CSE 676 Deep Learning
Project Report
Super SloMo

Sen Lin
senlin@buffalo.edu

Chong Tian
chongtia@buffalo.edu

Xinyi Song
xinyison@buffalo.edu

Abstract

In order to resolve the contradiction between the demand for high frame-rate qualified video and the inability of ordinary devices to meet the recording requirements, we build a system which can generate intermediate frame(s) to form both spatially and temporally coherent video sequences. Inspired by the CVPR 2018 spotlight paper [1]. This project implemented a end-to-end network model to do highly quality estimation of multiple intermediate frames for arbitrary-time video interpolation. Compared to the state-of-the-art methods focusing on single-frame interpolation, The methods proposed by the paper to use U-Net modeling the motion and occlusion for variable-length multi frames perform superiorly.

1 Introduction

1.1 Background

With the development of technology, the demand for high-frame-rate video is increasing. For example, FPS e-sports players rely on high-frame-rate animations to make accurate judgments and immediate responses to situations. Additionally, in the film industry, more and more directors try to express richer and more complex emotions by shooting with higher frame rates. Such as the *Billy Lynn's Long Halftime Walk* directed by Ang Lee, which used an unprecedented shooting and projection frame rate of 120 fps instead of the tranditional 24 fps, that gained great success.

However, nomal recording devices are limited in shooting high-frames video. for instance, digital camera or smart phones can only shot at 30fps or 60 fps in optimal resulotion. The newly advanced phone which can shot at high-frame-rate actually implemented through the compromise of resolution and image quality.

So, like our methods of coloring black and white photos through machine learning, the solution converting low-frame-rate videos to higher ones is ready to go.

Besides, in this project we will develop a model for video data rather than images separately, there are some concepts related to flows which enable us to get richer information and do more sophisticated work though it's challengeable. And this paper was just published last year which meant the content inside could be very cutting-edge and interesting.

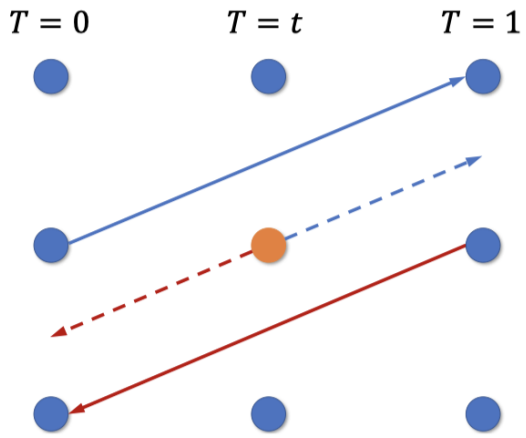

Figure 1: Illustration of intermediate optical flow approximation

1.2 Challenges

Due to the willing to interpolate variable-length multi frames at arbitrary time, the main idea is to warp the input two images to the specific time step and then adaptively fuse the two warped images to generate the intermediate image. However, there are two challenges risen naturally.

The first one is **motion interpretation**. Since in the video, almost everything are moved in-between two frames caused by the moving camera or the objects. So we need to find a way to determine which a specific pixel will be at the arbitrary time point. For example, it could be shifted of spinned to the neighbor locations. Thus we need to capture the motions.

The second one is **occlusion reasoning**. As we said before, videos are always in-moving. So the appearance of each pixel is a kind of probability with Bernoulli distribution, which also needs modeling jointly in order to estimate the arbitrary-time intermediate frames accurately.

2 Related Work

2.1 Optical Flow

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the motion between an observer and a scene [2] [3]. Optical flow can also be defined as the distribution of apparent velocities of movement of brightness pattern in an image [4].

It is kind of way to represent pixel movement.

2.2 Intermediate Time Optical Flow Synthesis

Consider the optical flow for one pixel in-between two reference frames are show in Figure.1

For a 2D+t dimensional case a voxel at location (x, y, z) with intensity $I(x, y, z)$

Assuming that the optical flow field is locally smooth, then we can synthesize the optical flow like complimentary filter [5] with co-efficient t. Specifically, $\hat{F}_{t\to1}$ can be approximated as:

$$\hat{F}_{t\to1} = (1-t)F_{0\to1} \tag{1}$$

OR

$$\hat{F}_{t\to1} = -(1-t)F_{1\to0} \tag{2}$$

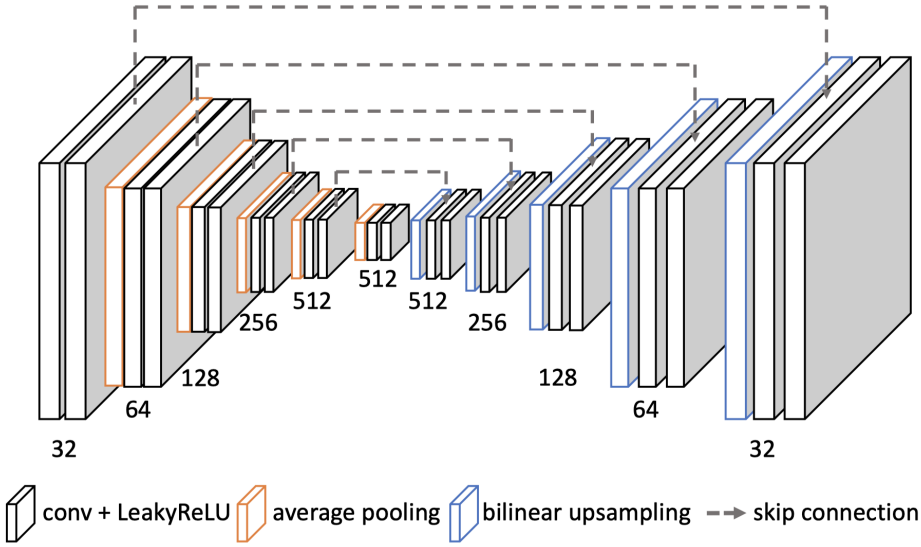

Figure 2: U-Net Architecture in This Project

Similarly, $\hat{F}_{t\to 0}$ can be approximated as:

$$\hat{F}_{t\to 0} = -tF_{0\to 1} \tag{3}$$

OR

$$\hat{F}_{t\to 0} = tF_{1\to 0} \tag{4}$$

Combining (1)(2)(3)and(4), we have:

$$\hat{F}_{t\to 0} = -(1-t)tF_{0\to 1} + t^2F_{1\to 0} \tag{5}$$

$$\hat{F}_{t\to 1} = (1-t)^2F_{0\to 1} - t(1-t)F_{1\to 0} \tag{6}$$

2.3 Backward Warping Function

Backward Warping Function is implemented by bi-linear interpolation with optical flow and reference frame. the target frame is generated by grid sampling the pixels of reference frame by the one-direction optical-flow from $T = t$.

2.4 U-Net

U-Net is one type of convolutional neural network models which was first introdued by the paper [6] written by Olaf Ronneberger et al for biomedical image segmentation. This net model got its name by its shape of "U" as shown in Figure.2. Based on the tranditional fully convolutional network model [7], some "skip connections" are added between the down-sampling part and the up-sampling part. Benefited from the end-to-end structure and skip connections, U-Net has shown a big progress in segmentation field which raised research's interests to apply it in other application. Consquently, this paper [1] purposed to use U-Net to model and compute the bi-directional optical flows between any two frame images.

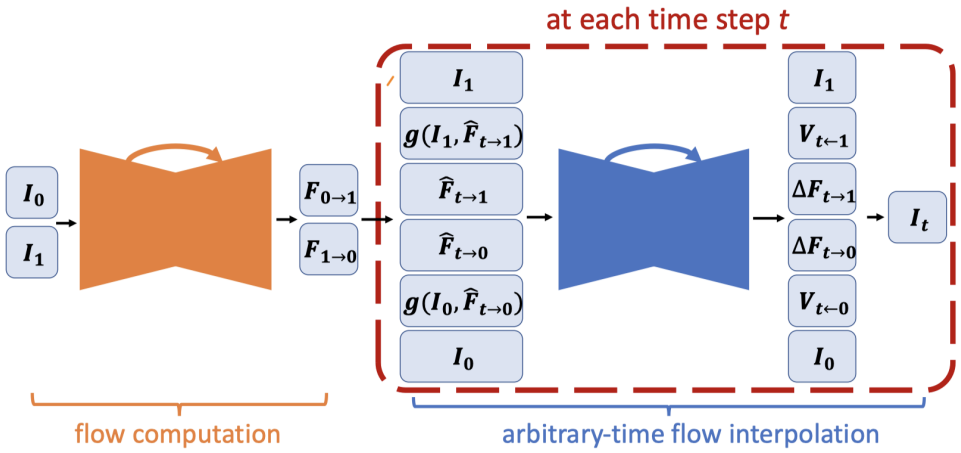

Figure 3: Model Architecture Overview

3 Model Overview

The architecture of our whole model is shown in Figure.3. As we can see, there are two part mainly both associated with an individual U-Net model.

In general, it's a computational graph with inputs of two reference frames and an arbitrary intermediate time t indicated the position in time domain to interpolate the generated frame and outputs such intermediate generated frame.

Details will be covered in the next section.

4 Detailed Design

4.1 Optical Flow Computation

Instead of tranditional linear ways to calcualte the optical flow, such as the Lucas–Kanade method [8], we use a U-Net to compute it. This U-Net has an input with 6 channels and output with 4 channels, because there are two RGB images I_0 and I_1 inputing and there are two channels for either optical flow in one direction where one is for vertical direction and the other is for horizontal direction.

