RTSR: Enhancing Real-time H.264 Video Streaming using Deep Learning based Video Super Resolution

Team 16

20155045	20155628	20163052	20163535	20165562
Soomin Kim	Leslie Tiong	Youngki Kwon	Insu Jang	Wonyoung Lee

1. Problem

With the advent of modern filmmaking technologies and high-density storage media, we are now enjoying high-quality video with the lowest cost. In real-time video streaming applications, however, the main challenging problems are related to bandwidth limitation, storage capability, and distributed video quality.

Fig 1 illustrates the existing situation, which is related to unstable and limited communication for transmitting video to end-user. Due to the limited capability of storage, most of the video source have been downsampled and compressed. This causes the video contains unknown noise level, motion blur and so on. Hence, several works **[8-10]** have been proposed for handling this problem by introducing video Super-Resolution (SR) technique, which recovers a high-resolution video based on a low-resolution input. Through this technique, we can recover a good quality of video without limitation of network transmission.

However, the existing SR techniques for video is still not applicable for real-time application, as they are time-consuming so delay the video streaming to end-user. In order to solve this problem, we introduce a new SR model with deep learning algorithm to estimate and reconstruct high-resolution frames in real-time streaming. Our proposed model can improve the quality of video with no additional network bandwidth consumption in real time, in consideration of the properties of a video compression technique and Convolutional Neural Network (CNN).

Fig 1. Modern video streaming environment and its constraints

2. Related Work

2-1. Super resolution

Several algorithms have been presented for single-image super-resolution. Traditionally, interpolation-based approaches **[1, 2]** tried to construct a prediction function by estimating the correlation between pixels, and reconstruction-based methods were suggested where various regularization terms are utilized **[3, 4]**. More recently, various deep learning-based techniques were proposed especially via CNNs. Super-Resolution CNN (SRCNN) **[5]** was designed that directly learn the nonlinear mapping from low-resolution to high-resolution image, and Cascade of Sparse Coding based Network (CSCN) **[6]** suggested a sparse coding model that is incarnated as a neural network, and trained in a cascaded structure from end to end. Very Deep Super Resolution (VDSR) **[7]** suggest

a way to accelerate network training. It makes residual network use high learning rate in training time. VDSR is relatively more accurate and faster than other models. Furthermore, some researches adopted Generative Adversarial Networks (GANs) to tackle this problem. Super-Resolution GAN (SRGAN) **[8]** combines deeper CNNs with an extension of the perceptual loss function that consists of an adversarial loss and a content loss to recover photo-realistic textures. Those algorithms have successfully enhanced the performance of single-image super-resolution.

There are also several contributions for video super resolution, however, they did not mainly focus on real time processing, due to its high computing power requirement. **[9]** used a Bayesian approach for video super resolution, however, it cannot be done in real-time. Video super resolution technique proposed in **[10]** used CNN, but its processing time is around 10 seconds, which is not suitable for real time video super-resolution processing. **[11]** proposed real-time video super-resolution by focusing on compensating the motions, and used CNN. To make video super resolution runnable in real time, they focused on motions, and spatial and temporal dimension.

2-2. H.264

H.264 is a standard video codec that adopted several compression techniques, one of which is to store the difference of consecutive frames. Compressed frames are grouped as a Group of Pictures (GOP), which consists of the following three kinds of frames: I-frame, B-frame, P-frame. An I-frame consists of the whole information of the frame. However, B-frame and P-frame does not have the whole information, but restore it by referencing previous or next frames. Each GOP must have at least one I-frame. Fig 2 shows an example of a GOP.

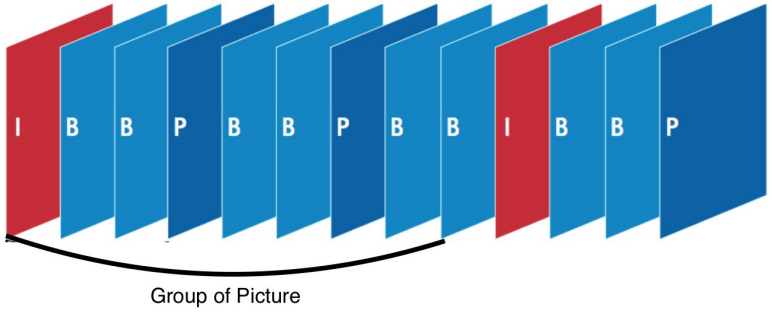

Fig 2. A sample of group of pictures. The GOP has one I- and three (B-, B-, P-) group frames.

