End-to-End Learned Image and Video Compression: Design, Implementation, and Computer Vision Applications

Wen-Hsiao Peng
National Yang Ming Chiao Tung University (NYCU), Taiwan

May 26, 2023
Santa Clara University
Outline

- Overview of Learned Image/Video Compression
- End-to-end Learned Image Compression
- End-to-end Learned Video Compression
- Computer Vision Applications
Outline

- Overview of Learned Image/Video Compression
  - Neural network-based image/video compression
  - Challenge on learned image compression (CLIC)
  - JPEG AI standardization activities
- End-to-end Learned Image Compression
- End-to-end Learned Video Compression
- Computer Vision Applications
Learned Image/Video Compression

- **Deep Learning (DL)-based** compression
  - Neural networks as **backbone** of the compression system
- **DL-assisted** compression
  - DL techniques for enhancing conventional codecs **without changing** their design
- **Hybrid** schemes
  - DL-based **tools** in traditional codecs
Deep Compression Papers

- Deep image/video compression is attracting attention

- **150** papers on deep image compression since 2017
  - Most adopt the autoencoder-based framework with hyperprior

- **40** papers on deep video compression since 2019
  - Potential techniques are still being researched
  - Pixel/feature-domain residual and conditional coding are popular approaches
Transform Coding vs. Deep Coding

[DCT-based Image Codecs]

Image \[\xrightarrow{\text{Block partition}}\] DCT \[\xrightarrow{\text{Quantization}}\] Entropy coding \[\xrightarrow{\text{Binary data}}\]

Image \[\xrightarrow{\text{Reconstruction}}\] IDCT \[\xrightarrow{\text{Inverse-Quant}}\] Entropy decoding

Encoder (DNN) \(f_\theta(x)\) \[\xrightarrow{z}\] Decoder (DNN) \(g_\phi(z)\)

Discriminator (DNN) \(d_\phi(x)\)

[Deep Coding]
Variational Autoencoder (VAE)

- A **generative** model that forms the basis of most learned image/video compression systems

encoding distribution: \( q_\phi(y|x) \)

Decoding distribution: \( p_\theta(x|y) \)

Prior distribution: \( p_\pi(y) \)

\[
\min_{\phi,\pi,\theta} E_{y \sim q_\phi(y|x)} [-\log p_\theta(x|y)] + E_{y \sim q_\phi(y|x)} [-\log p_\pi(y)]
\]

\( \text{Distortion} \)

\( \text{Rate} \)
Generative Adversarial Network (GAN)

- Two neural networks playing against each other
  - Generator (e.g. *learned image codec*)
  - Discriminator (evaluator)
- Latent variables $\rightarrow$ Generator $\rightarrow$ generated image
- Real/Fake inputs $\rightarrow$ Discriminator $\rightarrow$ identify fake!

![Diagram of GAN process]

Training data $X$ (Real)

Generated data $\hat{X}$ (Fake)

Discriminator

Input Image $\rightarrow$ Generator (Learned Codec) $\rightarrow$ Decoded Image

Is real or fake?
Neural Networks for Image Compression?

- Neural networks are good at *synthesizing* image details
- They are amenable to any *differentiable* quality metric

https://hific.github.io/

Source: Mentzer et al., “High-Fidelity Generative Image Compression (HIFIC),” NIPS 2020
How Good is Learned Image Compression?

Top performer (CVPR’23): \textbf{12\%} bit rate saving over VVC Intra
Reference

- Liu et al., “Learned Image Compression with Mixed Transformer-CNN Architectures,” CVPR 2023
- He et al., “ELIC: Efficient Learned Image Compression With Unevenly Grouped Space-Channel Contextual Adaptive Coding,” CVPR 2022
- Ma et al., “End-to-End Optimized Versatile Image Compression With Wavelet-Like Transform,” TPAMI 2020
- Cheng et al., “Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules,” CVPR 2020
How Good is Learned Video Compression?

