APPLICATIONS OF DEEP LEARNING TO VISUAL CONTENT PROCESSING AND ANALYSIS

APPLICATIONS OF DEEP LEARNING TO VISUAL CONTENT PROCESSING AND ANALYSIS

BY

XIAOHONG LIU, B.Eng., M.A.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

DOCTOR OF PHILOSOPHY

Doctor of Philosophy (2021) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Applications of Deep Learning to Visual Content Processing and Analysis
AUTHOR:	Xiaohong Liu M.A.Sc. (Electrical & Computer Engineering), University of Ottawa, Ottawa, Canada B.Eng. (Telecommunication Engineering), Southwest Jiaotong University, Chengdu, China
SUPERVISOR:	Dr. Jun Chen
NUMBER OF PAGES:	xix, 159

Abstract

The advancement of computer architecture and chip design has set the stage for the deep learning revolution by supplying enormous computational power. In general, deep learning is built upon neural networks that can be regarded as a universal approximator of any mathematical function. In contrast to model-based machine learning where the representative features are designed by human engineers, deep learning enables the automatic discovery of desirable feature representations based on a data-driven manner. In this thesis, the applications of deep learning to visual content processing and analysis are discussed.

For visual content processing, two novel approaches, named LCVSR and RawVSR, are proposed to address the common issues in the filed of Video Super-Resolution (VSR). In LCVSR, a new mechanism based on local dynamic filters via Locally Connected (LC) layers is proposed to implicitly estimate and compensate motions. It avoids the errors caused by the inaccurate explicit estimation of flow maps. Moreover, a global refinement network is proposed to exploit non-local correlations and enhance the spatial consistency of super-resolved frames. In RawVSR, the superiority of camera raw data (where the primitive radiance information is recorded) is harnessed to benefit the reconstruction of High-Resolution (HR) frames. The developed network is in line with the real imaging pipeline, where the super-resolution

process serves as a pre-processing unit of ISP. Moreover, a Successive Deep Inference (SDI) module is designed in accordance with the architectural principle suggested by a canonical decomposition result for Hidden Markov Model (HMM) inference, and a reconstruction module is built with elaborately designed Attention based Residual Dense Blocks (ARDBs).

For visual content analysis, a new approach, named PSCC-Net, is proposed to detect and localize image manipulations. It consists of two paths: a top-down path that extracts the local and global features from an input image, and a bottom-up path that first distinguishes manipulated images from pristine ones via a detection head, and then localizes forged regions via a progressive mechanism, where manipulation masks are estimated from small scales to large ones, each serving as a prior of the next-scale estimation. Moreover, a Spatio-Channel Correlation Module (SCCM) is proposed to capture both spatial and channel-wise correlations among extracted features, enabling the network to cope with a wide range of manipulation attacks.

Extensive experiments validate that the proposed methods in this thesis have achieved the SOTA results and partially addressed the existing issues in previous works.

To my family, for their love and support.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Jun Chen, for his academic enthusiasm, expert advice, insightful guidance, and constant encouragement throughout my Ph.D. study. I'm also deeply impressed by his personal character, especially for his wisdom and kindness. Undoubtedly, he is the best professor I have ever met.

Great appreciations are also expressed to my committee members Prof. Sorina Dumitrescu, Prof. Shahram Shirani, and Prof. Jiankang Zhang (dec.), for their valuable suggestions and comments on my research works. Prof. Zhang, we all miss you so much.

Moreover, I'd like to thank all my lab colleagues, especially to Zhihao Shi, Kangdi Shi, Linhui Dai, Yongrui Ma, Lingshi Kong, and Chenqi Li, for their assistance and friendship.

Last but not least, I would like to thank my parents for their love and support in my life. I would not have been able to complete this thesis without their continuous encouragement. I would also like to thank my wife, Dan Zhou, who gives me all her love and always backs me up in difficult times.

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
DSLR	Digital Single-Lens Reflex
ML	Machine Learning
DL	Deep Learning
DNN	Deep Neural Network
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural network
GAN	Generative Adversarial Network
GRU	Gated Recurrent Units
LSTM	Long Short-Term Memory
SGD	Stochastic Gradient Decent
SISR	Single Image Super-Resolution

VSR	Video Super-Resolution
LR	Low-Resolution
HR	High-Resolution
GT	Ground-Truth
ISP	Image Signal Processor
HMM	Hidden Markov Model
DLF	Dynamic Local Filters
IMDL	Image Manipulation Detection and Localization
MSE	Mean Squared Error
SSIM	Structural SIMilarity index
AUC	Area Under Curve
EER	Equal Error Rate
TPR	True Positive Rate
SOTA	State Of The Art

Chapter 1

Introduction

1.1 Deep Learning

As a sub-field of Machine Learning (ML), Deep Learning (DL) seeks to construct a universal approximator of any mathematical function. Different from the traditional ML methods where the representative features are designed by human engineers, DL is usually built on a hierarchical structure (*i.e.*, Deep Neural Network (DNN)) that enables the automatic discovery of needed feature representations via backpropagation. Here the term "deep" emphasizes that the corresponding network is composed of multiple layers, each acting as a non-linear transformation that maps the input data to a more abstract level. After stacking a plenty of such transformations (*i.e.*, layers), very complex functions can be approximated, thus a universal approximator is achieved. Note that the neural network was first proposed in 1943 [1] and designed to imitate the activities of human brains. Therefore, this kind of algorithms can be also termed Artificial Neural Network (ANN), and the learned knowledge is termed Artificial Intelligence (AI).

In comparison to traditional machine learning methods, the most significant advantage of DNN is its scalability, where the performance improves as more data and/or deeper networks are leveraged. In contrast, other approaches (*e.g.*, support vector machine) quickly reach the performance plateau and cannot be scalably improved. In summary, the rapid rise of deep learning (*i.e.*, DNN) relies on the following three key factors:

1. Sufficient computational power to support the enormous calculations in DNNs;
2. Collected and labelled large-scale datasets for network training;
3. The adoption of more advanced algorithms to make the DNN deeper without gradient vanishing and/or explosion.

Note that the computational power is the main constraint that limits the development of deep learning in the past few decades. Since deep learning leverages a data-driven approach to generate the representative features, collecting and annotating a large-scale dataset is of essential importance and usually determines the performance limit of the learned network. At last, the proposal of advanced modules (*e.g.*, the residual block [2]) and techniques (*e.g.*, batch normalization [3]) is beneficial to developing deeper DNNs with further enhanced performance.

In general, one of the common forms of deep learning is supervised learning [4], where each input to a network has a corresponding target. By computing an objective function, the distance (error) between the network output and the target can be measured. Subsequently, this distance is reduced via gradient decent, in which the relevant gradient for each network parameter is calculated to adjust this parameter in the opposite direction of the gradient, resulting in a better output closer to the

target. More specifically, to train the network, Stochastic Gradient Decent (SGD) and its variants [5] are usually adopted. These methods first split the whole training dataset into many small subsets and compute the average gradient over each subset, as a rough estimate of the gradient over the whole dataset, to update the network parameter. This procedure has been validated to be quite effective and efficient to find a good set of parameters.

To collect a dataset for network training, the generation of target (*i.e.*, labelling) usually requires upfront human intervention. However, the target changes in accordance with the objective of visual tasks. For example, in visual content processing (*e.g.*, image/video restoration), the target is in the low level (*e.g.*, a clear version of degraded data), thus does not involve semantic information. On the contrary, in visual content analysis (*e.g.*, image classification), the target is in the high level (*e.g.*, one specific category) and the tailored labels are needed to describe the semantic information in data. In both cases, the acquisition of a large-scale paired data is quite expensive and needs plenty of human effort.

To reduce the degree of human intervention, semi-supervised learning [6] and self-supervised learning [7, 8] are proposed. In semi-supervised learning, as the name implies, a small part of data is fully labelled but the rest majority is not. Therefore, it can be regarded as an integration of supervised learning and unsupervised learning, where the network with additional unlabelled data can be considerably improved. Since the unlabelled data is almost free to acquire, semi-supervised learning is widely considered when annotating a large set of data is not practical. Self-supervised learning, which is a subset of unsupervised learning, develops quickly in recent years. Note that the term "unsupervised" is not synonymous with "unlabelled". It only

implies that no human intervention is required. Self-supervised learning resorts to the intrinsic connection and correlation in data to *automatically* generate the labels to supervise the learning process. For example, in the task of video sequence sorting [9], a shuffled video sequence is given as input, and the objective is to sort this shuffled sequence. Here the "label" that serves as the supervisory signal is the temporal coherence, which can be trivially obtained without any human intervention. By exploiting the data correlation itself, self-supervised learning has the same cost as unsupervised learning but acts as the supervised learning. Recent advances illustrate that self-supervised learning can achieve the similar performance to supervised learning in several tasks [10]. However, in this thesis, since the training data in the fields of video super-resolution and image manipulation detection and localization is easy to synthesize, the proposed algorithms still adopt supervised learning for network training rather than using semi-supervised learning or self-supervised learning.

In summary, there are three typical categories of neural networks in deep learning:

1. Artificial Neural Network (ANN), also known as the feed-forward neural network;
2. Convolutional Neural Network (CNN);
3. Recurrent Neural network (RNN).

ANN is composed of multiple perceptrons/neurons, where each neuron performs similarly to the logistic regression. In general, it consists of 3 layers — input, hidden, and output layers, where the input and output layers are responsible for accepting input data and generating network results, and the hidden layer implicitly learns the desired mapping function based on the relation between the input and target. ANN

has been capitalized on many tasks including, among others, fraud detection [11], E-mail span filtering [12], and sales forecasting [13].

Different from ANN, CNN is composed of a variety of kernels, thus dramatically reduces the total number of learnable parameters in network. Since this kernel-based architecture is particularly suited to the processing of array-like data, CNN has achieved great success in computer vision. The typical applications of CNN include image classification [14], image segmentation [15], object detection [16], facial recognition [17], etc.

As for RNN, it can be imagined as a variant of ANN, where the hidden layer is recurrently connected along with continuous time stamps. This characteristic enables RNN to memorize the information in sequential data, thus is widely adopted in the areas of time series analysis, handwriting recognition [18], and natural language processing. Since RNN aims at dealing with sequential data, in backpropagation, the gradient from the eailier inputs may vanish, thus fails to adequately update the pertinent network parameters, causing the short-term memory issue. To address this issue, Gated Recurrent Units (GRU) [19] and Long Short-Term Memory (LSTM) [20] are proposed, where the so-called gates are introduced to alleviate gradient vanishing in RNN.

Recently, a novel network architecture is proposed, named *transformer* [21], which probably becomes a new category of neural networks. Different from the aforementioned three categories, this network architecture is based on a self-attention mechanism and has surpassed the performance of RNN family in natural language processing. As for how to use the transformer on vision tasks, it is an ongoing research topic [22].

Since this thesis mainly focuses on applications of deep learning to visual content processing and analysis, CNN is still the best choice to date.

1.2 Visual Content Processing

Visual content processing aims at exploring the recorded primitives (*e.g.*, edge, texture) in visual contents rather than inferring their semantics. This processing is commonly applied on low-level vision tasks such as Super-Resolution (SR) [23,24], deraining/dehazing [25–27], video interpolation [28,29], etc. Recently, extensive works have illustrated that visual content processing can benefit the downstream tasks. For instance, reference [30] claims that low-resolution faces after SR reconstruction achieves better recognition performance than the one without this process; reference [31] shows that image dehazing can be used as a pre-processing step to improve the accuracy of autonomous driving systems.

