Learned Image and Video Compression with Deep Neural Networks

Dong Xu  
University of Sydney, Australia

Guo Lu  
Beijing Institute of Technology, China

Ren Yang  
ETH Zurich, Switzerland

Radu Timofte  
ETH Zurich, Switzerland
Learned Image and Video Compression with Deep Neural Networks

Part 1  Learned Image Compression

Ren Yang
Ph.D. student

Radu Timofte
Lecturer, Group leader

Computer Vision Laboratory
ETH Zurich, Switzerland
Large amount of high-resolution images/videos

Limited bandwidth

Limited storage

7296 x 5472 = 39,923,712 pixels

Uncompressed image: 39,923,712 x 3 = 120 MB

Uncompressed video (60 fps): 120 MB x 60 = 7.2 GBps (18s needs 128 GB)

Lossless compression (.png): 44 MB

Lossy compression (.jpg): 9 MB

Image/video compression plays an important role in multimedia streaming, online conference, data storage, etc.
Rate-distortion trade-off

Metrics: PSNR, (MS-)SSIM, NIMA, LPIPS, user studies, etc.

Higher quality (lower distortion)

Target

Better quality at the same bit-rate

Lower bit-rate at the same quality

Higher bit-rate (file size)
Traditional Image Compression

• Classical Architecture:
  - Transform → Quantization → Entropy encoding → Bit-stream
  - Frequency domain (coefficients) → Quantized coefficients

• Standards: JPEG (DCT + Huffman), JPEG2000 (DWT + Arithmetic coding), BPG (HEVC), ...

• Example: JPEG compression framework

![diagram of JPEG compression process]
Entropy coding

Entropy:

\[ H(X) = E[I(X)] = E[-\log(P(X))] \]

\[ H(X) = -\sum_{i=1}^{n} P(x_i) \log_p P(x_i) \]

Cross entropy:

\[ H(p, q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x) \] (Eq.1)

(Adaptive) arithmetic coding is theoretically able to losslessly compress data at
• bit-rate \( \approx \) cross entropy (with little overhead)

Arithmetic coding:

• 60% chance of symbol NEUTRAL
• 20% chance of symbol POSITIVE
• 10% chance of symbol NEGATIVE
• 10% chance of symbol END-OF-DATA.

Adaptive arithmetic coding:
Changing the frequency (or probability) tables while processing the data.

\[ 0.10001010 \] (0.5390625 decimal)
8 bits
Learned Image Compression

- Basic architecture \(^1\): **End-to-end trainable**

1. **Deep neural network**
2. **Quantization**
3. **Latent representation**
4. **Differentiable entropy model** (to model \(P_q\))

![Diagram of learned image compression](image)

\[
L[g_a, g_s, P_q] = -\mathbb{E}[\log_2 P_q] + \lambda \mathbb{E}[d(x, \hat{x})]
\]

Learned Image Compression

- CNN transformer + **factorized** entropy model [1]

\[ \tilde{y} = y + \Delta y \sim \mathcal{U}(0, 1) \]

Training: differentiable quantization

\[ \hat{y} = \text{round}(y) \]

Inference: quantization (not differentiable)

\[ \tilde{y} \text{ or } \hat{y} \]

\[ f_k(x) = g_k(H^{(k)}x + b^{(k)}) \quad 1 \leq k < K \]

Optimized in an end-to-end manner

Learned Image Compression

- CNN transformer + **hyperpiror** entropy model [2]

![Diagram of CNN transformer and hyperprior entropy model](image)

**Discretized Gaussian Distribution**

\[
p_{y|z}(\hat{y} \mid \hat{z}) \leftarrow p_{\hat{y}_i}(\hat{y}_i \mid \hat{\sigma}_i) = \int_{\hat{y}_i-1/2}^{\hat{y}_i+1/2} \mathcal{N}(y \mid 0, \hat{\sigma}_i) \, dy
\]

Learned Image Compression

- CNN transformer + autoregressive entropy model \[^{[3]}\]