The length of GOP does heavily affects the quality of video. For example, if an I-frame is corrupted during transmission, all following frames within the same GOP cannot be decoded due to lack of I-frame, so the GOP's length is an important factor for video streaming. According to the paper **[12]**, the GOP's length should be 5~8 to get the best quality in LTE environment. One of famous streaming site, Youtube, recommends users to upload their video with GOP less than 15 to keep their quality good. Moreover, one of famous video streaming protocols, Apple HTTP Live Streaming (HLS), also recommends to use the GOP with the length between 12 and 30.

3. Approach

3-1. Super resolution based codec architecture

One of the effective ways to enhance the video quality while using small amount of network bandwidth is to use video super-resolution technology. However, as the period between two adjacent frames is less than 35ms for 30fps videos, existing super resolution techniques, which take more than 100ms for a single image, cannot be applied to real-time video streaming.

Our approach exploits the structure of H.264, GOP. It does not perform super resolution for all frames, but only I-frame, because the frequency and quality of I-frames do heavily affect the quality of whole video, as other B- and P-frames reference them. Fig 3 represents the difference between the previous works and ours.

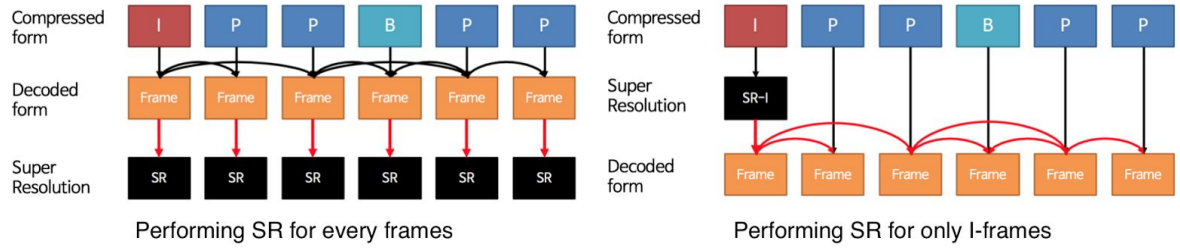

Fig 3. Data flow in decoding with super resolution for every frames (left) and I-frames (right)

In the existing works, they first decode to extract all image frames from the video, and perform super-resolution for all of these. However, in our approach, we perform super-resolution whenever the decoded frame is an I-frame. As following P- and B-frames require the information from the I-frame to be decoded, the decoding order must and always starts from I-frame. After getting super-resolution applied I-frame, we can continue video decoding by referencing this I-frame, so the effect of super-resolution is propagated to all frames.

There might be a delay for I-frame super resolution, because the following P- and B-frames cannot be decoded before super-resolution for I-frame is finished. To eliminate this delay, we make a forehead window to generate super-resolution applied I-frame in advance. For example, when a codec decodes an I-frame, an additional thread finds the next I-frame and performs super resolution, so when the main thread tries to decode this frame, there will be no delay.

Current video super-resolution techniques perform super resolution for every frames so highly compute intensive, so cannot be applied in real-time video streaming. In our approach, even without performing super resolution for all frames, however, the effect of super resolution is propagated to all frames. As P- and B-frames use the image data in the I-frame, they also show the result of super resolution without performing an additional computation. Also, typical recommended GOP size for video streaming is 15~30, which means the interval between two adjacent I-frames is 500ms ~ 1s for 30fps video, so that it makes the super resolution technique be available in real-time.

3-2. Deep learning based super resolution

We utilize VDSR [7], for i-frame super resolution in our method. The VDSR's main algorithm is utilizing the residual network for learning image detail for high-resolution image. The system takes an interpolated (for user-defined size) low-resolution image as an input and predicts image details for high-resolution output. Image details are trained by residual image data, which is calculated by subtracting interpolated low-resolution image from the high-resolution image in the dataset. Normally, deeper networks are better for complicated mapping learning. However, deeper networks bring about higher training error. Therefore, VDSR utilizes the residual network, so that the network keeps low training error with fast convergence rate as well. The network structures and details are explained in Section 5.

4. Implementation

We used PyTorch for a machine learning framework and VDSR implementation [14] running on it. For a H.264 codec, we modified a famous codec FFmpeg library [15]. We used a simple video player that uses FFmpeg as a decoder to show the video [16]. Fig 4 represents our implementation.