Among low-level vision tasks, SR has received special attentions owing to its broad applications in practice. Without resorting to hardware upgrades, SR aims at increasing the resolution of an image or a video sequence to provide users with a more attractive visual experience. Since the limitation of phone size impedes the manufacturers to adopt better but larger camera modules, this technique has been widely applied to improve the quality of phone-taken photographs of distant scenes, termed the *digital* zoom.

To train the SR methods, camera processed data produced by the Image Signal Processor (ISP) are commonly employed. However, such methods neglect the information loss incurred by non-invertible operations in ISP (*e.g.*, compression), and also fail to conform to the real imaging pipeline where SR should serve as a pre-processing

unit of ISP. Note that the advantage of untouched camera raw data over processed data has been recognized in several low-level vision tasks such as image relighting [32], camera ISP mapping [33], etc. As in SR field, reference [34] is the pioneer work that harnesses camera raw data for single image SR, and the proposed method [24] is the first attempt that exploits this type of data for Video Super-Resolution (VSR).

1.3 Visual Content Analysis

Different from visual content processing, visual content analysis aims at deriving the semantics in visual contents that enable machines to "think" as humans. Therefore, it is usually used to tackle high-level vision tasks such as image classification [35], object detection [36], semantic segmentation [16], etc.

Recently, the implementation of image manipulation dramatically reduces the difficulty of image editing, not only modifying part of image contents but fully synthesizing the image. For instance, a recent developed software[1] makes it possible to generate facial images that are too realistic to be distinguished by human. This kind of tools, usually implemented by deep neural networks, is named *Deepfake*, where the most common technique they adopt is Generative Adversarial Network (GAN). With these tools, image manipulation has become ubiquitous in social media, thus raises new concerns for their uses. Although a majority of image manipulations are just for entertainment, some attackers may utilize these tools to create deceitful contents and propagate them in social media. To alleviate this concern, a reliable "shield" is desired to expose the images that are manipulated.

For manipulation in facial images, there are four typical manipulations [37]: 1)

[1] https://thispersondoesnotexist.com/

identity swap, 2) expression swap, 3) attribute manipulation, and 4) entire face synthesis. In comparison to detecting manipulations in natural images, the detection of facial images largely benefits from the comprehensive studies in the face field (*e.g.*, facial landmark detection, face recognition), thus is relatively easier. However, the target of manipulation may not always aim at face regions. In fact, any interesting content can be potentially selected for manipulation. Therefore, in this thesis, a general method that enables the detection of manipulations in natural images is proposed [38].

Moreover, since the output of deep-learning based manipulation detection methods is an image-level score that indicates whether the input image is manipulated or not, it lacks interpretability and fails to provide concrete information to show the potential forged regions. To address this issue, the proposed method also localizes the forged regions to help us understand the logic behind. To the best of my knowledge, this is the first work that enables detecting and localizing manipulations in natural images.

1.4 Contributions and Thesis Organization

This thesis contributes to both visual content processing and analysis based on deep learning, which is composed of three articles in a *sandwich thesis* format following the terms and regulations of McMaster University. The independent contributions to each article are outlined in the preface of Chapter 2, Chapter 3, and Chapter 4. The reference information of these three articles is listed below:

1. **Xiaohong Liu**, Lingshi Kong, Yang Zhou, Jiying Zhao, and Jun Chen. "End-To-End Trainable Video Super-Resolution Based on a New Mechanism for Implicit Motion Estimation and Compensation." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2416-2425, 2020.

2. **Xiaohong Liu**, Kangdi Shi, Zhe Wang, and Jun Chen. "Exploit Camera Raw Data for Video Super-Resolution via Hidden Markov Model Inference." IEEE Transactions on Image Processing, Vol. 30, pp. 2127-2140, 2021.

3. **Xiaohong Liu**, Yaojie Liu, Xiaoming Liu, and Jun Chen. "PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization.", arXiv preprint arXiv:2103.10596, 2021.

The first two articles belong to the topic of visual content processing, and the last article involves visual content analysis.

More specifically, the first article is to tackle the VSR task, named Locally Connected VSR (LCVSR). A new mechanism that implicitly estimates and compensates motions among video frames is proposed based on a novel dynamic local filter network. In contrast to the explicit approach, this new mechanism avoids the introduction of additional errors coming from the generation of flow maps that delineate pixel displacements in horizontal and vertical directions, and the generated sample-specific and position-specific dynamic local filters reinforce the temporal consistency across video frames. Moreover, a global refinement network based on ResBlock and autoencoder structures is proposed to exploit non-local correlations and enhance the spatial consistency of super-resolved frames.

The second article is also about VSR. However, it is more practical and aims at the applications in real scenarios. Note that the existing VSR methods (including

the proposed method in the first article) exclusively make use of videos produced by the Image Signal Processor (ISP) of the camera system as inputs. However, this pipeline neglects the information loss incurred by non-invertible operations in ISP (*e.g.*, compression), and does not conform to the real image pipeline where VSR should serve as a pre-processing unit of ISP. To address these issues, a Raw Video Dataset (RawVD) is laboriously collected to train a novel method (named RawVSR) that can accommodate camera sensor data as inputs. More specifically, a Successive Deep Inference (SDI) module is designed in accordance with the architectural principle suggested by a canonical decomposition result for Hidden Markov Model (HMM) inference, and a reconstruction module is built with elaborately designed Attention-based Residual Dense Blocks (ARDBs).

As for the last article, it focuses on addressing the recent concern of image manipulation. A Progressive Spatio-Channel Correlation Network (PSCC-Net) is developed to detect and localize image manipulations. It consists of two paths: a top-down path that extracts local and global features and a bottom-up path that detects whether the input image is manipulated, and estimates its manipulation masks at 4 scales, where each mask is conditioned on the previous one. Different from the conventional encoder-decoder and no-pooling structures, PSCC-Net leverages features at different scales with dense cross-connections to produce manipulation masks in a coarse-to-fine fashion. Moreover, a Spatio-Channel Correlation Module (SCCM) is proposed to capture both spatial and channel-wise correlations among extracted features, enabling the network to cope with a wide range of manipulation attacks. The PSCC-Net is lightweight and can process $1,080$P images at 50+ FPS.

The rest of this thesis is organized as follows:

- **Chapter 2:** The details of the proposed LCVSR that is based on the new mechanism for implicit motion estimation and compensation.
- **Chapter 3:** The details of the proposed RawVSR and the demonstration of the collected RawVD.
- **Chapter 4:** The details of the proposed PSCC-Net for image manipulation detection and localization.
- **Chapter 5:** The conclusion of this thesis and the discussion of future work.

Bibliography

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*. PMLR, 2015, pp. 448–456.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[5] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[6] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–130, 2009.

[7] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1422–1430.

[8] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations (ICLR)*, 2018.

[9] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 667–676.

[10] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[11] R. Patidar, L. Sharma *et al.*, "Credit card fraud detection using neural network," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 1, no. 32-38, 2011.

[12] A. Alghoul, S. Al Ajrami, G. Al Jarousha, G. Harb, and S. S. Abu-Naser, "Email classification using artificial neural network," 2018.

[13] D. J. Livingstone, *Artificial Neural Networks: Methods and Applications*. Springer, 2008.

[14] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.

[15] 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 (CVPR)*, 2015, pp. 3431–3440.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2961–2969.

[17] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (ICCV)*, 2019, pp. 4690–4699.

[18] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2008.

[19] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, 2014, p. 17241734.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[22] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on visual transformer," *arXiv preprint arXiv:2012.12556*, 2020.

[23] X. Liu, L. Kong, Y. Zhou, J. Zhao, and J. Chen, "End-to-end trainable video super-resolution based on a new mechanism for implicit motion estimation and compensation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 2416–2425.

[24] X. Liu, K. Shi, Z. Wang, and J. Chen, "Exploit camera raw data for video super-resolution via hidden markov model inference," *IEEE Transactions on Image Processing*, vol. 30, pp. 2127–2140, 2021.

[25] X. Liu, Y. Ma, Z. Shi, L. Dai, and J. Chen, "Towards a unified approach to single image deraining and dehazing," *arXiv preprint arXiv:2103.14204*, 2021.

[26] X. Liu, Y. Ma, Z. Shi, and J. Chen, "Griddehazenet: Attention-based multi-scale network for image dehazing," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 7314–7323.

[27] X. Liu, Z. Shi, Z. Wu, and J. Chen, "Griddehazenet+: An enhanced multi-scale network with intra-task knowledge transfer for single image dehazing," *arXiv preprint arXiv:2103.13998*, 2021.

[28] Z. Shi, X. Liu, K. Shi, L. Dai, and J. Chen, "Video frame interpolation via generalized deformable convolution," *IEEE Transactions on Multimedia*, 2021.

[29] Z. Shi, C. Li, L. Dai, X. Liu, J. Chen, and T. N. Davidson, "Learning for unconstrained space-time video super-resolution," *arXiv preprint arXiv:2102.13011*, 2021.

[30] P. H. Hennings-Yeomans, S. Baker, and B. V. Kumar, "Simultaneous super-resolution and feature extraction for recognition of low-resolution faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008, pp. 1–8.

[31] B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng, "Aod-net: All-in-one dehazing network," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4770–4778.

[32] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to see in the dark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3291–3300.

[33] A. Ignatov, L. Van Gool, and R. Timofte, "Replacing mobile camera isp with a single deep learning model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 536–537.

[34] X. Xu, Y. Ma, and W. Sun, "Towards real scene super-resolution with raw images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1723–1731.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, pp. 1097–1105, 2012.

[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 21–37.

[37] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5781–5790.

[38] X. Liu, Y. Liu, J. Chen, and X. Liu, "Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization," *arXiv preprint arXiv:2103.10596*, 2021.

The following chapter is reproduced from an IEEE published paper:

Xiaohong Liu, Lingshi Kong, Yang Zhou, Jiying Zhao, and Jun Chen. "End-To-End Trainable Video Super-Resolution Based on a New Mechanism for Implicit Motion Estimation and Compensation." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2416-2425, 2020.

Contribution Declaration: Xiaohong Liu (the author of this thesis) is the first author and the main contributor of this paper (more than 95%). He proposed the algorithm, wrote the corresponding program, conduct all experiments, and composed the paper. Lingshi Kong helped organizing the experimental data; Yang Zhou participated in the discussions at the beginning; Prof. Jiying Zhao is the supervisor of Xiaohong Liu and Yang Zhou during their master's study; Prof. Jun Chen is the Ph.D. supervisor of Xiaohong Liu.

Chapter 2

End-To-End Trainable Video Super-Resolution Based on a New Mechanism for Implicit Motion Estimation and Compensation

2.1 Abstract

Video super-resolution aims at generating a high-resolution video from its low-resolution counterpart. With the rapid rise of deep learning, many recently proposed video super-resolution methods use convolutional neural networks in conjunction with explicit motion compensation to capitalize on statistical dependencies within and across low-resolution frames. Two common issues of such methods are noteworthy. Firstly, the quality of the final reconstructed HR video is often very sensitive to the accuracy

of motion estimation. Secondly, the warp grid needed for motion compensation, which is specified by the two flow maps delineating pixel displacements in horizontal and vertical directions, tends to introduce additional errors and jeopardize the temporal consistency across video frames. To address these issues, we propose a novel dynamic local filter network to perform implicit motion estimation and compensation by employing, via locally connected layers, sample-specific and position-specific dynamic local filters that are tailored to the target pixels. We also propose a global refinement network based on ResBlock and autoencoder structures to exploit non-local correlations and enhance the spatial consistency of super-resolved frames. The experimental results demonstrate that the proposed method outperforms the state-of-the-art, and validate its strength in terms of local transformation handling, temporal consistency as well as edge sharpness.

