Video Super-Resolution

VIDEO SUPER-RESOLUTION

BY

LINGSHI KONG, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2018) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Video Super-Resolution
AUTHOR:	Lingshi Kong B.Sc., (Electrical Engineering) McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Chen
NUMBER OF PAGES:	ix, 55

Abstract

Video super-resolution becomes significant desire recently to provide high-resolution contents for ultra high definition displays. Recent advances in video super-resolution have shown that convolutional neural networks combining with motion compensation, which can merge information from multiple low-resolution frames, to generate high-quality frames. But it has been demonstrated that most deep learning based video super-resolution methods heavily dependent on the accuracy of motion estimation and compensation. Other than before, here proposed a different end-to-end deep neural network that inexplicit compensates motion through the generates dynamic filters. The dynamic filters are computed depending on the local spatio-temporal neighborhood of each pixel. With this approach, a high-resolution frame has reconstructed directly from the low-resolution input frames by using a series networks combining with a dynamic local filter network. The proposed network can generate much sharper high-resolution videos with temporal consistency, compared to the previous methods.

Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Jun Chen, whose expertise, and understanding added considerably to my graduate experience. I appreciate his kindness, and patience as my supervisor. And I wish to thank Mr. Xiaohong Liu, for his advice and assistance in keeping my progress on schedule.

Contents

List of Figures

Chapter 1

Introduction and Problem Statement

There are millions of images and videos shared on the Internet every day, and people could use these multimedia stream to study what they are interested in or share the unforgettable pictures with their friends and family. High-resolution seems to be getting more and more desired, mostly because people are seeking reality and vivid visual experience. That is also one of a reason that TV manufactory companies successively produce Ultra High Definition TV in today's market. But the resource that you are usually watching are not high-resolution, because they either have been compressed due to the size of limitation during the uploading or restricted by the capability of devices. The high-resolution display has widely been used on the home and mobile devices now. Thus, the way transfer the low-resolution images and videos into the high-definition version has become a considerable demand.

Image or multi-frame super-resolution is the process that given the low-resolution version to reconstruct the corresponded high-resolution version. A video is a series

of images played in sequence at a specified frame rate. Other than single-image super-resolution where the details have to be generated base on only one figure, video super-resolution suppose to have more flexibility to choose various useful information to achieve a better result. The multiple frames of data will provide more information which should be used for higher quality of up-sampling. But the large motion between several consecutive frames will increase the difficulty in allocating the corresponding objects, the tasks of correctly and simultaneously extracting and fusing the details in multiple frames to assemble a new image are hard.

Thanks to parallelization of using GPU-accelerated computing, the calculating amount of convolutional neural network can efficiently be achieved. Since the deep learning method has led to a successful improvement in performance on the task of image super-resolution [4], Kappeler et al. [1] proposed the convolutional neural network for video super-resolution. To compensate for the ill-posed problem, they use the precomputed optical flow to calculate the interframe prediction, then using that prediction information to align all input frames to the reference one through backward warping to do the compensation. The super-resolved result has been outperformed the previous non-deep learning method, but people found the precomputed optical flow method seemed not an optimal choice for video super-resolution, Tao et al. [16] showed the higher quality video super-resolution results had been obtained by improving motion compensation.

In this work, the proposed video super-resolution network has inspired by Dynamic Filter Network [3] that the motion compensation method is neither through the optical flow nor a motion estimation network. Instead of explicitly calculating and compensating the motion of adjacent input frames, the motion information in

the proposed method is implicitly utilized by generating the dynamic filters as shown in Figure 1.1. The motion compensated feature maps that used to reconstruct high-resolution output are directly constructed by the dynamically filtered input frames (Figure 4.3). This method can generate much shaper and temporally consistent high-resolution videos. It is end-to-end trainable and does not require any pre-training stages, and achieve the state-of-the-art performance compared to the previous deep learning based video super-resolution algorithms.

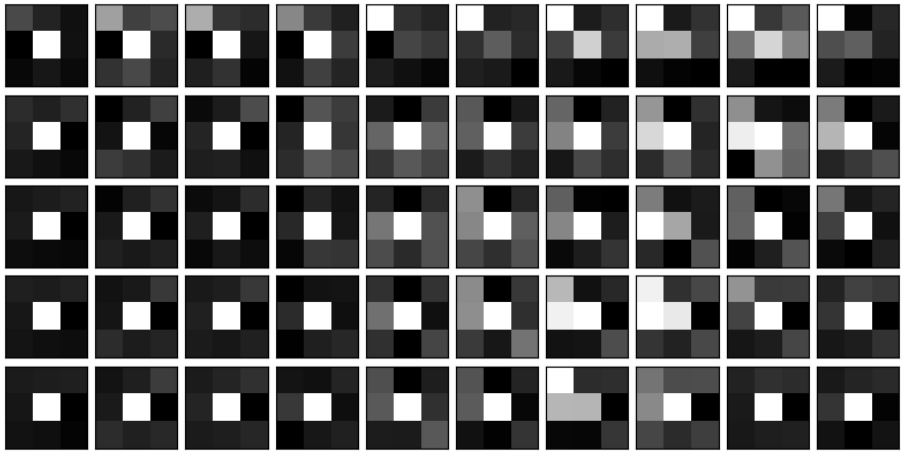

Figure 1.1: The motion information in the generated kernel.

1.0.1 Thesis Structure

In the second chapter, illustrated five video super-resolution methods based on deep learning method in detail. Some of the basic concepts of neural network and some super-resolution related components that relate to the proposed method have introduced in the third chapter. In the fourth chapter, the proposed video super-resolution

methods have explained in detail, and experimental results have demonstrated in pictures. The conclusion of this thesis is present in the fifth chapter.

Chapter 2

Previous Work

2.0.1 Real-Time Video Super-Resolution with Spatio-Temporal Networks and Motion Compensation

Caballero et al. [9] presented a real-time approach for video super-resolution based on sub-pixel convolution and spatio-temporal networks that effectively exploit temporal redundancies and improve reconstruction accuracy while maintaining fast processing speed.

The network uses three consecutive frames to combine a compensated module as inputs for super-resolution network as shown in Figure2.1. Since optical flow has been effectively encoding to describe motion for spatial transformer techniques, therefore, it is suitable for motion compensation. Optical flow represents a relationship between a new frame I_{t+1} and a reference current frame I_t. It contain two feature maps $\triangle_{t+1} = (\triangle_{t+1}x, \triangle_{t+1}y; \theta_{\triangle,t+1})$ relied on parameters $\theta_{\triangle,t+1}$ corresponding to the displacement for the x and y dimensions. Motion compensation is optical flow combine with a interpolation F that arrange resulting pixel back onto a regular grid, thus a

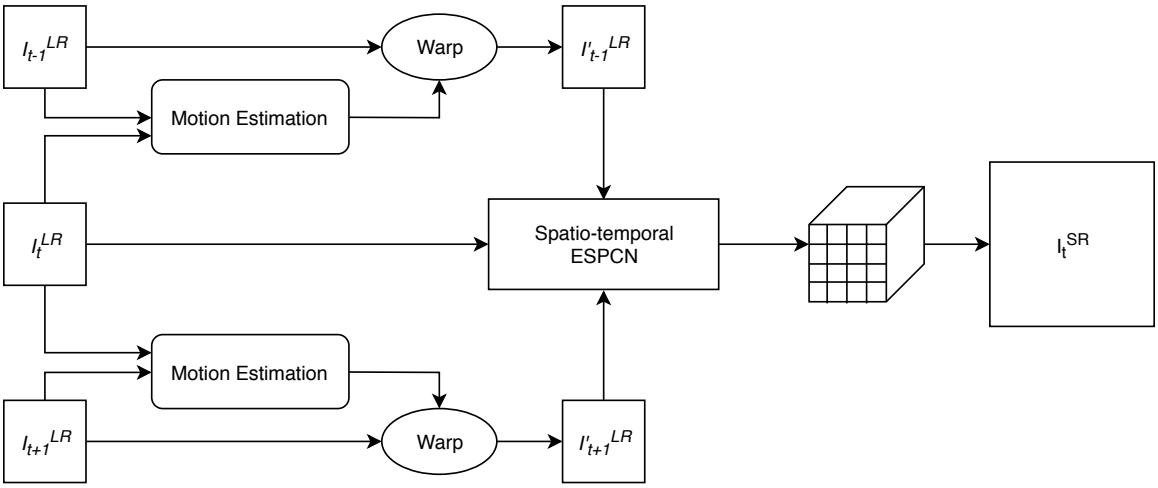

Figure 2.1: Designed video super-resolution architecture.

compensated image can be expressed as $I'_{t+1}(x, y) = F\{I_{t+1}(x + \triangle_{t+1}x, y + \triangle_{t+1}y)\}$. A schematic of the design for motion compensation is shown in Figure 2.2. By optimizing the parameters $\theta_{\triangle,t+1}$ to minimize the MSE between the transformed frame and the reference frame to train the spatial transformer to perform motion compensation. The same parameters can be used to model the motion of the outer two frames relative to the central frame.

The input of spatio-temporal networks is a block of spatio-temporal information, which is also a sequence of consecutive frames but motion compensated of original inputs. Supposed the output is a single frame, there are three types of fusion methods could be used for spatio-temporal networks. One of the most straightforward approaches is collapsed all temporal information in the first layer through matching the temporal depth of the input layer to the number of frames, and the remaining operations are identical to those in a single image super-resolution network, they called it early fusion. Another option is to merge temporal information in a hierarchical

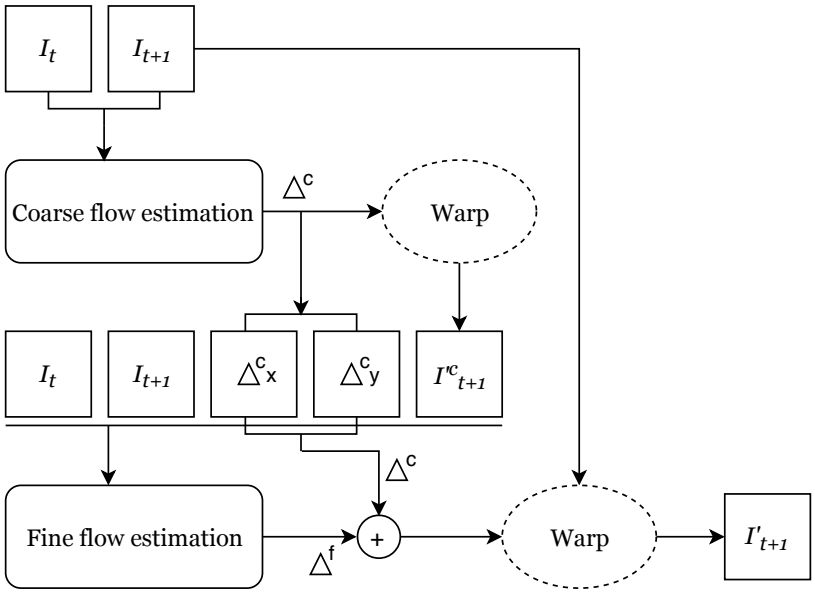

Figure 2.2: Spatial transformer motion compensation.

structure partially, so the temporal information is slowly fused through each layer of the network until the temporal depth of the last layer of the network become 1, they called it slow fusion. And the third method is using 3D convolution that force layer weights be share across the temporal dimension.

Other than the methods preprocess inputs from low-resolution to high-resolution through bicubic upsampling and do the mapping in high-resolution space, the sub-pixel convolution allows the network to process low-resolution inputs I_{LR} at low-resolution space directly. The sub-pixel convolution implies that the system could learn a better upscaling way than bicubic upsampling if well-trained network. The sub-pixel convolution will be explained in the next section. The spatial transformer and super-resolution modules are both differentiable and therefore end-to-end trainable.