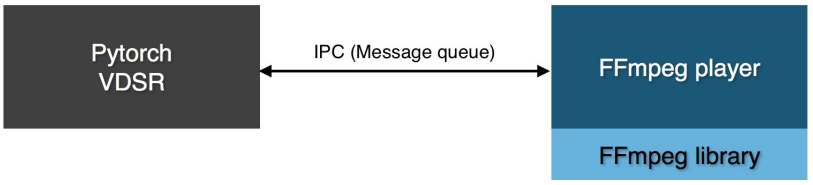

Fig 4. Implementation using Inter-Process Communication for Super Resolution

PyTorch and the video player are running as separate processes, and communicate with inter-process communication for transferring I-frame between them.

FFmpeg codec is modified to transfer I-frame data and asks PyTorch to perform super resolution for this data. During PyTorch is doing it, FFmpeg library keeps doing decoding the next P- and B-frames. When it receives the result of super-resolution from PyTorch, FFmpeg saves it in a temporal storage and uses it in a near future.

5. Experimental Setup

The network structure is summarized in Fig 5. It is based on VGGNet. The network consists of 20 Convolutional layers and ReLU layers and attaches skip connection to learn residual only. All layers except Input Low Resolution(ILR) layer and a High Resolution(HR) layer are same.

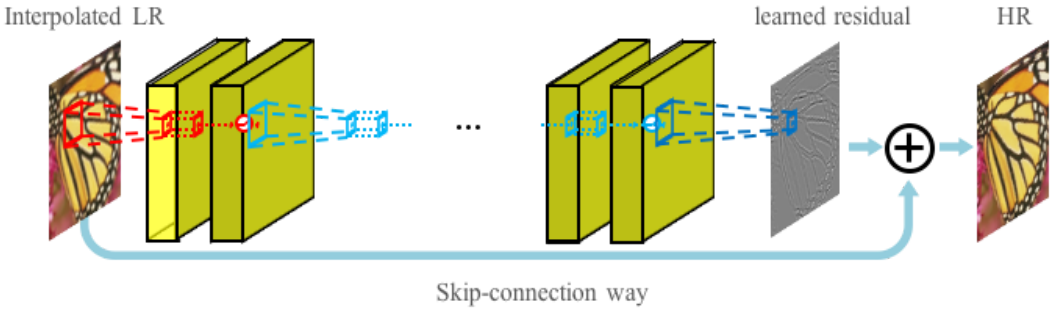

Fig 5. VDSR network structure [7]

For training, total 291 images are used for residual image learning. 91 images from Yang et al. [13] and 200 images from Berkeley Segmentation Dataset. Data augmentation is used for better training.

We compared our RTSR framework with the method which is applying super-resolution on every frame discussed in section 3 regarding video quality and time delay in video processing. We use several 24fps video which GOP size is 12 as experiment data. We have done all of our experiments in the system with Intel Xeon E2620 v3 CPU, 64GB DDR4 DRAM, and NVIDIA Geforce GTX 1080.

6. Result

This section shows the experimental results of our proposed approach in the real-time constraint. We selected two criteria: Peak Signal-to-Noise Ratio (PSNR) and time-delayed measurement to evaluate our approach. PSNR is commonly used to measure the quality of reconstruction of lossy compression codecs. The other is delayed video time when the existing works are used in real-time for super resolution, to verify how their super resolution policies are not suitable for real-time.

Fig 6: The PSNR comparison of high-resolution original video (left), our proposed approach (middle), and downsampling real-time video used in our experiment (right)

Fig 6 shows the reconstruction result and corresponding reference image. The PSNR result of our reconstructed image and corresponding reference image are 14.95 and 13.59, respectively; our result presents better result in terms of PSNR.

