PSNR
  - Accelerated Training: 2855 iter/hr (>1964 iter/hr)
  - Accelerated Inference: 2.7 × speed

20000 iterations

Phase 2 : Semantic Style Transfer

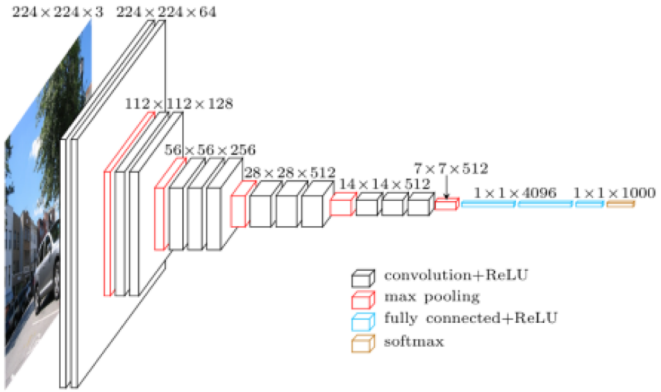
- Input: Generated Image I from DPED+, Reference Image R
- Weakly supervised manner, night-scene style tranfer network



Phase 2 : Semantic Style Transfer

- Passing both I & R into VGG-19 network and apply Gram Matrix to calculate the style loss

$$\mathcal{L}_s(O, \Lambda) = \sum_{r,l} \frac{\Lambda_r^l}{2N^{l^2}} \|G_r^l[\mathcal{O}] - G_r^l[\mathcal{R}]\|_2^2$$



- Semantically Adaptive Enhancement factor: special weights calculated by enhancement importance:

$$\Lambda_r^l = \frac{e^{\alpha_r^l H_r^l[\mathcal{O},\mathcal{R}]}}{\sum_r e^{\alpha_r^l H_r^l[\mathcal{O},\mathcal{R}]}}$$

Phase 2 : Semantic Style Transfer

- Photorealistic style transfer: A weighted combination of texture loss and content loss

$$\mathcal{L}_c(O) = \sum_l \alpha^l \frac{1}{2N^l D^l} \|F^l[\mathcal{O}] - F^l[\mathcal{I}]\|_2^2$$

$$\mathcal{L}_m(O) = \sum_{c \in r,g,b} V_c[\mathcal{O}]^T \mathcal{M}_\mathcal{I} V_c[\mathcal{O}]$$

```
313     # Content Loss
314     loss_content = content_loss(content_layer_const, content_layer_var, float(args.content_weight))
315     tf.summary.scalar('content loss', loss_content)
316
317     # Style Loss
318     layer_class_style_weights = [[float(args.r1), 1.0, 5.0, 1.0, 1.0, 1.0, 1.0, float(args.s1), 1.0], [float(args.r2), 1.0, float(args.b2), 1.0, 1.0, 1.
319                                  [float(args.r3), 1.0, float(args.b3), 1.0, 1.0, 1.0, 1.0, float(args.s3), 1.0], [float(args.r4), 1.0, float(args.b4), 1
320                                  [float(args.r5), 1.0, float(args.b5), 1.0, 1.0, 1.0, 1.0, float(args.s5), 1.0]]
321
```

Final Objective Function and Optimization Our final objective function is a weighted combination of semantic adaptive style loss $\mathcal{L}_s$, content loss $\mathcal{L}_c$ and photorealism regularization loss $\mathcal{L}_m$:

$$\mathcal{L}(\mathcal{O}, \Lambda) = \mathcal{L}_c(\mathcal{O}) + \beta \mathcal{L}_s(\mathcal{O}, \Lambda) + \gamma \mathcal{L}_m(\mathcal{O})$$

where β is the weight of the total style loss, γ is the weight to balance the photorealism regularization loss L_m.

```
# Affine Loss
if Matting:
    loss_affine = affine_loss(input_image_plus, M, args.affine_weight)
else:
    loss_affine = tf.constant(0.00001) # junk value

# Total Variational Loss
loss_tv = total_variation_loss(input_image, float(args.tv_weight))
```

Phase 2 : Semantic Style Transfer

- Semantic Enhancement Network Architecture:

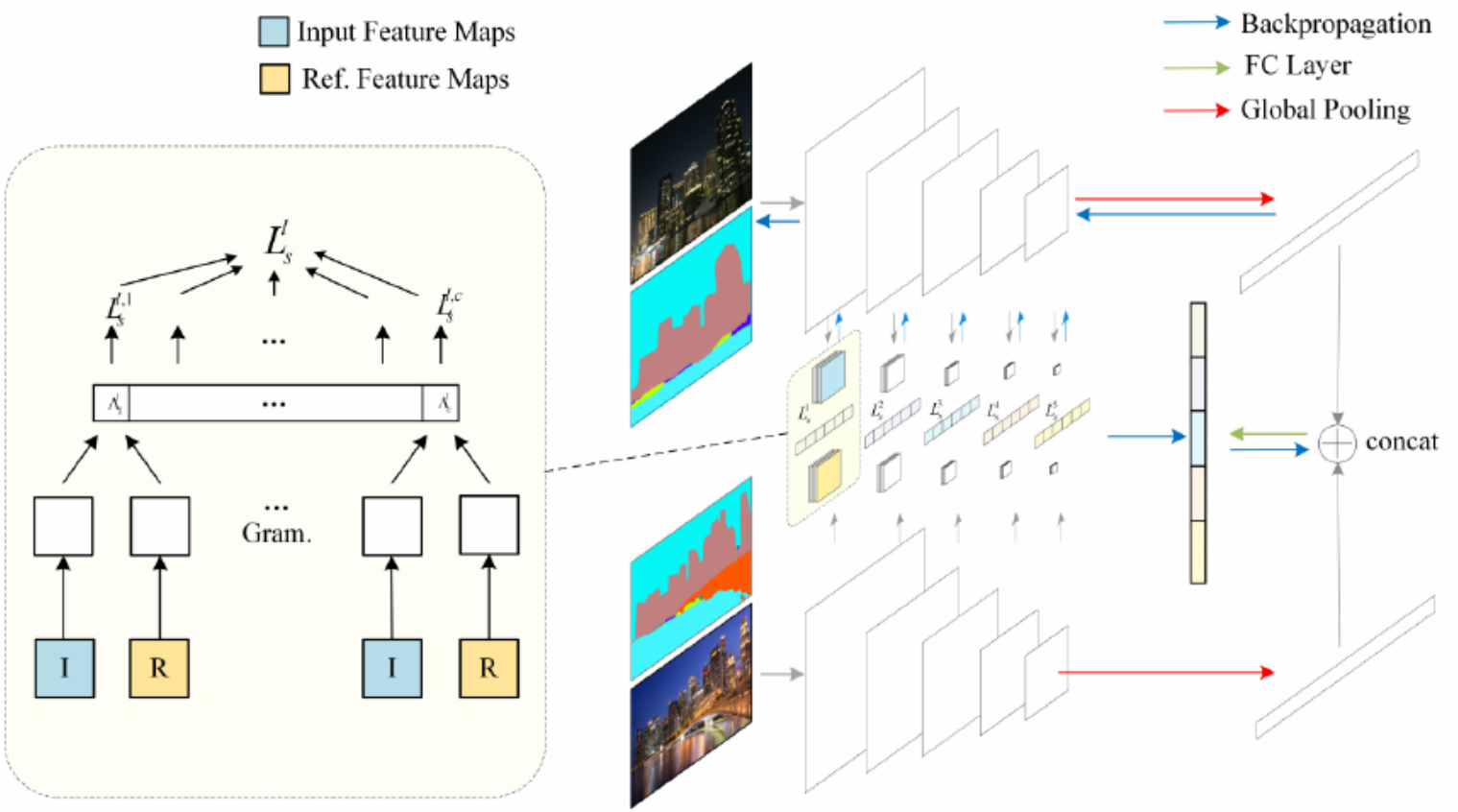


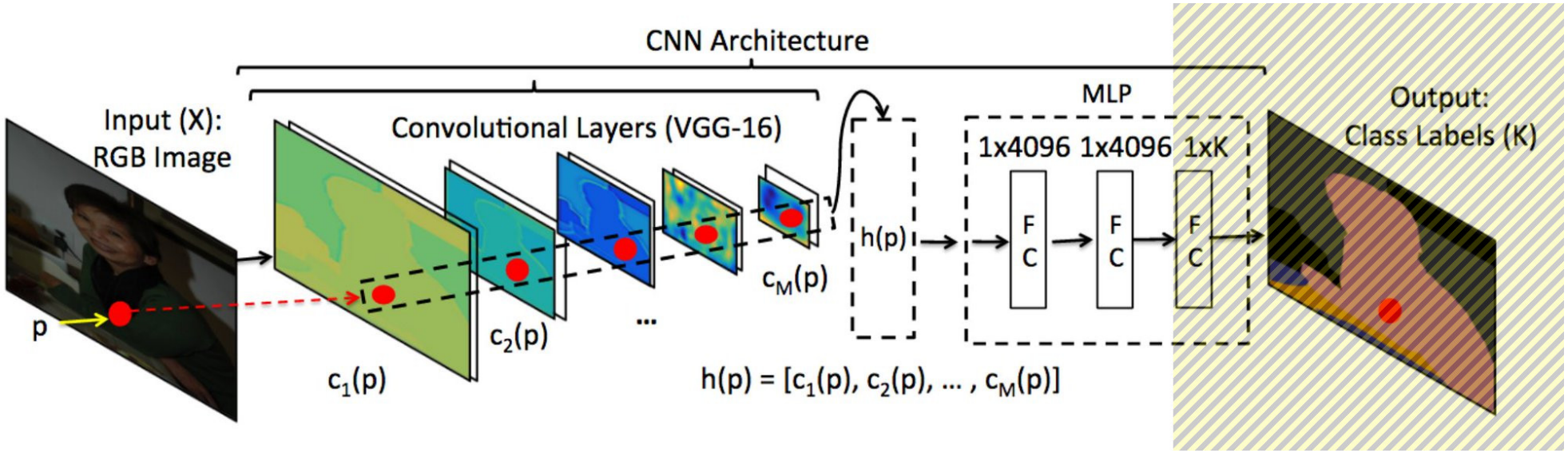
Figure 7. The network architecture of jointly enhancement factor Λ and the output image. The right side illustrates the network structure for style loss computing. We adaptively adjust our enhancement factor in regional basis and take into account the proportional information iteratively. The left side is the Gram matrix distance calculation from each region and each layer.

Phase 3 : Optimization & Post-processing

- Recommendation Network:

  FC-8 layer from the pre-trained VGG-16 network

$$D_{sem} = \|f_{fc8}(I_1 - f_{fc8}(I_1)\|_2$$



- Anchor x : minimize positive reference distances, maximize negative reference distances

$$\mathcal{L}_{ref} = \arg\min_{\mathcal{L}}[\sum_{j=1}^{10} \|f_(x_i) - f_(x_i^{p_j})\|_2 - \sum_{k=1}^{50} \|f_(x_i) - f_(x_i^{n_k})\|_2 + \alpha]_+$$

Phase 3 : Optimization & Post-processing

- Post-Processing:

  Histogram Equalization: maximize the information entropy by unify image histogram

$$T(k) = floor(L-1)\sum_{n=0}^{k} p_n$$

$$\frac{d}{dy}\left(\int_0^y p_Y(z)dz\right) = p_x(T^{-1}(y))\frac{d}{dy}(T^{-1}(y))$$

$$\left.\frac{d}{dx}\right|_{x=T^{-1}(y)} \frac{d}{dy}(T^{-1}(y)) = 1$$

- Gamma Correction:

  Human eyes-to-illuminance logarithm

  Curve

  1/2.2 gamma correction

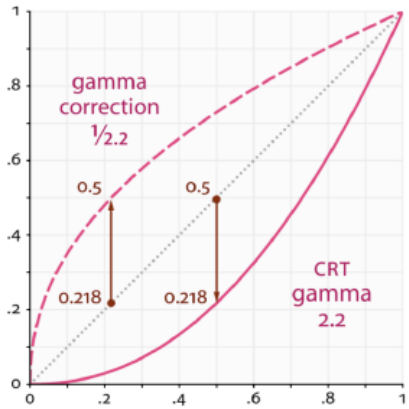
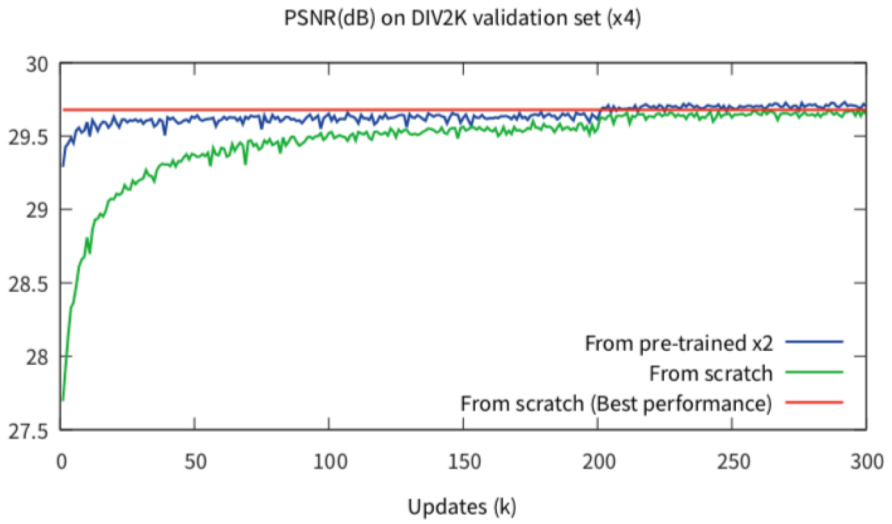


Figure 9. Comparison between responses to different light intensity of human eye (Gamma 1/2.2) and commonly-used cameras (Linear).

