Deep learning & applications

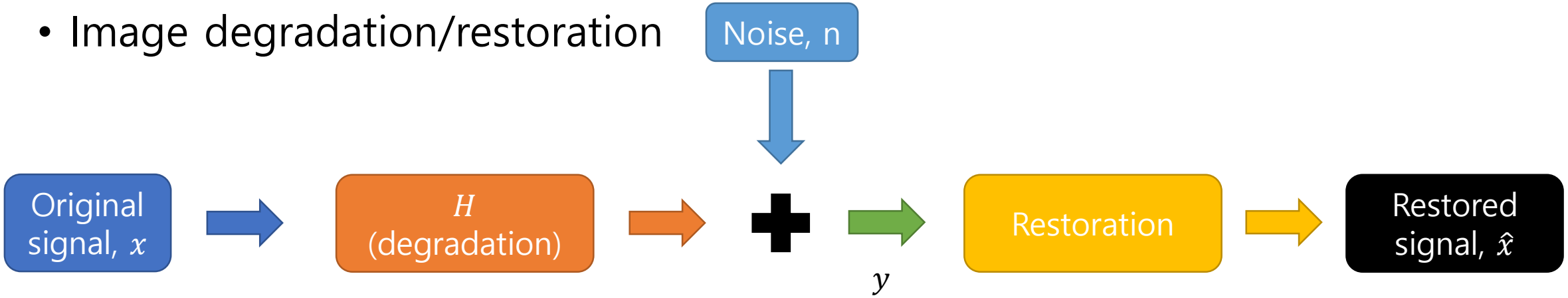
Practice#5

Tae Hyun Kim

What is Image Restoration?

What is Image Restoration?

- Image degradation/restoration



- Goal
  - Restore a degraded image (y) to its original signal (x)

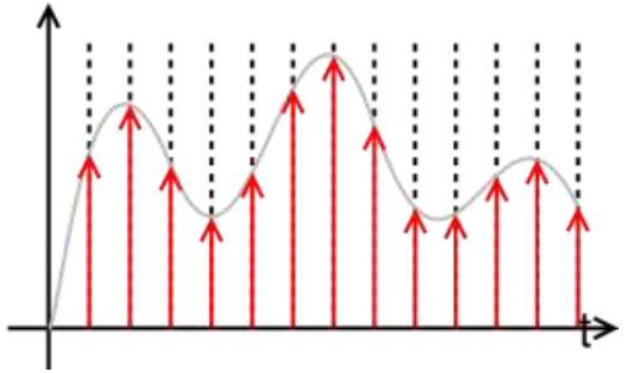
- Assumption
  - Degradation model is known or can be estimated

Common Degradations

- Image degradations
  - Resolution
  - Quantization
  - Blur
    - Motion blur
    - Focus blur
  - Noise
  - …

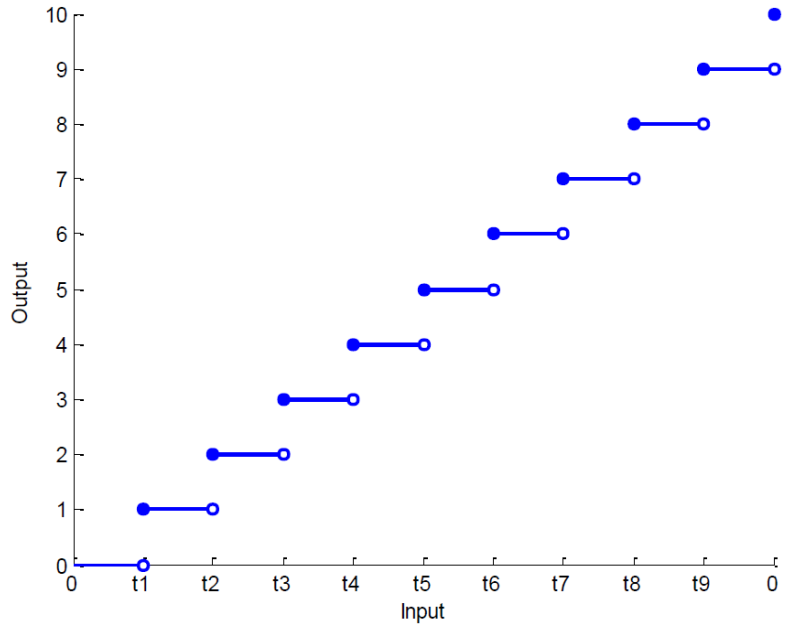
Degradation Sources: Sampling

- Sampling
  - Digitize
    - Analog to digital
  - Bicubic, Bilinear, Gaussian, Lanczos, …