To be noticed that one simple U-Net or FCN cannot compute the optical flows. However, with the later-phase computation and learning with our defined losses, this U-Net can indeed capture the optical flow features.

In practice, we concatenate inputs by according to the channel dimension, which will no longer state if no contradiction occurred.

4.2 Arbitrart-Time Flow Interpolation

Once we have the optical flows from $T = 0$ to $T = 1$ i.e. $F_{0\to1}$ and the one of reversed direction $F_{1\to0}$. we are able to synthesize the optical flows for $T = t$ to both directions. According to the formulas (5) and (6) in section 2.2, we can easily compute $\hat{F}_{t\to0}$ and $\hat{F}_{t\to1}$. This kind of approximation performs well in smooth region but poorly around motion boundaries. We therefore introduce another U-Net to refine the optical flows.

Besides, recalling the second challenge of occlusion reasoning talked in section 1.2, this paper introduced a tactic called **Visibility Maps** $V_{t\leftarrow0}$ and $V_{t\leftarrow1}$. $V_{t\leftarrow0}(p) \in [0, 1]$ denotes the probability

whether the pixel p is visible while moving from $T = 0$ to $T = t$. Combining the temporal consistency and occlusion reasoning, we have the interpolated intermediate frame $\hat{I}_t$ as:

$$\hat{I}_t = \frac{1}{Z} \odot [(1-t)V_{t\leftarrow 0} \odot g(I_0, F_{t\rightarrow 0}) + tV_{t\leftarrow 1} \odot g(I_1, F_{t\rightarrow 1})] \tag{7}$$

where $Z = (1-t)V_{t\leftarrow 0} + tV_{t\leftarrow 1}$ is a normalization factor.

These visibility maps are jointly captured in the U-Net model of Arbitrart-Time Flow Interpolation part.

More specifically, this U-Net inputs with 20 channels and outputs with 5 channels:

1. Input Channels:
   (a) I_0: 3 channels for RGB
   (b) I_1: 3 channels for RGB
   (c) $F_{0\rightarrow 1}$: 2 channels for linearly synthesized optical flows from $T = 0$ to $T = 1$ vertically and horizontally.
   (d) $F_{1\rightarrow 0}$: 2 channels for linearly synthesized optical flows from $T = 1$ to $T = 0$ vertically and horizontally.
   (e) $\hat{F}_{t\rightarrow 1}$: 2 channels for linearly synthesized optical flows from $T = t$ to $T = 1$ vertically and horizontally.
   (f) $\hat{F}_{t\rightarrow 0}$: 2 channels for linearly synthesized optical flows from $T = t$ to $T = 0$ vertically and horizontally.
   (g) $g(I_0, F_{t\rightarrow 0})$: 3 channels for RGB. Generated frame with optical flow by bi-linear interpolation.
   (h) $g(I_1, F_{t\rightarrow 1})$: 3 channels for RGB. Generated frame with optical flow by bi-linear interpolation.
2. Output Channels:
   (a) $\Delta F_{t\rightarrow 0}$: 2 channels for optical flow residuals.
   (b) $\Delta F_{t\rightarrow 1}$: 2 channels for optical flow residuals.
   (c) $V_{t\leftarrow 0}$: 1 channel for visibility maps from $T = 0$ to $T = t$.

We do not directly outputs refined optical flow for $T = t$ and predict the optical flow residuals instead. Because the paper [1] metioned that training with residuals will perform better in their experiments which I hold the same opinion. Obviously:

$$F_{t\rightarrow 1} = \hat{F}_{t\rightarrow 1} + \Delta F_{t\rightarrow 1} \tag{8}$$
$$F_{t\rightarrow 0} = \hat{F}_{t\rightarrow 0} + \Delta F_{t\rightarrow 0} \tag{9}$$
$$\tag{10}$$

Moreover, it's indisputable to say that $V_{t\leftarrow 0} + V_{t\leftarrow 1} = 1$. So we only output one of them and get both them by:

$$V_{t\leftarrow 0} = \sigma(V_{t\leftarrow 0}) \tag{11}$$
$$V_{t\leftarrow 1} = 1 - V_{t\leftarrow 0} \tag{12}$$

where $\sigma(\cdot)$ is a Sigmoid function compressing the value in the range $[0, 1]$.

Finally, we can generate the estimated intermediate frame $\hat{I}_t$ at $T = t$ from this model by applying backward warping function and visibility once again.