Due to the chain rule:

\[
p(y) = p(y_1) \cdot p(y_2|y_1) \cdot p(y_3|y_2, y_1) \cdots p(y_N|y_{<N})
\]

\[
p_{\hat{y}|z}(\hat{y} | \hat{z}) = \prod_{i=1}^{N} p_{\hat{y}_{<i}|\hat{y}_{<i}, \hat{z}}(\hat{y}_i | \hat{y}_{<i}, \hat{z})
\]

\[
p_{\hat{y} \mid \hat{z}, \theta_{hd}, \theta_{cm}, \theta_{ep}}(\hat{y}) = \prod_i \left( \mathcal{N}(\mu_i, \sigma^2_i) * \mathcal{U}(-\frac{1}{2}, \frac{1}{2}) \right)(\hat{y}_i)
\]

with \(\mu_i, \sigma_i = g_{ep}(\psi, \phi_i; \theta_{ep}), \psi = g_h(\hat{z}; \theta_{hd}), \text{and } \phi_i = g_{cm}(\hat{y}_{<i}; \theta_{cm})\)


Learned Image Compression

- CNN transformer + **autoregressive** entropy model [5]

Learned Image Compression

$y_i \in \{\text{hot coffee, hot tea, cold coffee, cold tea}\} \quad y = [y_1, y_2, y_3]

- **Factorized** entropy model

  \[ p_{y_i}(y_i) = 25\% \text{ for } y_i = \text{hot coffee, hot tea, cold coffee, cold tea} \]
  \[ H(p_{y_i}) = 4 \times (-0.25 \log_2 0.25) = 2 \]

  The expected number of bits to encode $y$ is 6

- **Hyperprior** entropy model \quad $z = [10\degree C, 15\degree C, 30\degree C]$

  \[ p_{y_i|z_i}(y_i|z_i < 20\degree C) = 50\% \text{ for } y_i = \text{hot coffee, hot tea} \quad H = 2 \times (-0.5 \log_2 0.5) = 1 \]
  \[ p_{y_i|z_i}(y_i|z_i \geq 20\degree C) = 50\% \text{ for } y_i = \text{cold coffee, cold tea} \quad H = 1 \]

  The expected number of bits to encode $y$ is 3

- **Autoregressive** entropy model (joint with hyperprior)

  Don’t drink coffee (or tea) in two consecutive days.
  \[ p(y) = [\text{hot coffee, hot tea, cold coffee}] = 0.5 \]
  \[ p(y) = [\text{hot tea, hot coffee, cold tea}] = 0.5 \]

  $z = [10\degree C, 15\degree C, 30\degree C]$

  The expected number of bits to encode $y$ is $H(y) = 2 \times (-0.5 \log_2 0.5) = 1$
Learned Image Compression

• Another differentiable quantization method \cite{Mentzer2018}

\[
\hat{z}_i = Q(z_i) := \arg \min_j \|z_i - c_j\|
\]
\[
\tilde{z}_i = \sum_{j=1}^{L} \frac{\exp(-\sigma \|z_i - c_j\|)}{\sum_{l=1}^{L} \exp(-\sigma \|z_i - c_l\|)} c_j
\]

• Importance map \cite{Mentzer2018}

\[\bar{z}_i = \text{tf.stopgradient}(\hat{z}_i - \tilde{z}_i) + \tilde{z}_i\]

Learned Image Compression

• Another differentiable quantization method \cite{Mentzer2018}

\[
\hat{z}_i = Q(z_i) := \arg \min_j \| z_i - c_j \|
\]

\[
\tilde{z}_i = \frac{\sum_{j=1}^{L} \exp(-\sigma \| z_i - c_j \|) c_j}{\sum_{l=1}^{L} \exp(-\sigma \| z_i - c_l \|)} c_j
\]

Inference

\[
\bar{z}_i = \text{tf.stopgradient}(\hat{z}_i - \tilde{z}_i) + \tilde{z}_i
\]

Training: differentiable

• Importance map \cite{Mentzer2018}

• Gaussian Mixture Model (GMM) for entropy \cite{Cheng2020}

\[
p_{\hat{y}|z}(\hat{y} | \hat{z}) \sim \sum_{k=1}^{K} w^{(k)} \mathcal{N}(\mu^{(k)}, \sigma^{2(k)})
\]


\cite{Cheng2020} Cheng et al. "Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules", in CVPR. 2020.
Learned Image Compression

- CNN transformer + **coarse-to-fine** model [7]

Learned Image Compression

- Performance

The context (autoregressive) and coarse-to-fine models outperform BPG 4:4:4 (latest traditional standard)

