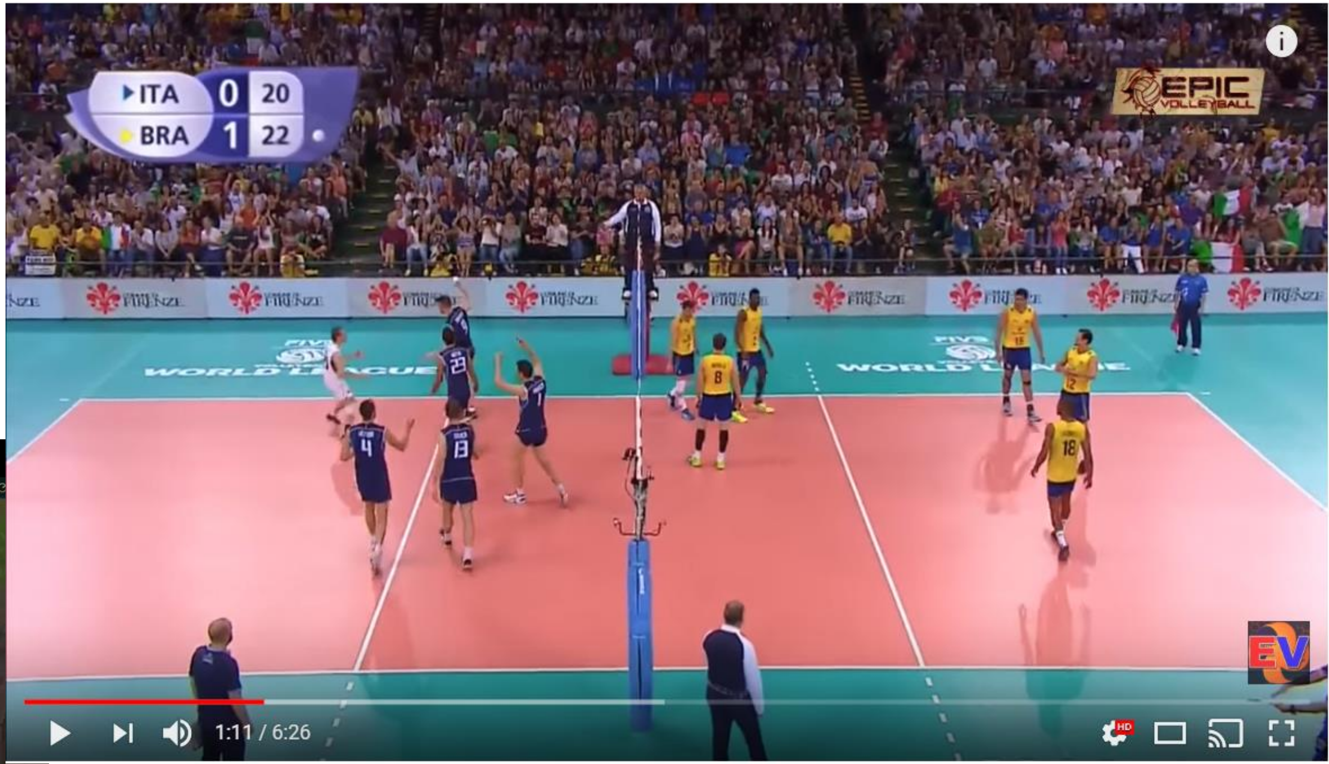
# Residual Coding for Domain-specific Video

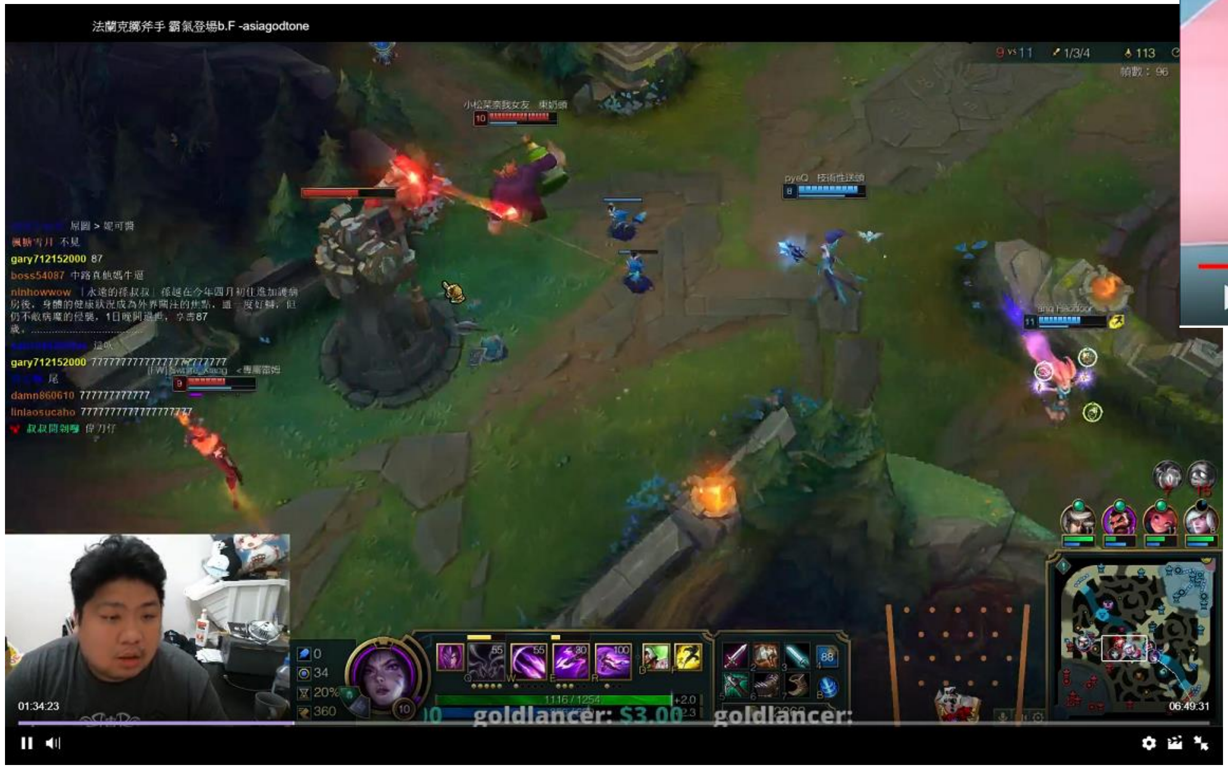
第四組
杜俊毅 B04504028
黃漢威 B03611035

Tsai, Y.H., Liu, M.Y., Sun, D., Yang, M.H., Kautz, J.: Learning binary residual representations for domain-specific video streaming. In: AAAI