4.3 Loss Function

Our model is an integrating model modeling both motion and occlusion. So the total losses is a combination of four tems:

$$l = \lambda_r l_r + \lambda_p l_p + \lambda_w l_w + \lambda_s l_s \tag{13}$$

4.3.1 Reconstruction Loss

Reconstruction Loss l_r models how good the reconstruction of the intermediate frames at $T = t$ with the ground truth in measured by L_1 loss:

$$l_r = \frac{1}{N} \sum_{i=1}^{N} ||\hat{I}_t - I_t||_1 \tag{14}$$

4.3.2 Perceptual Loss

Interpolated frame may be blur during the reconstruction process. So Perceptual Loss preserves details of the predictions and make interpolated frames sharper by measuring the L_2 loss between the "conv4_3" features features extracted by pre-trained VGG-16 model on both the reconstruction frame and the ground truth:

$$l_p = \frac{1}{N} \sum_{i=1}^{N} ||\phi(\hat{I}_t) - \phi(I_t)||_2 \tag{15}$$

where $\phi(\cdot)$ represents the "conv4_3" features from VGG-16 model pre-trained with ImageNet Dataset.

4.3.3 Warping Loss

Warping Loss models the quality of computed optical flow, which defined as:

$$l_w = ||I_0 - g(I_1, F_{0\to1})||_1 + ||I_1 - g(I_0, F_{1\to0})||_1 \tag{16}$$

$$+ \frac{1}{N} \sum_{i=1}^{N} ||\hat{I}_t - g(I_0, \hat{F}_{t\to0})||_1 + \frac{1}{N} \sum_{i=1}^{N} ||\hat{I}_t - g(I_1, \hat{F}_{t\to1})||_1 \tag{17}$$

4.3.4 Smoothness Loss

The last term Smoothness Loss encorage neighbor pixels to have samiliar optical flows which is very intuitive:

$$l_s = ||\nabla F_{0\to1}||_1 + ||\nabla F_{1\to0}||_1 \tag{18}$$

5 Experiments

All the implementation based on the advanced and user-friendly deep learning framework PyTorch[1] with newest api offered in version 1.1. For the hardware part, cause its highly computation load. we trained the model and convert demo videos to super-slomo one on the Google Cloud Services[2]. The configuration of instance we creaeted for our experimentations is an GPU instance with 4 vCPUs, 15 GB memory and one Nvidia Tesla P100 Graphic Card with CUDA units.

[1] https://pytorch.org/

[2] https://cloud.google.com/

Figure 4: Demo Screenshoot

For the training, we adapted the popular Adam optimizer [9] with initial learning rate of 0.001 which will be decreased by a factorof 10 every 200 epochs. We divides the Adobe 240-fps dataset[3] to be clips one of which consists 12 consecutive frames. Then we augumented data by randomly reversing the frame sequences order. Additionally, we resized each image to 360×360 and croped it to a 352×352 area as well as an perform horizontal flip randomly.

The coefficients for losses are:

1. $\lambda_r = 0.8$
2. $\lambda_p = 0.005$
3. $\lambda_w = 0.4$
4. $\lambda_s = 0.1$

We did the convertion between videos and frame images by the open-source library FFmpeg[4].

Compared to the 500-epoch training in the paper [1], we only train it 200 epochs due to the limited time. However, it perform better than the standard method.

Here, we give a short demo video[5]. We created a super slomo video from original video inputing to speed it up 3 times. i.e., interpolate 2 frames between any two frames from the original video. Then play it under same duration. The left one is the original and the right one is the super slomo generated by our trained model. Additionally, Figure.4 is a screenshoot of the demo video. And you can find the clear improvement we got.

6 Conclusion

This project creatively explored a new application field for U-Net to compute optical flow. By the end-to-end model to capture motion and occlusion, it also successfully give a method to do variable-length multi-frame video interpolation which has great prospects and practical value.

[3] http://www.cs.ubc.ca/labs/imager/tr/2017/DeepVideoDeblurring/
[4] https://ffmpeg.org/
[5] https://youtu.be/zjJfnVBU84g

Chosing this new hard project is also an challenge for us. Because this paper just publicated last year and the author didn't offer the implementation. So we need start from scratch to understand every details. But this is a truly great experience for us to explore deep learning area.

References

[1] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9000–9008.

[2] A. Burton and J. Radford, *Thinking in perspective: critical essays in the study of thought processes*. Routledge, 1978, vol. 646.

[3] D. H. Warren and E. R. Strelow, *Electronic spatial sensing for the blind: contributions from perception, rehabilitation, and computer vision*. Springer Science & Business Media, 2013, vol. 99.

[4] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[5] H.-J. Lee and S. Jung, "Gyro sensor drift compensation by kalman filter to control a mobile inverted pendulum robot system," in *2009 IEEE International Conference on Industrial Technology*. IEEE, 2009, pp. 1–6.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[8] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.