2.2 Introduction

Super-Resolution (SR) is considered as a promising technique to produce High-Resolution (HR) pictorial data using Low-Resolution (LR) sensors without resorting to hardware upgrades. Over the past few decades, it has received significant attention in a wide range of areas, including, among others, medical imaging [1, 2], satellite imaging [3–5] and surveillance [6–8]. Recently, it has also been used as a pre-processing step to facilitate various recognition tasks by enhancing the raw data [9, 10]. Super-resolution can be divided into two categories: Single Image Super-Resolution (SISR) and Video Super-Resolution (VSR). SISR can be viewed as a certain sophisticated image interpolation operation, which attempts to supply the missing details by strategically exploiting the spatial patterns in LR inputs. In contrast, VSR takes advantage of

both spatial and temporal relationships among consecutive frames to improve the quality of reconstructed videos. The traditional approaches to the VSR problem typically consist of three sub-tasks: sub-pixel motion estimation, motion compensation and up-sampling [11–14]. In general, motions across LR frames are estimated explicitly and the estimated LR displacements are employed to compensate sub-pixel motions by warping relevant LR frames to the target frame; the compensated LR frames are then fused to reconstruct the corresponding HR frame. The existing deep-learning-based VSR methods largely follow similar approaches [15–18]. One common limitation of such methods is that the super-resolved frames are very sensitive to the accuracy of the motion estimation, rendering the quality of SR outputs unstable.

In this paper, we propose a new approach to VSR using local dynamic filters via Locally Connected (LC) layers for implicit motion compensation and demonstrate its competitive advantages over the existing ones. The effectiveness of this LCVSR approach can be attributed to three major factors: 1) The overall system is end-to-end trainable and does not require any pre-training; the accuracy of motion estimation improves progressively through the training process. 2) Local motion estimation and compensation is performed implicitly by a novel Dynamic Local Filter Network (DLFN) with LC layers. There are at least two benefits of using the DLFN. Firstly, the implicit motion estimation, realized by sample-specific and position-specific dynamic local filters generated on-the-fly according to the target pixels, can deal with complicated local transformations in video frames such as regional blurring, irregular local movement and photometric changes. Secondly, the simultaneous action of dynamic local filters on all input LR frames via LC layers helps to maintain the temporal consistency. 3) The spatial consistency of super-resolved outputs is enforced

(a) Bicubic (b) Our Result (c) Ground Truth

Figure 2.1: The comparison of the bicubic interpolation, our result and the ground truth with the scale ratio set to 4.

by a novel Global Refinement Network (GRN) constructed using ResBlock and autoencoder structures. Since the implicit motion estimation performed by the DLFN is spatially localized, it may cause inconsistencies across neighboring areas. As such, the GRN plays a critical role of restoring the spatial consistency. Moreover, the GRN has the capability of exploiting non-local correlations due to its constituent autoencoder structure, which makes up for the lack of global motion estimation in the DLFN. Fig. 2.1 shows the comparison of the bicubic interpolation (the LR input), our result (the HR output) and the ground truth for a sample video frame.

2.3 Related Work

Many SISR and VSR methods have been proposed over the past few decades. The traditional methods typically solve the SR problem, which is inherently under-determined, by formulating it as a certain regularized optimization problem [11,12,14,19–22]. The

recent years, however, have witnessed the increasing dominance of deep-learning-based SR methods. The work by Dong *et al.* [23] is among the earliest ones that brought Convolutional Neural Networks (CNNs) to bear upon SISR. In their proposed SRCNN, a very shallow network is used to extract LR features, which are subsequently leveraged to generate HR images via non-linear mapping. To avoid time-consuming operations in the HR space, Shi *et al.* [24] proposed an Efficient Sub-Pixel Convolution Network (ESPCN) to extract and map features from the LR space to the HR space using convolutional layers instead of naive pre-defined interpolations such as bilinear or bicubic. Zhang *et al.* [25] designed a residual dense block with direct connections for the purpose of a more thorough extraction of local features from LR images.

Compared with SISR, VSR is inherently more complex due to the additional challenge of harnessing the relevant information in the temporal domain. To cope with this challenge, Kappeler *et al.* [15] proposed to employ the handcrafted optical flow method by Drulea and Nedevschi [26] to compensate motions across input frames and then feed the compensated frames into a pre-trained CNN to perform the SR operation. Huang *et al.* [27] developed a new VSR method based on the so-called Bidirectional Recurrent Convolutional Network (BRCN). The BRCN is a variant of Recurrent Neural Network (RNN) with commonly-used recurrent connections replaced by weight-sharing convolutional connections; as a consequence, it inherits the strength of RNN in terms of capturing long-term temporal dependencies and at the same time admits a more efficient implementation. Liao *et al.* [28] introduced a SR draft-ensemble approach in which multiple SR drafts are generated using an optical flow method with different estimation settings and then synthesized by a carefully

constructed CNN to produce the final HR output.

To avoid inaccurate motion estimation caused by fixed temporal radius, Liu *et al.* [17] proposed a temporal adaptive neural network. This network has several SR inference branches, each with a different temporal radius; the final HR output is obtained by adaptively aggregating all SR inferences. They also introduced a spatial alignment network that can efficiently estimate motions between neighboring frames. The VESPCN developed by Caballero *et al.* [16] combines spatio-temporal networks with an end-to-end trainable spatial transformer module to generate the super-resolved video. Based on the motion compensation module in the VESPCN [16], Tao *et al.* [13] designed a Sub-Pixel Motion Compensation (SPMC) layer to compensate motion and up-sample video frames simultaneously. Moreover, they advocated the use of an encoder-decoder style structure with ConvLSTM [29] and skip-connections [30] for effectively processing sequential videos and reducing the training time.

To improve the temporal consistency of super-resolved videos, Sajjadi *et al.* [18] proposed a frame recurrent VSR method, which enables the processing of the current frame to benefit from the inferred SR results for the previous frames. This method is more efficient than those treating the VSR problem as a sequence of multi-frame SR problems due to the recurrent nature of its operations.

Jo *et al.* [31] developed a novel VSR method, known as VSRDUF, which works as follows: A deep neural network is employed to generate dynamic upsampling filters and frame residuals; certain provisional HR frames are constructed from their LR counterparts through dynamic upsampling filters, and the inferred residuals are then added to such frames to produce the final output. This work is most related to ours

in the sense that motion compensation is only performed implicitly. However, it will be seen that the underlying mechanisms are fundamentally different.

2.4 Method

In this section we give a detailed description of the proposed VSR method with an emphasis on its most prominent feature, namely, the use of local dynamic filters via LC layers to implicitly compensate motions across video frames. The process starts with converting the input LR frames from RGB to YCbCr color space. Only the Y channels are fed into the proposed VSR system, which helps to reduce the computational complexity. The Cb and the Cr channels are upsampled via bicubic interpolation and merged with the super-resolved Y channels to generate HR frames in YCbCr, from which the final result in RGB is obtained. Let $Y_t \in \mathbb{R}^{H\times W}$ denote the t-th LR frame in the Y channel degraded by blurring and down-sampling operations from the corresponding HR frame $X_t \in \mathbb{R}^{rH\times rW}$, where r is the scale ratio. The given LR video sequence of C consecutive frames centered at Y_t is denoted as $\{Y_{t-T:t+T}\}$, where T is the temporal radius and $C = 2T + 1$. The corresponding HR video sequence is $\{X_{t-T:t+T}\}$. We use F_{LCVSR} as the functional representation of the proposed LCVSR system with the end-to-end relation

$$\hat{X}_t = F_{LCVSR}(\{Y_{t-T:t+T}\}; \theta_{LCVSR}), \tag{2.4.1}$$

where $\hat{X}_t$ is the reconstructed HR frame and θ_{LCVSR} denotes the ensemble of the system parameters. Regardless of the batch size, the shape of the input tensor is set to be $C \times H \times W$ while that of the output is set to be $1 \times rH \times rW$. The

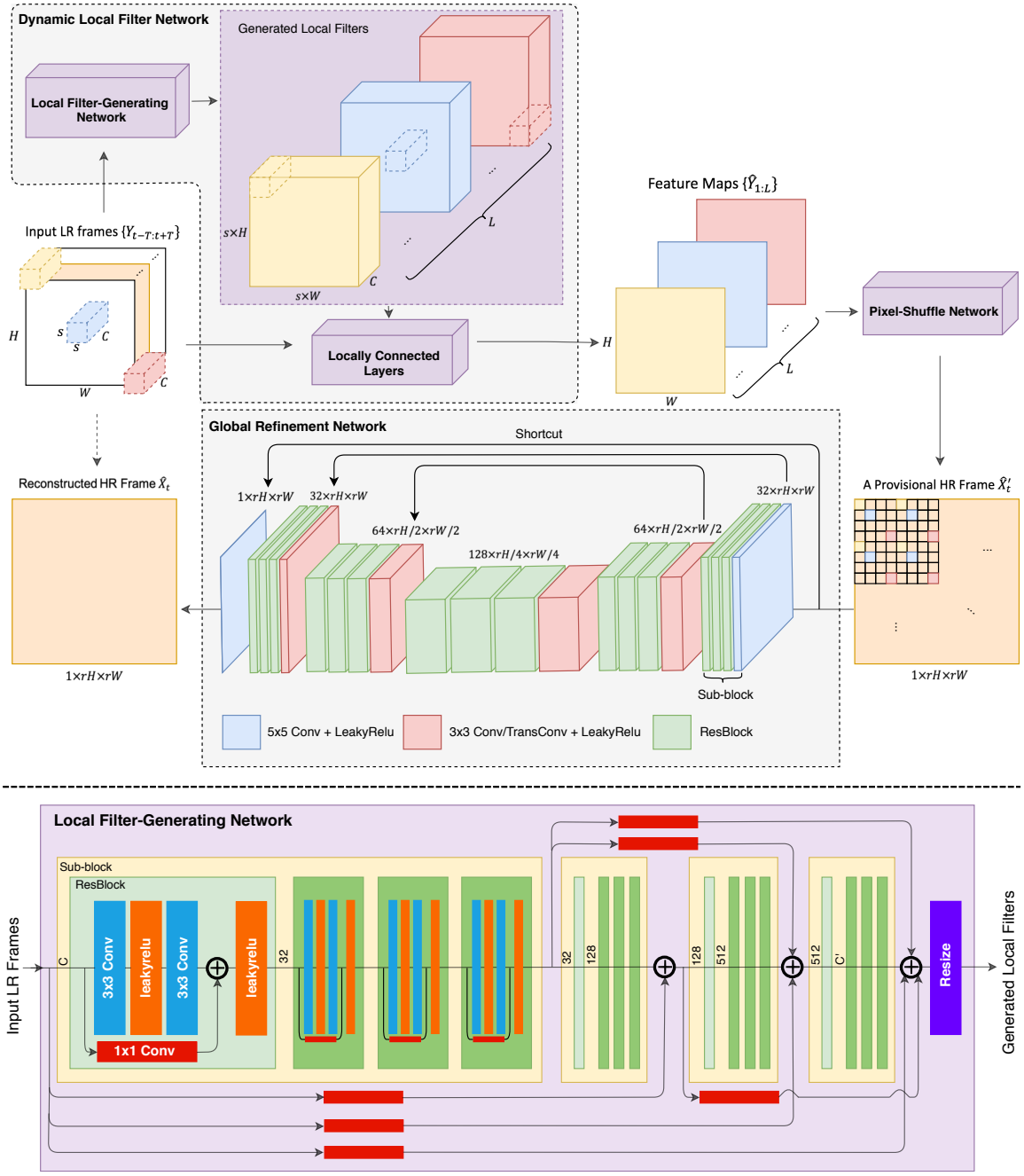


Figure 2.2: The upper part shows the overall architecture of the proposed LCVSR system. The bottom part provides a detailed illustration of the LFGN.