# Motivation

- I love watching streaming



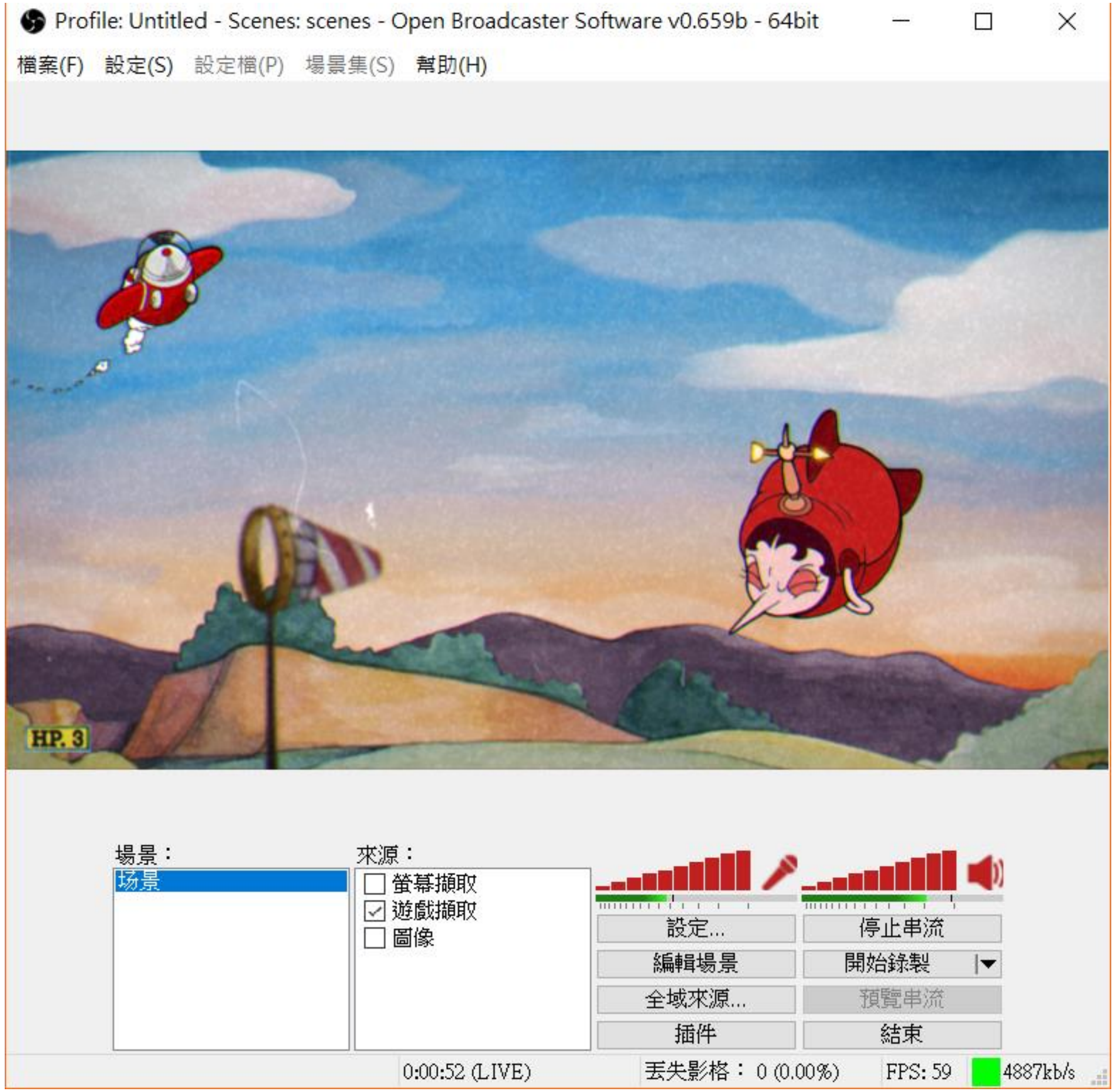



# Motivation

- I hate lagging

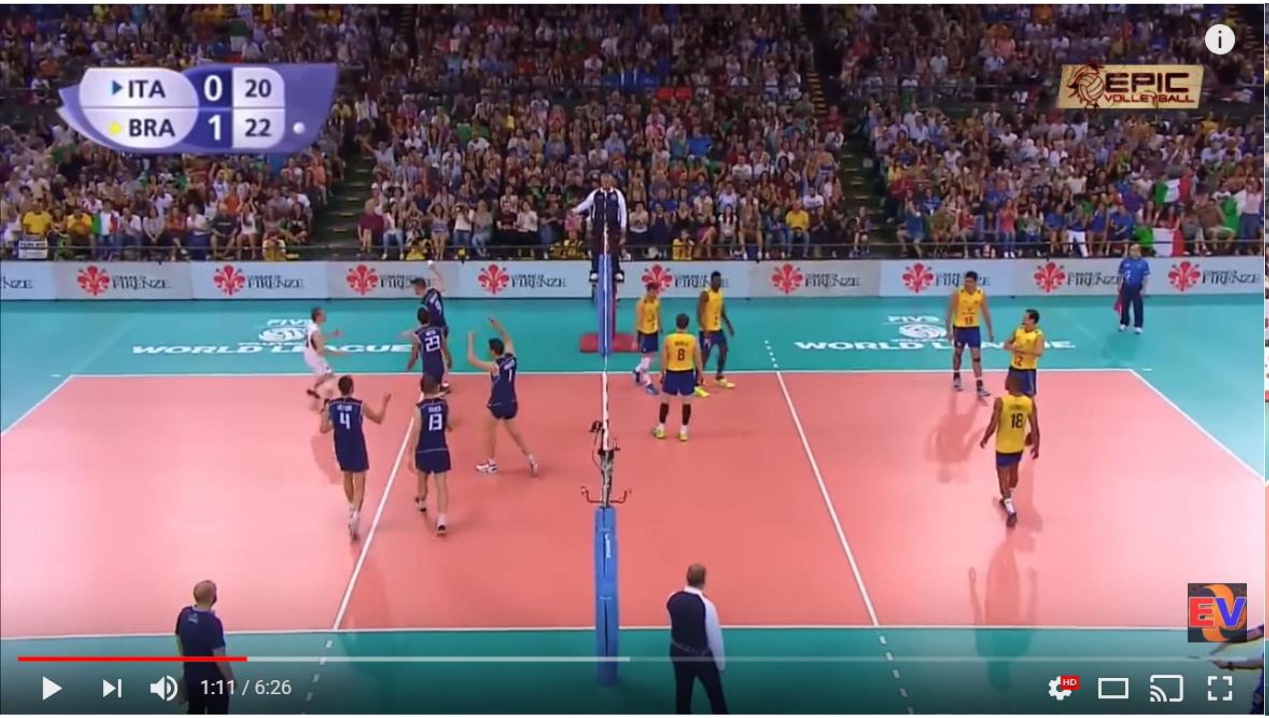
# Motivation

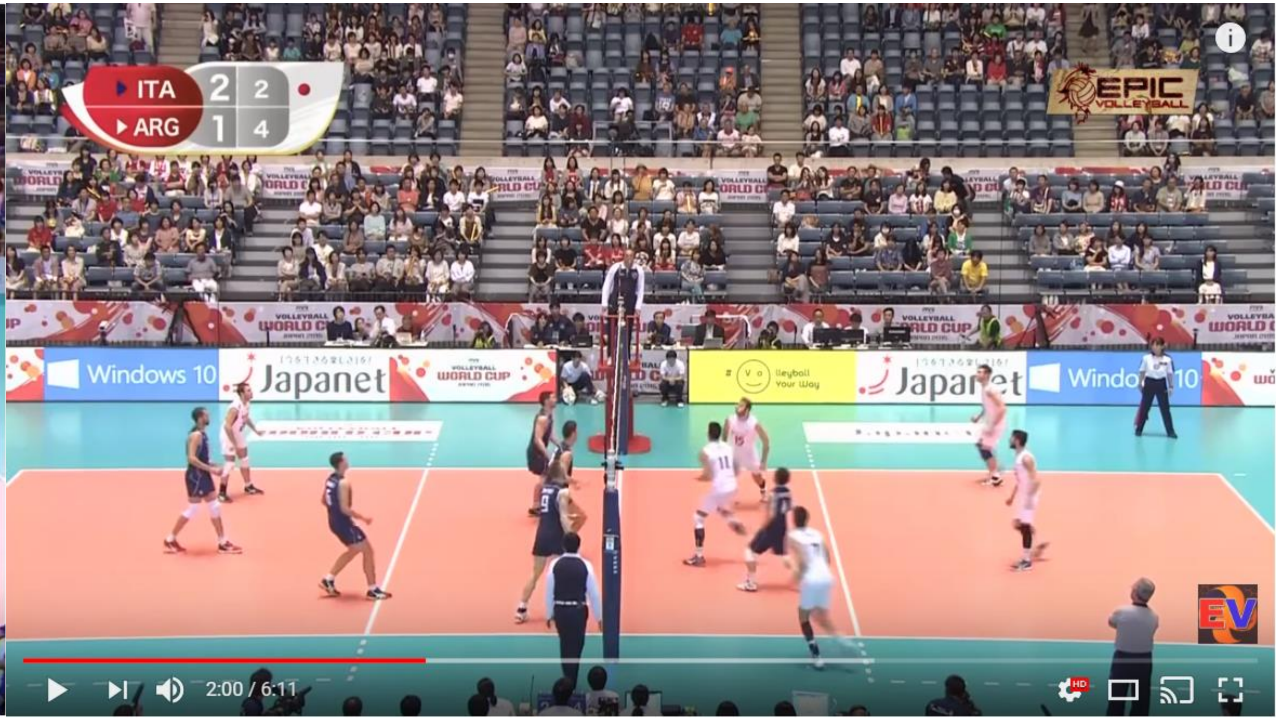
- Sometimes I stream