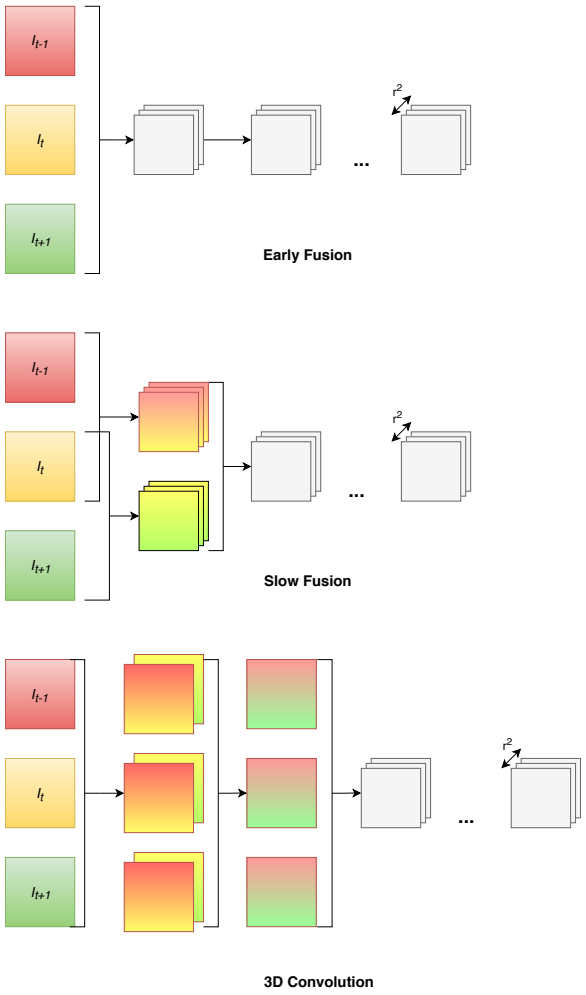

Figure 2.3: Spatio-temporal models

2.0.2 Detail-revealing Deep Video Super-resolution

Tao et al. [16] proposed a sub-pixel motion compensation layer in a convolutional neural network and achieved a high-quality result shown that proper frame alignment and motion compensation is crucial for video super-resolution. The network comprises motion estimation, motion compensation, and detail fusion; it takes a sequence of $N_F = (2n + 1)$ low-resolution images as input to produce a high-resolution image corresponds to center reference frame of inputs.

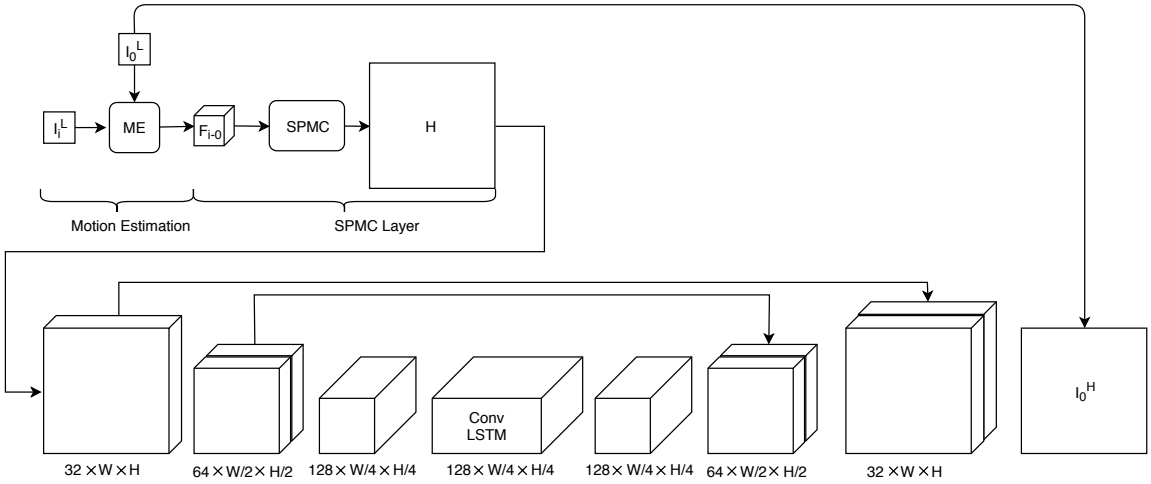

Figure 2.4: Convolutional neural network for video super-resolution with sub-pixel motion compensation layer.

The network using motion compensation transformer module to perform motion estimation, then using a proposed layer called sub-pixel motion compensation(SPMC) layer simultaneously achieve sub-pixel motion compensation and resolution enhancement through utilized sub-pixel information from motion. It is defined as:

$$I^H = Layer_{SPMC}(I^L, F; \alpha) \tag{2.1}$$

where I^L and I^H denote input in low demension and output in high dimension respectively, F denotes optical flow used for transposed warping and α denotes the scaling factor. And the layer contains two submodules, which are sampling grid generator and differentiable image sampler.

In sampling grid generator, transformed coordinates are first calculated according to estimated flow as:

$$\begin{pmatrix} x_p^s \\ y_p^s \end{pmatrix} = W_{F;\alpha} \begin{pmatrix} x_p \\ y_p \end{pmatrix} = \alpha \begin{pmatrix} x_p + u_p \\ y_p + v_p \end{pmatrix} \tag{2.2}$$

where p denotes pixels' indexes in low-resolution image space. x_p and y_p denote the two coordinates of p. $W_{F;\alpha}$ denotes the operator of coordinates transformer. u_p and v_p denote estimated flow vector from flow field $F = (u, v)$. x_p^s and y_p^s denote transformed coordinates in high-resolution image space.

In differentiable image sampler, the output image I_q^H is constructed in high-resolution image space according to x_p^s and y_p^s as:

$$I_q^H = \sum_{p=1} I_p^L M(x_p^s - x_q) M(y_p^s - y_q) \tag{2.3}$$

where q denotes pixels' indexes in high-resolution image space. x_q and y_q denote the two coordinates of q. $M(\cdot)$ denotes the sampling kernel, which defines the image interpolation methods such as bicubic interpolation or bilinear interpolation etc.

Due to the property of forwarding warping and zero-upsampling, I^H is sparse, and the majority of the pixels are zero-valued. It requires the network to have large receptive fields to capture image patterns in I^H. Using simple interpolation to fill these

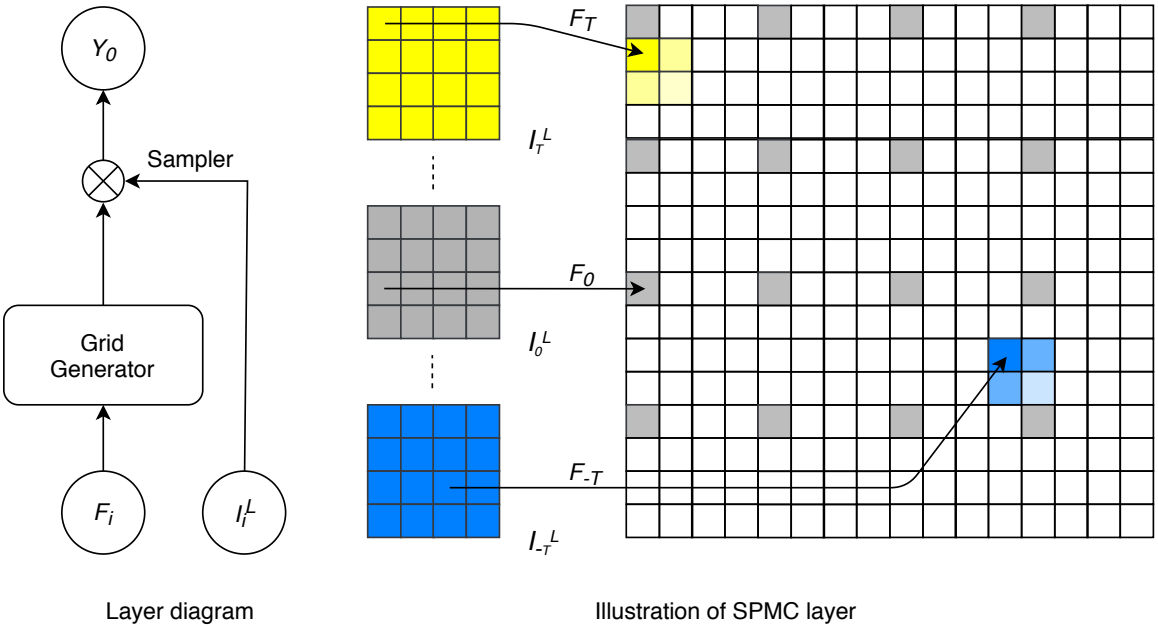

Figure 2.5: Sub-pixel motion compensation layer.

holes is not a good solution because interpolated values would dominate during training. A detail fusion net base on encoder-decoder architecture and skip-connections was designed to overcome the issue.

In the detail fusion net, the encoder part reduces the size of the constructed high-resolution image makes the feature maps less sparse leading to minimizing computation cost, which means it doesn't need a very deep network. A ConvLSTM module [17] is inserted in the middle stage as a natural choice for sequential input because of the property of selectively remembering patterns for long durations of time. The skip-connection are used for training acceleration.

The convolutional neural network framework effectively incorporates the sub-pixel motion compensation layer, and the fusion net is an end-to-end trainable network. It can generate visually, and quantitatively high-quality results compare to previous works.

2.0.3 End-to-End Learning of Video Super-Resolution with Motion Compensation

Compared to the previous convolutional neural network for video super-resolution methods, the main difference is Makansi et al. [14] proposed an operation that combined warping and upsampling operations, which named Joint Upsampling and Backward Warping($JUBW$) further proved the benefit from optical flow. The procedure is strongly related to Tao et al. [16] proposed the sub-pixel motion compensation layer that performs forward warping and upsampling jointly. Other than sub-pixel motion compensation layer has sub-pixel interpolation for the target position in the high-resolution grid and insertion between values. In Joint Upsampling and Backward Warping, if multiple flow vectors point to the same target location, the $JUBW$ only outputs sub-pixel distances, which means it doesn't perform any simple interpolation at all. $JUBW$ let the network itself finding a meaningful insertion.

Let x_p^s and y_p^s denote the coordinates in high-resolution space relate to a pixel p, while x_p and y_p denote the source coordinates in low-resolution space. First, using high-resolution flow estimations (u_p, v_p), which used to warp all frames to the center frame, to compute mapping from high to low-resolution space according to the following equation as:

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} = \frac{1}{\alpha} \begin{pmatrix} x_p^s + u_p + 0.5 \\ y_p^s + v_p + 0.5 \end{pmatrix} - \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} \tag{2.4}$$

where α denotes the scaling factor. Subtraction and addition of 0.5 place the origin at the top left corner of the first pixel. Then computing the warped images as:

$$I_w(p) = \begin{cases} I(\lfloor x_p \rceil, \lfloor y_p \rceil) & \text{if } \lfloor x_p \rceil, \lfloor y_p \rceil \text{ is inside I,} \\ 0 & \text{otherwise,} \end{cases} \tag{2.5}$$

where $\lfloor \cdot \rceil$ denotes the round to nearest operation. But it is no interpolation between pixels. The operation then additionally outputs the following distances per pixel as:

$$\begin{pmatrix} d_p^x \\ d_p^y \end{pmatrix} = \begin{pmatrix} \lfloor x_p \rceil - x_p \\ \lfloor y_p \rceil - y_p \end{pmatrix} if \ \lfloor x_p \rceil, \lfloor y_p \rceil \ is \ inside \ I \ and \ \begin{pmatrix} 0 \\ 0 \end{pmatrix} \ otherwise \tag{2.6}$$

The whole idea about the network is upsampling the low-resolution inputs through bicubic interpolation in the beginning, then feed into flow estimation module FlowNet2-SD [7], which can be learned end-to-end with a convolutional neural network, to get the flow information in high-resolution space. Using flow information and original low-resolution inputs as the inputs for joining upsampling and backward warping operation to get dense images. Stacking all frames feed into the convolutional neural network to perform the fusion interpolation. Here used the fusion module proposed by Tao et al. [16] for comparison. The architecture of the network is shown in Figure 2.7.

Compared to the sub-pixel motion compensation layer, joint upsampling and backward warping layer achieved a better result. Makansi et al. concluded from that video super-resolution should generally avoid any interpolation and leave it to the network. Besides, including sub-pixel distances has gained a small additional improvement.