Experiment : Statistics about Training Details

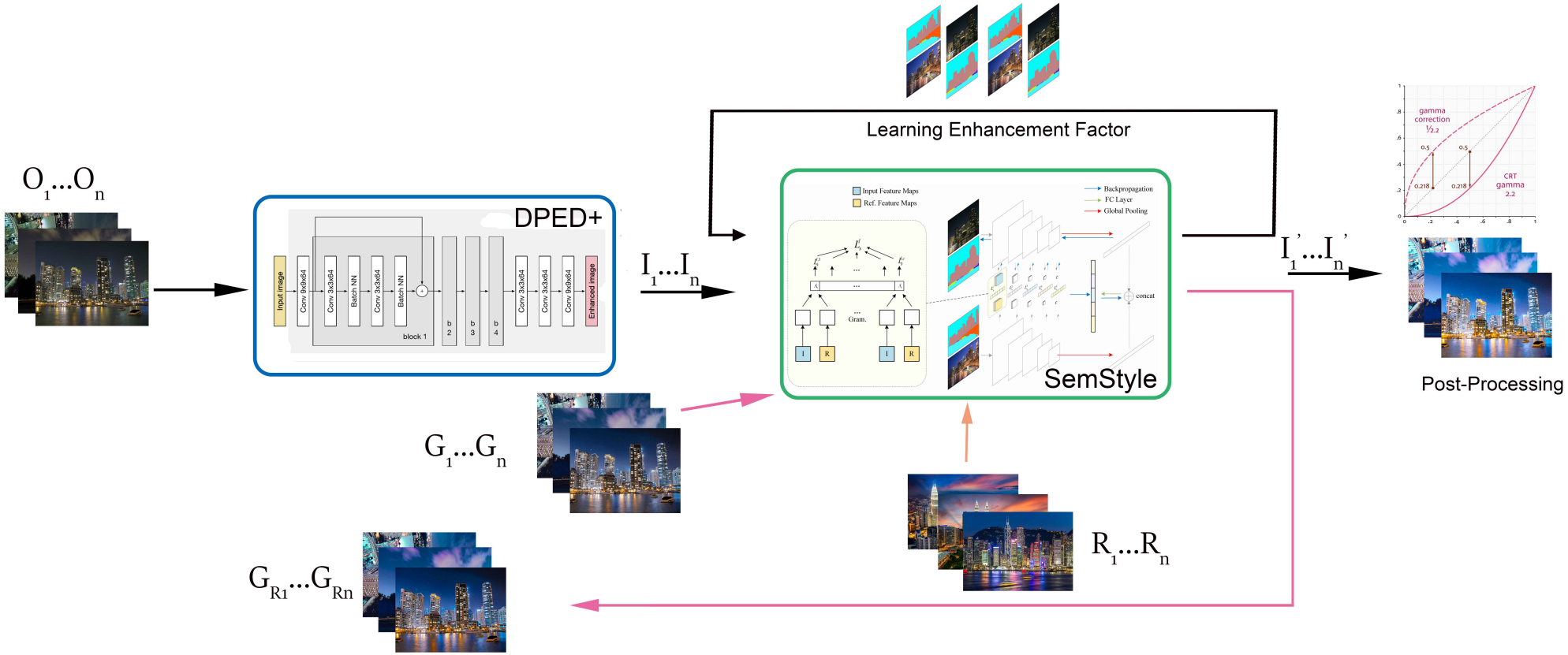
- Experimental Details

Quantitative Measurement(2×10^4 and 7×10^4 iters)				
Revised Models	PSNR	SSIM	PSNR	SSIM
Baseline	26.3455	0.9112	27.8025	0.9338
VGG Loss(VL)	27.4288	0.9493	28.8042	0.9703
$l1 + l2$	26.2453	0.8954	26.8251	0.9122
MobileNet	25.2410	0.9321	27.4123	0.9416
LReLU(LR)	26.5448	0.9107	27.7239	0.9293
Contrast(HE)	27.0145	0.9220	28.0112	0.9445
VL+LR+HE	27.4338	0.9420	29.2002	0.9754




```
#Placeholder for image inputs
self.input = x = tf.placeholder(tf.float32,[None,img_size,img_size,output_channels])
#Placeholder for upscaled image ground-truth
self.target = y = tf.placeholder(tf.float32,[None,img_size*scale,img_size*scale,output_channels])

"""
Preprocessing as mentioned in the paper, by subtracting the mean
However, the subtract the mean of the entire dataset they use. As of
now, I am subtracting the mean of each batch
"""
mean_x = tf.reduce_mean(self.input)
image_input =x- mean_x
mean_y = tf.reduce_mean(self.target)
image_target =y- mean_y

#One convolution before res blocks and to convert to required feature depth
x = slim.conv2d(image_input,feature_size,[3,3])

#Store the output of the first convolution to add later
conv_1 = x

scaling_factor = 0.1

#Add the residual blocks to the model
for i in range(num_layers):
    x = utils.resBlock(x,feature_size,scale=scaling_factor)

#One more convolution, and then we add the output of our first conv layer
x = slim.conv2d(x,feature_size,[3,3])
x += conv_1

#Upsample output of the convolution
x = utils.upsample(x,scale,feature_size,None)

#One final convolution on the upsampling output
output =x# slim.conv2d(x,output_channels,[3,3])
self.out = tf.clip_by_value(output+mean_x,0.0,255.0)

self.loss = loss = tf.reduce_mean(tf.losses.absolute_difference(image_target,output))
```

Entire Pipeline: 3 Phases of MPSE



- Phase 1: PR Learning DPED+ [detail & texture augmentation]
- Phase 2: UPR Learning [semantic style transfer, structural enhancement]
- Phase 3: Post-processing [global contrast, illuminance map adjustment]

MPSE Entire Pipeline: Result

04 Multi-Domain Intrinsic Enhancement (MDIE)

LIME

- The task of low-light enhancement can be viewed as ‘finding a better illuminance map’

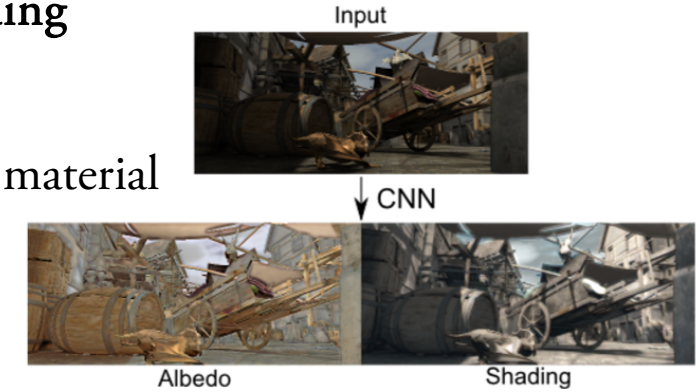
$$\mathbf{L} = \mathbf{I} \circ \mathbf{T},$$