proposed LCVSR system, as shown in Fig. 2.2, consists of three modules, which are respectively the Dynamic Local Filter Network (DLFN), the pixel-shuffle network [24] and the Global Refinement Network (GRN). The input LR frames $\{Y_{t-T:t+T}\}$ are first fed into the DLFN, which has two sub-modules: Local Filter-Generating Network (LFGN) and LC layers. The LFGN produces sample-specific and position-specific dynamic local filters filters on-the-fly according to the spatio-temporal relationship among the inputs. These dynamic local filters, each of size $s \times s \times C$ (with $s = 2d+1$, where d is the spatial radius), then act on the LR frames via LC layers to generate the feature maps $\{\hat{Y}_{1:L}\}$ (we set $L = r^2$ in this work). These feature maps are forwarded to the pixel-shuffle network to construct a provisional HR frame $\hat{X}'_t$, which is subsequently fed into the GRN to enhance the spatial consistency and cope with global transformations. The output of the GRN is the reconstructed HR frame $\hat{X}_t$.

2.4.1 Dynamic Local Filter Network

The existing VSR methods typically compensate motions across video frames by explicitly estimating pixel displacements in horizontal and vertical directions. This error-prone estimation step may potentially jeopardize the quality of SR results. Therefore, it is of considerable interest to develop deep-learning-based techniques for implicit motion estimation and compensation. One possible approach is to use the conventional CNNs with weight-sharing filters, which have been shown to achieve outstanding performance in image classification and segmentation tasks [32,33]. However, motion, blur and photometric changes encountered in the VSR problem are usually sample-specific and position-specific, in other words, each pixel in a video frame may exhibit a unique degradation pattern, which cannot be effectively exploited by

the weight-sharing filters. For this reason, we propose a DLFN with LC layers that can perform local operations tailored to the spatio-temporal characteristics of the target pixels. Specifically, a sample-specific and position-specific dynamic local filter is generated for each pixel in the input LR frames; these dynamic local filters then collectively act on the input frames via LC layers to generate feature maps by compensating motions and other transforms in an implicit manner.

Let $\Theta_l \in \mathbb{R}^{sH \times sW \times C}$ denote the lth set of (unbiased) local filters. Each local filter in Θ_l (say, $\Theta_{i,j,l} \coloneqq \{\Theta_{i,j,l}^{(m,n,k)} : m,n = 1,2,3; k = 1,\cdots,C\}$) is associated with a specific pixel (say, the (i,j)-pixel) in the tth LR frame. The lth feature map $\hat{Y}_l$ is obtained by applying Θ_l on the input LR frames $\{Y_{t-T:t+T}\}$. More precisely, we have

$$\hat{Y}_l^{(i,j)} = \sum_{m=i-d}^{i+d} \sum_{n=j-d}^{j+d} \sum_{k=t-T}^{t+T} \Theta_{i,j,l}^{(m-i+d+1,n-j+d+1,k-t+T+1)} \cdot Y_k^{(m,n)}, \tag{2.4.2}$$

where $\hat{Y}_l^{(i,j)}$ represents the value of the (i,j)-pixel in the lth feature map, and $Y_k^{(m,n)}$ denotes the (m,n)-pixel in the kth LR frame. It is worth pointing out that both Θ_l and $\hat{Y}_l$ depend on t and should actually be written as $\Theta_{t,l}$ and $\hat{Y}_{t,l}$ respectively; here we suppress the subscript t for notational simplicity.

To generate dynamic local filters, we build a novel LFGN based on ResBlocks [34]. Its input-output relationship can be expressed as

$$\Theta = F_{LFGN}(\{Y_{t-T:t+T}\}; \theta_{LFGN}), \tag{2.4.3}$$

where F_{LFGN} is the functional representation of the LFGN and θ_{LFGN} denotes the ensemble of its parameters. Note that the output $\Theta \coloneqq \{\Theta_{1:L}\} \in \mathbb{R}^{C \times sH \times sW \times L}$ is a 4-D tensor (which consists of all dynamic local filters) whereas the input of the

LFGN is a 3-D tensor of shape $C \times H \times W$. To generate Θ based on $\{Y_{t-T:t+T}\}$, we employ modified ResBlocks in concatenation to progressively increase the depth of the input tensor from C to C', where $C' = C \times s^2 \times L$, then resize the resulting 3-D tensor of shape $C' \times H \times W$ to a 4-D tensor of shape $C \times sH \times sW \times L$. The LFGN consists of one Resize module and four sub-blocks, each of which is built using ResBlocks. We find that using more ResBlocks in each sub-block leads to better performance. However, to strike a balance between system performance and computational complexity, in each sub-block we deploy one ResBlock for depth enlargement and three ResBlocks with no shape change. Besides, grouped convolutions [35] are also utilized. The four sub-blocks are densely connected by shortcuts to facilitate information exchange among them. If the tensors at the two sides of a shortcut have different shapes, a 1×1 convolution is performed to make the shape compatible; otherwise, we directly connect the two sides without modification. Each ResBlock consists of two 3×3 convolutional layers, two LeakyReLU layers [36] and an inner shortcut. The output of the DLFN consists of L feature maps, each of which is generated by exploiting, via implicit motion compensation, the relevant spatio-temporal information in all LR frames. These feature maps are then fed into the pixel-shuffle network to construct a provisional HR frame $\hat{X}'_t$.

It is worth emphasizing that dynamic local filters are intermediate computational results produced within the proposed system and should not be viewed as the parameters of the system itself. They are generated by the LFGN based on the input LR frames, then act back on the input frames, via LC layers, to perform pixel-level fine-grained motion estimation and compensation. More generally, this is an effective mechanism for leveraging the learning capability of a deep neural network (say, the

LFGN in the current setting) to realize dynamic localized functionalities. See Sections 2.5.1 and 2.5.4 for some supporting experimental results.

2.4.2 Global Refinement Network

Since the proposed DLFN performs localized motion estimation and compensation across LR frames, it can potentially cause inconsistencies among neighboring areas. To address this issue, we propose a GRN (see Fig. 2.2) employing ResBlock and autoencoder structures to improve the spatial consistency of super-resolved frames. The autoencoder structure enlarges the receptive field so that the GRN also has the ability to deal with global transformations, which makes up for the lack of global motion estimation in the DLFN. The GRN mainly consists of five sub-blocks connected by shortcuts. Each sub-block contains a convolutional layer or a transposed convolutional layer with LeakyReLu as the activation function, followed by three ResBlocks that are structurally the same as those in the DLFN. The encoder, formed by the second and third sub-blocks, reduces the spatial dimension but increases the depth dimension to enlarge the receptive field progressively. In contrast, the decoder, formed by the last two sub-blocks, reduces the depth dimension but increases the spatial dimension to perform global refinement. Finally, a 5×5 convolutional layer activated by LeakyRelu produces the reconstructed HR frame. The input-output relationship of the proposed GRN is given by

$$\hat{X}_t = F_{GRN}(\hat{X}'_t; \theta_{GRN}), \tag{2.4.4}$$

where F_{GRN} is the functional representation of the GRN and θ_{GRN} denotes the ensemble of its parameters.

2.4.3 Data Preparation

Deep-learning-based VSR methods rely heavily on the quality and the quantity of the training datasets. Unfortunately, so far there is no standard training dataset for VSR. To build our own, we totally collect 100k ground-truth sequences, each with 7 consecutive frames of size 252×444, where 70k sequences are selected from the Vimeo-90k dataset recently built by Xue *et al.* [37] and the rest 30k sequences are extracted from several videos provided by Harmonic[1]; as a comparison, the current state-of-the-art VSRDUF uses 160k sequences for training. We adopt the Vid4 dataset [21] and the SPMCS dataset [13] for testing. Our input LR frames are generated from the ground-truth sequences via Gaussian blur and downsampling. For the Gaussian blur, we set the standard deviation to be 1 and the kernel size to be 3×3. As to the downsampling operation, we choose the scale ratio $r = 3, 4$ (considered to be the most challenging cases in the VSR task).

2.4.4 Implementation

The proposed LCVSR system is end-to-end trainable and no pre-training is needed for sub-networks. Our training is carried out on a PC with two NVIDIA GeForce GTX 1080 Ti, but only one GPU is used for testing. We adopt Xavier initialization [38] and set the mini-batch size to be 12. The $\mathcal{L}_2$ loss function is used to calculate the reconstruction error as follows:

$$\mathcal{L}_2(X_t, \hat{X}_t) = \left\| X_t - \hat{X}_t \right\|_2^2. \tag{2.4.5}$$

[1] https://www.harmonicinc.com/free-4k-demo-footage/

We train the proposed system for about 0.8 million iterations using the Adam optimizer [39] with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The learning rate is set to 10^{-4} at the beginning and decays to 10^{-5} after 0.7 million iterations. Our source code will be made publicly available.

2.5 Experimental Results

In our experiments, we set $C = 7$, $T = 3$, $s = 3$ and $d = 1$. As such, one super-resolved frame is generated based on 7 consecutive LR frames with the middle one as the reference, and the size of generated dynamic local filters is $3 \times 3 \times 7$. We use PSNR and SSIM for quantitative assessment of the SR results. All PSNR values are calculated based on the Y channel using the ITU-R BT.601 standard to make fair comparisons [16]. In addition to the aforementioned quantitative performance metrics, we also consider qualitative measures such as edge sharpness and temporal consistency. The following existing VSR methods are chosen as benchmarks: *Bayesian* [21], *VSR-Net* [15], *VESPCN* [16], $B_{1,2,3} + T$ [17], *SPMC* [13], *FRVSR* [18] and *VSRDUF* [31]. For the *VSRDUF*, both its basic version with 16 layers (*DUF-16L*) and the enhanced version with 52 layers (*DUF-52L*) are used for comparisons. The quantitative experimental results of these benchmarks are obtained using the provided source codes (if available) or cited from the original papers.