The rank may vary on different datasets
Learned Image Compression

- Variable rate image compression: RNN-based methods [8, 9]

Learned Image Compression

- Variable rate image compression: RNN-based methods [10]

Learned Image Compression


Loss function:

\[
\min_{\phi, \theta} \left\{ D_{\phi, \theta} + \lambda R_{\phi} \right\}
\]

\[
\min_{\phi, \theta} \sum_{\lambda \in \Lambda} (D_{\phi, \theta}(\lambda) + \lambda R_{\phi, \theta}(\lambda))
\]

\[
\min_{\phi, \theta} \sum_{\lambda \in \Lambda} \mathbb{E}_{p(\Delta)}[D_{\phi, \theta}(\lambda, \Delta) + \lambda R_{\phi, \theta}(\lambda, \Delta)]
\]

Learned Image Compression

- Variable rate image compression: Conditional autoencoder \cite{choi2019variable}

Learned Image Compression

• Variable rate image compression: Wavelet-like transformer [12]

Invertible: achieving lossy and lossless compression by the same framework

Learned Image Compression

• Variable rate image compression: Wavelet-like transformer \[12\]
Learned Image Compression

- Generative image compression: GAN-based methods [13]

$$\min_{E,G} \max_D \left[ \mathbb{E}[f(D(\tilde{w}))] + \mathbb{E}[g(D(G(\tilde{w}))) + \lambda \mathbb{E}[d(x, G(\tilde{w}))] + \beta H(\tilde{w}), \right]$$

GAN loss

RD loss

Learned Image Compression

- Generative image compression: GAN-based methods [13]

**Conditional GAN:** \( \mathcal{L}_{\text{cGAN}} := \max_D \mathbb{E}[f(D(x, s))] + \mathbb{E}[g(D(G(z, s), s))] \)

**Selective generative compression (SC):** binary heatmap \( m \)

Learned Image Compression

• Generative image compression: GAN-based methods \[^{[14]}\]

**High-Fidelity Generative Image Compression**

**Conditional discriminator:**

\[
\mathcal{L}_{EGP} = \mathbb{E}_{x \sim p_X} \left[ \lambda r(y) + d(x, x') - \beta \log(D(x', y)) \right],
\]

\[
\mathcal{L}_D = \mathbb{E}_{x \sim p_X} \left[ -\log(1 - D(x', y)) \right] + \mathbb{E}_{x \sim p_X} \left[ -\log(D(x, y)) \right].
\]

Learned Image Compression

• Generative image compression: GAN-based methods \cite{14}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{image.png}
\caption{Comparison of original and compressed images using different methods.}
\end{figure}

\begin{itemize}
\item \textbf{HIFiC}^{Lo}: 0.198bpp
\item BPG: 0.224bpp
\item HIFiC^{Lo} (Ours): 0.198bpp
\item Original
\end{itemize}

\textsuperscript{14} Mentzer, Fabian, et al., "High-Fidelity Generative Image Compression." in NeurIPS. 2020.
Learned Image Compression

Conclusion:
• CNN-based methods
  • Factorized entropy model
  • Hyperprior entropy model
  • Autoregressive entropy model
  • Coarse-to-fine entropy model
  • Conditional auto-encoder (variable bit-rates)
  • Invertible auto-encoder (lossy and lossless by one framework)
• RNN-based methods
  • Variable bit-rate
• GAN-based methods
  • Photo-realistic compressed image with low bit-rate

The state-of-the-art learned image compression methods successfully outperform the latest traditional compression standard BPG 4:4:4
Learned Image Compression

• Will learning-based compression be standardized?
• Can learning-based method be compatible with traditional standards (e.g., JPEG)?

**JPEG initiates standardisation of image compression based on AI**

The 89th JPEG meeting was held online from 5 to 9 October 2020.

During this meeting multiple JPEG standardisation activities and explorations were discussed and progressed. Notably, the call for evidence on learning-based image coding was successfully completed and evidence was found that this technology promises several new functionalities while offering at the same time superior compression efficiency, beyond the state of the art.