Top performer (CVPR’23): **55%** bit rate saving over **VVC LDP**
Reference

- **DCVC-DC**: Li et al., “Neural Video Compression with Diverse Contexts,” CVPR 2023.
- **DCVC-HEM**: Li et al., “Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression,” ACM MM 2022
- **DCVC-TCM**: Sheng et al. “Temporal Context Mining for Learned Video Compression,” IEEE TMM 2022
- **DCVC**: Li et al., “Deep Contextual Video Compression,” NeurIPS 2021
Outline

• Overview of Learned Image/Video Compression
  • Neural network-based image/video compression
  • Challenge on learned image compression (CLIC)
  • JPEG AI standardization activities
• End-to-end Learned Image Compression
• End-to-end Learned Video Compression
• Computer Vision Applications
Challenge on Learned Image Compression

- CLIC 2018 targeted image coding @ 0.15bpp
- CLIC 2019 included transparent track (PSNR>40dB)
- CLIC 2020 introduced P-frame track (1 P-frame@0.075 bpp)
- CLIC 2021 introduced video coding track (2-sec videos@30Hz) and multi-rate image coding track (0.075, 0.15, 0.3 bpp)
- CLIC 2022 introduced video coding track (1 mbps & 0.1 mbps for 720p/1080p@15-60fps) and multi-rate image coding track (0.075, 0.15, 0.3 bpp)
Outline

• Overview of Learned Image / Video Compression
  • Neural network-based image / video compression
  • Challenge on learned image compression (CLIC)
  • JPEG AI standardization activities
• End-to-end Learned Image Compression
• End-to-end Learned Video Compression
• Computer Vision Applications
JPEG AI Call-for-Proposals (CFP)

- To complete a **learning-based** image coding standard by **2024**, targeting **human perception** and effective performance for image processing and computer vision tasks.

"Final Call for Proposals for JPEG AI," ISO/IEC JTC 1/SC29/WG1 N100095, Jan 2022
Status and Timeline

- Call-for-Proposals concluded in July 2022
  - 10 responses evaluated objectively and subjectively
  - MS-SSIM, IW-SSIM, VIF, NLPD, PSNR-HVS-M, VMAF, FSIM
  - DSCQS method, 280+ subjects
- 4 parts to be included
  - Part 1 – Core Coding System
  - Part 2 – Profiling
  - Part 3 – Reference Software
  - Part 4 – Conformance
- 2 versions to be standardized
  - v1 focuses on image reconstruction (Int’l Standard: Apr. 2024)
  - v2 addresses compressed-domain vision/processing tasks and better coding efficiency (Int’l Standard: Jan. 2026)
### JPEG AI VM-2.1

5 points BD-rate (0.06, 0.12, 0.25, 0.5, 0.75) vs VVC

<table>
<thead>
<tr>
<th>Test</th>
<th>AVG</th>
<th>msssim Torch</th>
<th>vif</th>
<th>fsim</th>
<th>nlpd</th>
<th>iw-ssim</th>
<th>vmaf</th>
<th>psnrHV</th>
<th>Monotonicity</th>
<th>Max Bit Dev.</th>
<th>Dec. complexity</th>
<th>Enc. complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMv2.1-tools-off-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3%</td>
<td>784</td>
</tr>
<tr>
<td>BRM</td>
<td></td>
<td>-25.1%</td>
<td>-38.0%</td>
<td>-17.9%</td>
<td>-28.8%</td>
<td>-23.4%</td>
<td>-35.0%</td>
<td>-22.7%</td>
<td>-9.7%</td>
<td>TRUE</td>
<td>1.1</td>
<td>0.004</td>
</tr>
<tr>
<td>VMv2.1-tools-on-</td>
<td></td>
<td>-28.3%</td>
<td>-38.7%</td>
<td>-18.8%</td>
<td>-33.8%</td>
<td>-26.8%</td>
<td>-35.6%</td>
<td>-32.6%</td>
<td>-11.5%</td>
<td>TRUE</td>
<td>8%</td>
<td>897</td>
</tr>
<tr>
<td>BRM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.3</td>
<td>0.038</td>
</tr>
</tbody>
</table>

**Decoding runtime (JPEG AI on GPU vs. VVC on CPU)** -- **1.3 : 1**

**Encoding runtime (JPEG AI on GPU vs. VVC on CPU)** -- **0.038 : 1**

25%-28% rate savings over VVC Intra
Outline

• Overview of Learned Image / Video Compression
• End-to-end Learned Image Compression
  • VAE-based framework with hyperprior
  • Parallel-friendly entropy coding
  • Non-linear transform
• End-to-end Learned Video Compression
• Computer Vision Applications
Notable Works

- J. Ballé et al., "Variational Image Compression with a Scale Hyperprior," *ICLR 2018.*
- D. He et al., “Checkerboard context model for efficient learned image compression,” *CVPR 2021.*
Main Encoder and Decoder

- **Encoder** works as an analysis transform to condense the image information

- **Decoder** - “inverse of Encoder” - synthesizes an approximation of the original input
Hyperprior Encoder and Decoder

- **Hyper-encoder** produces hyperprior $z = \{z_i\}$ from image latents $y = \{y_i\}$ as side information.