2.5.1 Visualization of Dynamical Local Filters

Although the performance of the proposed LCVSR system benefit from many contributing factors, arguably the most crucial one is the use of dynamic local filters, via

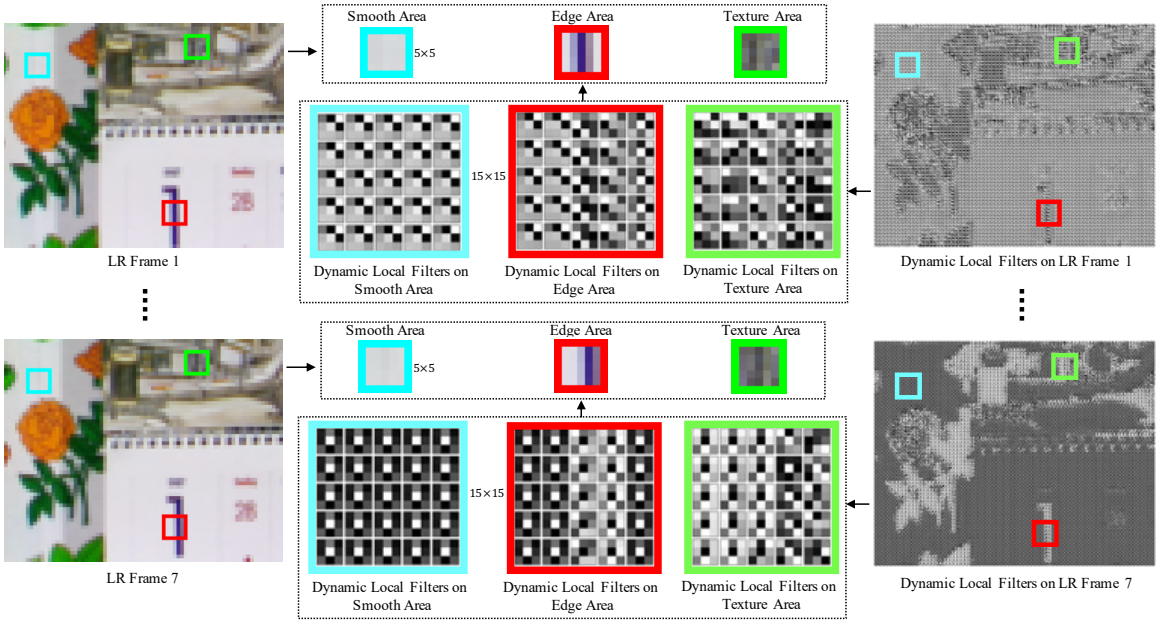


Figure 2.3: Dynamic local filters applied on smooth, edge and texture areas respectively.

LC layers, for implicit motion estimation and compensation. To gain a better understanding, it is instructive to distinguish generated filters from learned filters [40]. The learned filters such as those in the LFGN update themselves only during the training process and become static afterwards. On the contrary, the generated filters are adaptive in the sense that they are not fully specified until the input is given. The dynamic local filters in the proposed system belong to this category. They are computed based on the input LR frames and used as the kernel weights in LC layers; moreover, to reduce the computational complexity, they are applied to the LR space rather than the HR space.

Fig. 2.3 illustrates the dynamic local filters that are used to generate the first feature map $\hat{Y}_1$, shown as the yellow cube in Fig. 2.2. It also shows some sample patches of size 5×5 extracted from smooth, edge and texture areas in the first and the last

LR frames together with their corresponding dynamic local filters, which are of size 15×15 (since $s = 3$). It can be seen from Fig. 2.3 that the dynamic local filters are spatially content-adaptive within each frame and temporally distinct (even when the associated pixels are of similar nature) across different frames. This provides supporting evidence for their ability to adapt according to the spatio-temporal characteristics of the target pixels and perform implicit motion estimation and compensation. It can also been seen from Fig. 2.3 that, in a given frame, the dynamic local filters for pixels with a homogeneous neighborhood tend to have similar patterns, which helps to retain intra-frame spatial consistency. For example, the local filters associated with the pixels in the smooth area and those in the edge area (except the exact edge pixels) are quite alike. In contrast, each pixel in the texture area has a distinct dynamic local filter, which is essential since this area is very sensitive to motions.

2.5.2 Analysis of Temporal Consistency

The Lack of temporal consistency may cause flickering artifacts that manifest visually, for instance, in the form of jagged edges. To validate that our proposed method maintains temporal consistency, we extract spatially co-located rows from consecutive super-resolved frames of *Calendar* in the Vid4 dataset with the scale ratio $r = 4$, and arrange them vertically to compose a temporal profile [18]; we also compose a temporal profile for the *City* frames in the same dataset. Fig. 2.4 shows the comparison of our temporal profiles and the corresponding ones generated by five existing VSR methods. Overall, the proposed method presents the most consistent temporal profiles. In fact, its maintenance of temporal consistency remains good even in some cases where the ground-truth frames have certain artifacts in this respect (see, e.g.,

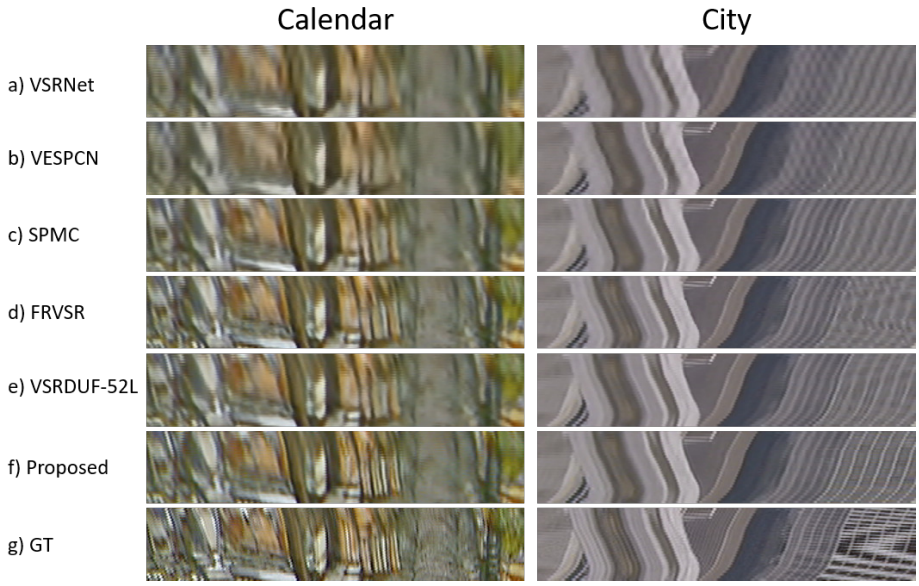


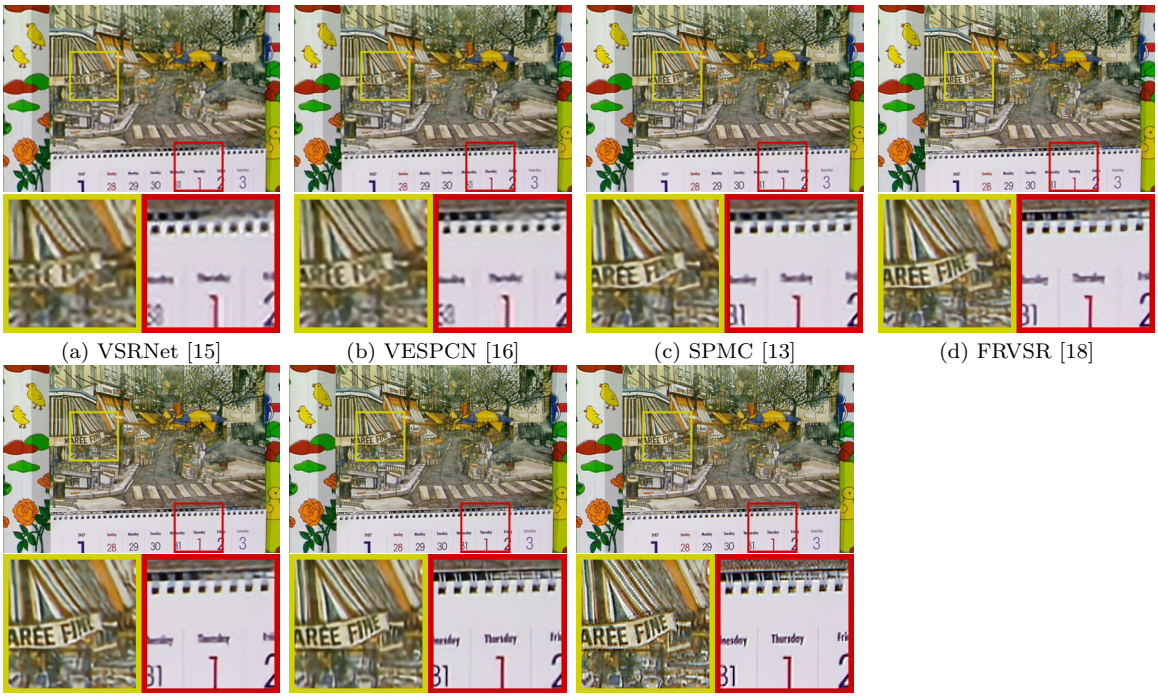
Figure 2.4: Visual comparison of the temporal profiles generated by the proposed VSR method and some existing ones for super-resolved *Calendar* and *City* frames in Vid4.

City). Moreover, it can be seen from the temporal profiles that our method produces the sharpest edges, as compared to the other ones, in all *Calendar* and *City* frames.

2.5.3 Qualitative and Quantitative Comparisons

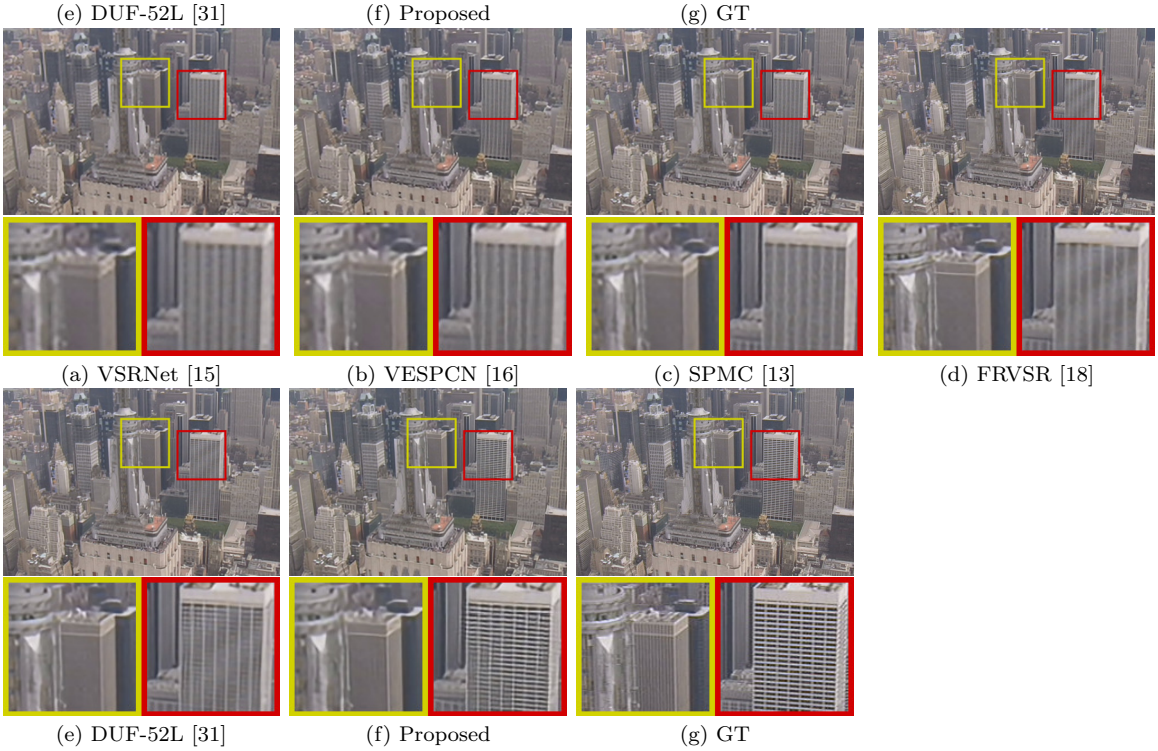
The proposed method is compared to several existing VSR methods qualitatively and quantitatively with a particular focus on the *VSRDUF*, which is the current state-of-the-art.