**JPEG AI**

At the 89th meeting the submissions to the Call for Evidence on learning-based image coding were presented and discussed. Four submissions were received in response to the Call for Evidence. The results of the subjective evaluation of the submissions to the Call for Evidence were reported and discussed in detail by experts. It was agreed that there is strong evidence that learning-based image coding solutions can outperform the already defined anchors in terms of compression efficiency, when compared to state-of-the-art conventional image coding architecture. Thus, it was decided to create a new standardisation activity for a JPEG AI on learning-based image coding system, that applies machine learning tools to achieve substantially better compression efficiency compared to current image coding systems, while offering unique features desirable for an efficient distribution and consumption of images. This type of approach should allow to obtain an efficient compressed domain representation not only for visualisation, but also for machine learning based image processing and computer vision. JPEG AI releases to the public the results of the objective and subjective evaluations as well as a first version of common test conditions for assessing the performance of leaning-based image coding systems.
Learned Image Compression

- Will learning-based compression be standardized?
- Can learning-based method be compatible with traditional standards (e.g., JPEG)?

**We made an attempt:** [15]

---

Learned Image Compression

• Will learning-based compression be standardized?
• Can learning-based method be compatible with traditional standards (e.g., JPEG)?

We made an attempt: [15]

Learned Image Compression

• Will learning-based compression be standardized?
• Can learning-based method be compatible with traditional standards (e.g., JPEG)?

**We made an attempt:** [15]

**Frequency domain pre-editing**

We achieve better rate-distortion performance **without changing the standard decoder**

The compressed image can be decoded (viewed) on **any common device**, e.g., mobile, ipad, PC, etc.

Learned Image Compression

• Open source codes:

  • Ballé et al., (factorized), Ballé et al., (hyperprior):
    https://github.com/tensorflow/compression (TensorFlow)
  • Ballé et al., (factorized), Ballé et al., (hyperprior), Minnen et al., (autoregressive):
    https://interdigitalinc.github.io/CompressAI/index.html (PyTorch)
  • Lee et al., (context-adaptive):
    https://github.com/JooyoungLeeETRI/CA_Entropy_Model
  • Mentzer et al., (autoregressive + importance map):
    https://github.com/fab-jul/imgcomp-cvpr
  • Cheng et al., (GMM entropy model):
  • Hu et al., (coarse-to-fine):
    https://github.com/huzi96/Coarse2Fine-imaComp
  • Ma et al., (wavelet-like transformer):
    https://github.com/mahaichuan/Versatile-Image-Compression
  • Mentzer et al., (generative compression):
    https://github.com/tensorflow/compression/tree/master/models/hific
Learned Image Compression

Thanks for your attention

Q & A

Dong Xu  
University of Sydney, Australia  
dong.xu@sydney.edu.au

Guo Lu  
Beijing Institute of Technology, China  
luguo2014@sjtu.edu.cn

Ren Yang  
ETH Zurich, Switzerland  
ren.yang@vision.ee.ethz.ch

Radu Timofte  
ETH Zurich, Switzerland  
radu.timofte@vision.ee.ethz.ch
Learned Image and Video Compression with Deep Neural Networks

Dong Xu
University of Sydney, Australia

Guo Lu
Beijing Institute of Technology, China

Ren Yang
ETH Zurich, Switzerland

Radu Timofte
ETH Zurich, Switzerland

VCIP, December 1-4, 2020
Learned Image and Video Compression with Deep Neural Networks

PART 2: Learned Video Compression

**Guo Lu**  
Assistant Professor  
Beijing Institute of Technology, China

**Dong Xu**  
Professor  
University of Sydney, Australia

VCIP, December 1-4, 2020
Outline

• Background for Video Compression
• End-to-end Learned P-frame Compression
• End-to-end Learned B-frame Compression
• Learned Autoencoder based Video Compression
• Discussion
Background for Video Compression

Traditional codecs rely on classical prediction-transform architecture and hand-crafted techniques.

Deep learning has been widely used for a lot of vision tasks for its powerful representation ability.