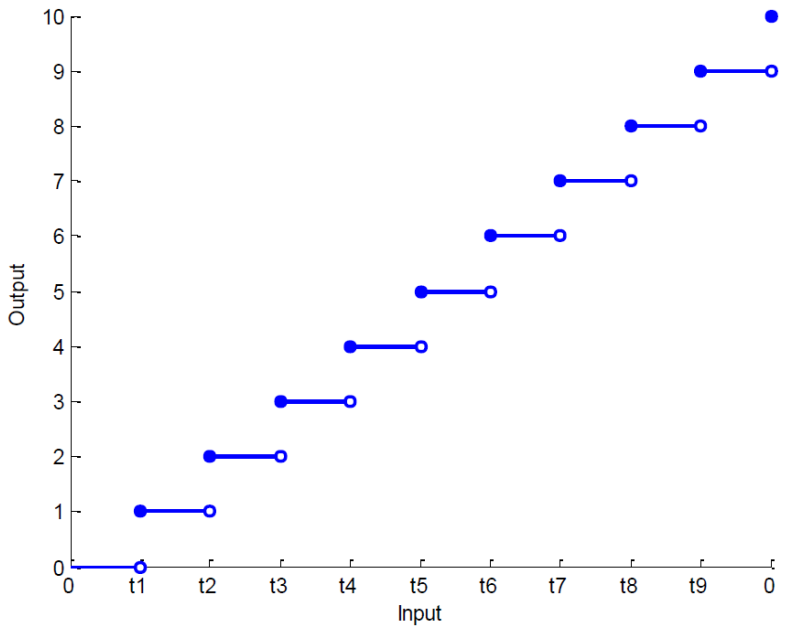
Degradation Sources: Quantization

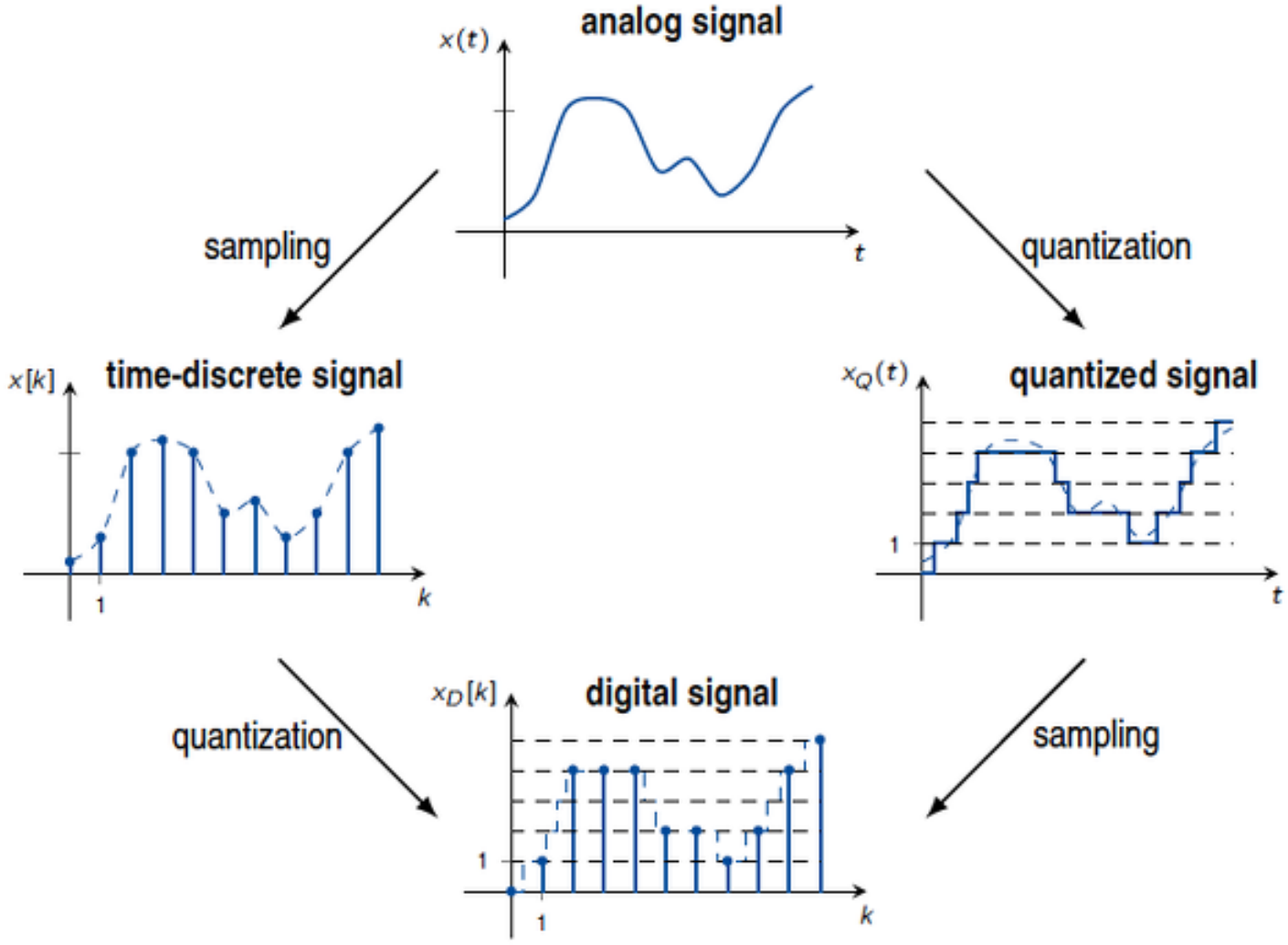
- Quantization
  - Lossy
  - Non-invertible transform



Degradation Sources: Quantization

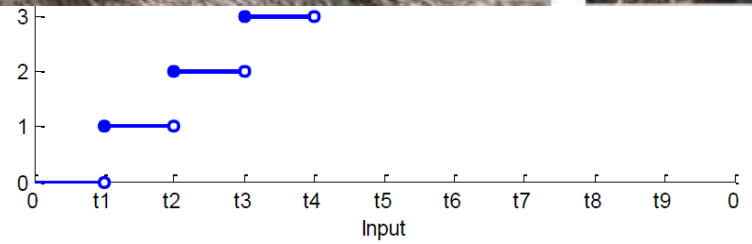
- Quantization
  - Lossy
  - Non-invertible transform

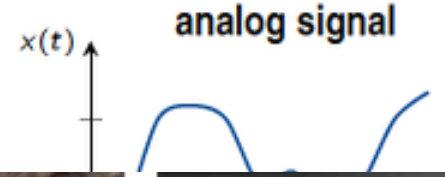


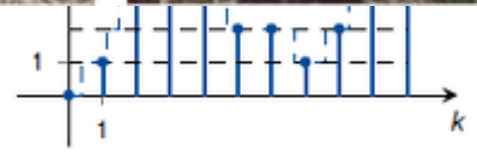


Degradation Sources: Quantization

- Quantization







Degradation Sources: Blur

- Blur
  - Image smoothing (low-pass filter)
  - Mainly caused by
    - Camera shake
    - Object motion
    - Defocus
    - Turbulence
    - …

Camera shake (Camera motion blur)

Object movement (Object motion blur)

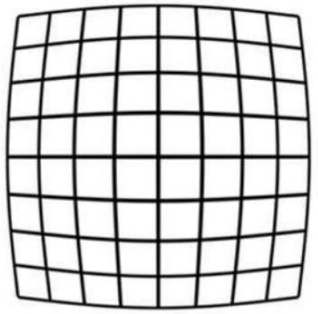
Out of focus (Defocus blur)

Combinations (vibration & motion, …)