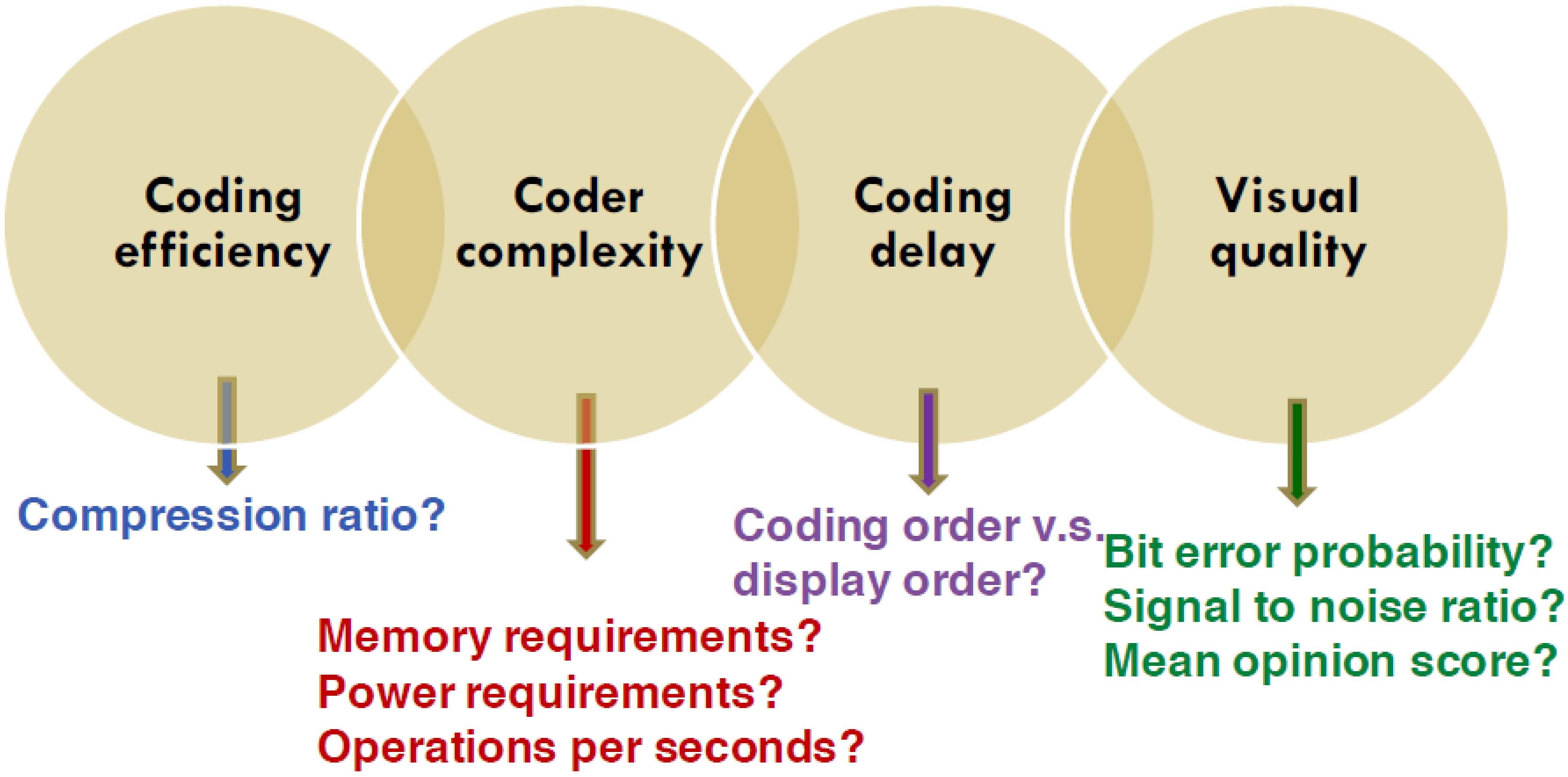
# Motivation

- In YouTube, Twitch…, etc. Videos are categorized.
- Special codecs can be applied.

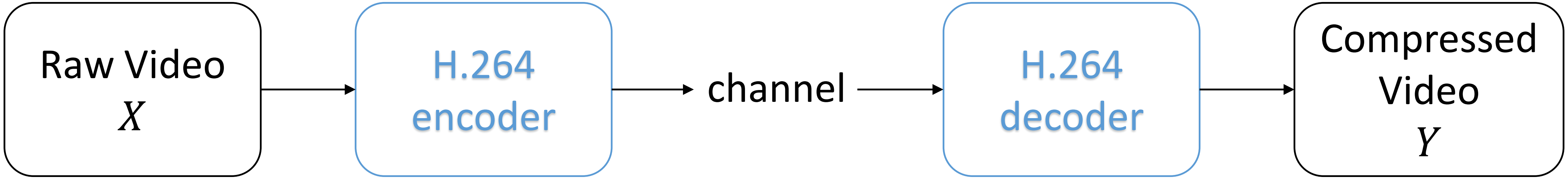




# Motivation

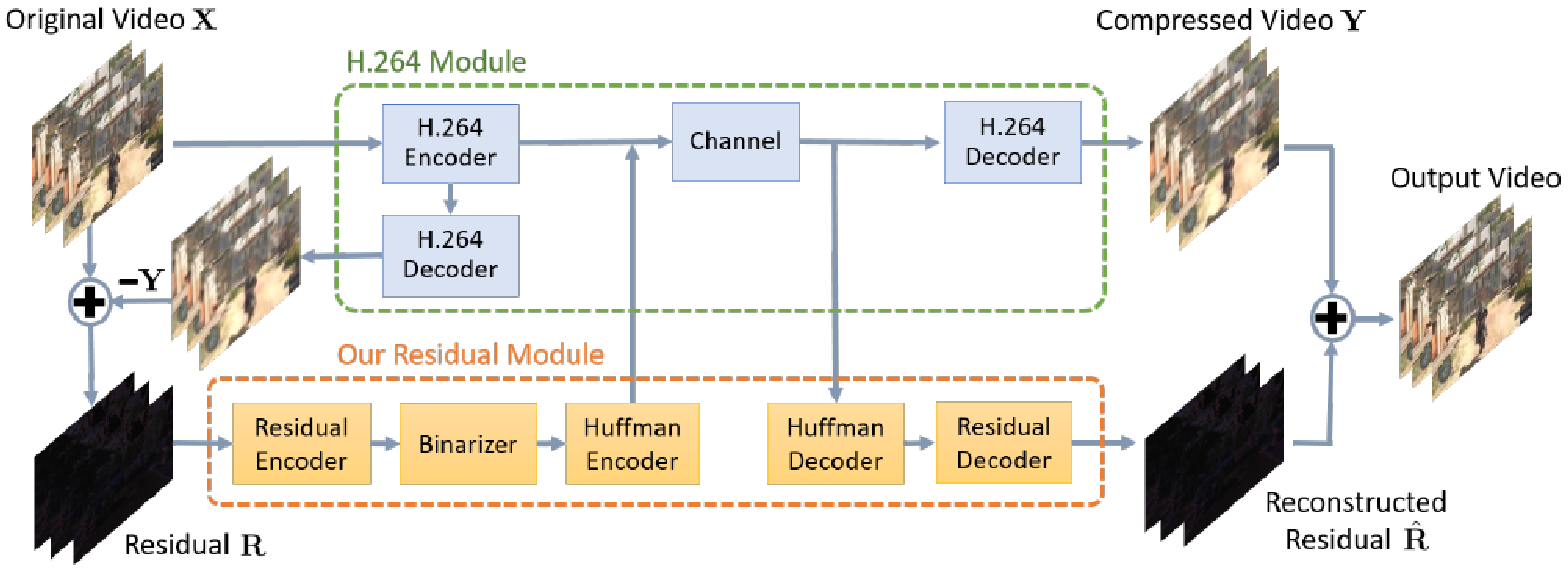

Coding efficiency
Coder complexity
Coding delay
Visual quality
Compression ratio?
Coding order v.s. display order?
Bit error probability?
Signal to noise ratio?
Mean opinion score?
Memory requirements?
Power requirements?
Operations per seconds?