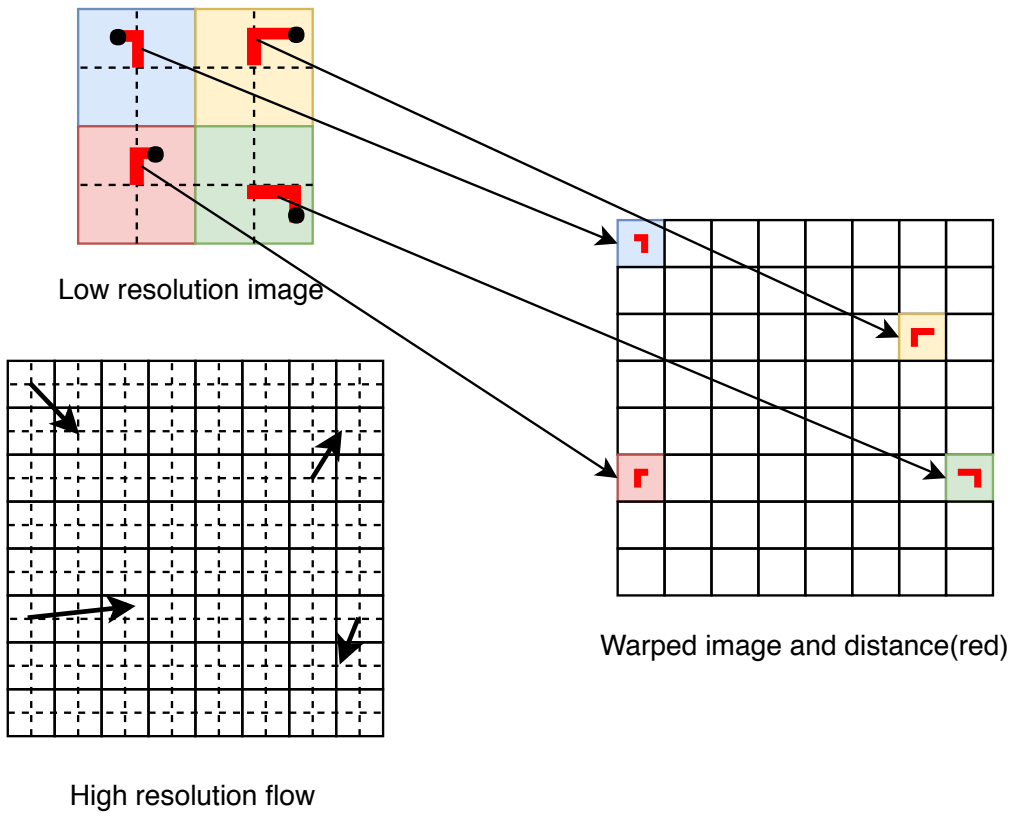

Figure 2.6: Illustration of the Joint Upsampling and Backward Warping operation.

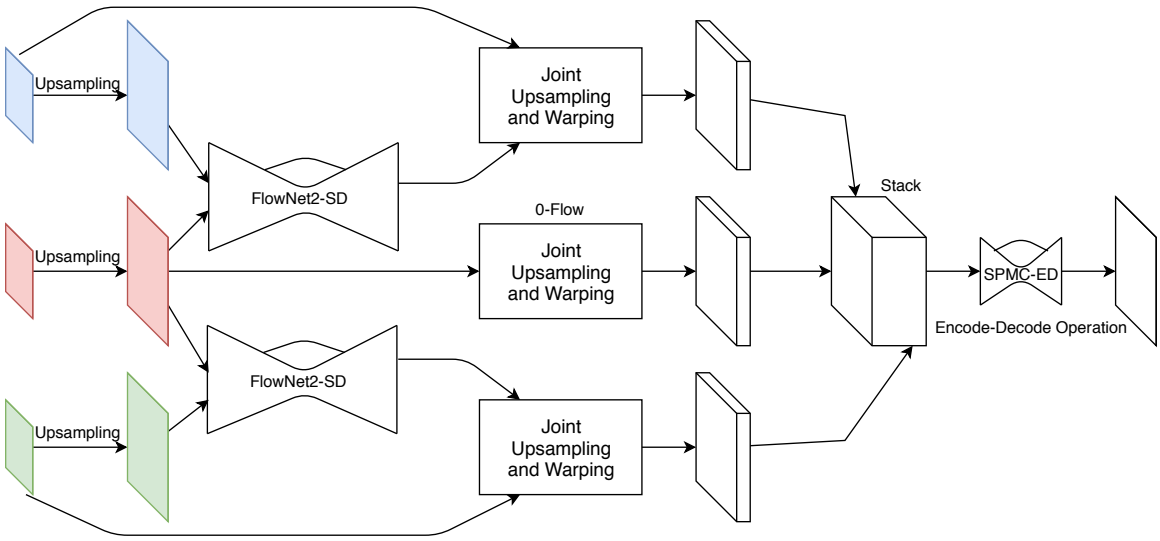

Figure 2.7: Network architecture.

2.0.4 Robust Video Super-Resolution with Learned Temporal Dynamics

Liu et al. [6] proposed a model combined with a spatial alignment network and a temporal adaptive neural network to deal with complex motion and effectively and efficiently utilize temporal information. The spatial alignment design reduced the complexity of motion between consecutive frames thus increased robustness of image alignment with complex movement. The temporal adaptive design demonstrated a clear advantage in handling complex motion over other methods using fixed length temporal filters.

In the primary flow, the network takes consecutive low-resolution inputs as input into a spatial alignment network to do the motion estimation, through takes spatial transform of the adjacent frames to produce low-resolution aligned frames with

the reference frame. After that, many aligned low-resolution frames as inputs are brought into a temporal adaptive neural network to apply several filters of different temporal sizes to generate multiple high-resolution frame estimates. The resultant high-resolution forecast is adaptively aggregated according to motion information.

In spatial alignment network, two low-resolution frames stacked together feed into a localization network to predict the spatial transform parameter $\hat{\theta}_{ST}$. The localization network only needs to infer two translation parameters, which are the estimation of the integer translation along the horizontal direction by rounding the average horizontal displacement of all pixels in a patch, and the vertical direction respectively. This scheme termed rectified optical flow alignment, which simplifies the motion in patch level to integer translations for avoiding interpolation which may cause blur or aliasing. After that, applying the spatial transform parameter $\hat{\theta}_{ST}$ to the source frame in the spatial transform layer, to produce the low-resolution aligned frame through minimizing the mean squared error between $\hat{\theta}_{ST}$ and the ground truth θ_{ST}. Finally, the low-resolution reference frame and all the resultant aligned neighboring frames from this network are used together as input to the temporal adaptive network. The architecture of the spatial alignment network cascade with the temporal adaptive network is shown in Figure 2.8.

In temporal adaptive neural network, the network has multiple super-resolution inference branches $\{B_i\}_{i=1}^{N}$ with different temporal scale i. The branch B_i is the branch works on a $2i-1$ aligned frames from the output of spatial alignment network as input and uses its temporal dependency on its scale through a convolutional neural network to predict a high-resolution estimation. And an extra-temporal modulation branch T is to determine the pixel-wise aggregation weights and adaptively combine

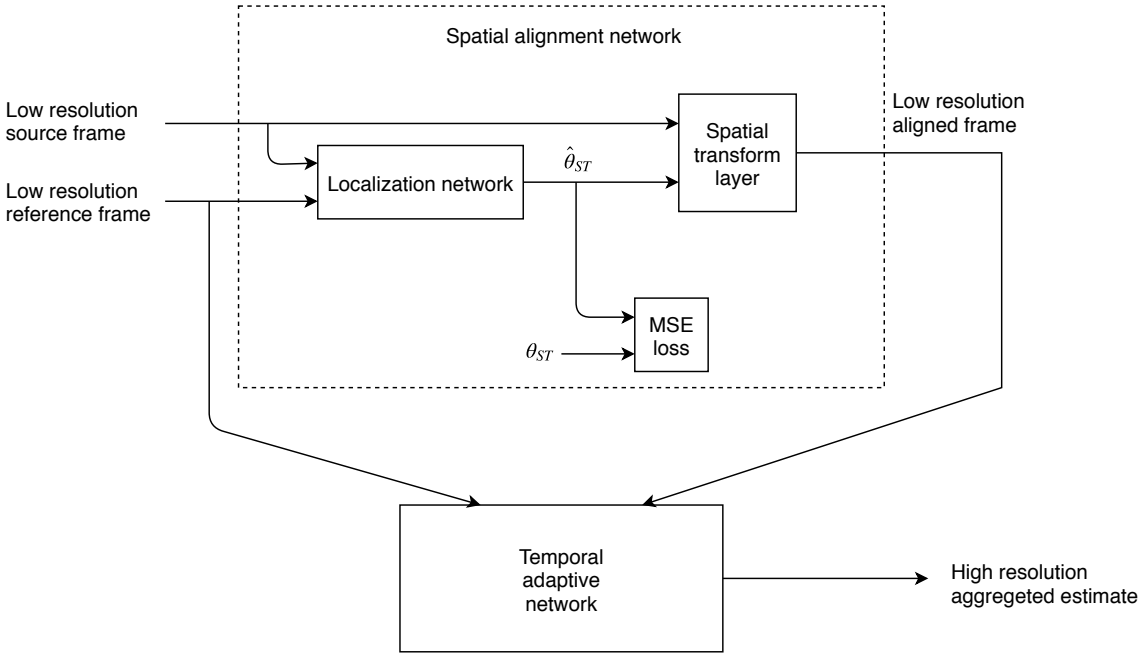

Figure 2.8: The architecture of spatial alignment network cascade with temporal adaptive network.

all the high-resolution estimates based on motion information. For a model of N super-resolution inference branches, the temporal modulation branch takes $2N-1$ consecutive frames as input to outputs the pixel-level weight maps on all N possible temporal scales. The final estimated high-resolution frame is aggregated from the estimates from all super-resolution inference branches and multiplied with its corresponding weight maps from temporal modulation branch. All super-resolution inference branches and the temporal modulation branch can be trained in the same neural network. The architecture of the temporal adaptive network is shown in Figure 2.9.

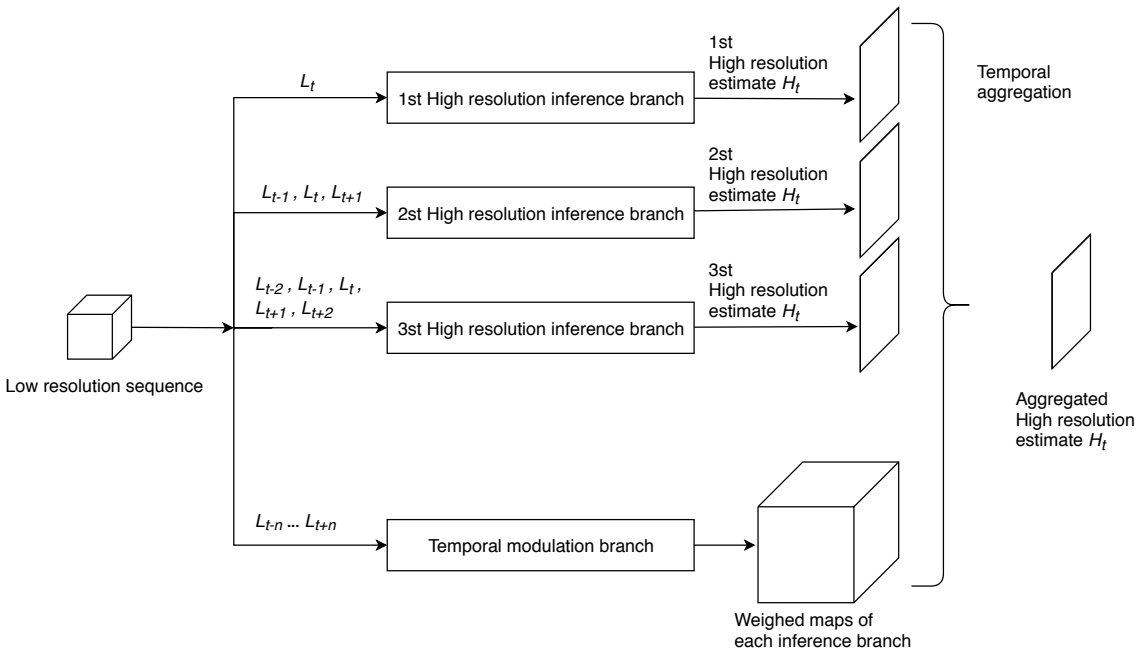

Figure 2.9: Network architecture.

Spatial alignment network and temporal adaptive neural network are joint training through minimize the weighted sum of the mean square loss of them, thus the whole

network is end-to-end fashion.

2.0.5 Frame-Recurrent Video Super-Resolution

Previously, video super-resolution methods generate a single high-resolution frame through processing a batch of low-resolution inputs, repeat this processing until made every frame of entire video. The computational cost is relatively high because each input frame is processed and warped multiple times. And each output frame is estimated independently conditioned on the input frames that limited temporal consistency of results. To minimize the computational cost and efficiently enhance temporal consistency results, Sajjadi et al.[13] proposed a frame recurrent video super-resolution network.