- **Hyper-decoder** decodes quantized $z$ to output distribution parameters of image latents $y$. 

Distribution parameters
Distributions of Hyperprior \( z \)

- **Assumptions**
  - \( p(z) \) is factorial
  - \( p(z_i) \) is identically distributed
- **Cumulative Distribution Function (CDF) of** \( z_i \) **is learned**

\[
p(\hat{z}_i) = \text{CDF}(\hat{z}_i + \frac{1}{2}) - \text{CDF}(\hat{z}_i - \frac{1}{2})
\]
Conditional Distributions of Image Latents $\mathbf{y}$

- Assumptions
  - $p(y | \hat{z})$ is **fully factorial** along channel and spatial dimensions
  - $p(y_i | \hat{z})$ is **Gaussian** with mean zero and scale derived from $\hat{z}$

Prob. of quantized latent $\hat{y}_i$: $p(\hat{y}_i | \hat{z}) = \int_{\hat{y}_i - 1/2}^{\hat{y}_i + 1/2} \mathcal{N}(y_i; 0, \sigma_i(\hat{z})) dy_i$

---

Quantization

- Rounding to the **nearest integer** at inference time, i.e. uniform quantization with step size 1

- Differentiable approximation during **training**
  - **Additive uniform noise**: to mimic quantization noise; e.g., *End-to-end optimized image compression,* ICLR2017
  - **Stochastic rounding**: quantized value + noise; e.g., *Lossy image compression with compressive autoencoders,* ICLR2017
  - **Soft quantization**: continuous approximation to hard quantizer; e.g., *Soft-to-Hard Vector Quantization for End-to-End Learning Compressible Representations,* NIPS2017
Training Objective

• A weighted sum of distortion $L_D$ and rate $L_R$

\[ L = L_D(x, \hat{x}) + \lambda \times L_R(\hat{y}, \hat{z}) \]

• $L_D$ — any differentiable metric, e.g. MSE, MS-SSIM, perceptual loss, and adversarial loss

• $L_R$ — the rate needed to signal the image latents
  $\log p(\hat{y}|\hat{z})$ and hyperprior $\log p(\hat{z})$
Context Model (1/2)

- $p(y|\hat{z})$ is **non-factorial** along the **spatial** dimension
- $p(y_i|\hat{z}, \hat{y}_{<i})$ is Gaussian with **mean and scale** derived from **hyperprior** $\hat{z}$ and previously **coded image latents** $\hat{y}_{<i}$

To condition the mean and scale predictions based on hyperprior and previously coded latents $\hat{y}_{<i}$.

Source: A. van den Oord et al., “Conditional image generation with pixelcnn decoders,” NIPS16
Rate-Distortion Comparison

Issues of the Context Model

- **NOT parallel-friendly** for decoding
  - Mean and variance prediction has to be done one-by-one
- **Uni-directional context** due to causality
  - Samples can only be referenced from one direction

Previously coded latents $\hat{y}_i$
Outline

- Overview of Learned Image / Video Compression
- End-to-end Learned Image Compression
  - VAE-based framework with hyperprior
  - Parallel-friendly entropy coding
  - Non-linear transform
- End-to-end Learned Video Compression
- Computer Vision Applications
Checkerboard Context Model

- Split latents into **two slices** along the **spatial** dimension
- Slice 1 (anchor) uses **hyperprior** to derive coding probabilities
- Slice 2 (non-anchor) refers to **Slice 1, hyperprior** as context

Source: D. He et al., “Checkerboard context model for efficient learned image compression,” CVPR 2021
Remarks