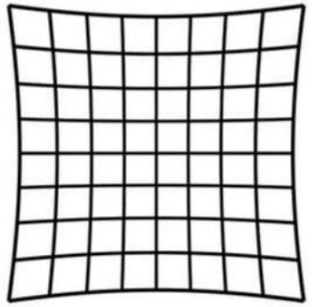
Degradation Sources: Deformation

- Spatial deformation
  - Spatial transformation
  - Invertible in most cases

  - Lenz distortion
    - Non-standard camera
      - Omni-vision
      - Fish-eye



Barrel



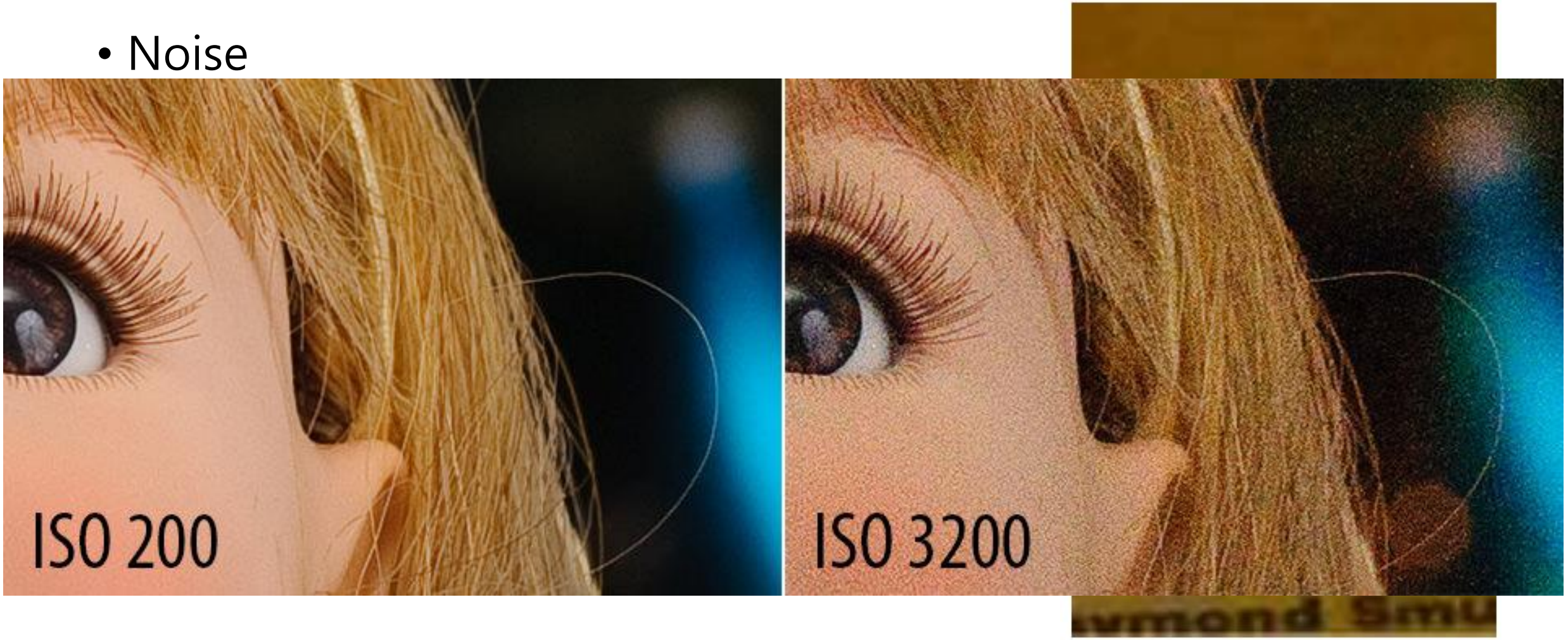
Pincushion

Degradation Sources: Noise

- Noise
  - Apparent in low-light scenes taken at high ISO
- Where it comes from
  - Background and thermal noise (additive Gaussian)
  - Shot noise (photon noise) – Poisson, approx.
  - Random errors of A/D converter, transmission errors (impulse noise)
  - Wavefront interference noise (speckle noise)