The framework of frame recurrent video super-resolution network is using low-resolution of the current input frame I_t^{LR}, the previous low-resolution input frame I_{t-1}^{LR} and high-resolution of the previous estimate I_{t-1}^{est} to produce the current high-resolution estimate I_t^{est}. There are two trainable components included in the network, which are the optical flow estimation network FNet and the super-resolution network SRNet. The architecture of frame recurrent video super-resolution network is shown in Figure 2.10.

Firstly, taking the low-resolution previous input I_{t-1}^{LR} and current input I_t^{LR} into the optical flow estimation network FNet to estimate the low-resolution flow map F^{LR}, which assigns a position in I_{t-1}^{LR} to each pixel location in I_t^{LR} and normalize it as:

$$F^{LR} = FNet\left(I_{t-1}^{LR}, I_t^{LR}\right) \in [-1, 1]^{H \times W \times 2} \tag{2.7}$$

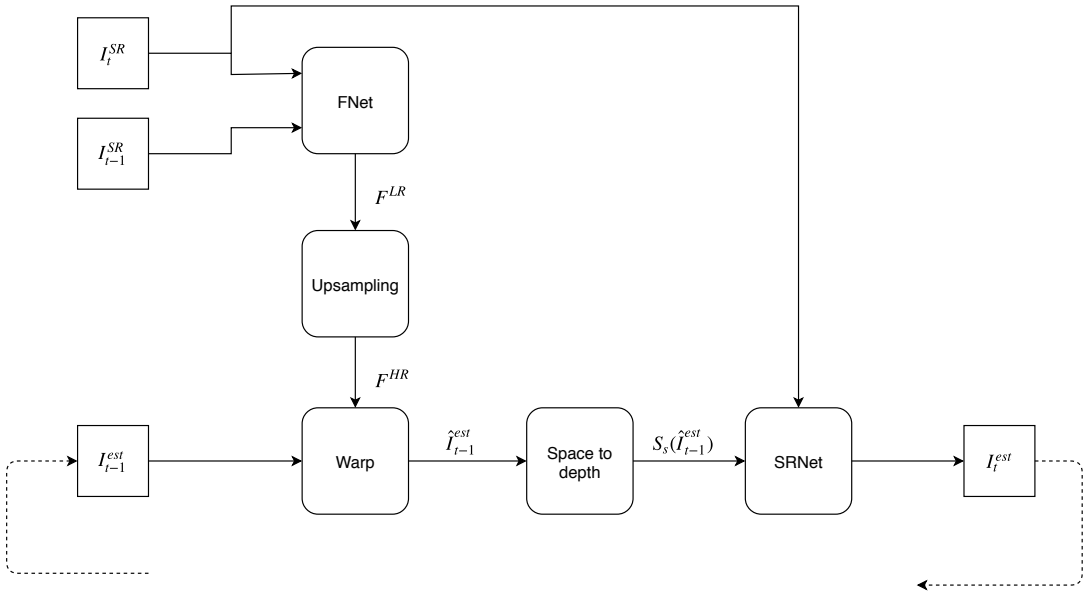

Figure 2.10: The architecture of frame recurrent video super-resolution network.

Then using bilinear interpolation to upscale the low-resolution flow map F^{LR} ti high-resolution flow map F^{HR} with scale s and using the flow map F^{HR}, which is the optical flow from the previous frame onto the current frame, to warp the previously estimated image I_{t-1}^{HR} as:

$$F^{HR} = UP\left(F^{LR}\right) \in [-1, 1]^{sH \times sW \times 2} \tag{2.8}$$

$$\hat{I}_{t-1}^{est} = WP_{bilinear}\left(I_{t-1}^{est}, F^{HR}\right) \tag{2.9}$$

Next, using a space-to-depth transformation operator to map warped previous estimated image $\hat{I}_{t-1}^{est}$ to low-resolution space as:

$$S : [0,1]^{sH \times sW \times C} \rightarrow [0,1]^{H \times W \times s^2 C} \tag{2.10}$$

The space-to-depth transformation operator is the inverse of Shi et al[15] pixel shuffle convolution layer, and I will explain it in the next chapter.

Finally, concatenating the low-resolution mapping of the warped previous estimated image with the current low-resolution input frame $I_t^{LR} \oplus S_s(\hat{I}_{t-1}^{est})$ in channel dimension, and take it into the super-resolution network SRNet.

There are two loss terms used for training, and using mean square error to minimize them. The loss of motion compensation is calculating the error between warped low-resolution input $WP_{bilinear}(I_{t-1}^{LR}, F^{LR})$ and low-resolution input I_t^{LR} due to there is no ground truth optical flow. The loss of super-resolution network is calculating the estimated output with the ground truth then backpropagated through both SRNet and FNet.

Chapter 3

Related Work

3.0.1 Activate Function

The convolutional neural network has achieved great success to outperform traditional methods in various computer vision tasks, such as image classification, object detect and super-resolution. Despite increased the depth of networks, one of the critical characteristics of the modern deep learning system is to use non-saturated activation function (e.g., ReLU) to replace saturated activation function (e.g., sigmoid, tanh).

There is two main advantage of using a non-saturated activation function. The first advantage is to solve the so-called exploding/vanishing gradient. The second advantage is the convergence speed. In all of these non-saturated activation functions, the most notable one is rectified linear unit (ReLU). Briefly speaking, it is a piecewise linear function which let the negative part to be zero and retains the positive part.

$$ReLU(x) = \begin{cases} a & \text{if } x \geq 0, \\ 0 & \text{if } x < 0, \end{cases} \tag{3.1}$$

ReLU is the reduced likelihood of the gradient to vanish. When $x \geq 0$. In this regime, the gradient has a constant value. Even there is a way for saturated activation function like Sigmoid to compensate vanishing gradient by proportionally increasing learning rate; the computational cost is too high. Normally, in training of neural networks, the data are normalized to have mean 0 and variance 1. In a sigmoid function the bigger absolute value of the input the smaller the gradient has been received, then the network refuses to learn further or is drastically slow. If there have many layers, the output need multiplication of these gradients, and the product of many smaller than one values goes to zero very quickly. The constant gradient of ReLU results in faster learning.

It has a desirable property that the activations are sparse after passing ReLU. Sparsity arises when $x < 0$. The more such units that exist in a layer, the more dispersed the resulting representation. Sigmoid, on the other hand, is always likely to generate some non-zero value resulting in dense representations. Sparse representations seem to be more beneficial than dense representations.

Unfortunately, ReLU units can be fragile during training and can die (output zero). For example, the gradients will be zero after one negative value or negative weight has been inputted to the ReLU function or a large negative bias term is learned. The ReLU will hence output the zero value for almost all of the activation functions flowing through the unit in the model.

In contrast to ReLU, in which the negative part is dropped, Leaky Rectified Linear Units (Leaky ReLU) are ones that have a very small gradient instead of a zero gradient when the input is negative, giving a chance for the net to continue its learning. Leaky ReLU assigns a non-zero slope to it. The leak helps to increase the range of the ReLU

function. Some people report success with this form of activation function, but the results are not always consistent.

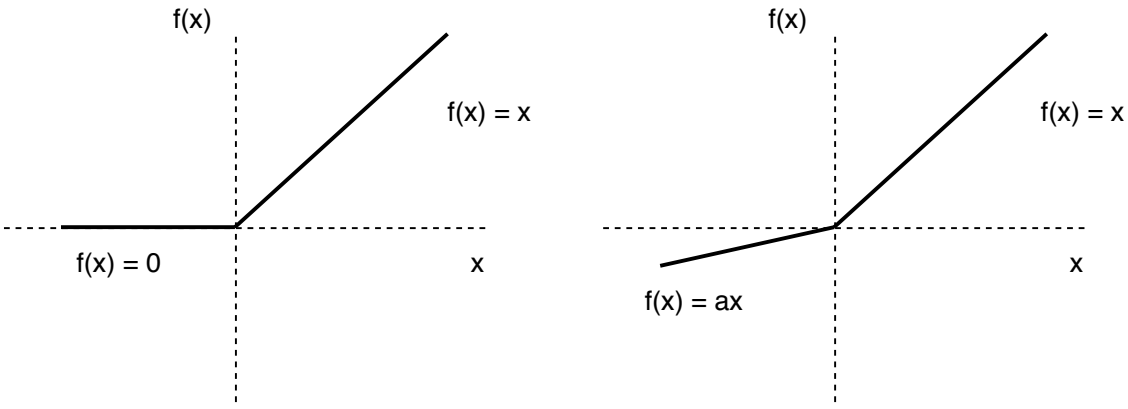

Figure 3.1: ReLU vs Leaky ReLU.

3.0.2 Convolutional Neural Network For Super-Resolution

Since the success of Dong et al. [4] proposed image super-resolution method basing on a convolutional neural network, nowadays, almost all image and video super-resolution methods using the convolutional neural network.

The framework of their method is considering a single low-resolution image, first, upscale it before getting into the convolutional neural network. Then denoting the interpolated image as X and still calling it the low-resolution image, even X is already upsampled. The goal is to learn the mapping F to map an image $F(X)$ that recovered from X to the ground truth high-resolution image Y as similar as possible. The mapping conceptually consists of three operations.

Patch extraction and representation this operation extracts patches from the low-resolution image X and represents each patch as a high-dimensional vector through convolving the image by a set of filters as:

$$F_1(X) = max(0, W_1 * X + B_1) \tag{3.2}$$

where W_1 is the filters, W_1 applies n_1 convolutions on the image, and each convolution has a kernel size $c \times f_1 \times f_1$, where c is the number of channels in the input image, f_1 is the spatial size of a filter. The output is composed of n_1 feature maps. B_1 is the biases, an n_1-dimensional vector, which each element is associated with a filter. and $*$ is convolution operation. And apply the Rectified Linear Unit $ReLU, max(0; x)$ on the filter responses.

Non-linear mapping this operation nonlinearly maps each high-dimensional vector onto another high-dimensional vector, which is mapping each of n_1-dimensional feature into an n_2-dimensional feature space as:

$$F_2(X) = max(0, W_2 * F_1(X) + B_2) \tag{3.3}$$

where W_2 contains n_2 filters of size $n_1 \times f_2 \times f_2$ and applies n_2 convolution on the outputs of first layer. The output is composed of n_2 feature maps. B2 is n_2-dimensional bias. Each of the output n_2-dimensional vectors is conceptually a representation of a high-resolution patch that will be used to reconstruct super-resolved result. And it is possible to add more convolutional layers to increase the non-linearity.

Reconstruction this operation aggregates the previous generated high-resolution patch-wise features to construct the final high-resolution image as:

$$F_3(X) = max(0, W_3 * F_2(X) + B_3) \tag{3.4}$$

where W_3 contains c filters of size $n_2 \times f_2 \times f_2$. B2 is c-dimensional bias. This image is expected to be similar to the ground truth Y and all these operations form a convolutional neural network.

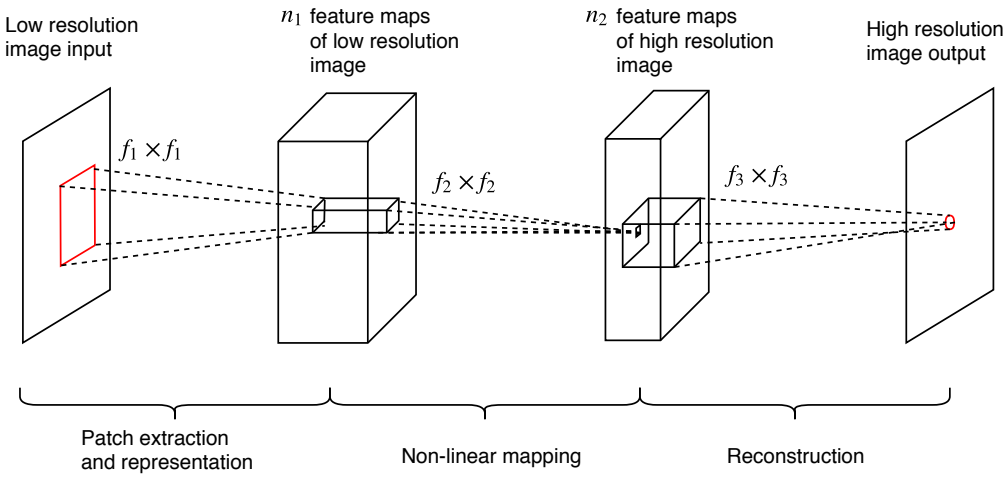

Figure 3.2: An illustration of the super-resolution using convolutional neural network.