# Problem Definition




- Residual = Raw – Compressed Video $R = X - Y$
- Goal: Minimize $R$ without lowering H.264 compression ratio
  - Lowering streaming bitrate
  - Increasing video quality
  - Focusing on specific content

# Algorithm



Original Video $\mathbf{X}$
H.264 Module
H.264 Encoder
Channel
H.264 Decoder
Compressed Video $\mathbf{Y}$
Output Video
H.264 Decoder
$-\mathbf{Y}$
Our Residual Module
Residual Encoder
Binarizer
Huffman Encoder
Huffman Decoder
Residual Decoder
Residual $\mathbf{R}$
Reconstructed Residual $\hat{\mathbf{R}}$

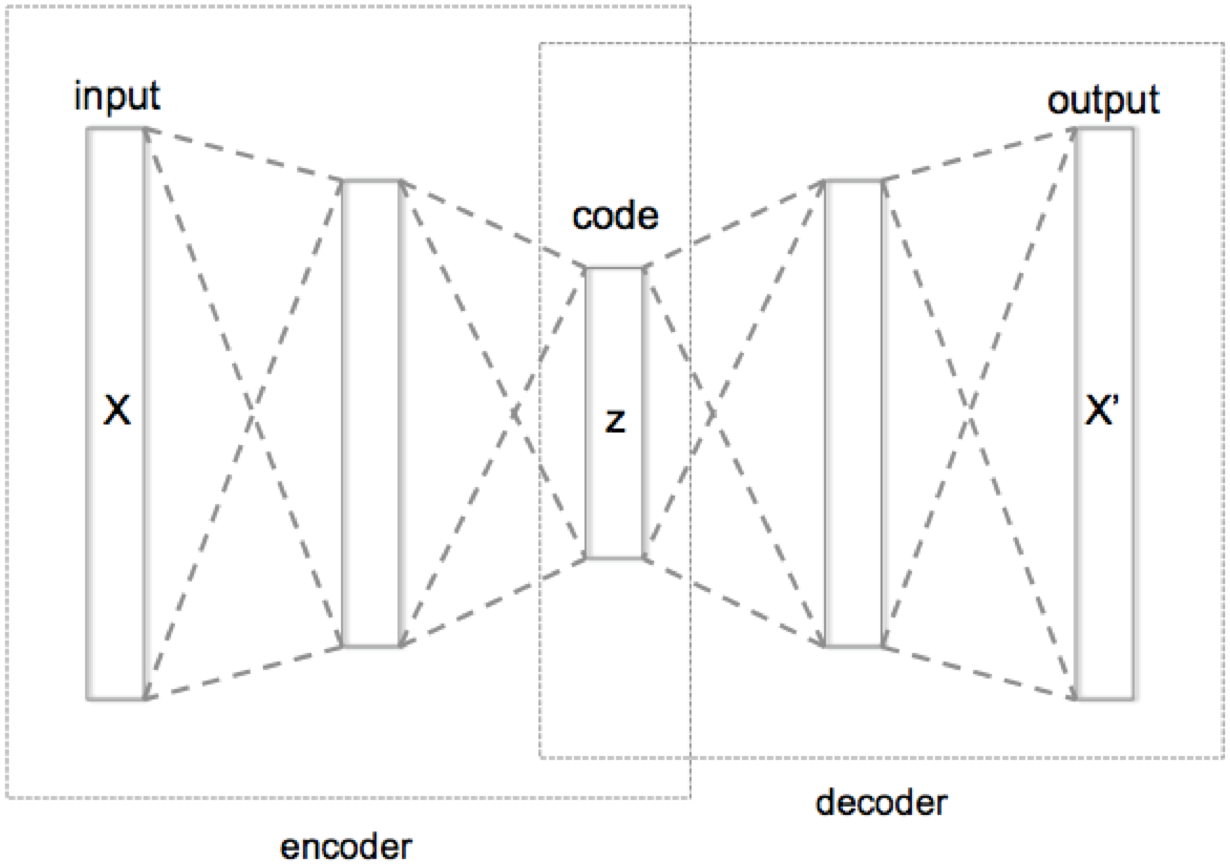
# Autoencoder

- Information-preserving encoding
- Approximate identity function
- $d - \tilde{d} - d$ neural network
- $\tilde{d} < d$: compressed representation