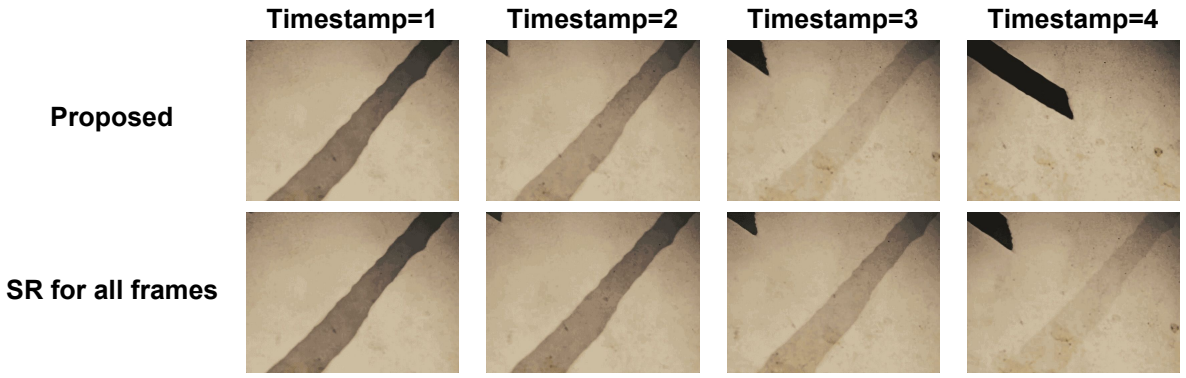

Fig 7: The time delay comparison of encoded video of our approach and the way to super-resolution on every frame

Moreover, we conducted another experimental result to measure the time delay in the real-time environment. The captured sequences of the video were illustrated in Fig 7, which showed that our proposed approach has virtually no delay, compared to other approaches that super-resolution is applied to every frame. In addition, Fig 8 demonstrated our approach in real-time video streaming. The result showed that our approach is fit to a real-time constraint. However, the referenced approach had 38% time delay.

Fig 8: Real-time reconstruction result (left) and corresponding reference image (right)

7. Conclusion

We confirmed that our proposed approach that applied SR with deep learning is applicable in a real-time streaming application. This is because our approach is able to reconstruct high-resolution videos without time-delayed in real time streaming, by applying SR technique to I-frames instead of every frame during the streaming process. However, the limitation of our approach is depending on the original video sources to reconstruct the high-resolution video. Due to the low-quality video sources, SR technique does not guarantee to recover the detailed shapes of the objects in the video.

References

[1] Li, Xin, and Michael T. Orchard. "New edge-directed interpolation." *IEEE transactions on image processing* 10.10 (2001): 1521-1527.

[2] Zhang, Lei, and Xiaolin Wu. "An edge-guided image interpolation algorithm via directional filtering and data fusion." *IEEE transactions on Image Processing* 15.8 (2006): 2226-2238.

[3] Sun, Jian, Zongben Xu, and Heung-Yeung Shum. "Gradient profile prior and its applications in image super-resolution and enhancement." *IEEE Transactions on Image Processing* 20.6 (2011): 1529-1542.

[4] Marquina, Antonio, and Stanley J. Osher. "Image super-resolution by TV-regularization and Bregman iteration." *Journal of Scientific Computing* 37.3 (2008): 367-382.

[5] Dong, Chao, et al. "Learning a deep convolutional network for image super-resolution." *European Conference on Computer Vision*. Springer International Publishing, (2014).

[6] Wang, Zhaowen, et al. "Deep networks for image super-resolution with sparse prior." *Proceedings of the IEEE International Conference on Computer Vision*. (2015).

[7] Kim, Jiwon, Jung Kwon Lee, and Kyoung Mu Lee. "Accurate image super-resolution using very deep convolutional networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[8] Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." *arXiv preprint arXiv:1609.04802* (2016).

[9] Liu, Ce, and Deqing Sun. "A Bayesian approach to adaptive video super resolution." *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011.

[10] Kappeler, Armin, et al. "Video super-resolution with convolutional neural networks." *IEEE Transactions on Computational Imaging* 2.2 (2016): 109-122.

[11] Caballero, Jose, et al. "Real-Time Video Super-Resolution with Spatio-Temporal Networks and Motion Compensation." *arXiv preprint arXiv:1611.05250* (2016).

[12] Zulpratita, Ulil S. "GOP length effect analysis on H. 264/AVC video streaming transmission quality over LTE network." (2013).

[13] Yang, Jianchao, et al. "Image super-resolution via sparse representation." *IEEE transactions on image processing* 19.11 (2010): 2861-2873.

[14]. Github. Pytoch-VDSR. [Online]. https://github.com/twtygqyy/pytorch-vdsr

[15]. Github. FFmpeg. [Online]. https://github.com/FFmpeg/FFmpeg

[16]. Github. FFmpeg player. [Online] https://github.com/Akagi201/ffmpeg-player