3.0.3 Upsampling

Dong et al.[4] found there are no efficient implementations of a convolution layer whose output size is larger than the input size. Before Shi et al.[15] proposed the sub-pixel convolution layer to learn the upsampling operation, most of existing image or video super-resolution methods using bicubic interpolation to upscale the low-resolution

input before it applies a convolutional neural network. Shi et al. call the type of network using bicubic interpolation as high-resolution networks because the size of data before getting into the convolutional neural network are already upsampled to the desired output size.

Other than bicubic interpolation, transposed convolution is also an ordinary upsampling operation, which adds zero value in between non-zero pixels followed by convolution with kernel rotated 180 degrees. But these zero values are meaningless because they have no gradient information that can be backpropagated through.

In sub-pixel convolution layer, no meaningless zeros are necessary. Each input image is directly fed into the network and do the feature extraction through nonlinear convolution in low-resolution space, and super-resolve the high-resolution data from low-resolution feature maps at the end of the network through a specific type of image reshaping called a phase shift. Shi et al.[15] proposed a network with sub-pixel convolution layers to solve super-resolution problems and called it an efficient sub-pixel convolutional neural network (ESPCN) as shown in Figure 3.3.

For a network with L layers, the system learns n_{L-1} upscaling filters for the n_{L-1} feature maps. Compared to a single fixed filter upscaling at the first layer and transposed convolution put zeros in between pixels, the network is capable of learning a better and more complex low-resolution to high-resolution mapping.

In the sub-pixel convolutional neural network, the output number of feature maps at the second before the last layer is $C \cdot r^2$ instead of one high-resolution image. Then at the final layer of the network, using periodic shuffling to recreate the high-resolution image. Let r denote the upscaling ratio, for example, if the input low-resolution image of sub-pixel convolution layer is $H \times W \times C \cdot r^2$ then the high-resolution output image

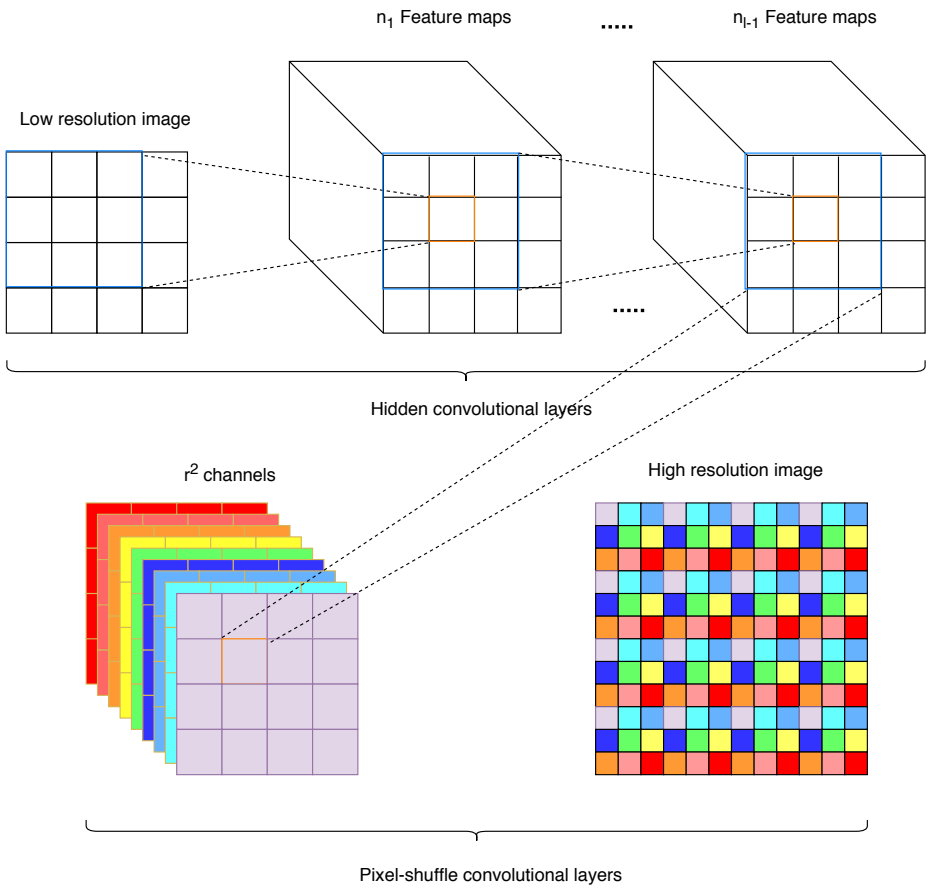

Figure 3.3: An illustration of the ESCPN framework where r denotes the upscaling ratio.

will be $H \cdot r \times W \cdot r \times C$.

$$I^{SR} = f^{L}(I^{LR}) = PS(W_L * f^{L-1}(I^{LR}) + b_L) \tag{3.5}$$

$$PS(T)_{x,y,c} = T_{\lfloor x/r \rfloor, \lfloor y/r \rfloor, c \cdot r \cdot mod(y,r) + c \cdot mod(x,r)} \tag{3.6}$$

The convolution operator W_L has shape $n_{L-1} \times r^2C \times k_L \times k_L$, where r^2C is the number of feature maps and k_L is the filter size at layer L before the periodic shuffle. PS is periodic shuffling operator, where x, y are the output pixel coordinates in high-resolution space and c is the output channels in high-resolution space.

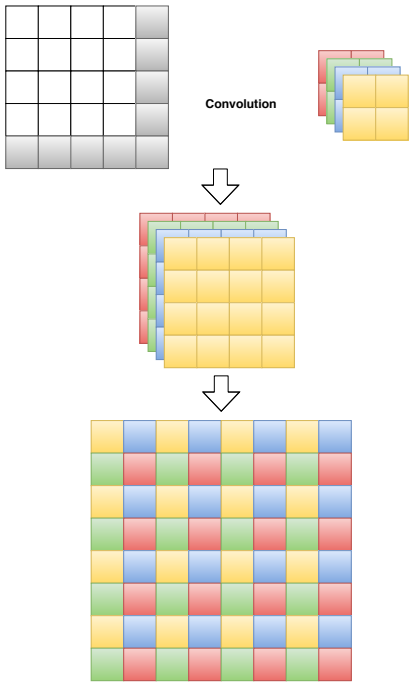

Figure 3.4: An demonstration of the pixel shuffle convolution layer operation.

3.0.4 Residual Learning

Since initialization and intermediate normalization layers mostly addressed the problem of vanishing gradients, the evidence reveals an achievement that increases the depth of convolution neural network lead the results of both ImageNet challenging task and many other non-trivial visual recognition tasks, ie. VGG nets. But He et al. [10] exposed the degradation problem happened when the networks start converging especially for the deep network, which means with the network depth increasing, accuracy gets saturated and then degrades rapidly. Adding more layers to a suitable model will get higher training error, i.e., the result of 56-layers architecture is worse than 20-layers architecture on the CIFAR-10 dataset, which proved degradation problem is not causing by overfitting. The degradation indicates that not all network is similarly easy to optimize. Solvers might have difficulties in approximating identity mapping by multiple nonlinear layers. They figured out that add more identity mapping layers onto a pre-trained shallow architecture to construct deep model are not produce higher training error. Motivated by counterintuitive phenomena about the degradation problem, they proposed deep residual network to address the degradation problem. The weights of multiple nonlinear layers will approach zero if the identity mapping is optimal.

Considering x denote the inputs and $H(x)$ as an underlying mapping to be fit by a few stacked convolutional layers. Assume x and $H(x)$ have same dimension, rather than expect the stacked layers to approximate $H(x)$, they let them to approximate a residual function $F(x, W_i) = H(x) - x$ thus the original function become $F(x, W_i) + x$. $F(x, W_i)$ represent the residual mapping to be learned and x is a shortcut connection directly from input, then do the element-wise addition to get the output of layers.

The residual network has achieved an excellent performance in both low-level and high-level computer vision tasks, Lim et al. [2] analyzed previously published image super-resolution network and figure out a better residual network architecture than SRResNet proposed by Ledig et al. [5] to further improve the performance. SRResNet removed rectified linear unit at the end of each residual block, a small improvement in test performance compared to the original residual block that place Rectified Linear Unit after the addition. Since batch normalization layers normalize the features that limited flexibility of the network, Lim et al. [2] removed batch normalization layers from their work as shown in Figure 3.5 and successfully reconstruct the higher detailed textures and sharper edges in results.

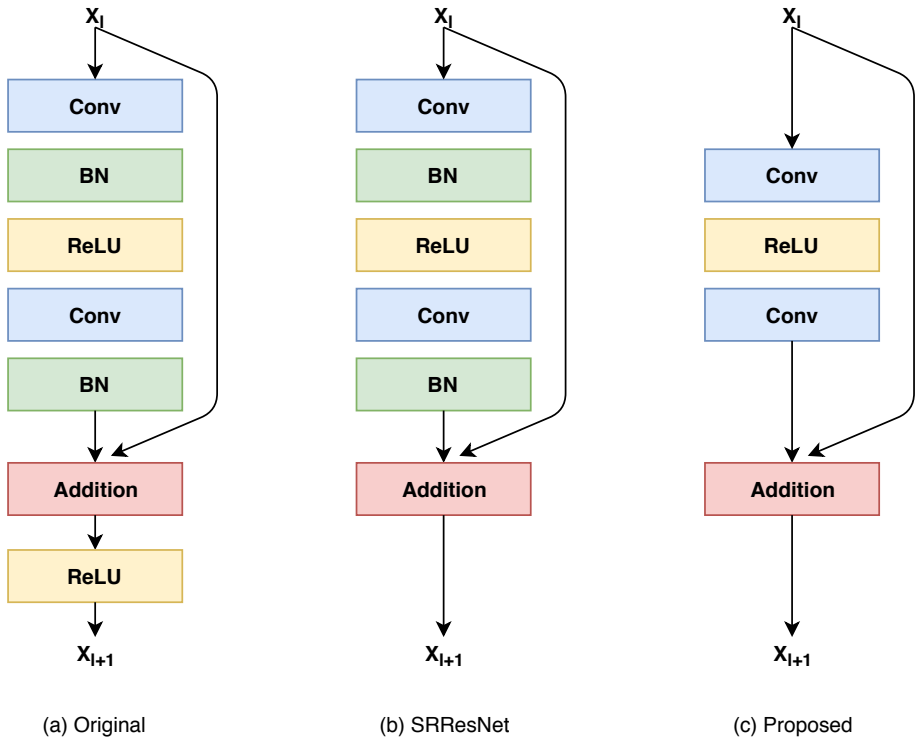

Figure 3.5: Comparision of the different residual block architectures.

3.0.5 Dynamic Filter Networks

In the traditional convolutional layer, the learned filters stay fixed after training. In video super-resolution, one of the vital parts is motion compensation due to an ill-posed problem. Different video clips contain different motion patterns, how to accurately locate the corresponding image region between consecutive frames will benefit to restore the detail information of the current frame from future frames and previous frames. The same set of filtering operations apply on every input seems suboptimal for the tasks such as video super-resolution; thus Brabandere et al. [3] proposed a dynamic filter module as shown in Figure 3.6. And the experiment results proved dynamic filter networks worked well on video prediction and able to correctly learn the individual motions of digits.

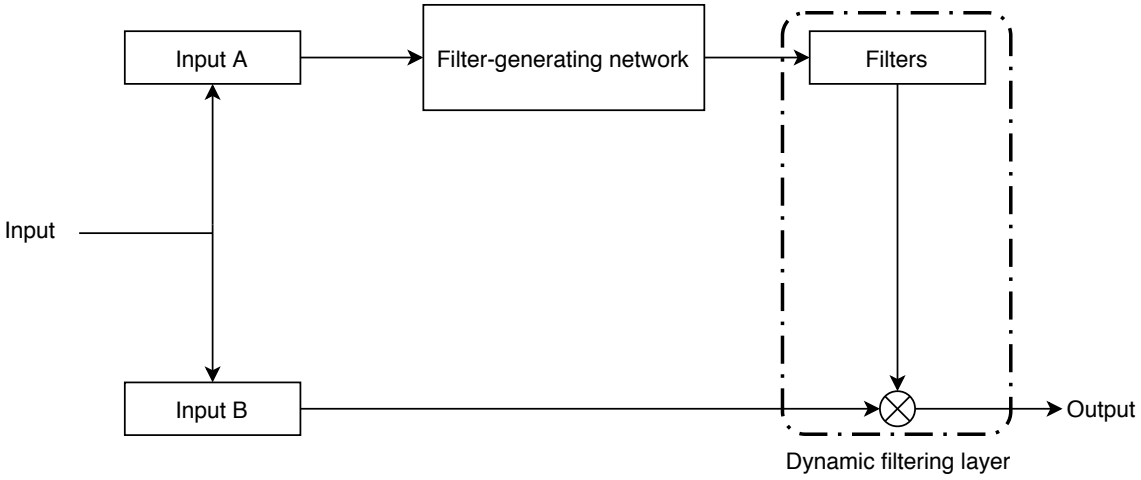

Figure 3.6: A general architecture of dynamic filter network.