$$z = \sigma(Wx + b)$$

$$X' = \sigma'(W'z + b')$$

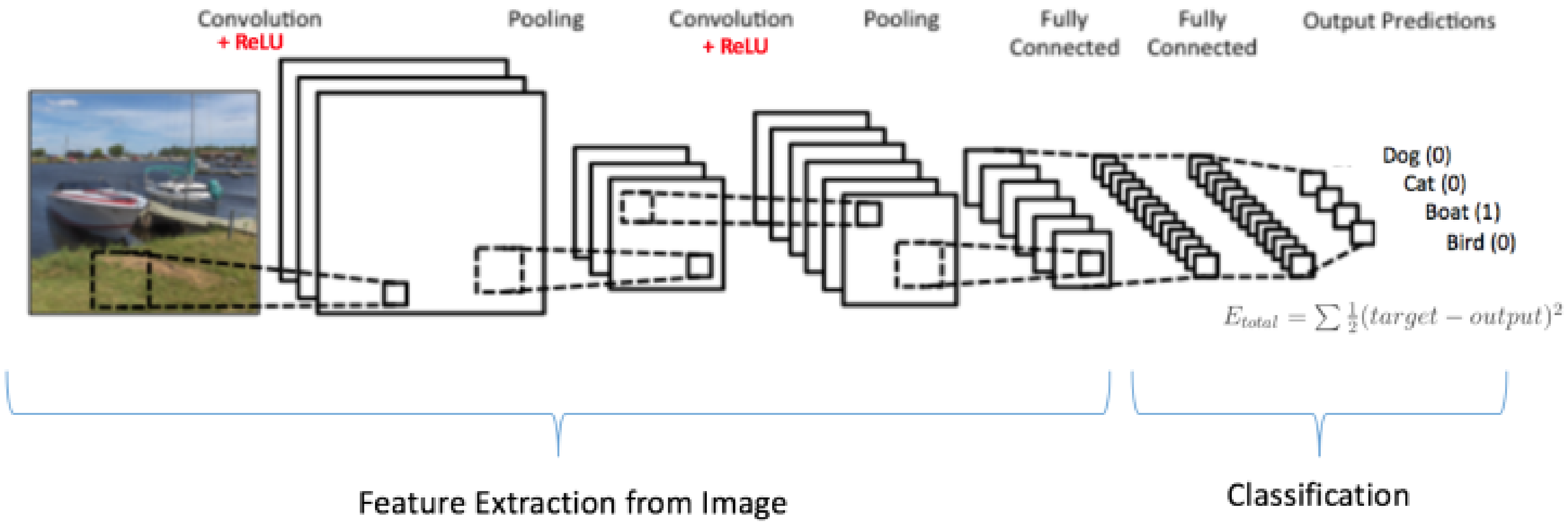
$$L = ||X - X'||^2$$




https://en.wikipedia.org/wiki/Autoencoder

# Convolutional Neural Network

- ConvNet can be seen as a great feature extractor for image.