- Mean and variance prediction in each slice is done in parallel
- Slice 1 (50% of samples) refers to **hyperprior** only
- Slice 2 (remaining 50%) refers to **bi-directional context**
- Decoding time is very close to using **hyperprior only**
- Rate saving decreases by 2-3% (w.r.t. **the context model**)
Channel-wise Context Model

- Split latents into two slices along the channel dimension
- Slice 1 uses hyperprior to derive coding probabilities
- Slice 2 refers to Slice 1 and hyperprior as context

Spatial-channel Context Model

- Split latents into **non-even slices** along the **channels**
- Apply **spatial-channel context** in each slice
- Ex: Checkerboard-channel context

Outline

• Overview of Learned Image / Video Compression
• End-to-end Learned Image Compression
  • VAE-based framework with hyperprior
  • Parallel-friendly entropy coding
  • Non-linear transform
• End-to-end Learned Video Compression
• Computer Vision Applications
Swin-Transformer as Non-linear Transform

- As compared to ConvNet, Swin-Transformer offers
  - Flexible receptive fields
  - Non-stationary, content-adaptive convolution
  - Short- and long-range attentions with shifted windows

Rate saving vs. MACs/pixel for decoding

Flow-based Coding Frameworks


ANFIC in a Nutshell

- Learn image distribution $p(x)$

- Employ a stack of additive autoencoding transforms

- Use augmented noise $e_z$ to convert input $x$ into $x_2 \approx 0$

- Encode the latents $\hat{z}_2, \hat{h}_2$ into a bitstream

[Ho et al. OJCAS 2021]
Additive Autoencoding Transform: Forward

- Additive autoencoding transform + Hyperprior
- Use augmented noise $e_z$ to convert input $x$ into $x_2 \approx 0$
- Encode the latents $\hat{z}_2, \hat{h}_2$ into a bitstream

[Ho et al. OJCAS 2021]
Additive Autoencoding Transform: Inverse

- Addition $\rightarrow$ Subtraction
- Subtraction $\rightarrow$ Addition

[Ho et al. OJCAS 2021]
\[ x_2 = x_1 - \mu_{\pi_1}^{dec} \]

\[ x_3 = x_2 - \mu_{\pi_3}^{dec} \]

\[ g_{\pi_1}^{dec} \]

\[ g_{\pi_2}^{enc} \]

\[ g_{\pi_2}^{dec} \]

\[ g_{\pi_3}^{enc} \]

\[ \mu_{\pi_1}^{dec} \]

\[ \mu_{\pi_2}^{dec} \]

\[ \mu_{\pi_3}^{dec} \]
Rate-Distortion Performance

Kodak dataset

PSNR-RGB (dB)

Bit-rate (bpp)

ANFIC

VVC

BPG

NIPS’18 (VAE-based)

CVPR’20 (VAE-based)

Ours

VTM-444

CVPR’20 (VAE-based)

NIPS’18+GMM (VAE-based)

BPG-444
Outline

- Overview of Learned Image / Video Compression
- End-to-end Learned Image Compression
- End-to-end Learned Video Compression
  - Residual-based video compression
  - Conditional video compression
- Computer Vision Applications
Residual-based Coding Framework

Source: Lu et al., “DVC: An end-to-end deep video compression framework,” CVPR 2019
Notable Works


- **Scale-space**: Agustsson et al., “Scale-space Flow for End-to-End Optimized Video Compression,” CVPR 2020

- **FVC**: Z. Hu et al., “FVC: A New Framework towards Deep Video Compression in Feature Space,” CVPR 2021

- **C2F-FVC**: Z. Hu et al., “Coarse-to-fine Deep Video Coding with Hyperprior-guided Mode Prediction,” CVPR 2022
Outline

- Overview of Learned Image / Video Compression
- End-to-end Learned Image Compression
- End-to-end Learned Video Compression
  - Residual-based video compression
  - Conditional video compression
- Computer Vision Applications
Residual vs. Conditional Coding

Residual coding

Conditional Coding

No evaluation of residual signals!
Conditional Coding

- **Residual coding** is sub-optimal from the information-theoretic perspective

\[ H(I_t - I_c) \geq H(I_t - I_c | I_c) = H(I_t | I_c) \]

- **Conditional coding** aims to approach \( H(I_t | I_c) \)
  - Need to learn the conditional distribution \( p(I_t | I_c) \)