- $\mathbf{L}$ is the original input and $\mathbf{I}$ is the desired recovery, $\mathbf{T}$ is the illuminance map
- Given the same original image, different estimated illuminance map results diverting enhancement results
- The wanted illuminance map, smooth, local-consistent and texture-free
- The wanted illuminance map should exploit the maximum potential of the original image

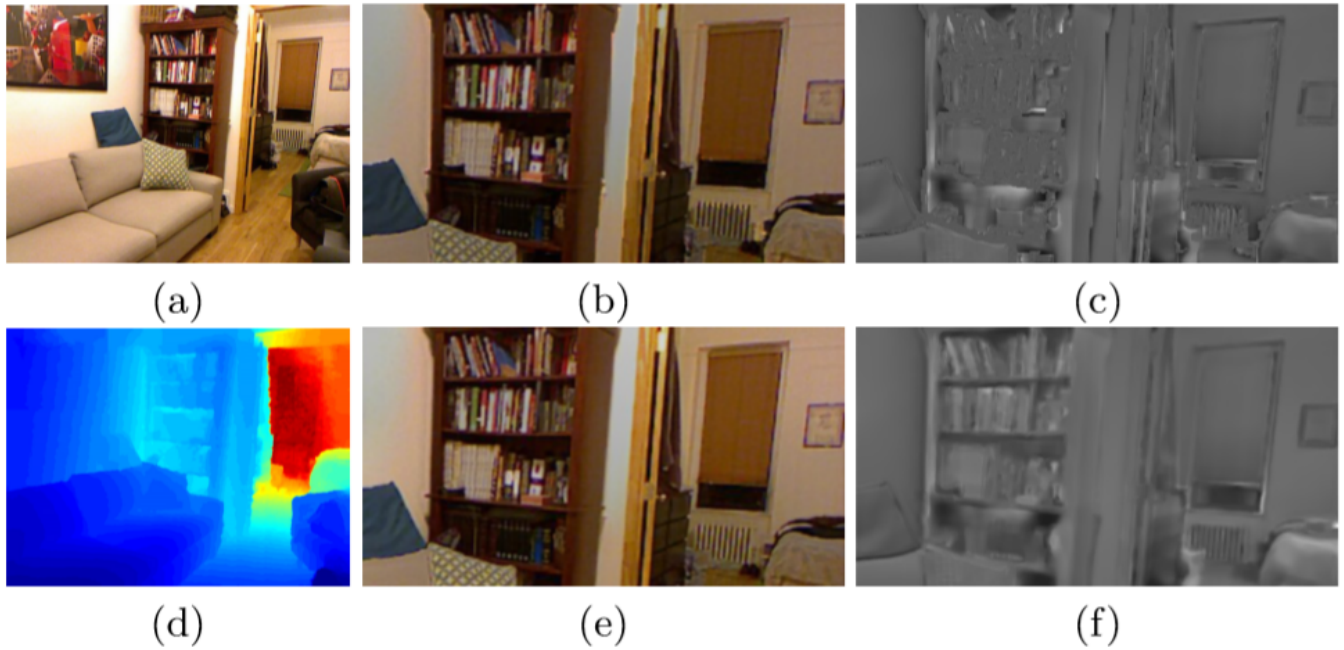
Intrinsic Image Decomposition

- Decompose the Image into **Albedo** (reflectance) and **Shading**

$$\mathcal{I} = \mathcal{A} \cdot \mathcal{S}$$



- **Shading** gives important cues on the object geometry and material
- **Albedo**(reflectance) is a lighting invariant quantity
- Albedo is similar to the previous illuminance map
- Lighting-texture-structure separation can be possible (supervision)

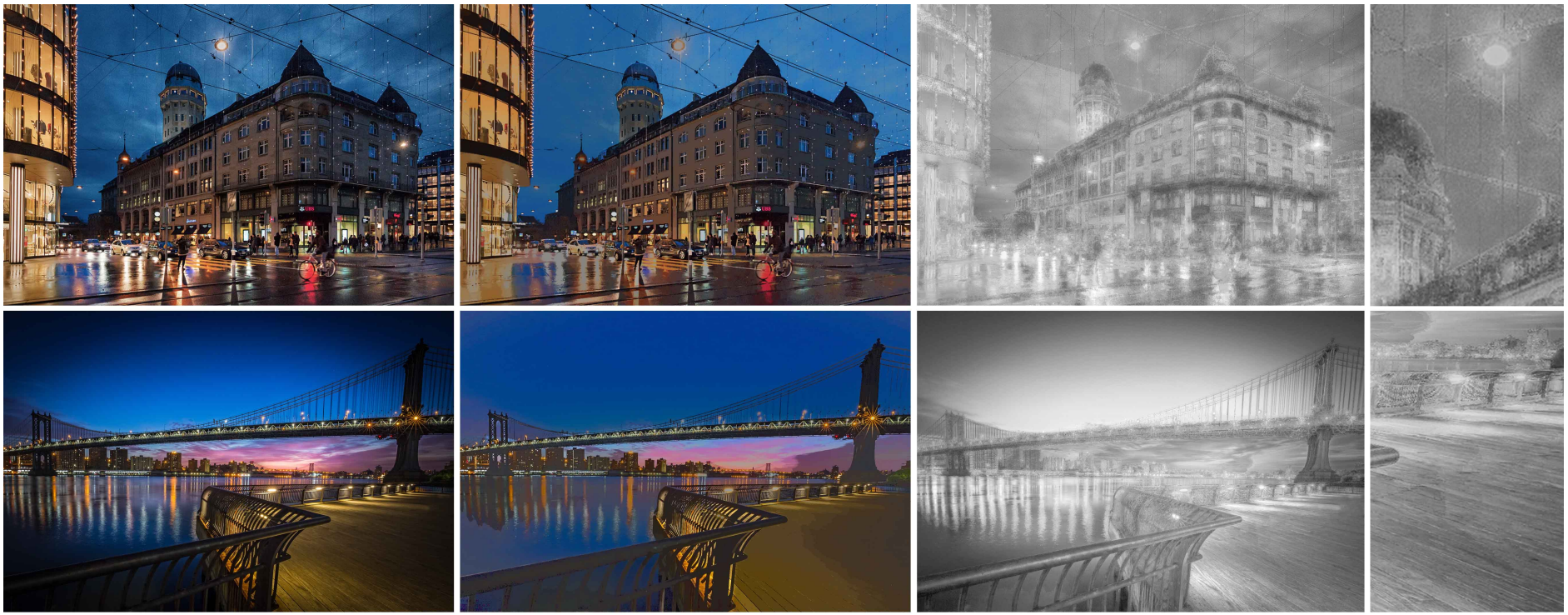


Observation on Real-World Intrinsic

- In daylight condition, the ambient light is relatively abundant:

  DSLR-Mobile is mainly **Shading-different**
- In night-scene, with poor lighting condition:

  DSLR-Mobile is mainly **Illuminance-different** (minor **Shading-different**)
- Structure information mainly exists in **Reflectance**
- Texture information mainly exists in **Shading**
- Noise pattern mainly exists in **Shading**



(a) (b) (c) (d)

Self-Supervised Intrinsic Decomposition

Problem and Motivation:

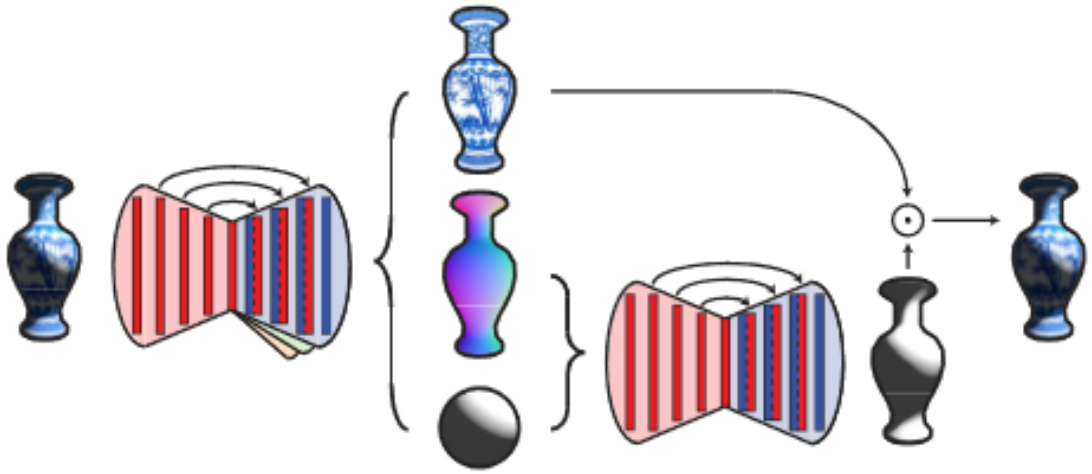
- Real-world intrinsic dataset is rare, manually labelling is extremely expensive
- CG-based dataset (e.g. MPI Sintel) performs disappointingly on real-world task

Solutions:

- Symmetric network structure, two encoder-decoders
- Reconstruction loss, similar to cycle loss

$$\|\mathcal{I} - \mathcal{A} \cdot \mathcal{S}\|_2^2 \qquad \mathcal{A} = \mathcal{I}, \mathcal{S} = 11^T \qquad \mathcal{L}_{rec} = \|\mathcal{I} - \mathcal{F}'(\mathcal{A}, \mathcal{S})\|^2$$

- Pretrain the first encoder-decoder on synthesized dataset
- Train the entire network with reconstruction loss on real-world unlabeled dataset



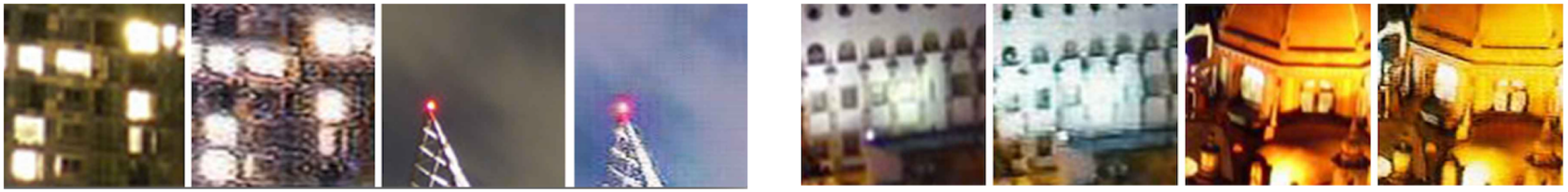
Multi-Domains Translation: Data-Driven Initiatives

Absence of Overall Enhancement Dataset:

- Alignment Issue: Overall DSLR-Mobile Dataset is still absent
- Partial enhancement datasets largely exist (denoise, color, contrast, texture, resolution)
- Denoise (DND), Color and Contrast (Adobe 5K), Resolution (DIV 2K)

Unclear Training Directions:

- The actual contributing factors behind 'high-quality' is hidden
- In overall enhancement training, noise and texture is sometimes indistinguishable



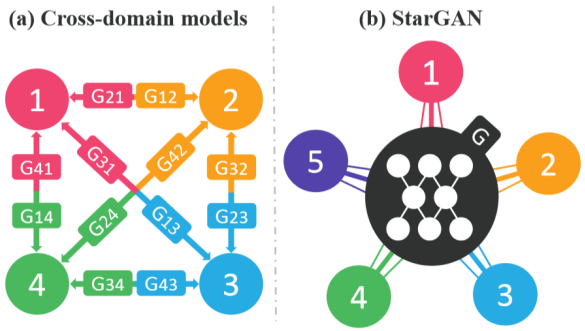
Solution:

- Three datasets dedicated for denoise, color & contrast, texture
- Multi-domain image translation GAN, even if the overall enhancement dataset is lacked

Star-GAN: Unified Solution for Multi-Domains

Previous Image-to-Image Translation Network:

- General solutions require training N(N-1)/2 dual-domain image translation network
- Previous solution cannot translate to a domain with no paired data



Star-GAN

- Only one universal generator is needed

- **Attribute**: enhancement factor inherent in an image such as color, texture, or noise level.
- **Attribute Value**: a particular level of the attribute, such as low-quality or high-quality.
- **Domain**: a set of images sharing the same attribute value. E.g., images of low-quality in color perspective.

Our Goal:

- Train G to translate an input image x into an output image y conditioned on the target domain label c

$$G(x, c) \rightarrow y.$$

- Train D to produce probability distributions over both sources and domain labels

$$D : x \rightarrow \{D_{src}(x), D_{cls}(x)\}.$$

Network Design

Masked Vector:

- We use an n-dimensional **one-hot vector** to denotes attribute value vector ***m***
- If here we input the i-th dataset, then only c_i is assigned to 1 and others is assigned to 0
- We have three datasets dedicated to color, texture, and noise. Then the n is set to be 3.
- In each dataset, we only have two attribute value: low-quality(0) and high quality(1).