What happens when video compression meets deep learning?
Background for Video Compression

• Traditional Video Compression

Transform

\( x_t \) Current Frame

Motion Estimation

\( \hat{x}_t \) Prediction

Linear Transform (DCT)

Quantization

Linear Inv-Transform (IDCT)

Motion Compensation

Reconstructed Residual

Decoded Frames Buffer

\( \hat{x}_{t-1} \)
End-to-End Learned P-Frame Video Compression

- The first end-to-end optimized video compression system\(^1\)

\[
\min \lambda D + R
\]

End-to-End Learned P-Frame Video Compression

• Flexible Framework\textsuperscript{[1,2]}
  • Advanced entropy model for motion/residual compression with auto-regressive entropy \textit{\rightarrow} reduce bitrate
  • Motion and Residual Refinement \textit{\rightarrow} reduce distortion

• Variable Bitrate Solution
  • Adaptive quantization layer \textit{\rightarrow} feature transformation between different rates

End-to-End Learned P-Frame Video Compression

- PSNR results on JCT-VC

End-to-End Learned P-Frame Video Compression

• RDO is the fundamental technique for video compression
  • Choose the optimal mode in the coding stage, such as motion vector, block size, etc.

• Learning based video compression
  • RDO is ignored in the inference stage
  • the “modes” are fixed after the training stage

• How to apply the RDO technique in learned video compression
  • Directly optimize the encoder
  • Introduce more “modes” and select the optimal
End-to-End Learned P-Frame Video Compression

1. Lack of adaptiveness to different video content\cite{3}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figures.png}
\caption{Comparison of PSNR between different methods on HEVC Class B and Class C datasets.}
\end{figure}

DVC, Not as adaptive to different content as traditional methods

Once the training procedure is finished, \textbf{the parameters in the learning based encoder are fixed}

---

2. Error propagation in inter predictive coding

Quality keeps decreasing because **the error propagation is not considered in the training procedure**

End-to-End Learned P-Frame Video Compression

1. Online Encoder Updating Scheme
2. Error Propagation Aware Training Strategy

End-to-End Learned P-Frame Video Compression

1. Online Encoder Updating Scheme

- Update encoder online
- Keep the decoder unchanged

\[ L_t = \lambda D_t + R_t \]
\[ = \lambda d(x_t, \hat{x}_t) + [H(\hat{y}_t) + H(\hat{m}_t)] \]

End-to-End Learned P-Frame Video Compression

1. Online Encoder Updating Scheme

Improve adaptiveness with Same decoding time

End-to-End Learned P-Frame Video Compression

2. Error Propagation Aware Training Strategy

Combine the rate-distortion loss of several frames to optimize the learned codec

\[ L^T = \frac{1}{T} \sum L_t = \frac{1}{T} \sum \{ \lambda d(x_t, \hat{x}_t) + [H(\hat{y}_t) + H(\hat{m}_t)] \} \]
End-to-End Learned P-Frame Video Compression

All methods use same GoP settings

End-to-End Learned P-Frame Video Compression

### End-to-End Learned P-Frame Video Compression

Table 1: The BDBR and BD-PSNR results of different algorithms when compared with H.264. Negative values in BDBR represent the bitrate saving.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BDBR(%)</th>
<th>BD-PSNR(dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>H.265</td>
<td>DVC</td>
</tr>
<tr>
<td>Class B</td>
<td>-32.0</td>
<td>-27.9</td>
</tr>
<tr>
<td>Class C</td>
<td>-20.8</td>
<td>-3.5</td>
</tr>
<tr>
<td>Class D</td>
<td>-12.3</td>
<td>-6.2</td>
</tr>
</tbody>
</table>

All methods use same GoP settings

End-to-End Learned P-Frame Video Compression

Error Propagation Aware training brings 0.2dB BD-PSNR improvement

End-to-End Learned P-Frame Video Compression

Online Encoder Updating brings 0.5dB BD-PSNR improvement

Overall, near 1dB improvement

End-to-End Learned P-Frame Video Compression

- Various block sizes are adopted in traditional video compression
  - Large block size for smooth region
  - Small block size for complex region

- Fixed optical flow resolution and motion representation resolutions are used in existing work, like DVC.
  - Generates more mode -> flow resolutions or representation resolutions
  - Choose the optimal mode using RDO
End-to-End Learned P-Frame Video Compression

- (a) Overview of the proposed video compression system.
- (b) RaFC-Frame: decides the **Global Optimal Flow Map** resolution for each video frame.
- (c) RaFC-Block: select the optimal resolution for each **Local Block** of motion feature

End-to-End Learned P-Frame Video Compression

[Image of a diagram showing the process of video compression using deep neural networks, including motion estimation, encoder and decoder nets, and upsampling.]
End-to-End Learned P-Frame Video Compression

Motion Estimation Net

\[ V_t^1 \]

\[ V_t^2 \]