[Li et al. NIPS’21][Ladune et al. MMSP’20]
Conditional vs. Residual Coding in 2022

UVG Dataset

CANF-VC (Conditional), ECCV’22

C2F (Residual-based), CVPR’22

X265

HM as anchor
Notable Works

- **DCVC-TCM**: X. Sheng et al., “Temporal Context Mining for Learned Video Compression,” IEEE TMM, 2022
- **DCVC-HEM**: J. Li et al., “Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression,” ACM MM 2022
- **DCVC-DC**: J. Li et al., “Neural Video Compression with Diverse Contexts,” CVPR 2023
- **MIMT**: J. Xiang et al., “MIMT: Masked Image Modeling Transformer for Video Compression,” ICLR, 2023
DCVC: Deep Contextual Video Coding

- J. Li, B. Li, Y. Lu (MSRA), NeurIPS 2021
- Use a **conditional variational autoencoder (CVAE)** for contextual encoding and decoding

Source: J. Li et al., “Deep Contextual Video Compression,” NIPS 2021
Conditional Variational Autoencoder

- Use signal concatenation as a means to achieve contextual encoding and decoding.

Source: J. Li et al., “Deep Contextual Video Compression,” NIPS 2021
DCVC-TCM: Temporal Context Mining

- Extend DCVC by learning **multi-scale conditioning factors** from **previously stored frame features**

DCVC-HEM

- An extended version of DCVC-TCM
- **Spatial-channel context** model for entropy coding
- **Multi-granularity quantization for variable-rate coding**

Source: J. Li et al., “Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression,” ACM MM 2022
CANF-VC: Conditional Augmented Normalizing Flows for Video Compression

- Y. H. Ho (NYCU), C. P. Chang (NYCU), P. Y. Chen (NYCU), A. Gnutti (Univ. Brescia), W. H. Peng (NYCU), ECCV 2022
- Adopt conditional augmented normalizing flows (CANF) for conditional coding
- Apply conditional coding to both motion and inter-frame coding
Conditional ANF (CANF)

- Turn ANFIC into a **conditional video generator**

Idea: To generate $I_t$ conditionally based on $I_{t-1}$
Conditional Video Frame Generation

$I_c$  Reference frame

$I_t$  Current frame

$\approx I_c$

$g^{\text{dec}}_{\pi_2}$

$g^{\text{enc}}_{\pi_2}$

$h^{\text{dec}}_{\pi_3}$

$h^{\text{enc}}_{\pi_3}$

Dec.2($\hat{z}_2$)

Dec.1($z_1$)

Autoencoding

Autoencoding of hyper-prior

Quantization

$x_2 \approx I_c$
CANF for Motion and Inter-frame Coding

Input Frame

$I_t \xrightarrow{G_{\pi}} I_c \xrightarrow{G_{-\pi}^{-1}} \hat{I}_t$

Motion Compensation

$\hat{I}_{t-1}$

$\hat{f}_t \xrightarrow{F_{\pi}^{-1}} \hat{f}_{t-1}, \hat{f}_{t-2}$

Motion Extrapolation

$\hat{f}_{t-1}, \hat{f}_{t-2}, \hat{f}_{t-3}$

Motion Estimation

$I_t \xrightarrow{\hat{I}_{t-1}}$
Learned Video Codecs vs. X265

- **Learned codec** - Encoding : Decoding ~ 1.6 : 1
- **X265 (HEVC)** - Encoding : Decoding ~ 17.8 : 1
- **Decoding time** - Learned : X265 ~ 37.3 : 1
- **Encoding time** - Learned : X265 ~ 3.3 : 1

Per-frame Runtime on CPU

<table>
<thead>
<tr>
<th></th>
<th>Encode</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>x265</strong></td>
<td>7.81</td>
<td>0.44</td>
</tr>
<tr>
<td><strong>Learned (DVC)</strong></td>
<td>25.78</td>
<td>16.42</td>
</tr>
</tbody>
</table>

CPU: i7-9700K
RAM: 16G
Multiply-Accumulate Operations (MAC)

- BD-rate saving (vs. HM-16.20): The higher the better
- kMAC/pixel for encoding: The lower the better
- Encoding : Decoding = $1.5 : 1$