The dynamic filter module consists of two parts: a filter generating network that generates sample-specified filter parameters conditioned on input, and a dynamic filtering layer that applies the generated filters to another input. Both components

of dynamic filter module are differentiable that gradients can be backpropagated throughout the network. The two inputs of the module can be either identical or different, depending on the task.

There have different between model parameters and dynamically generated parameters, model parameters are same for all samples that are initialized in advance and only updated during training, but dynamically generated parameters are sample-specific. The filter-generating network outputs dynamically generated parameters, while the parameters of the filter-generating network itself are part of the model parameters.

Filter-Generating Network

Supposed filter-generating network has an input $I_A \in \mathbb{R}^{h \times w \times c_A}$, where h, w and c_A are height, width and number of channels of input A respectively. It outputs filters F_θ parameterized by parameters $\theta \in \mathbb{R}^{s \times s \times c_B \times n \times d}$, where s is the filter size, which used to determine the respective field and is chosen depending on the designed network. c_B is the number of channels in input B, n is the number of filters, and d is equal to 1 for dynamic convolution and $h \times w$ for dynamic local filtering. The filters are applied to input $I_B \in \mathbb{R}^{h \times w \times c_B}$ to generate an output. They declared a convolutional network using for the filter-generating network is particularly suitable when using images as inputs.

Dynamic Filtering Layer

The dynamic filtering layer takes I_B as input and outputs the filtered result $G = F_\theta(I_B)$, with $G \in \mathbb{R}^{h \times w \times n}$. And there are two type of architectures of dynamic

filtering layer were proposed, the dynamic convolutional layer and dynamic local filtering layer.

Dynamic convolutional layer Same as a traditional convolutional layer, a dynamic convolutional layer also apply same filter at every position of input I_B. But the filter weights of traditional convolutional layer are model parameters, the filter parameters θ in a dynamic convolutional layer are dynamically generated by a filter-generating network:

$$G(i,j) = F_\theta(I_B(i,j)) \tag{3.7}$$

Thus the filters are sample-specific and conditional on the input of the filter-generating network as shown in Figure 3.7.

Dynamic local filtering layer Other than dynamic convoltuional layer use same dynamically generated filter parameters on every position of input I_B, depending on position of input I_B, dynamic local filtering layer use different filters generated by filter-generating network to apply different position of input. For each position (i,j) of the input I_B, a specific local filter $F_\theta^{(i,j)}$ is applied to the region centered around $I_B(i,j)$:

$$G(i,j) = F_\theta^{(i,j)}(I_B(i,j)) \tag{3.8}$$

The dynamic local filtering layer is both sample-specific and position-specific, the filter varies from position to position and from sample to sample allowing more sophisticated operations on inputs as shown in Figure 3.8. The dynamic local filtering layer can perform both a single transformation like dynamic convolutional layer and position-specific transformations like local deformation.

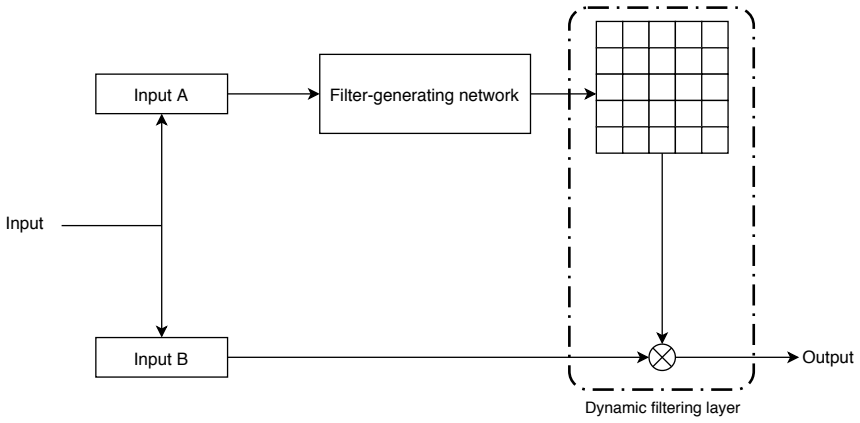

Figure 3.7: Dynamic convolition: the filter-generating network produces a single filter.

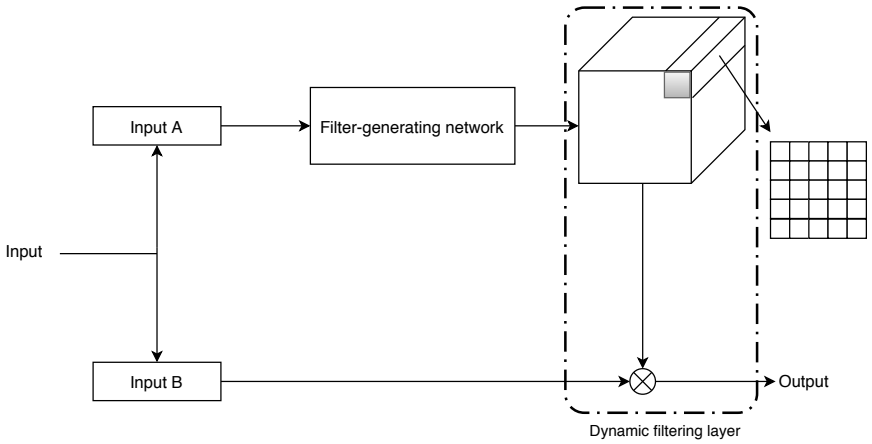

Figure 3.8: Dynamic local filtering: each location is filtered with a location-specific dynamically generated filter.

Chapter 4

Method

4.0.1 Introduction

The goal of video super-resolution is to estimate high-resolution frames from given low-resolution frames. There are different methods to reconstruct high-resolution frame base on different input frames [6] [13] [14]. In my method, the high-resolution result $\hat{Y}_t$ is reconstructed from corresponded low-resolution frame X_t and its low-resolution neighboring frames. The low-resolution frame X_t and its low-resolution neighboring frames are downsampled version from the corresponding ground truth frames Y_t and its neighboring frames, where t denotes the time step. With the proposed video super-resolution network VSR and the network parameters θ, the video super-resolution problem is defined as:

$$\hat{Y}_t = VSR_\theta(X_{t-N:t+N}) \tag{4.1}$$

where N is the temporal radius. An input tensor shape is $T \times H \times W \times C$ and the upscaling factor r, the corresponding output tensor shape will become $1 \times rH \times rW \times C$,

where $T = 2N+1$, H and W are the height and the width of the input low-resolution frames, and C is the number of color channels.

4.0.2 Proposed Method

There are three subnetworks contained in my super-resolution network F, which are used to reconstruct high-resolution frame $\hat{Y}_t$ from input X_t. The F consists of three subnetworks are:

1. Dynamic Local Filter Network.
    - Dynamically capturing the motion information.
    - Applying motion compensation.
2. Pixel-Shuffle Network.
    - Capturing the detail information about the generated feature maps.
    - Upsampling the generated feature maps to reconstruct the high-resolution result.
3. Refinement Network
    - Refining the textures.

In most of the super-resolution methods, the input images or video frames are segmented into the patches due to the limit ability of the device. It has been shown that for PSNR scores there is a little improvement in image-based training than patch-based training [14], and there is some artificial effect between patches alignment as shown in Figure 4.1. But the memory of GPU will be run out if the input images

are too large. In my method, to maximize the capability of my device and reduced artifacts, the size of the input frames is cropped to one-quarter of validation frames size.

Figure 4.1: Left: smaller patches combined output Right: larger patches combined output. From PSNR scores perspective the left side even slightly larger than the right side, but there is the artificial effect on windows of the building. The upscaling factor is set to 4.

Specifically, for the input $X_{t-N:t+N}$, first downsampled the segmented frames through bicubic interpolation by the desired scale then transform the color frames into the $YCbCr$ space. The super-resolution algorithms are only applied to the Y channel, while bicubic interpolation upsamples the Cb, Cr channels.

First, putting low-resolution inputs $X_{t-N:t+N}$ into a dynamic local filter network to get motion compensated feature maps that related to center frame X_t. Then upsampling the feature maps to high-resolution frame $\hat{Y}_t^{est}$ by a pixel-shuffle network. Finally, using a refinement network to obtain the final result $\hat{Y}_t$. The architecture of video super-resolution network is shown in Figure 4.2. Here, I'm making a specific explanation of upscaling ratio of four (r=4) with input frames of three (N=1).

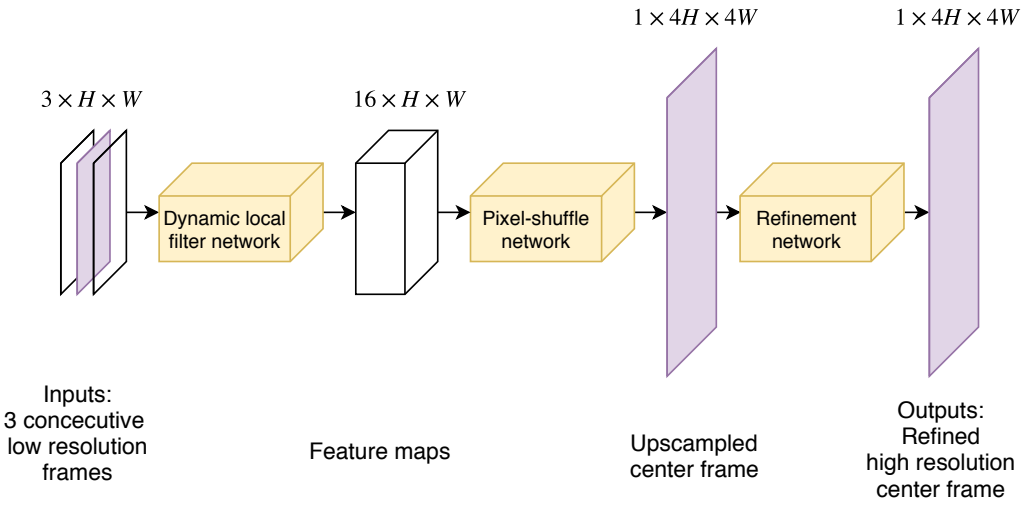

Figure 4.2: The architecture of proposed video super-resolution network.

Motion compensation

The filter kernels for the traditional convolutional layer are basically fixed, the learned filters stay fixed after training. But there different motion patterns within different video frames. Contrary to this, we propose to use the dynamic local filter network inspired by the Dynamic filter network [3]. The filters before upsampling are generated locally and dynamically depending on the spatial-temporal neighborhood of each pixel in low-resolution frames.