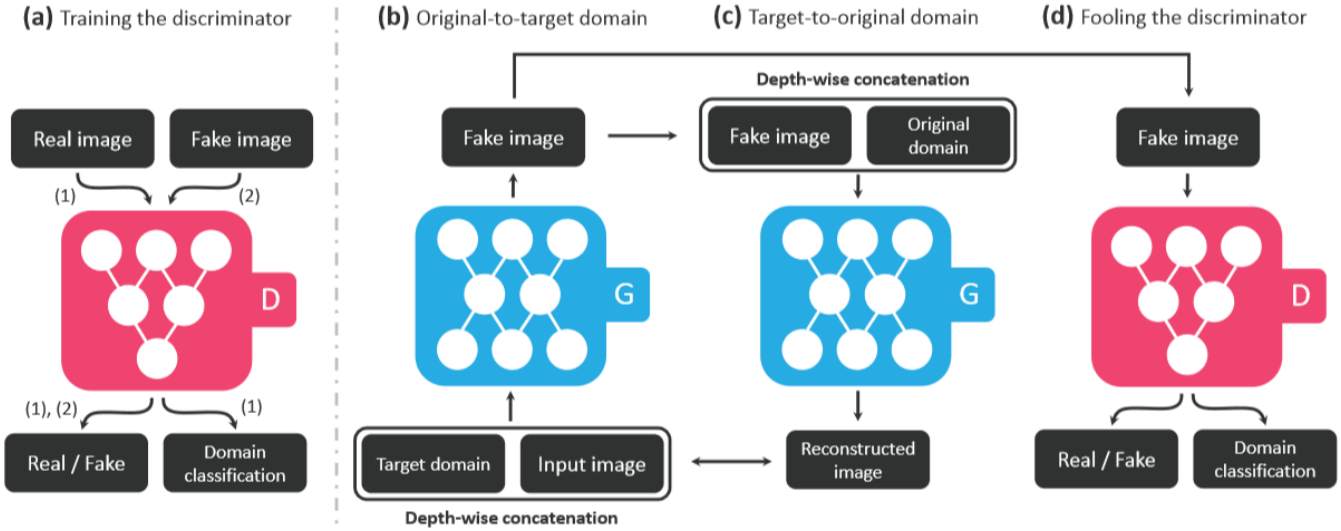
$$\widetilde{c} = [c_1, ., c_i, ., c_n, m]$$

Domain Classification Loss:

- An auxiliary classifier is added on top of the discriminator D to impose domain classification

$$\mathcal{L}_{cls}^{r} = \mathbb{E}_{x,c'}[-logD_{cls}(c'|x)]$$

- D_{cls} (c′|x) denotes a probability distribution over domain labels calculated by D.
- D learns to classify a real image x to its corresponding original domain c'



Network Design

Cycle Construction Loss:

- G takes the translated image G(x, c) and the domain label of the original input c′ as input and attempts to reconstruct the original image x.

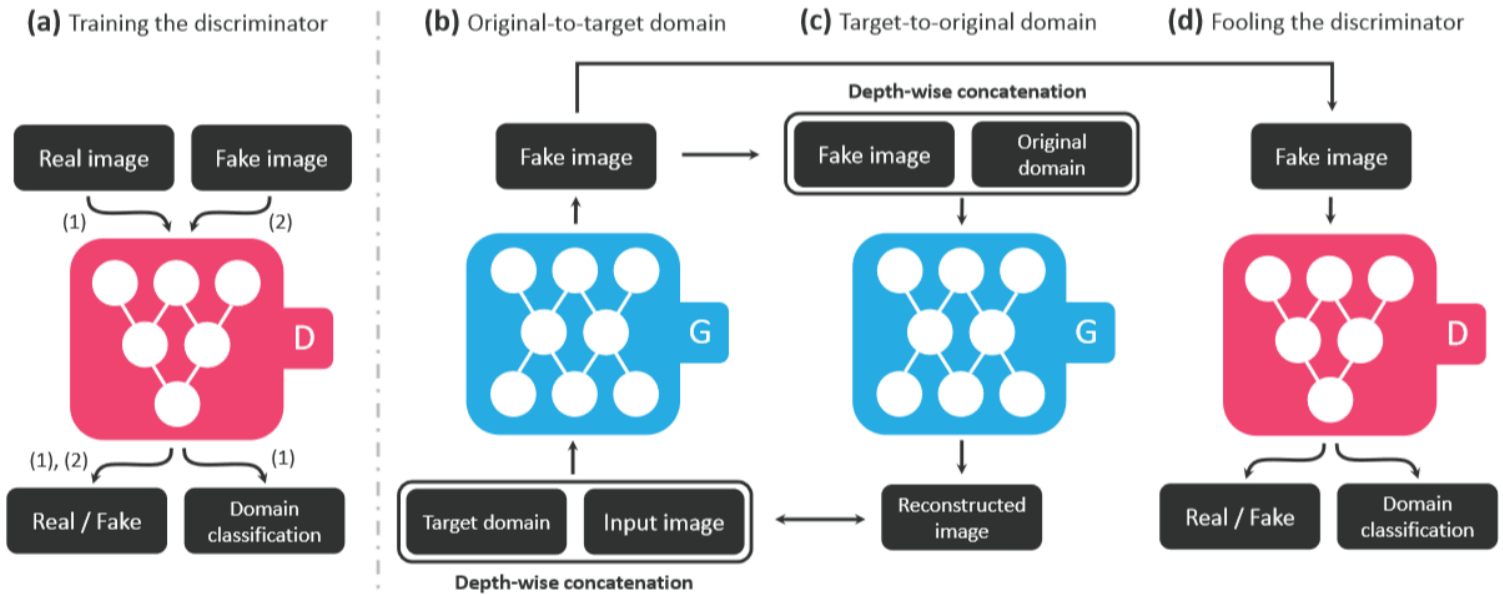
$$\mathcal{L}_{cyc} = \mathbb{E}_{x,c,c'}[\|x - G(G(x,c),c')\|]_1$$

Total Objective Function

- λ_{cls} and λ_{cyc} are hyper-parameters that balance the relative weight of domain classification loss and cycle re- construction loss.

$$\mathcal{L}_D = -\mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^r_{cls}$$

$$\mathcal{L}_G = \mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^f_{cls} + \lambda_{cyc}\mathcal{L}_{cyc}$$



Training Strategy

Our Previous Observations:

- Color and contrast information exist both in **Reflectance** and **Shading**
- Texture information mainly exists in **Shading**
- Noise pattern mainly exists in **Shading**

Three Partially Enhanced Datasets:

- DIV2K: resolution and texture enhancement (train on **Shading**)
- Adobe-5K: color and contrast enhancement (train on **Reflectance** and **Shading**)
- DND Denoise Benchmark Dataset: noise reduction (train on **Shading**)

Inference Setting:

- We Set all c_i to 1 with the m value of 1, indicating the translation from one LQ domain with all 'bad factors' to the HQ domain with all 'good' factors.

