Learning based fast H.264/AVC to H.265/HEVC transcoding

Qingxiong Huangyuan¹, Li Song¹, Yue Ma¹, Rong Xie¹, Zhengyi Luo²

¹Shanghai Jiao Tong University
²Shanghai University of Electric Power
2015.12
Outline

- Background
- Prior works
- The proposed method
- Experimental results
- Conclusion
Transcoding

Video transcoding is the operation of converting video from one format to another to meet various requirements. A format is defined by characteristics such as bit-rate, spatial/temporal resolution etc.
H.264 and H.265

- H.264/AVC is widely deployed today and a matured solution for most existing video infrastructure.
- H.265/HEVC is the latest video coding standard, aims at high coding efficiency while retaining the video quality.

H.264 to H.265 Transcoding

- Transcoding from H.264 to HEVC will enable lowering the bitrate resulting in a more efficient compression.
- AVC and HEVC share a similar prediction, transform, quantization, and entropy coding architecture.
Background

Pixel domain H.264-HEVC transcoder

To simplify, both spatial resolution and frame rate is kept, only bitrate is reduced by setting same QP for HEVC encoder as input H.264.

Fig from D. Zhang, B. Li, J. Xu, and H. Li, ‘Fast Transcoding from H.264/AVC to High Efficiency Video Coding’ IEEE ICME, pp. 651-656, July, 2012
Background

**H.264/AVC**

- 16x16 Macroblock

**Block coding structure**

- 3 Intra partitioning
- 4 Inter partitioning
- +4 sub-partitioning 8x8
- 2 Transform sizes: 4x4, 8x8

- Up to 9 Intra prediction directions

**HEVC**

- Coding Unit 64x64 to 8x8

**Quadtrees coding structure**

- Prediction Unit and Transform Unit partitioning

- Multiples sizes/forms: 64x64 to 4x4

- 35 Intra prediction directions

Efficient spatio-temporal mv prediction

http://www.worldbroadcastingunions.org/wbuarea/library/docs/isog/presentations/2012B/2.4_Vieron%20ATEME.pdf
We are focusing on CU depth decision for inter frame

- Observation: “coding patterns” is common for MB or CU
- Methodology: learn this pattern offline, use them online

![Diagram showing the process of converting H.264 to HEVC]

Input H.264 Bitstream → H.264 Decoder → Prediction Model → HEVC Re-encoder → Output HEVC Bitstream

MB partition features → CU Depth
Outline

- Background
- Prior works
- The proposed method
- Experimental results
- Conclusion
Prior works

- Z. Chen, C. Tseng, P. Chang, “Fast Inter prediction for H.264 to HEVC transcoding”, ICMT 2013
Prior works

- Learning based H.264 to HEVC [E. Peixoto, 2014]
  - Training and transcoding: training is performed at the beginning of the sequence (12 frames) to get a model. Then this model is used for later frames
  - Content modeling transcoder using linear discriminant functions (LDA) as classifier
  - Rich features from MV, DCT Coefficients and mode distribution are used
  - A higher speed-up (~3 times) at the cost of a higher loss in bitrate (~10%)

Prior works

- Learn based HEVC encoding [Y. Zhang, 2015]
  - Model quad-tree CU depth decision process as a three-level of hierarchical binary decision problem, and a RD-complexity model is derived to determine the optimal parameters for the joint classifier, which is capable of minimizing the complexity in each CU depth at given RD degradation constraints.
  - Get state-of-the-art results by reducing computational complexity by about 50% on average with negligible loss (0.061 dB, in terms of BD-PSNR)

Outline

- Background
- Prior works
- The proposed method
- Experimental results
- Conclusion
The proposed method

- **Observation 1: Similarity between H.264 and HEVC**
  - H.264 has quite similar partition and splitting modes with HEVC, we can make full use of H.264’s information to accelerate HEVC encoding.
Observation 2: splitting patterns are regularized by limited coding modes for similar neighboring block
The proposed method

- **SVM is used to train a mode which can predict CU depth according to available H.264 MB level features for HEVC encoder to accelerate the transcoding speed**
The proposed method

- Constructing features
  - Take H.264 MB(16x16) as a unit
  - Should reflect neighboring MBs partition info., up to LCU size.