In the dynamic local filter(DLF) network, as a first step, filter-generating network generates both inputs specified and location specified filters F parameterized by parameter θ_{DLF} of shape $W \times H \times c \times n \times f \times f$, where W and H are the width and height sizes of input frames, n is the number of output channels, c is the input channels and f in the filter size of filter-generating network. Here, each position (i, j) in inputs has a location specified filter of shape $c \times n \times f \times f$ apply on it, the width

size W and height size H are the sum of positions in the horizontal i and vertical j direction respectively. Because each frame is applied only on the luminance channel for my method, the input channels c is equal to the number of inputs frames. The number of output channels is equal to the square of upscale ratio $n = r^2$ for the up-sampling purpose in next step. Then the filters $F_{\theta_{DLF}}$ are applied to inputs $X_{t-N:t+N}$ of shape $W \times H \times c$ to generate outputs G_t of shape $W \times H \times r^2$. The generated filters are estimates the optical flow between the low-resolution inputs X_{t-1}, X_{t+1} and X_t yielding the normalized low-resolution flow map

$$G_t = F_{\theta_{DLF}}(X_{t-1:t+1}) \tag{4.2}$$

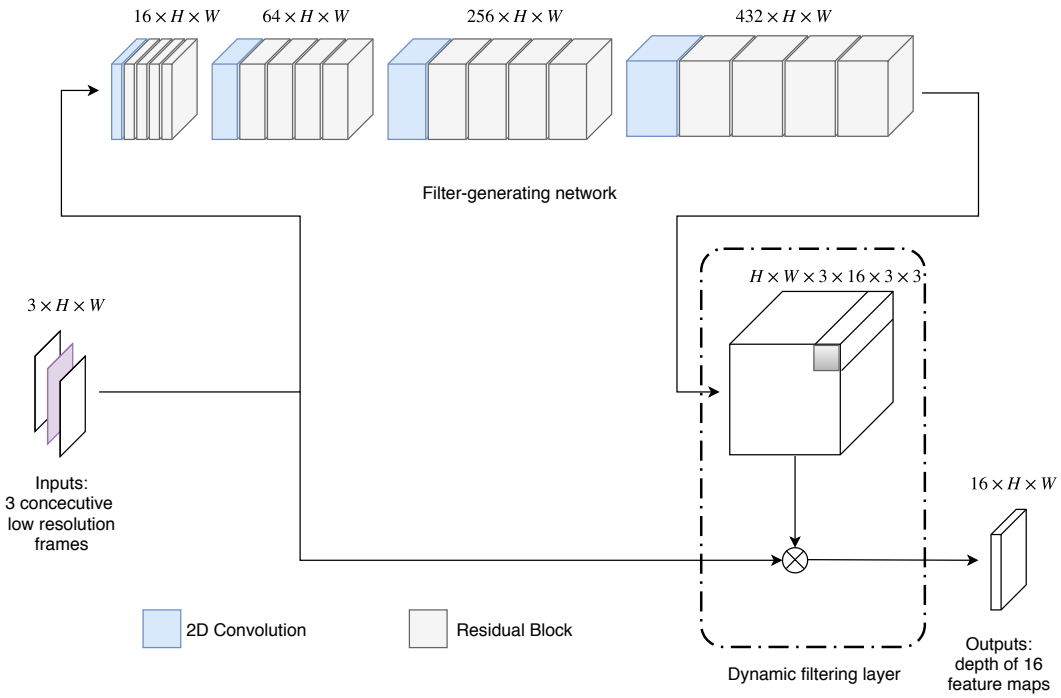

Figure 4.3: The architecture of the dynamic local filter network.

Upscaling

Other than before, Shi et al. [15] proposed a novel network architecture, which upsampling is handled by the last layer of the network, to perform single image super-resolution. It means each low-resolution image can directly feed to the system and feature extraction occurs through nonlinear convolutions in low-resolution image space. Thus, the network is capable of learning a better and more complex low-resolution to high-resolution mapping compared to a single fixed filter such as bicubic interpolation upsampling at the first layer [1] [4] [14]. It efficiently restored sharp and textured regions than traditional interpolation.

Network depth is of crucial importance in neural network architectures, but deeper networks are more difficult to train. The residual learning framework eases the training of these networks and enables them to be substantially deeper. Here using residual block proposed by He et al.[10] to deepen the network but removed all batch normalization layer and activation function at the end of the residual block, based on successful of Lim et al.[2].

In the pixel-shuffle(PS) network, the output feature maps that generated from the dynamic local filter network as the input are first though eight residual block layers to capturing the detail information about the generated feature maps to improve performance. Then using a pixel-shuffle layer to upsample low-resolution feature maps G_t of shape $W \times H \times r^2$ to high-resolution output $\hat{Y}_t^{est}$

$$\hat{Y}_t^{est} = PS(G_t) \tag{4.3}$$

where $\hat{Y}_t^{est}$ has shape $rW \times rH \times 1$

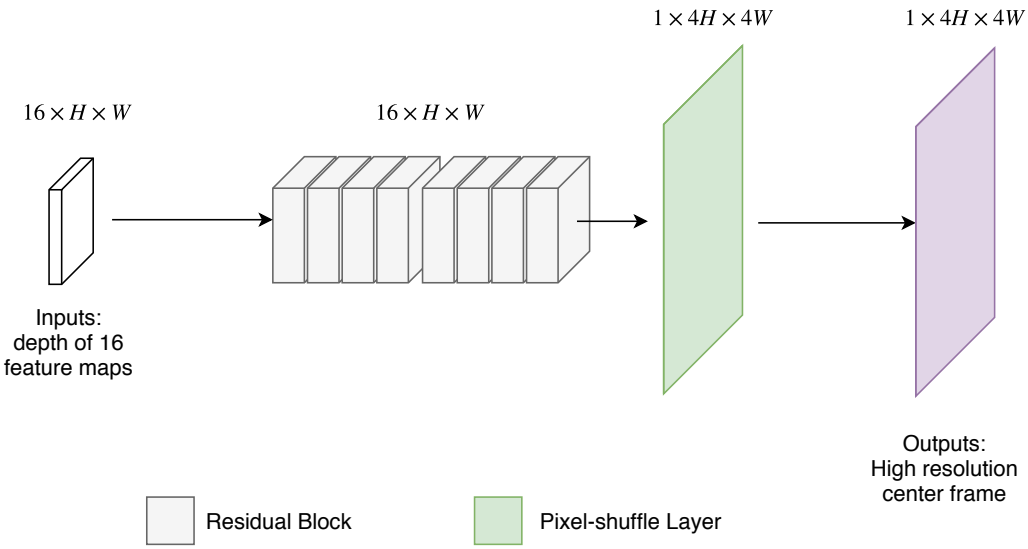

Figure 4.4: The architecture of the pixel-shuffle network.

Refinement

Even the first two subnetworks already can generate satisfactory results, but after experimentation, an additional refinement network is still has a benefit for the results. The architecture of the refinement network is designed as an encoder-decoder style with skip connections, which is inspired by the tailored detail fusion network [16]. I have replaced the ConvLSTM module [17] with several residual blocks in the middle stage. As the experiment results, the addition refinement network not only achieved better PSNR score but also fasten the convergent speed during the training.

In the refinement(Ref) network, the high-resolution output $\hat{Y}_t^{est}$ of shape $rW \times rH \times 1$ as the input are shrank using convolution layers with stride of 2 (The amount by which the filter shifts is the stride) to size $W \times H \times 1$, then upsampled twice using transposed convolution layers to get refined high-resolution result $\hat{Y}_t$

$$\hat{Y}_t = Ref(\hat{Y}_t^{est}) \tag{4.4}$$

where $\hat{Y}_t$ has shape $rW \times rH \times 1$.

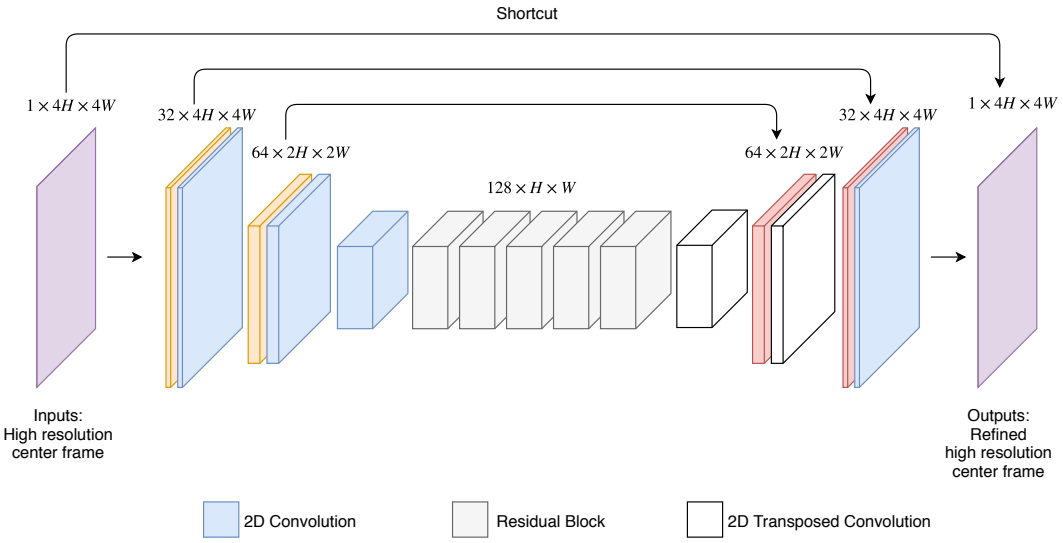

Figure 4.5: The architecture of the refinement network.

4.0.3 Implementation

Dataset

One of the essential elements of deep learning is the quantity and the quality of training data. To achieve good generalization results, videos in the training dataset must contain various and complex real-world motions. Therefore, the collected videos must include various textures and motions. But for the video super-resolution task, a dataset like ImageNet[8] does not exist. There is a total of 59 videos downloaded from the Internet with various contents including wildlife, activity, and landscape. Then

cutting downloaded videos to 10789 video clips, each video clips has 3-5 seconds with the spatial resolution of 1280×720 by selecting the scenes with sufficient amount of motion. For every clip choose only the first 30 frames as the training dataset. For the comparison purpose, the validation set using standard **Vid4** benchmark dataset[12] to compare with state-of-the-art methods.

Training

Easily for training, the mini-batch size is set to 4, and the number of consecutive frames set to 3. The filter size set to 3×3 for every convolution layers in all three subnetworks. To obtain low-resolution inputs, in the first, the ground truth data are randomly cut to the fixed patch size of spatial resolution 384×384 in training and divided evenly into four pieces for validation. This leverage both device capability and reducing artificial effects. Then downsampled using bicubic interpolation for the upscaling factor r.

The majority of super-resolution algorithms focus on gray-scale or single-channel image or video super-resolution [10] [1] [14]. For color images or video frames, first transform the problem to a different color space (YCbCr or YUV), and super-resolution is applied only on the luminance channel. He et al. [10] proved that the Cb, Cr channels could decrease the performance of the Y channel when training performed in a unified network. There are also works attempting to super-resolve all channels simultaneously [13]. Appling each RGB channel into the model and combined them to produce the final results. However, none of them has analyzed the super-resolution performance of different channels, and the necessity of recovering all three channels. Thus, the super-resolution algorithms are only applied to the Y

channel, while bicubic interpolation upsamples the Cb, Cr channels.

Following Sajjadi et al.[13], the computing of video PSNR score is on the luminance channel Y ($ITU - R\ BT.601\ YCbCr$ standard). Using the mean square loss to measure the loss between final output $\hat{Y}_t$ and ground truth Y_t as the cost function. Using Adam optimizer [11] to train the network with a fixed learning rate 1×10^{-5}. And replacing every activation function from rectified linear units (ReLU) to leaky rectified linear units (Leaky ReLU).

4.0.4 Experimental Results

Quantitative Evaluation

Quantitative comparison with other state-of-the-art video super-resolution methods on **Vid4** for scale factor $r = 3$ is shown in Table 4.1 and for scale factor $r = 4$ is shown in Table 4.2. The proposed method outperforms all other methods by a large margin in terms of PSNR and SSIM for both upscale factors. And the size comparison between downsampled by a factor of 4 with actual size is shown in Figure 4.6.

Metric	Bicubic	Bayesian	VSRnet [1]	B1,2,3+T [6]	VESPCN [9]	SPMC [16]	proposed
PSNR	25.28	25.82	26.79		27.25	27.49	29.03
SSIM	0.7329	0.8323	0.8098		0.8447	0.84	

Table 4.1: $\times 3$ upsampling : Compared proposed method with state-of-the-art.

Metric	Bicubic	Bayesian	VSRnet [1]	B1,2,3+T [6]	VESPCN [9]	SPMC [16]	proposed
PSNR	23.79	25.06	24.84	25.35	25.35	25.52	26.72
SSIM	0.6332	0.7466	0.7049	0.738	0.7557	0.76	0.803

Table 4.2: $4\times$ upsampling: Compared proposed method with state-of-the-art.

(a) Downsampled (b) Ground truth

Figure 4.6: Comparing downsampled 4× figure with ground truth figure side by side

Qualitative Comparisons

Some qualitative examples are shown in Figure 4.7. The proposed models are able to recover fine details and produce visually pleasing results In Fig. 10, we also compare a result from [34] with the results using our network with different depth. We outperform the previous work and can also observe the increase in the performance with more depth. More qualitative comparisons with other state-of-the-art video super-resolution methods on **Vid4** is shown in Figure 4.8, Figure 4.9, Figure 4.10 and Figure 4.11. The results of proposed network show sharper outputs with more smooth temporal transition compared to other works.