MV Encoder Net

\[ \hat{M}_t^1 \]

\[ \hat{M}_t^2 \]

MV Decoder Net

\[ Q \]

Upsampling

\[ \hat{V}_t^1 \]

\[ \hat{V}_t^2 \]

End-to-End Learned P-Frame Video Compression

- Generate multi-scale motion features
- Select the optimal resolution of the motion features for each block

End-to-End Learned P-Frame Video Compression

End-to-End Learned P-Frame Video Compression

End-to-End Learned P-Frame Video Compression

End-to-End Learned P-Frame Video Compression

MCL-JCV:
BDBR: Ours: -35.0% vs. DVC: -14.5%

VTL:
BDBR: Ours: -30.0% vs. DVC: -21.9%

End-to-End Learned P-Frame Video Compression

- UVG: BDBR:
  Ours: -35.7% vs. DVC: -19.4%

- Class E:
  BDBR:
  Ours: -50.3% vs. DVC: -34.8%

End-to-End Learned P-Frame Video Compression

Traditional Motion Compensation Procedure

Reference → Current

Optical Flow

Warped Frame

Formulations

\[ x' := \text{Bilinear-Warp}(x, f) \quad \text{s.t.} \]
\[ x'[x, y] = x[x + f_x[x, y], y + f_y[x, y]] \]

Limitations
1. Rely on existing network architecture
2. May need pretrain
3. Large residual due to inaccurate warp operation

End-to-End Learned P-Frame Video Compression

Traditional Motion Compensation Procedure

- Reference
- Current
- Optical Flow
- Warped Frame

Scale-space-warp\cite{5} motion compensation procedure

Displacement Field \((g_x, g_y)\)

Scale Field \((g_z)\)

Scale Space Volume \((X)\)  
Trilinearly Interpolated Output

Formulations

\[
x' := \text{Scale-Space-Warp}(x, g) \quad \text{s.t.} \\
x'[x, y] = X[x + g_x[x, y], y + g_y[x, y], g_z[x, y]]
\]

End-to-End Learned P-Frame Video Compression

• Overall Architecture

Hybrid Coding Approach:
1. Scale Space Flow Encoder & Decoder
2. Scale Space Warping based Motion Compensation
3. Residual Encoder & Decoder

End-to-End Learned P-Frame Video Compression

• Scale-space flow visualization

Previous reconstruction $\hat{x}_{i-1}$  Displacement Field $(g_x, g_y)$  Scale Field $g_z$

Scale Space Warped Prediction $\tilde{x}_i$  Decoded Residual $\hat{r}_i$  Final Reconstruction $\check{x}_i$

End-to-End Learned P-Frame Video Compression

End-to-End Learned P-Frame Video Compression

• Existing methods use one previous reference frame
• Exploiting multiple reference frames for learned video compression
  • Directly use multiple frames for motion estimation or motion compensation.
  • Explore the long-range temporal information in latent space
End-to-End Learned P-Frame Video Compression

- Exploiting multiple reference frames for learned video compression

---

End-to-End Learned P-Frame Video Compression

- M-LVC: Multiple Frames Prediction for Learned Video Compression

End-to-End Learned P-Frame Video Compression

• Maintains a state of arbitrary information learned by the model and jointly compressing all transmitted signals\cite{rippel2019learned};

\[ S_{t-1} \] represents the state from previous time steps and includes the information from both residual and motion.

End-to-End Learned P-Frame Video Compression

• The latent representations are generated based on limited reference frames;
• Existing work focus on the independent context information only;
  • motion compression and residual compression

-> exploiting the temporal redundancy to generate latent representations and more accurate context information
End-to-End Learned P-Frame Video Compression

- Implicitly explore temporal information in multiple frames


This slide is provided by Ren Yang.
End-to-End Learned P-Frame Video Compression

- Implicitly explore temporal information in multiple frames

\[
H(p_t, q_t) = \mathbb{E}_{y_t \sim p_t}[-\log_2 q_t(y_t | y_1, \ldots, y_{t-1})]
\]

\[
q_t(y_t | y_1, \ldots, y_{t-1}) = \prod_{i=1}^{N} q_{it}(y_{it} | y_1, \ldots, y_{t-1})
\]

\[
q_{it}(y_{it} | y_1, \ldots, y_{t-1}) = \int_{y_{it} - 0.5}^{y_{it} + 0.5} \text{Logistic}(y; \mu_{it}, s_{it}) dy
\]
End-to-End Learned B-Frame Video Compression