- SKIP(0)
- 16x16(1)
- 16x8(2)
- 8x16(3)
- 8x8 (4)
- <8x8(5)
- ...

- Sum_D1
- Var_D1

- Sum_D0
- Var_D0
The proposed method

- Thus, for each MB, we have a feature vector like

\[ F = \begin{bmatrix} MB_{Type}, & Sum_{D1}, & Var_{D1}, & Sum_{D0}, & Var_{D0} \end{bmatrix}^T \]

- Many prior works verify that depth is greatly affected by Qps or bit rate, thus we need train models for different Qps.

- In practice, one can set some discrete anchor points across certain Qps or bit rate.
The proposed method

- **Post-processing—merge after model predicting**
  - To compensate model inaccuracy with strong prior information: remove outliers with surrounding values.
The proposed method

- *Post-processing-adjust for special case as shown in the following picture*
- The current model cannot discriminate this case well, thus we need calculate both depth 2 and depth 3 for all four MBs.

- Predicted Depth
  - 2
  - 2
  - 3
  - 3

- Real Depth
  - 3
  - 2
  - 3
  - 2
The proposed method

The transcoding processing

- Input H.264 Stream
- H.264 Decoder
  - QP(i)
  - MB & Partition
  - Feature Construction
  - SVM Model(i)
- YUV
- Post-processing
- CU Depth & control flag
- HEVC Encoder
- Output H.265 Stream
Outline

- Background
- Prior works
- The proposed method
- Experimental results
- Conclusion
Experimental results

Benchmark – The cascaded H.264-HEVC transcoder

- JM Encoder
- Input H.264 Bitstream
- JM Decoder
- Output HEVC Bitstream
- x265 Encoder
- PSNR
- QP
Experimental results

- **Experiments set**
  - JM decoder + x.265 1.4
  - x265 is open source implementation of HEVC encoding, highly optimized for multicore CPUs, which make it more close to real-time application.
  - X265 default preset with RDO=6(highest) with typical low delay setting.
  - We train models for four QPs(22,27,32,37)
  - Class B and class E sequences are used training and testing, where one is used for testing, others are used to train model to validate our method.
## Experimental results

### Preliminary results

<table>
<thead>
<tr>
<th>Sequences</th>
<th>$\Delta$Time(%)</th>
<th>$\Delta$PSNR(dB)</th>
<th>$\Delta$Bitrate(%)</th>
<th>BD-PSNR(dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cactus (1920x1080)</td>
<td>43.07</td>
<td>0.0163</td>
<td>2.07</td>
<td>-0.0572</td>
</tr>
<tr>
<td>BQTerrace (1920x1080)</td>
<td>44.47</td>
<td>0.0175</td>
<td>1.79</td>
<td>-0.06</td>
</tr>
<tr>
<td>BasketDrive (1920x1080)</td>
<td>45.90</td>
<td>-0.005</td>
<td>3.26</td>
<td>-0.0569</td>
</tr>
<tr>
<td>Jonny (1080x720)</td>
<td>36.73</td>
<td>0.0083</td>
<td>3.22</td>
<td>-0.068</td>
</tr>
<tr>
<td>FourPeople (1080x720)</td>
<td>38.94</td>
<td>-0.0075</td>
<td>2.60</td>
<td>-0.078</td>
</tr>
<tr>
<td>Avg</td>
<td>41.16</td>
<td>0.003</td>
<td>2.79</td>
<td>-0.064</td>
</tr>
</tbody>
</table>
Conclusion

- We propose a new fast transcoding algorithm based on SVM
  - 41% time saving and 0.061 dB degradation compared with the benchmark cascaded solution.
- Experiments validate the proposed method
- More work need be exploited in future
Thank you!

song_li@sjtu.edu.cn
http://medialab.sjtu.edu.cn