https://www.clarifai.com/technology

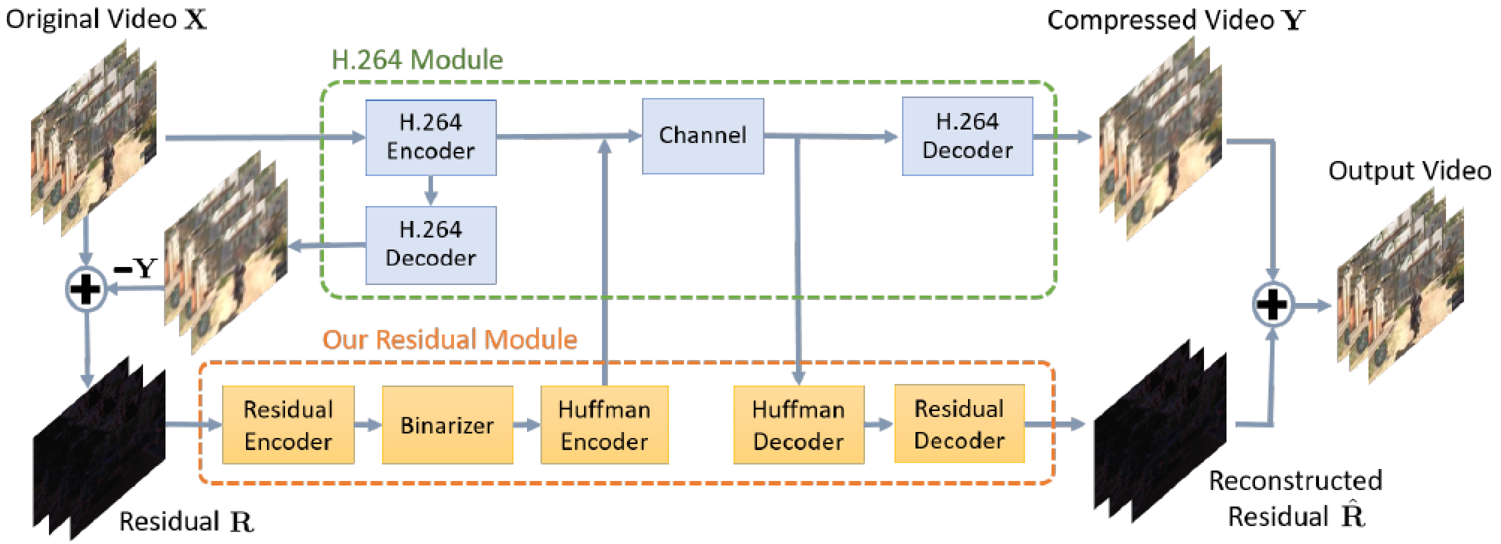
# Binary Residual Autoencoder

Autoencoder consists of encoder $\varepsilon$, binarizer $\beta$ and decoder $D$

**Encoder:** extract feature representation for binarizer

**Binarizer:** convert the output from encoder into a binary map

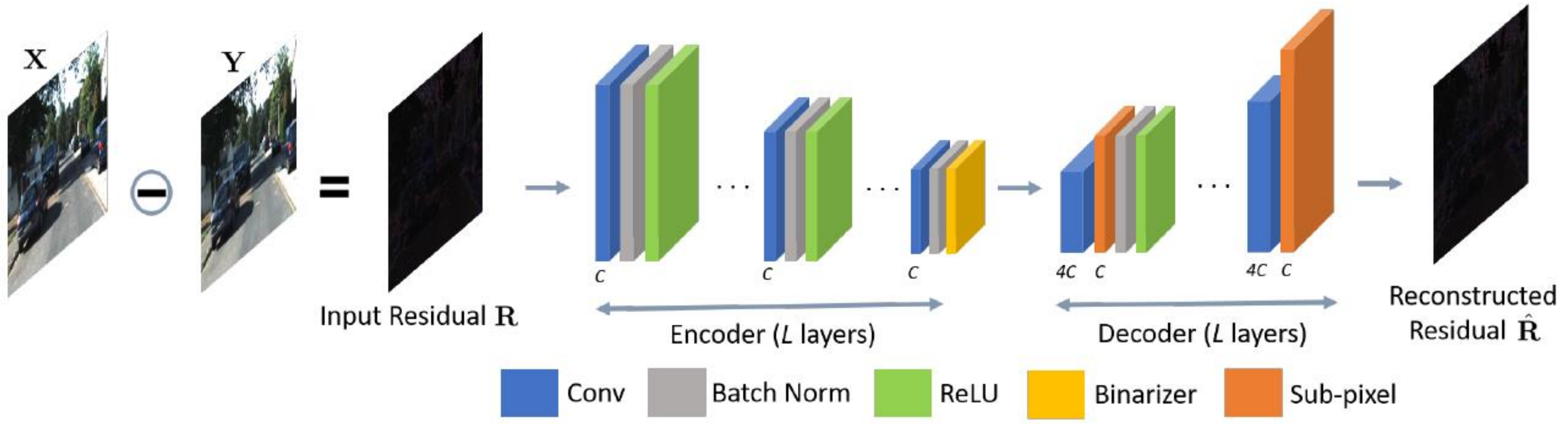
**Decoder:** up-sample the binary map back to the original input



# Binary Residual Autoencoder

Encode and decode residual signal $\mathbf{R}$ frame by frame.
Let $\{r_i\}$ be a set of residual frames after applying H.264.
Objective function: $min_{D,\varepsilon} \sum ||r_i - D(\beta(\varepsilon(r_i)))||^2$

Sub-pixel layer: used for up-sampling [1] Shi et al 2016
Batch normalization and ReLU: facilitate the learning process.




[1] https://arxiv.org/abs/1609.05158

# Binarizer

Let output feature of encoder be $e_i = \varepsilon(r_i)$
Applying activation: $z = \sigma(e_i)$, where $\sigma$ can be tanh or hardtanh.

$$b(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ -1, & \text{if } z < 0, \end{cases}$$

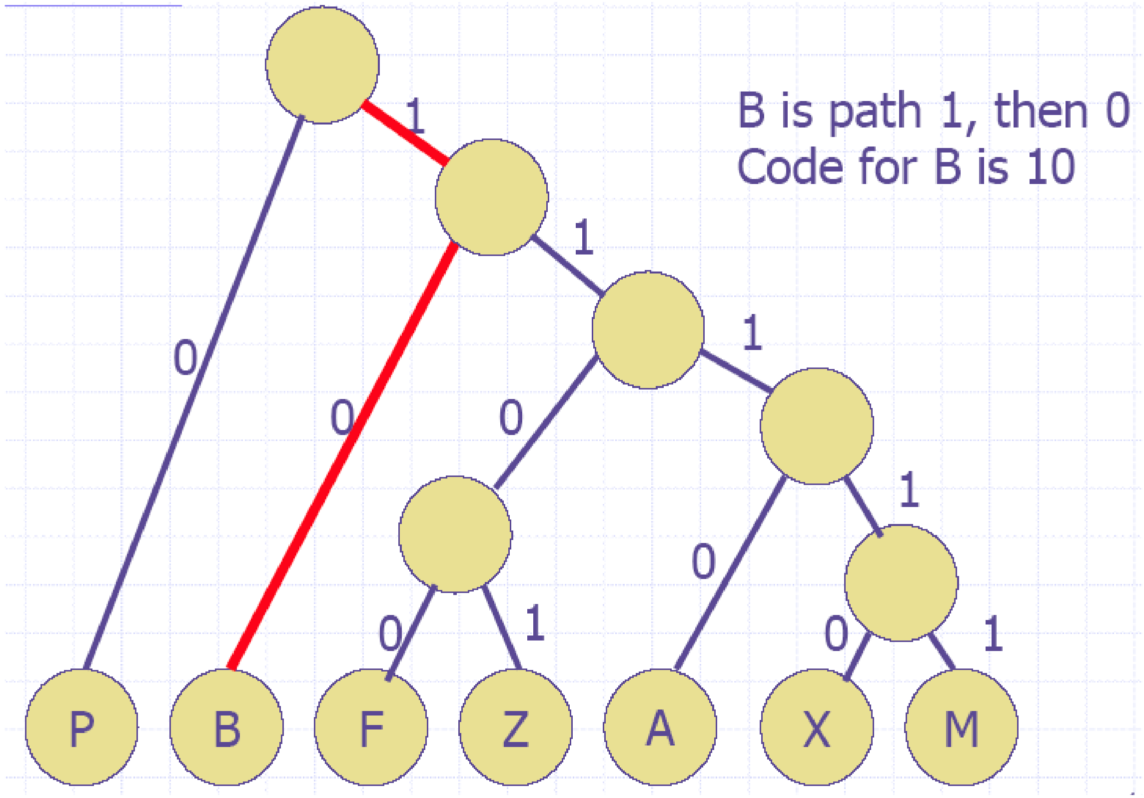
However, since binarization is not differentiable,
we cannot train the autoencoder by back-propagation.

Adopting piecewise function $b_{bp}$ during back-propagation.

$$b_{bp}(z) = \begin{cases} 1, & \text{if } z > 1 \\ z, & \text{if } -1 \leq z \leq 1 \\ -1, & \text{if } z < -1. \end{cases}$$

# Lossless Compression

After generating the binary feature map, we use lossless compression to reduce the size of the binary representation: Huffman coding