$$\widetilde{c} = [c_1, ., c_i, ., c_n, m]$$

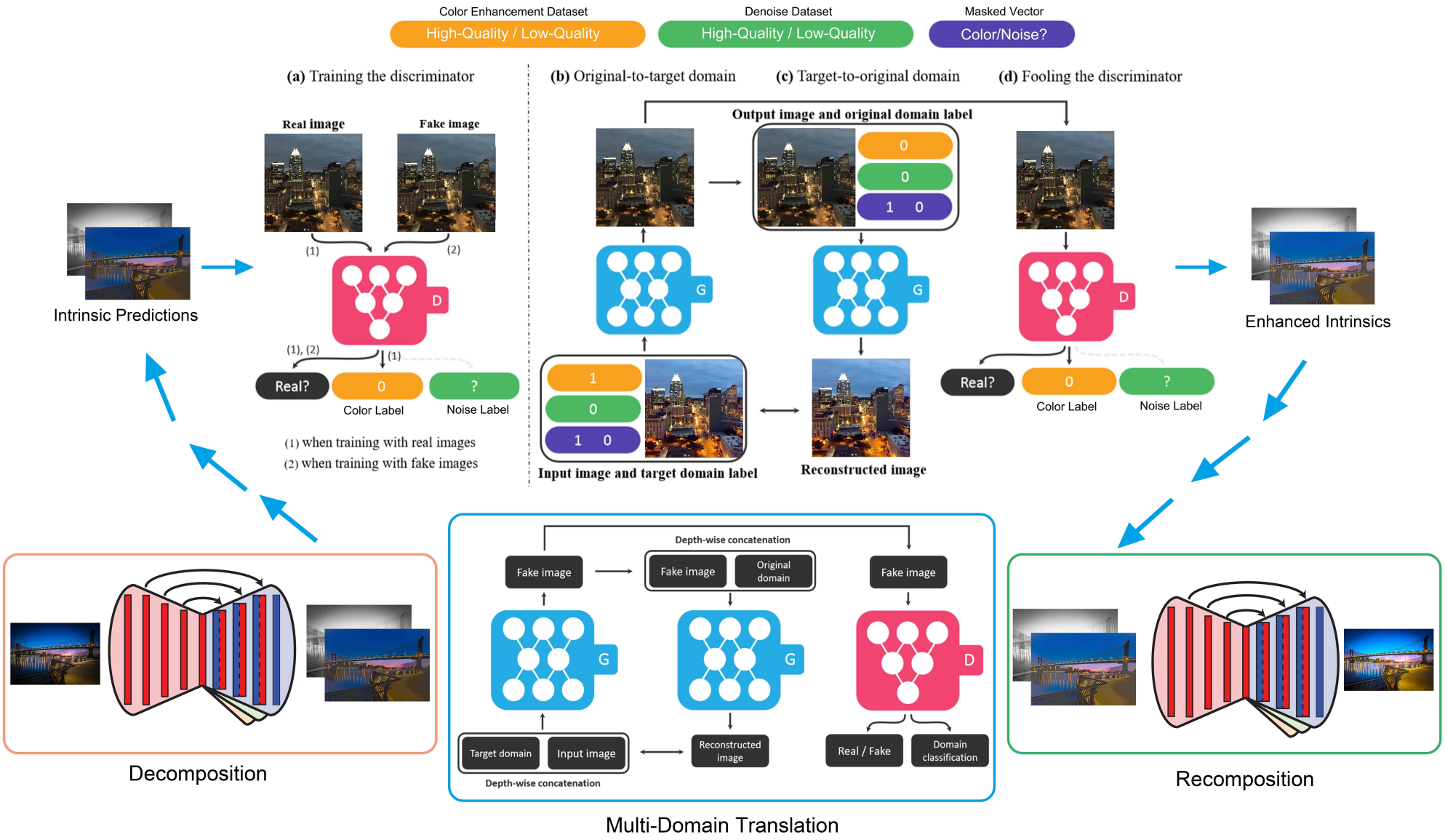
Total Objective:

$$\mathcal{L}_D = -\mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^r_{cls}$$

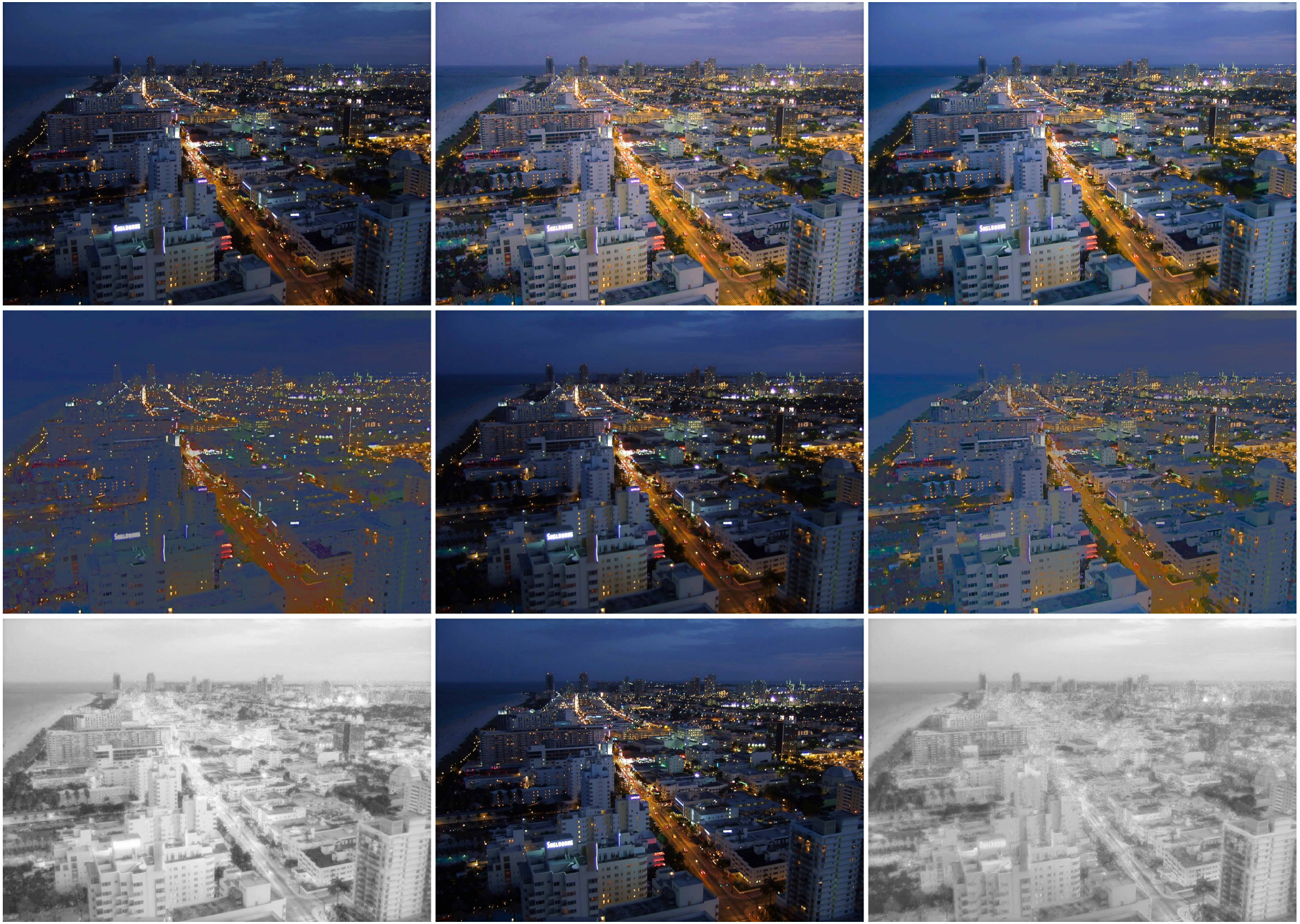
$$\mathcal{L}_G = \mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^f_{cls} + \lambda_{cyc}\mathcal{L}_{cyc}$$