• Frame Interpolation based Video Compression

---

End-to-End Learned B-Frame Video Compression

- Frame Interpolation based Video Compression

1. Extract features from reference images
2. Use block based motion estimation
3. Interpolate the current frame
4. Compress residual using learned image codec
5. Compress motion using traditional image codec

Limitations:
1. Not end-to-end optimized
2. Motion compression is not learnt

End-to-End Learned B-Frame Video Compression

- Hierarchical Learned Video Compression (HLVC) with recurrent enhancement [1]

The benefits of hierarchical quality are two-fold:

- **At encoder side**, the high quality frames provide high quality references to improve the compression performance of other frames.

- **At decoder side**, the low quality frames can be enhanced by taking advantage of high quality frames without bit-rate overhead. It is equivalent to reducing bit-rate on low quality frames.


This slides is provided by Ren Yang
End-to-End Learned B-Frame Video Compression

- Hierarchical Learned Video Compression (HLVC) with recurrent enhancement \cite{1}

**Layer 1:**
Compressed by BPG for PSNR model, and by Lee et al. ICLR 2019 for MS-SSIM model.

**Layer 2:**
Compressed by the proposed Bi-Directional Deep Compression (BDDC) network


This slides is provided by Ren Yang
End-to-End Learned B-Frame Video Compression

• Hierarchical Learned Video Compression (HLVC) with recurrent enhancement \(^1\)

Layer 3:
Compressed by the proposed
Single Motion Deep
Compression (SMDC)

Due to the correlation of motions among multiple neighboring frames, we propose using the motion between \(x_0^C\) and \(x_2\) to predict the motions between \(x_1\) and \(x_0^C\) or \(x_2\). That is,

\[
\hat{f}_{1 \rightarrow 0} = \text{Inverse}(0.5 \times \text{Inverse}(\hat{f}_{2 \rightarrow 0})).
\]

As such, \(x_1\) can be compressed with the reference frames of \(x_0^C\) and \(x_2\), without bits consumed for motion map, thus improving the rate-distortion performance.
End-to-End Learned B-Frame Video Compression

• Previous works use separate interpolation network and motion compression module
  -> combine interpolation network and motion compression
• The residual is compressed in the pixel domain and it is a non-trivial task.
  -> Feature space residual compression
End-to-End Learned B-Frame Video Compression

• Combine interpolation and flow compression and decode the flow and interpolation coefficients simultaneously.

\[ x_{\text{intrp}} = \sum_{i=1}^{k} \hat{\alpha}_i w(x_i, \hat{f}_i) \quad \text{with} \quad \sum_{i=1}^{k} \hat{\alpha}_i = 1. \]

End-to-End Learned B-Frame Video Compression

• Residual Compression in Latent Space
Learned Autoencoder based Video Compression

• Previous works follow the hybrid coding framework, i.e., motion compensation and residual coding.
• Using optical flow for explicitly motion estimation
• Separately motion and residual compression
Learned Autoencoder based Video Compression

- Use 3D autoencoders to compress video frames without explicitly motion estimation.

1. The input is a chunk of video
2. Resblock based 3D autoencoder
3. Auto-regressive prior model

results on UVG

Discussion

• Open-Source Project
  • Pytorch Data Compression
    • Learned Image Compression
      • Balle, ICLR2017
      • Balle, ICLR2018
      • Minne, NeurIPS2018
    • Learned Video Compression
      • DVC, CVPR2019
      • HU[4], ECCV2020(ongoing)
    • Learned Point Cloud Compression
      • OctSqueeze, CVPR2020

https://github.com/ZhihaoHu/PyTorchDataCompression/
Reference

Acknowledge

• Co-authors
  • Chunlei Cai(SJTU)
  • Zhihao Hu(BUAA)
  • Zhenghao Chen(Usyd)
  • …

• Benchmark results from
  • Ren Yang(ETH Zurich)
  • Chao-yuan Wu(UT Austin)
VCIP2020 Tutorial Learned Image and Video Compression with Deep Neural Networks

Q&A

VCIP, December 1-4, 2020  

IEEE VCIP 2020