RTX 3080: 4K@30Hz ~ 128kMAC/pixel
Peak Memory Requirements

- **BD-rate saving (vs. HM-16.20):** The higher the better
- **Peak memory:** The lower the better
Model Size

- **BD-rate saving (vs. HM-16.20):** The higher the better
- **Model size (I-frame NOT counted):** The lower the better
Outline

- Overview of Learned Image / Video Compression
- End-to-end Learned Image Compression
- End-to-end Learned Video Compression
- Computer Vision Applications
Notable Works

- **Single-task** bitstream
  - J. Liu et al., “Improving Multiple Machine Vision Tasks in the Compressed Domain,” ICPR 2022

- **Scalable** bitstream
  - H Choi et al., ”Scalable Image Coding for Humans and Machines,” TIP 2022

- **Multi-task (or many-task)** bitstream
  - R. Feng et al., “Image Coding for Machines with Omnipotent Feature Learning,” ECCV2022
Improving Multiple Machine Vision Tasks in the Compressed Domain

- Train a base codec with multi-task loss (e.g. recon. + seg.)
- Encoder adopts a gate module for task-specific feature coding
- Decoder uses a transform module to adapt features to the task

Encoder = Image Compressor + Feature Extractor

Source: J. Liu et al., “Improving Multiple Machine Vision Tasks in the Compressed Domain,” ICPR 2022
Scalable Image Coding for Humans and Machines

- Divide image latents $y$ along channel dimension into $y_1$ and $y_2$
- Decode $y_1$ (base layer) for machine perception
- Decode $y_1 + y_2$ (enhancement layer) for human perception

Source: H Choi et al., "Scalable Image Coding for Humans and Machines," TIP 2022
Image Coding for Machines with Omnipotent Feature Learning

- Use **contrastive learning** to learn **omnipotent features**, i.e. features suitable for **many vision tasks**
- Encode omnipotent features with learned codecs
- Fine-tune recognition networks with omnipotent features

Source: R. Feng et al., “Image Coding for Machines with Omnipotent Feature Learning,” ECCV 2022
Omnipotent Feature Learning

- Information Filtering (IF) + Contrastive learning
- IF: rate constrained representation learning

Source: R. Feng et al., “Image Coding for Machines with Omnipotent Feature Learning,” ECCV 2022
Transfer Learning for Machine Perception

- **Task**: To **transfer a learned codec** from human perception to machine perception
- Adapt the **feature distributions** of the learned codec using **task-specific conditions** for machine perception

---

![Diagram of Transfer Learning Process](image)

- **Input**
- **Task 1 condition**
- **Encoder**
- **Decoder**
- **Task 1**: Classification
- **Task 2**
- **Encoder**
- **Decoder**
- **Task 2 condition**
- **Single-task or multi-task bitstreams**
Machine Task: Classification

- **Base codec** (Human)
- **Adapted base codec** (Machine)
- **Full finetuning** (Machine)

Decoded Image

Effective Receptive Field
Rate-Accuracy Performance

**Classification**

- **Full finetuning**
- **Adapted**
- **Base**

**Object Detection**

- **Full finetuning**
- **Adapted**
- **Base**

Graphs showing the relationship between bit-rate (bpp) and Top-1 Accuracy or mAP for both Classification and Object Detection tasks. The graphs illustrate the performance of Uncompressed, Full fine-tuning, Base codec, and Adapted base codec models.
Conclusion

- End-to-end learned image/video compression is progressing at light speed
- For image coding,
  - VAE-based compression with hyperprior is currently the most popular approach
  - JPEG AI is progressing fast towards version 1
  - PSNR-RGB: > VVC intra, MS-SSIM: >> VVC
- For video coding,
  - Research is still ongoing, with conditional coding emerging as an attractive alternative to residual coding
  - More content-adaptive coding is expected
  - PSNR-RGB: > VVC (LDP), MS-SSIM: >> VVC
Conclusion

- The complexity of learned codecs is still high
  - 2 to 3 orders of magnitude higher than traditional codecs in terms of kMAC/pixel and CPU decoding time
- Many issues remain widely open
  - Performance and complexity assessment
  - YUV content coding
  - Encoder optimization
  - Rate control
  - Generalization
  - Coding for machines
  - etc.
Neural Image Compression on FPGA

Thank you for your attention