$$\mathcal{L}_{rec} = \alpha\|\mathcal{I}_{LQ} - \mathcal{F}'(\mathcal{A}_{LQ}, \mathcal{S}_{LQ})\|^2 + \beta\|\mathcal{I}_{HQ} - \mathcal{F}'(\mathcal{A}_{HQ}, \mathcal{S}_{HQ})\|^2$$

MDIE: Decompose-Enhance-Reconstruct



MDIE Experiment



05 Future Works

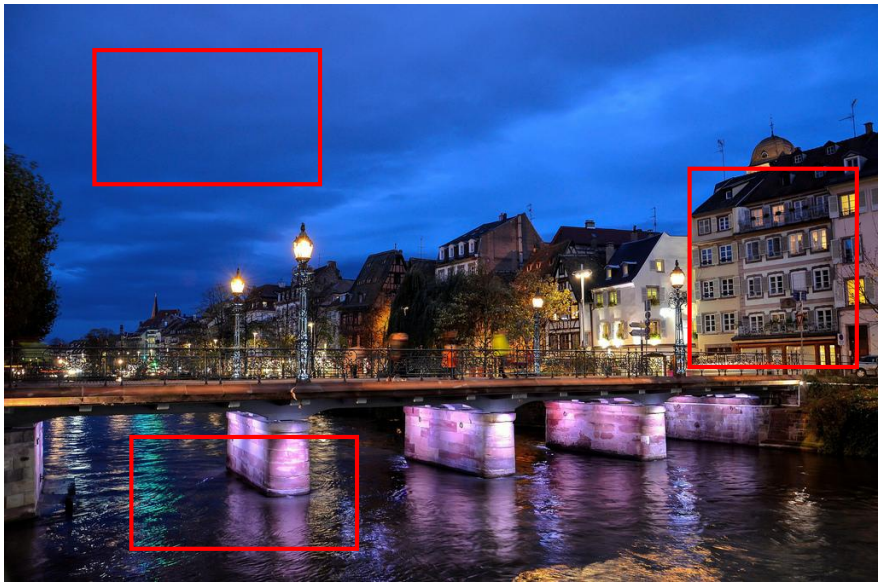
Future Works - MPSE

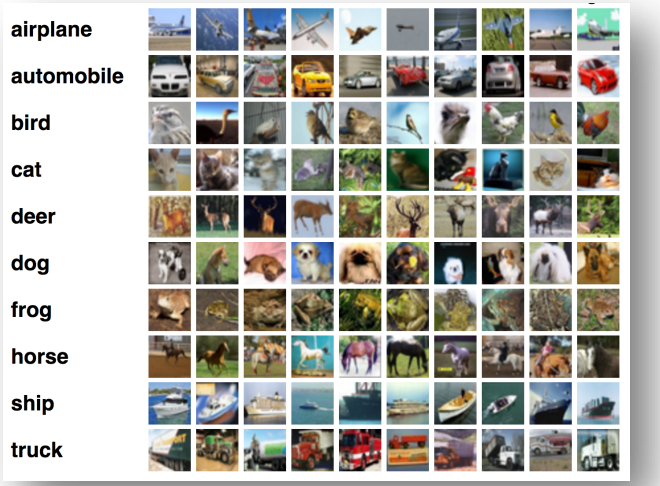
- **Augmented Semantic Regions**: current three main regions ~ building, river, sky

```
        loss_style = 0.0
        for loss in loss_styles_list:
            loss_style += loss
        tf.summary.scalar('style loss', loss_style)
        tf.summary.scalar('building style loss', object_loss_style_list[2])      # 0 river 1 tree 2 building 6 road 7 sky
        tf.summary.scalar('sky style loss', object_loss_style_list[7])
        tf.summary.scalar('river style loss', object_loss_style_list[0])
        tf.summary.scalar('building l2 loss', object_loss_l2_list[2])  # 0 river 1 tree 2 building 6 road 7 sky
        tf.summary.scalar('sky l2 loss', object_loss_l2_list[7])
        tf.summary.scalar('river l2 loss', object_loss_l2_list[0])

        input_image_plus = tf.squeeze(input_image + mean_pixel, [0])
```

- **Semantic Enhancement Factor**: Weight Selection

Enhancement Importance: Softmax, Texture Calculation, Total Variation





Future Works - MDIE

Texture Enrichment:

- Not limited to resolution, but using texture dataset for training

Joint Training of the Intrinsic and Translation Network:

- Pretrain the first encoder-decoder using MPI Sintel
- Input three partially enhanced datasets, joint training and balancing the two networks
- The intrinsic network should learn real-life decomposition
- The translation network should learn the partial enhancement for a overall enhancement

$$\mathcal{L}_D = -\mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^r_{cls}$$

$$\mathcal{L}_G = \mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}^f_{cls} + \lambda_{cyc}\mathcal{L}_{cyc}$$

$$\mathcal{L}_{rec} = \alpha\|\mathcal{I}_{LQ} - \mathcal{F}'(\mathcal{A}_{LQ}, \mathcal{S}_{LQ})\|^2 + \beta\|\mathcal{I}_{HQ} - \mathcal{F}'(\mathcal{A}_{HQ}, \mathcal{S}_{HQ})\|^2$$

Minimizing the Separation Loss:

- Using more sophisticated network for intrinsic decomposition
- Purify the separation of albedo and shading in real world data

Improve Mobile Phone's Image Quality by Deep Learning

· Introduction & Review ·

· Augmented Dataset ·

· MPSE(Multi-Phase Semantic Enhancement) ·

· MDIE(Multi-Domain Intrinsic Enhancement) ·

· Future Work·

· Q&A·

HUANG Cen | LIU Yicun