Degradation Sources: Noise

- Noise



Degradation Sources: Noise

- Noise types (distribution)

Gaussian

Impulse

Poisson

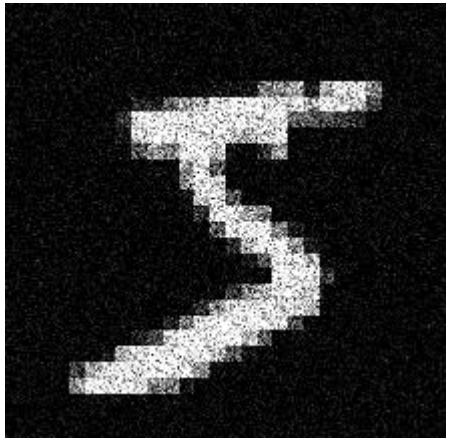
Image denoising

- Typical regression problem
  - Need to predict the output of a continuous value unlike the classification problem
    - Other example
      - https://www.tensorflow.org/tutorials/keras/regression

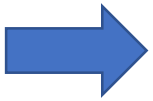
- Refernce
  - DnCNN
    - TIP2017
    - https://ieeexplore.ieee.org/document/7839189

Our task

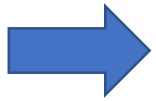
- Input
  - ~~Corrupted MNIST image~~
  - Corrupted CIFAR10 image (color)
- Output
  - ~~Clean MNIST image~~
  - Clean CIFAR10 image (color)

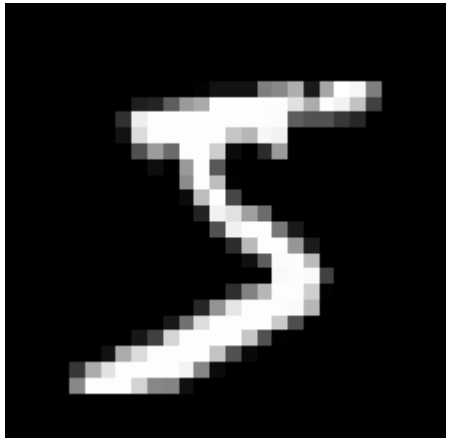


Noisy input



Denoiser





Denoised output

Network configuration

- 5-layered CNN
  - Model1
    - in → conv(3x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x3) → out
  - Model2 (w/ skip connection)
    - in → conv(3x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x64) → relu → conv(64x3x3x3) + in→ out
  - Model3 (w/ skip connection & batch normalization)
    - in → conv(3x3x3x64) → bn → relu → conv(64x3x3x64) → bn → relu → conv(64x3x3x64) → bn → relu → conv(64x3x3x64) → bn → relu → conv(64x3x3x3) + in→ out

Train/test set

- Train set
  - ~~60000 mnist train set~~
  - CIFAR-10 train set
- Test set
  - ~~10000 mnist test set~~
  - CIFAR-10 test set

```
mnist = tf.keras.datasets.mnist
(x_train, _), (x_test, _) = mnist.load_data()
(x_train, _), (x_test, _) = datasets.cifar10.load_data()

x_train, x_test = x_train / 255.0, x_test / 255.0 #scaling; pre-processing
y_train, y_test  = x_train, x_test # ground-truth data

model = tf.keras.models.Sequential([
  tf.keras.layers.GaussianNoise(0.1, input_shape=(28, 28)), #add gaussian noise here
  ...
```

Implementation details

- Loss
  - MeanSquaredError
  - https://keras.io/api/losses/regression_losses/#meansquarederror-class
- Mini-batch size
  - 32
- Epochs
  - 100

Report

- **Due: (6/8, 11:59pm)**
  - English only
  - Late submission will not be counted
- **Submissions: (through blackboard system)**
  - 3 output image files (denoising results when input = ‘noisy.png’)
    - Model1.png, Model2.png, Model3.png
    - Many solutions are available to handle an arbitrary input, and these are one of skills you need to learn in this practice. Please find it out for yourself!
  - 3 source files including training and inference code (i.e., generate ModelX.png)
    - allows only .py or .pynb
    - Model1.py, Model2.py, Model3.py