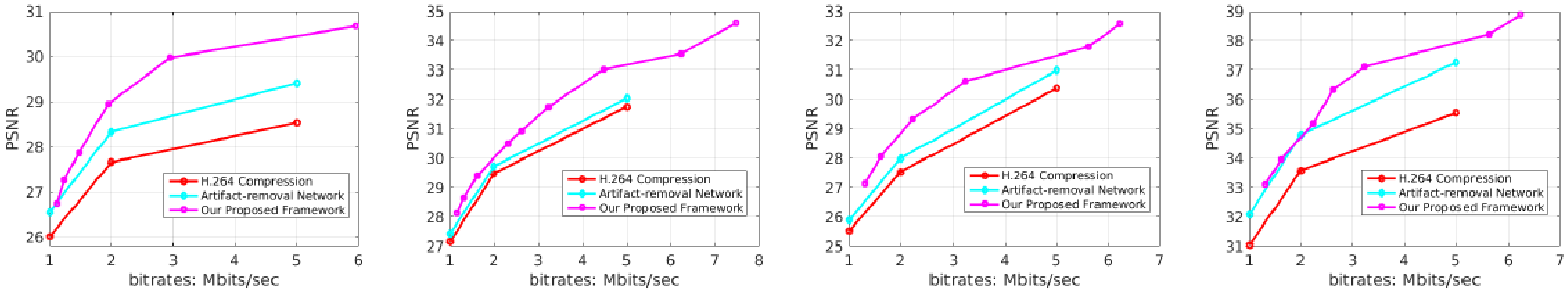


# Expected Result

- The author use KITTI and 3 popular video games: Assassins Creed, Skyrim and Borderlands as datasets.

Table 2: Number of videos and frames on the datasets.

| | KITTI | Assassins Creed | Skyrim | Borderlands |
|---|---|---|---|---|
| Videos | 50 | 50 | 9 | 19 |
| Frames | 19,057 | 34,448 | 9,337 | 8,752 |




(a) KITTI (b) Assassins Creed (c) Skyrim (d) Borderlands

Figure 3: PSNR comparisons on four datasets at different bandwidths. We compare our pipeline with H.264 and an artifact-removal method based on (Kim, Lee, and Lee 2016; Zhang et al. 2017).

# Expected Result

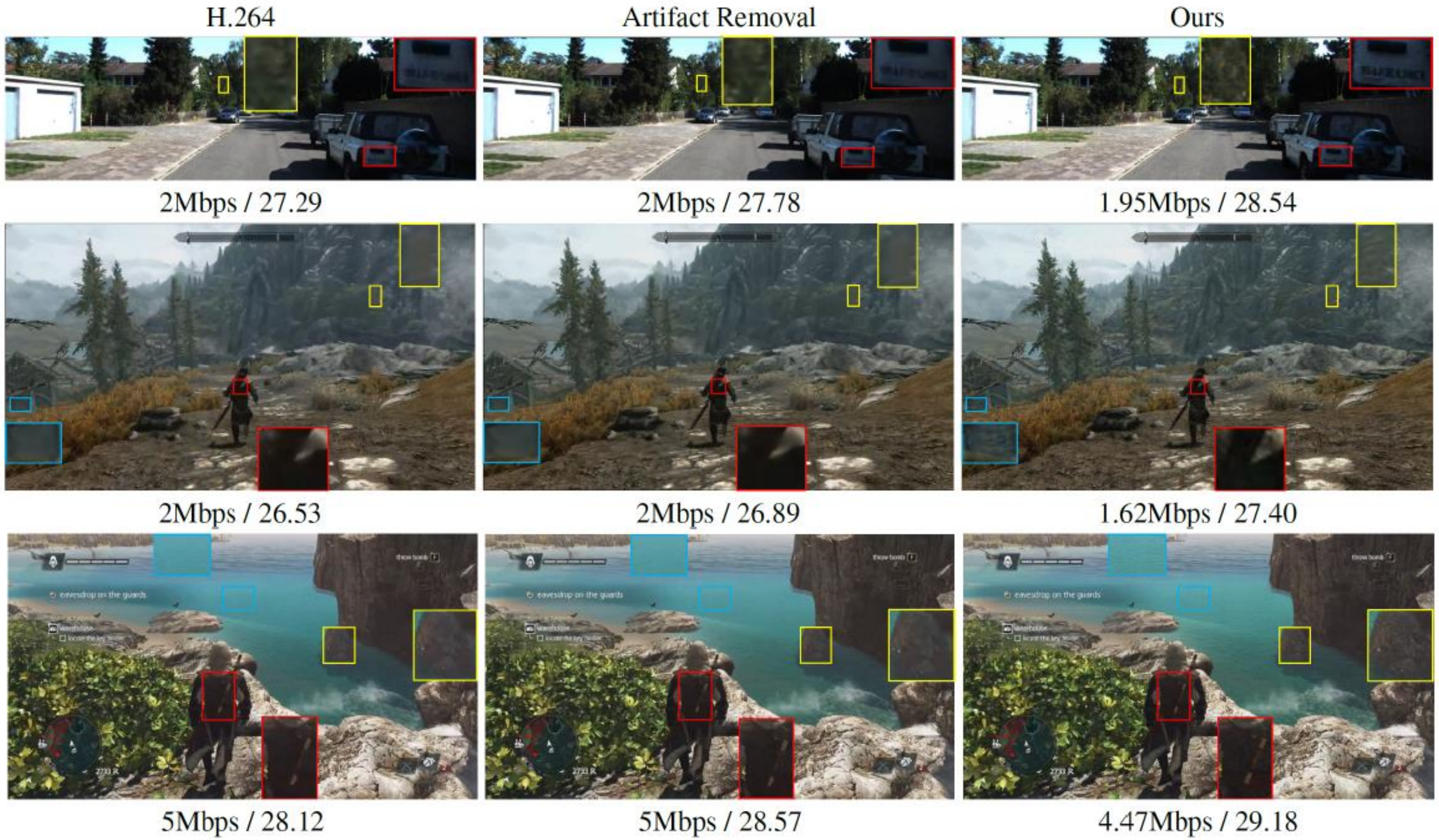



Figure 5: Example results on the KITTI and video game datasets. We compare our pipeline with H.264 and an artifact-removal method. The corresponding bit rate and PSNR are shown next to the images. Best viewed with enlarged images.

# Reference

[1] Tsai, Y.H., Liu, M.Y., Sun, D., Yang, M.H., Kautz, J.: Learning binary residual representations for domain-specific video streaming. In: AAAI (2018)

[2] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1874–1883, 2016.

[3] https://www.clarifai.com/technology

[4] https://en.wikipedia.org/wiki/Autoencoder

[5] MING-SUI (AMY) LEE Lecture