In the *VSRDUF*, the generated Dynamic Upsampling Filters (DUF) are only applied on the central LR frame to reconstruct the SR one without joint consideration of all input frames. This mechanism would cause fuzzy edges and temporal inconsistencies (see Fig. 2.4 (e)). In contrast, the generated Dynamic Local Filters (DLF) are applied on all input LR frames to reconstruct the SR one, yielding better visual



(a) VSRNet [15] (b) VESPCN [16] (c) SPMC [13] (d) FRVSR [18]

(e) DUF-52L [31] (f) Proposed (g) GT



(a) VSRNet [15] (b) VESPCN [16] (c) SPMC [13] (d) FRVSR [18]

(e) DUF-52L [31] (f) Proposed (g) GT

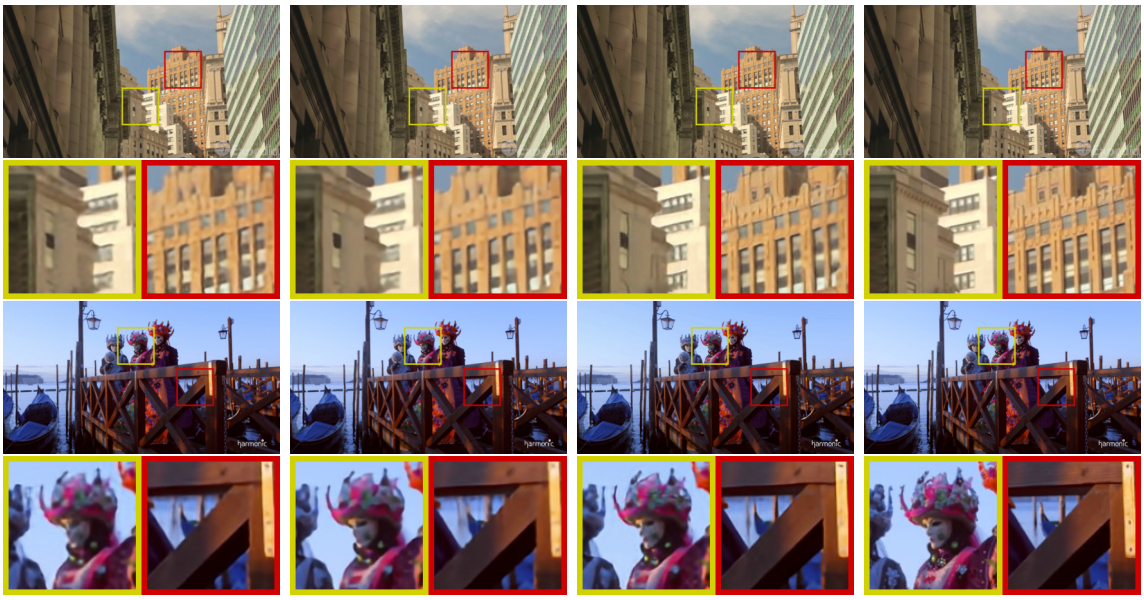
(a) VSRNet [15] (b) VESPCN [16] (c) SPMC [13] (d) FRVSR [18]

(e) DUF-52L [31] (f) Proposed (g) GT

(a) VSRNet [15] (b) VESPCN [16] (c) SPMC [13] (d) FRVSR [18]

(e) DUF-52L [31] (f) Proposed (g) GT

Figure 2.5: Visual comparisons on the Vid4 dataset with $r = 4$.



(a) SPMC [13] (b) DUF52 [31] (c) Proposed (d) GT

Figure 2.6: Visual comparisons for *NewYork* and *Venice* from the SPMCS dataset with $r = 4$.

quality in terms of edge sharpness and temporal consistency (see Fig. 2.4 (f)), and achieving higher PSNR and SSIM values. Moreover, the essence of the DUF is still a weight-sharing filter that has been widely used in conventional CNNs. However, the proposed DLF is spatially content adaptive (position-specific) within each frame and temporally distinct (sample-specific) among different frames. This is a new mechanism and is not like conventional CNNs. The reason we employ the DLF in VSR is based on the observation that each pixel in a video frame may exhibit a unique degradation pattern, which cannot be effectively exploited by the weight-sharing filters.

Fig. 2.5 shows the qualitative comparisons of different methods on the Vid4 dataset with $r = 4$, and Table 2.1 demonstrates the quantitative comparisons in terms of average PSNR and SSIM values on the same dataset with $r = 3, 4$. Note that our SR results contain more fine details and restore sharper edges. Moreover, the PSNR

value achieved by the proposed method is 0.61 dB higher than that by the *DUF-16L* when $r = 3$ and 0.66 dB higher when $r = 4$. Even compared with the *DUF-52L*, our result is still 0.13 dB higher when $r = 4$. We have also performed the test on the SPMCS dataset, which contains 31 video clips, for further qualitative and quantitative comparisons. It can be seen from Fig. 2.6 and Table 2.2 that the proposed method performs competitively on this dataset as well.

2.5.4 Ablation Study

To gain a better understanding of the effectiveness of the new mechanism for implicit motion estimation and compensation, we conduct an ablation study by directly feeding the output of the LFGN (via a convolutional layer) to the pixel-shuffle network; the resulting system is trained in the same way as before. Compared to the original system, this new system has (essentially) the same number of parameters but bypasses the action of dynamic local filters, via LC layers, on the input LR frames. As shown in Table 2.3, this modification leads to significant performance degradation on the Vid4 dataset, suggesting that the mechanism adopted by the original system is more effective in terms of exploiting the learning capability of the LFGN to realize dynamic localized functionalities. We also conducted the analysis on the role of the GFN by 1) removing it from the proposed system and 2) replacing it with the U-Net [41]. It can be seen from Table 2.3 that these two variants incur significant performance loss compared with our full model.

We further investigate the performance-complexity trade-off for the proposed system by varying the input length and the network size. Specifically, by employing 1, 2

Table 2.1: Quantitative comparisons on the Vid4 dataset with $r = 3, 4$. The best one is highlighted in **Bold**.

Vid4	Metric	*Bicubic*	*Bayesian*	*VSRNet*	*VESPCN*	$B_{1,2,3} + T$	*SPMC*	*FRVSR*	*DUF-16L*	*DUF-52L*	*Proposed*
x3	PSNR	25.28	25.82	26.79	27.25	-	27.49	-	28.90	-	**29.51**
	SSIM	0.7329	0.8323	0.8098	0.8447	-	0.8400	-	0.8898	-	**0.8964**
x4	PSNR	23.79	25.06	24.84	25.35	25.39	25.52	26.69	26.81	27.34	**27.47**
	SSIM	0.6332	0.7466	0.7049	0.7557	0.7490	0.7600	0.8220	0.8145	0.8327	**0.8394**

Table 2.2: Quantitative comparisons on the SPMCS dataset with $r = 3, 4$. The best one is highlighted in **Bold**.

SPMCS	Metric	*SPMC*	*DUF-16L*	*DUF-52L*	*Proposed*
x3	PSNR	32.10	-	-	**33.91**
	SSIM	0.9000	-	-	**0.9358**
x4	PSNR	29.89	30.01	30.39	**30.66**
	SSIM	0.8400	0.8355	0.8646	**0.8711**

Table 2.3: Quantitative comparisons on the Vid4 dataset for different variants of the proposed method with $r = 3, 4$. The best one is highlighted in **Bold**.

Vid4	Metric	w/o LC Layers	w/o GRN	w/ U-Net [41]	Our full model
x3	PSNR	27.27	28.13	29.20	**29.51**
	SSIM	0.8471	0.8752	0.8896	**0.8964**
x4	PSNR	25.45	26.20	27.29	**27.47**
	SSIM	0.7530	0.8134	0.8352	**0.8394**

and 3 ResBlocks (the ones without shape change) in both DLFN and GRN, we construct three different configurations of the proposed system, denoted by LCVSR-Res1, LCVSR-Res2 and LCVSR-Res3, respectively.

Fig. 2.7 plots the PSNR against the average runtime for each configuration on the Vid4 dataset with the number of input LR frames set to be 1, 3, 5 and 7. It can be seen that increasing the input length leads to higher PSNRs at the cost of longer runtimes. When the input length is 1, the VSR task degenerates to the SISR task, which only exploits intra-frame dependencies. Increasing the input length from 1 to 3 significantly improves the PSNR values due to the additional freedom of exploring inter-frame dependencies via motion estimation and compensation. Further increasing the input length provides more spatial and temporal information that can be capitalized on, which helps to generate better SR results. However the improvement

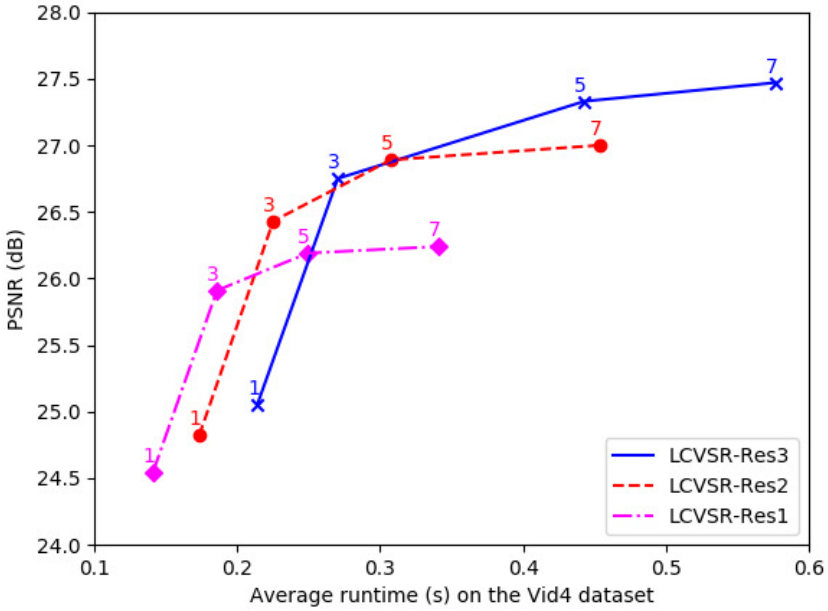


Figure 2.7: PSNR vs. runtime for different configurations of the LCVSR system on the Vid4 dataset with the input length set to be 1, 3, 5 and 7.

becomes negligible when the input length goes beyond 7. Employing more ResBlocks in DLFN and GRN has a similar effect. Indeed, it can be seen from Fig. 2.7 that the PSNR value increases progressively from LCVSR-Res1 to LCVSR-Res3 for the same input length, and the runtime follows the same trend. Note that the proposed LCVSR system corresponds to LCVSR-Res3 with input length 7. The above experimental results provide certain justifications for the design of the proposed system in consideration of the performance-complexity trade-off.