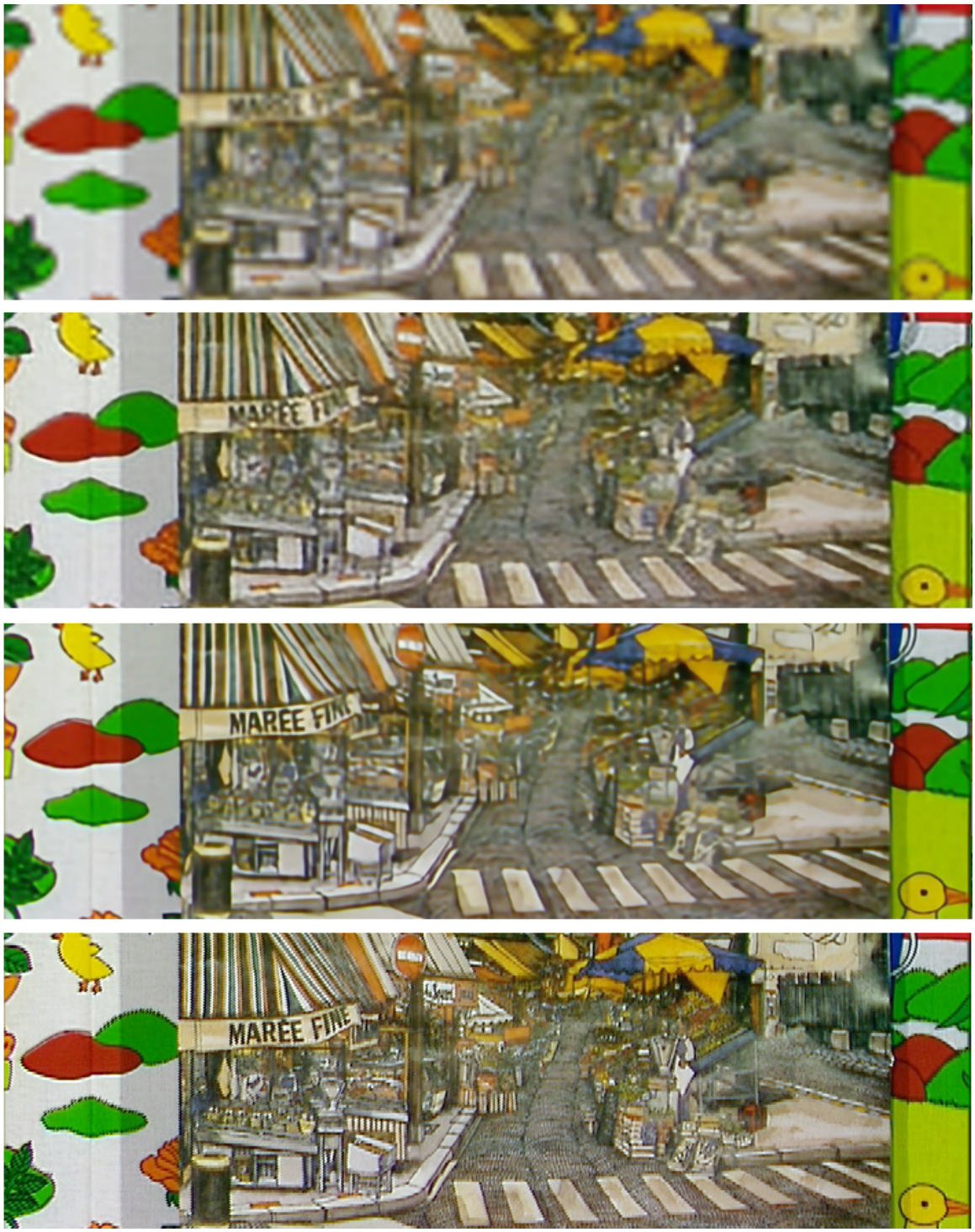

Figure 4.7: Calendar in **Vid4** for 4× upscaling. From top to bottom: Bicubic, B1,2,3+T, Proposed, Ground truth.

Figure 4.8: Calendar in **Vid4** for 4× upscaling. From top to bottom: Bicubic, B1,2,3+T, Proposed, Ground truth.

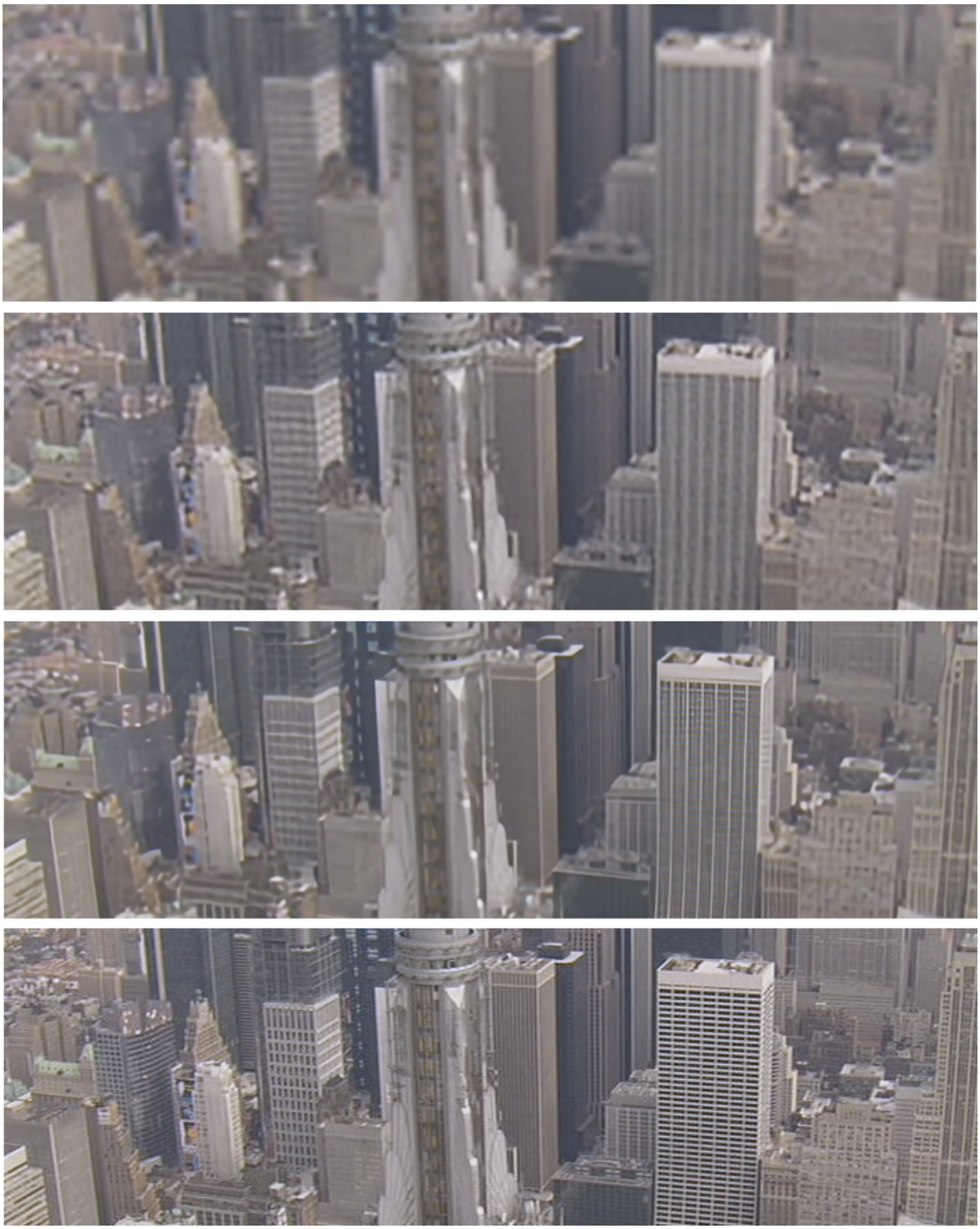

Figure 4.9: City in **Vid4** for 4× upscaling. From top to bottom: Bicubic, B1,2,3+T, Proposed, Ground truth.

Figure 4.10: Foliage in **Vid4** for 4× upscaling. From top to bottom: Bicubic, B1,2,3+T, Proposed, Ground truth.

Figure 4.11: Walk in **Vid4** for 4× upscaling. From top to bottom: Bicubic, B1,2,3+T, Proposed, Ground truth.

Chapter 5

Conclusion and Future Works

The proposed flexible end-to-end trainable framework for video super-resolution can recover sharp higher quality results. And the network can implicitly handle the motion via dynamic local filter network without explicit motion estimation and compensation. The proposed model significantly outperforms state-of-the-art video super-resolution approaches both quantitatively and qualitatively on a standard benchmark dataset.

A further extension of the framework would be the increasing size of the dataset and the inclusion of more advanced loss terms which have recently been shown to produce more visually pleasing results [5]. Adding a convolutional encoder-decoder as the filter-generate network module in the dynamic local filter network where the encoder consists of several strided convolutional layers, and the decoder includes several convolutional layers and upsampling layers. The convolutional encoder-decoder may able to exploit the spatial correlation within a frame and temporal relation between frames.

Bibliography

[1] Armin Kappeler, Seunghwan Yoo, Q. D. and Katsaggelos, A. K. (2016). Video super-resolution with convolutional neural networks. *IEEE Transactions on Computational Imaging (TCI) 2016*, **Volume: 2**(2).

[2] Bee Lim, Sanghyun Son, H. K. S. N. and Lee, K. M. (2017). Enhanced deep residual networks for single image super-resolution. *Computer Vision and Pattern Recognition Workshops (CVPRW) 2017*.

[3] Bert De Brabandere, Xu Jia, T. T. and Gool, L. V. (2016). Dynamic filter networks. *Advances In Neural Information Processing Systems (NIPS) 2016*.

[4] Chao Dong, Chen Change Loy, K. H. and Tang, X. (2015). Image super-resolution using deep convolutional networks. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 2015*, **Volume: 38**(2).

[5] Christian Ledig, Lucas Theis, F. H. J. C. A. C. A. A. A. A. A. T. J. T. Z. W. and Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. *Computer Vision and Pattern Recognition (CVPR) 2017*.

[6] Ding Liu, Zhaowen Wang, Y. F. X. L. Z. W. S. C. and Huang, T. (2017). Robust

video super-resolution with learned temporal dynamics. *International Conference on Computer Vision (ICCV) 2017*.

[7] Eddy Ilg, Nikolaus Mayer, T. S. M. K. A. D. and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. *Computer Vision and Pattern Recognition (CVPR) 2017*.

[8] Jia Deng, Wei Dong, R. S. L.-J. L. K. L. and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *Conference on Computer Vision and Pattern Recognition (CVPR) 2009*.

[9] Jose Caballero, Christian Ledig, A. A. A. A. J. T. Z. W. and Shi, W. (2017). Real-time video super-resolution with spatio-temporal networks and motion compensation. *Computer Vision and Pattern Recognition (CVPR) 2017*.

[10] Kaiming He, Xiangyu Zhang, S. R. and Sun, J. (2016). Deep residual learning for image recognition. *Computer Vision and Pattern Recognition (CVPR) 2016*.

[11] Kingma, D. P. and Bal, J. (2015). Adam: A method for stochastic optimization. *International Conference for Learning Representations (ICLR) 2015*.

[12] Liu, C. and Sun, D. (2014). On bayesian adaptive video super resolution. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 2014*, **Volume: 36**(2).

[13] Mehdi S. M. Sajjadi, R. V. and Brown, M. (2018). Frame-recurrent video super-resolution. *Computer Vision and Pattern Recognition (CVPR) 2018*.

[14] O. Makansi, E. I. and Brox, T. (2017). End-to-end learning of video super-resolution with motion compensation. *German Conference on Pattern Recognition (GCPR) 2017*.

[15] Wenzhe Shi, Jose Caballero, F. H. J. T. A. P. A. R. B. D. R. and Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *Computer Vision and Pattern Recognition (CVPR) 2016*.

[16] Xin Tao, Hongyun Gao, R. L. J. W. and Jia, J. (2017). Detail-revealing deep video super-resolution. *International Conference on Computer Vision (ICCV) 2017*.

[17] Xingjian Shi, Zhourong Chen, H. W. D.-Y. Y. W.-k. W. and chun Woo, W. (2015). Convolutional lstm network: a machine learning approach for precipitation nowcasting. *Neural Information Processing Systems (NIPS) 2015*.