2.5.5 Network Parameters and Runtime Analysis

In table 2.4, the number of parameters and the runtime of different methods are demonstrated. Our method has slightly less parameters and faster runtime than the current state-of-the-art DUF-52L. Although the VSRNet has the least number of parameters and is close to real-time, it has the worst VSR performance.

Table 2.4: The number of parameters and average runtime of different methods for 1080p frames. The best one is highlighted in **Bold**.

Method	*VSRNet*	*VESPCN*	*SPMC*	*DUF-52L*	*Proposed*
Params.(M)	**0.39**	0.89	2.17	5.82	5.81
Time (s)	**0.23**	0.29	2.80	2.68	2.32

2.6 Conclusion

In this paper, we have proposed an end-to-end trainable VSR method based on a new mechanism for implicit motion estimation and compensation (realized through dynamic local filters and LC layers). Our experimental results demonstrate that the proposed method outperforms the current state-of-the-art in terms of local transformation handling, edge sharpness and temporal consistency. As part of our future work, we intend to explore the application of LC layers in deep neural network architectures for image/video restoration tasks.

Bibliography

[1] H. Greenspan, "Super-resolution in medical imaging," *The Computer Journal*, vol. 52, no. 1, pp. 43–63, 2008.

[2] D. H. Trinh, M. Luong, F. Dibos, J.-M. Rocchisani, C. D. Pham, and T. Q. Nguyen, "Novel example-based method for super-resolution and denoising of medical images," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1882–1895, 2014.

[3] H. Demirel and G. Anbarjafari, "Discrete wavelet transform-based satellite image resolution enhancement," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 1997–2004, 2011.

[4] F. Li, X. Jia, D. Fraser, and A. Lambert, "Super resolution for remote sensing images based on a universal hidden markov tree model," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 3, pp. 1270–1278, 2010.

[5] A. J. Tatem, H. G. Lewis, P. M. Atkinson, and M. S. Nixon, "Super-resolution target identification from remotely sensed images using a hopfield neural network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 4, pp. 781–796, 2001.

[6] B. K. Gunturk, A. U. Batur, Y. Altunbasak, M. H. Hayes, and R. M. Mersereau, "Eigenface-domain super-resolution for face recognition," *IEEE Transactions on Image Processing*, vol. 12, no. 5, pp. 597–606, 2003.

[7] K. Jia and S. Gong, "Generalized face super-resolution," *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 873–886, 2008.

[8] L. Zhang, H. Zhang, H. Shen, and P. Li, "A super-resolution reconstruction algorithm for surveillance images," *Signal Processing*, vol. 90, no. 3, pp. 848–859, 2010.

[9] P. H. Hennings-Yeomans, S. Baker, and B. V. Kumar, "Simultaneous super-resolution and feature extraction for recognition of low-resolution faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[10] D. Capel and A. Zisserman, "Super-resolution enhancement of text image sequences," in *15th International Conference on Pattern Recognition (ICPR)*, vol. 1, 2000, pp. 600–605.

[11] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multi-frame super resolution," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1327–1344, 2004.

[12] Z. Ma, R. Liao, X. Tao, L. Xu, J. Jia, and E. Wu, "Handling motion blur in multi-frame super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5224–5232.

[13] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia, "Detail-revealing deep video super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 22–29.

[14] X. Liu, L. Chen, W. Wang, and J. Zhao, "Robust multi-frame super-resolution based on spatially weighted half-quadratic estimation and adaptive btv regularization," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 4971–4986,

2018.

[15] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos, "Video super-resolution with convolutional neural networks," *IEEE Transactions on Computational Imaging*, vol. 2, no. 2, pp. 109–122, 2016.

[16] J. Caballero, C. Ledig, A. P. Aitken, A. Acosta, J. Totz, Z. Wang, and W. Shi, "Real-time video super-resolution with spatio-temporal networks and motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 7.

[17] D. Liu, Z. Wang, Y. Fan, X. Liu, Z. Wang, S. Chang, and T. Huang, "Robust video super-resolution with learned temporal dynamics," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2526–2534.

[18] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6626–6634.

[19] M. Elad and Y. Hel-Or, "A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1187–1193, 2001.

[20] T. Köhler, X. Huang, F. Schebesch, A. Aichert, A. Maier, and J. Hornegger, "Robust multiframe super-resolution employing iteratively re-weighted minimization," *IEEE Transactions on Computational Imaging*, vol. 2, no. 1, pp. 42–58, 2016.

[21] C. Liu and D. Sun, "On bayesian adaptive video super resolution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 2, pp. 346–360, 2014.

[22] X. Liu and J. Zhao, "Robust multi-frame super-resolution with adaptive norm choice and difference curvature based btv regularization," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017, pp. 388–392.

[23] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 184–199.

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1874–1883.

[25] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[26] M. Drulea and S. Nedevschi, "Total variation regularization of local-global optical flow," in *14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011, pp. 318–323.

[27] Y. Huang, W. Wang, and L. Wang, "Video super-resolution via bidirectional recurrent convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 1015–1028, 2018.

[28] R. Liao, X. Tao, R. Li, Z. Ma, and J. Jia, "Video super-resolution via deep draft-ensemble learning," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 531–539.

[29] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 802–810.

[30] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016, pp. 2802–2810.

[31] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3224–3232.

[32] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[33] R. C. Vijay Badrinarayanan, Alex Kendall, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, vol. 39, no. 12, pp. 2481 – 2495, 2017.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012, pp. 1097–1105.

[36] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[37] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp. 1106–1125, 2019.

[38] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010, pp. 249–256.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[40] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016, pp. 667–675.

[41] 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.

The following chapter is reproduced from an IEEE published paper:

Xiaohong Liu, Kangdi Shi, Zhe Wang, and Jun Chen. "Exploit Camera Raw Data for Video Super-Resolution via Hidden Markov Model Inference." IEEE Transactions on Image Processing, Vol. 30, pp. 2127-2140, 2021.

Contribution Declaration: Xiaohong Liu (the author of this thesis) is the first author and the main contributor of this paper (more than 90%). He proposed the algorithm, collect part of raw videos, wrote the corresponding program, conduct all experiments, and composed the paper. Kangdi Shi helped the data collection; Zhe Wang helped organizing the experimental data; Prof. Jun Chen is the Ph.D. supervisor of Xiaohong Liu.

Chapter 3

Exploit Camera Raw Data for Video Super-Resolution via Hidden Markov Model Inference

3.1 Abstract

To the best of our knowledge, the existing deep-learning-based Video Super-Resolution (VSR) methods exclusively make use of videos produced by the Image Signal Processor (ISP) of the camera system as inputs. Such methods are 1) inherently suboptimal due to information loss incurred by non-invertible operations in ISP, and 2) inconsistent with the real imaging pipeline where VSR in fact serves as a pre-processing unit of ISP. To address this issue, we propose a new VSR method that can directly exploit camera sensor data, accompanied by a carefully built Raw Video Dataset (RawVD) for training, validation, and testing. This method consists of a Successive Deep Inference (SDI) module and a reconstruction module, among others. The SDI

module is designed according to the architectural principle suggested by a canonical decomposition result for Hidden Markov Model (HMM) inference; it estimates the target high-resolution frame by repeatedly performing pairwise feature fusion using deformable convolutions. The reconstruction module, built with elaborately designed Attention-based Residual Dense Blocks (ARDBs), serves the purpose of 1) refining the fused feature and 2) learning the color information needed to generate a spatial-specific transformation for accurate color correction. Extensive experiments demonstrate that owing to the informativeness of the camera raw data, the effectiveness of the network architecture, and the separation of super-resolution and color correction processes, the proposed method achieves superior VSR results compared to the state-of-the-art and can be adapted to any specific camera-ISP. Code and dataset are available at `https://github.com/proteus1991/RawVSR`.

Index Terms — Video super-resolution, camera raw data, hidden Markov model inference

3.2 Introduction

Super-Resolution (SR) is a promising technique that can restore High-Resolution (HR) pictorial data from their Low-Resolution (LR) counterpart without requiring hardware upgrades. Over the past few decades, it has found applications in a wide range of areas such as medical imaging [1, 2], satellite imaging [3, 4] and surveillance [5, 6]. SR is also useful for improving the quality of data dedicated to high-level vision tasks [7–9]. There are two major categories of SR: Single Image Super-Resolution (SISR) and Video Super-Resolution (VSR). Although early studies [10–12] largely treat VSR as a simple extension of SISR by focusing on intra-frame spatial

correlation, more attention has been paid in recent works [13–16] to strategically exploiting inter-frame temporal correlation (typically in the form of motion estimation and compensation) to further improve the VSR results. The availability of an extra dimension in VSR creates both opportunities and challenges. Indeed, there is considerable freedom in the ways that multiple LR frames can be leveraged to reconstruct one target HR frame, especially with the advent of deep learning techniques. Even though many effective heuristics have been proposed, a theoretical guideline for the fusion process is still lacking.

The data-driven approach has become increasingly popular in VSR research. To the best of our knowledge, the existing data-driven VSR methods [13–16] exclusively make use of camera processed data as inputs, despite the fact that modern cameras are capable of providing raw data which are potentially more informative. Note that camera processed data are produced by the Image Signal Processor (ISP) from raw data through several operations, including image demosaicing, denoising, sharpening, color converting, tone adjustment, compression, among others [17–19]. Overall, the operations in ISP are non-invertible and tend to degrade the information content of the original data. For example, the bit-depth is reduced from 12-14 bits to 8 bits [20] due to quantization, and various artifacts are introduced by compression (JPEG is the default image format for many cameras). Therefore, using camera processed data as inputs is inherently suboptimal. Moreover, such methods are inconsistent with the real imaging pipeline where VSR in fact serves as a pre-processing unit rather than a post-processing unit of ISP.

The main contributions of this paper are summarized as follows:

1. To avoid information loss and better fit the imaging pipeline, we propose a new

Raw VSR method, named RawVSR, that can directly exploit camera sensor data. Furthermore, we carefully build the first Raw Video Dataset (RawVD) for training, validation, and testing.

2. Assuming a Hidden Markov Model (HMM), we show that the minimal sufficient statistic of LR frames with respect to the target HR frame can be computed via an iterative pairwise fusion process; this result provides the architectural principle for the design of RawVSR, especially its Successive Deep Inference (SDI) module.

3. We design an Attention-based Residual Dense Block (ARDB) and leverage it to build the reconstruction module of RawVSR; this module is capable of simultaneously refining the fused feature and learning the color information needed to generate a spatial-specific transformation for accurate color correction.

Our extensive experimental results demonstrate that owing to the informativeness of the camera raw data, the effectiveness of the neural network architecture, and the separation of super-resolution and color correction processes, the proposed RawVSR achieves superior performance compared to the state-of-the-art and can be adapted to any specific camera-ISP.

(a) Train (b) w/ processed data (c) w/ raw data (d) ground truth

Figure 3.1: Illustrations of the proposed RawVSR based on camera processed data and raw data, respectively, for 4× VSR using the testing video "Train" in RawVD.

Fig. 3.1 illustrates the proposed RawVSR based on camera processed data and

raw data, respectively, for 4× VSR using the testing video "Train" in RawVD. It can be seen that the latter yields clearer texture and sharper edges.

3.3 Related Work

To place our work in a proper context, we give a review of the existing VSR methods and discuss some relevant issues.

VSR can be viewed as an underdetermined problem. Many traditional methods tackle this problem via regularized optimization, aided by certain image priors [21–26]. The advent of deep learning has led to a paradigm shift in VSR and more generally low-level vision research [27–30]. Indeed, following the pioneering work by Dong *et al.* [31] on SISR, the recently proposed VSR methods are predominantly deep-learning-based [10, 11, 13–16, 32–35], achieving performances out of reach of the traditional prior-based approach. Here we just describe a few representative ones. Kappeler *et al.* [11] resort to the hand-crafted optical flow [36] for motion compensation across input frames, and adopt a pre-trained CNN for SR operation. Caballero *et al.* [33] improve the quality of estimated optical flow using a spatial transformer module to facilitate the subsequent spatial-temporal network [37] for VSR. Tao *et al.* [34] design a Sub-Pixel Motion Compensation (SPMC) layer to perform motion compensation and resolution upsampling simultaneously. A novel ConvLSTM is put forward in [32], which can be integrated into the autoencoder structure to effectively exploit temporal correlations among input frames. The 3D convolution finds its first application to motion estimation and compensation in [14], where the spatial and temporal correlations are explored coherently via 3D convolving operations. Inspired by back-projection [38], Haris *et al.* [15] develop a fusion strategy that treats each

input frame as a stand-alone source carrying distinct information and progressively embeds extracted correlation from each source-target pair into the back-projection network for motion compensation. In consideration of the fact that inter-frame displacements could spread over a wide range, Tian *et al.* [39] substitute the conventional convolution with a deformable one that is capable of exploring spatial correlation via learnable sampling points tailored to each pixel, not constrained by any prescribed kernel shape. Wang *et al.* [16] propose a refined version of deformable convolution and construct a Pyramid, Cascading and Deformable (PCD) module for motion alignment; the resulting method is the winner of the NTIRE 2019 challenge on video deblurring and super-resolution [40], and achieves the best VSR performance to date.

Below we identify several issues with the existing VSR methods that will be addressed in the present work.

3.3.1 Raw Data Processing

The advantage of untouched camera raw data over processed data has been recognized in several areas of low-level vision. For instance, Chen *et al.* [27] make use of raw data to perform fast imaging in low-light and achieved favorable results in terms of texture detail. This success can be attributed to the primitive radiance information retained by raw data. Indeed, using camera processed data as a substitute of raw data fails to deliver comparable results due to the information loss caused by quantization in ISP, which obscures subtle differences in pixel values that are essential for exhibiting delicate textures. Ignatov *et al.* [41] demonstrate that color images suitably converted from smartphone's raw data are visually comparable to those produced by a professional camera. A comprehensive study of raw-data-based SISR is conducted by Xu

et al. [19], showing that the benefits of raw data are intrinsic, *i.e.*, not specific to a particular SISR method [42]. However, to the best of our knowledge, the existing VSR methods are still exclusively based on camera processed data, and there is a lack of understanding of the potential gain offered by raw data (beyond that seen in the SISR setting).

3.3.2 Motion Estimation

Due to the temporal dependencies among video frames, VSR can benefit substantially from precise motion estimation, which can be performed either explicitly or implicitly. Explicit motion estimation usually relies on optical flows [11, 13, 33, 34]. However, for video frames with large pixel displacement and object occlusion, it could be extremely challenging and arguably impossible to obtain precise optical flows. Moreover, as indicated by [43], even completely accurate optical flows might not be adequate for motion estimation since they do not fully capture all possible motion effects. Therefore, many recent VSR methods such as DUF [14], TDAN [39], and EDVR [16] choose to estimate motion implicitly, resulting in better final reconstructions compared to the flow-based ones. In a certain sense, implicit motion estimation provides a new mechanism for multi-frame fusion. Nevertheless, many aspects of the fusion process remain unspecified, and a theoretical guideline is much needed.

3.3.3 Color Correction

Different from camera processed images, raw images only record the primitive radiance information, which has not been projected to any color space. In [27], an end-to-end mapping is learned that transforms raw images to colored images directly.

However, this mapping does not have the desired flexibilities due to its camera-specific nature. [18] addresses this issue by generating a global color transformation that can cope with a variety of cameras. But it has been observed that such a transformation may cause significant local color distortions. This problem is solved by [19] through the introduction of a spatial-specific color transformation. Note that [19] makes use of a stand-alone deep convolutional network for this purpose, which increases the overall model size and computational load. In contrast, we show that spatial-specific color transformation can actually be accomplished together with HR image reconstruction by a single network. Our new design leads to more efficient implementation and facilitates joint learning for both tasks.

3.4 Raw Video Dataset

The widely-used VSR training datasets such as Vimeo-90K [43] and REDS [40] are constructed with post-ISP videos (*e.g.*, H.264 videos) and, as a consequence, are inherently unsuitable for raw-data-based VSR. We are not aware of the existence of publicly available raw video datasets for VSR. An important contribution of this work is a new Raw Video Dataset (RawVD) for training and benchmarking. In fact, it will be seen that RawVD consists of both HR/LR raw video pairs and their processed counterparts, thus is suitable for essentially all VSR methods, regardless whether they are raw-data-based or not. To build this dataset, we use a Canon 5D3 camera with the third-party upgrade[1] to film videos in Magic Lantern Video (MLV) format (which is a raw video format), and utilize the MLV App[2] to obtain corresponding raw

[1]https://magiclantern.fm/
[2]https://mlv.app/

frames in DNG format (which is a raw image format). Note that different cameras might have different Bayer patterns. For Canon 5D3, the Bayer pattern is RGGB, which means 50% sensors for green color, 25% for red color, and 25% for blue color. A degraded LR raw frame $Y_{raw} \in \mathbb{R}^{H/S \times W/S \times 1}$ is generated for each filmed HR raw frame $X_{raw} \in \mathbb{R}^{H \times W \times 1}$, where H and W stand respectively for frame height and width, and S represents the VSR scale ratio. Specifically, inspired by [19,27], we first use AHD [44] to produce a demosaiced frame $X_{lin} \in \mathbb{R}^{H \times W \times 3}$ based on X_{raw} without any post-processing such as white balance and gamma correction (where the subscript of X_{lin} reflects the fact that each of its pixel value is a linear measurement of some radiance information contained in X_{raw}), then generate Y_{raw} from X_{lin} through a sequence of degradation operations involving, among others, blurring, downsampling, and noising. More precisely, we have

$$Y_{raw} = f_{Bayer}(f_{Down}(X_{lin} * K_{Blur})) + n. \tag{3.4.1}$$

Here K_{Blur} is a uniformly distributed defocus blur kernel, which simulates the out-of-focus effect in camera; $*$ denotes the convolutional operation; n is a heteroscedastic Gaussian noise [19,45] with its variance specified by two parameters σ_1^2 and σ_2^2; f_{Down} and f_{Bayer} stand respectively for downsampling and mosaicing operations (in particular, f_{Bayer} converts a three-channel frame back to a one-channel frame that obeys the RGGB pattern). Finally, given X_{raw} and Y_{raw}, we use Rawpy, a Python version of LibRaw (with parameters adjusted to closely approximate Canon-ISP), to generate their corresponding color frames $X_{rgb} \in \mathbb{R}^{H \times W \times 3}$ and $Y_{rgb} \in \mathbb{R}^{H/S \times W/S \times 3}$. An illustration of Y_{raw}, Y_{rgb}, X_{raw}, X_{lin}, and X_{rgb} can be found in Fig. 3.2; note that the color of X_{lin} is biased towards green because the dominant measurement recorded in raw

(a) Y_{raw} (b) Y_{rgb} (c) X_{raw} (d) X_{lin} (e) X_{rgb}

Figure 3.2: Examples of videos in RawVD with the brightness and contrast of raw frames adjusted for better visualization.

data is from green sensors. In total, 110 videos (with a resolution of 1920×1080 and at least 100 frames each) are filmed. These videos capture a wide variety of scenes. We divide these 110 videos into three groups: 100 videos for training, 5 for validation, and the rest 5 for testing. The 5 test videos are named "Store", "Painting", "Train", "City", and "Walk" according to their respective content.

As mentioned above, each video has four different versions (in addition to the linear measurement version (X_{lin})): the original HR raw version (X_{raw}), the LR raw version (Y_{raw}), the HR color version (X_{rgb}), and the LR color version (Y_{rgb}). The LR raw version and the LR color version serve as the inputs of raw-data-based VSR methods and processed-data-based VSR methods, respectively. We adopt the HR color version as the ground truth for both types of methods to facilitate direct comparisons of their outputs. The linear measure version is not directly used for supervised training in the present work; nevertheless, we choose to include it in our RawVD as it visually reveals the intimate connections among the other four versions

and potentially provides valuable insights that can be fruitfully exploited in future research.

Remark: We have also considered collecting real-world raw LR-HR pairs through optical zooming [46]. However, this approach has several limitations. First, each LR-HR pair should have exactly the same motion (otherwise the pixel-level alignment is difficult to accomplish), which severely limits the varieties of suitable videos that can be collected in practice. Second, since raw data only record primitive radiance information, some back-and-forth conversions between raw data and processed data (*e.g.*, sRGB) are typically needed for raw LR-HR pair alignment, causing information loss. As such, this approach is tailored to SISR (without inter-frame motion) with sRGB data. In contrast, our approach does not suffer from these problems and is better suited to raw VSR. It is also worth noting that even though AHD is not the state-of-the-art for demosaicing, it performs very stable and is widely adopted in practice [19]. Indeed, upon close scrutiny, no ground-truth video in the constructed RawVD suffers evident demosaicing artifacts.

3.5 Method

Now we are in a position to present the proposed raw-data-based VSR method, *i.e.*, RawVSR. A general overview is provided in Section 3.5.1. Sections 3.5.2 and 3.5.3 are devoted to describing two key components of RawVSR: the SDI module and the reconstruction module. We address some potential practical concerns regarding raw-data-based VSR in Section 3.5.4.

3.5.1 Overview

To effectively exploit LR raw videos, one might be inclined to build a deep neural network to learn an end-to-end mapping that can generate HR color videos directly. However, in reality, raw videos only record the radiance information read from camera sensors and the associated color videos are in fact ISP-dependent. That is to say, different ISPs may produce different color videos based on the same raw video. As such, an end-to-end mapping can only be tailored to a specific type of camera ISP and lacks the flexibilities needed for accommodating diverse VSR requirements. To tackle this problem, we use one certain color reference generated by an ISP-adaptive operation to guide the transformation from the raw space to the color space.

The proposed RawVSR consists of several components: the demosaicing module, the feature extraction module, the SDI module, the reconstruction module, the upsampling module, and the color-transformation generating module. It takes 7 consecutive LR raw frames $Y_{raw}^{t-3}, \cdots, Y_{raw}^{t+3}$ and outputs a SR color frame $X_{sr}^{t} \in \mathbb{R}^{H \times W \times 3}$. The goal is to make X_{sr}^{t} as close to the HR color frame X_{rgb}^{t} as possible. We use Y_{rgb}^{t} as the color reference frame to ensure that the provided color information is consistent with that of X_{rgb}^{t}. Note that the color reference frame is derived from the middle frame in the raw video sequence through an ISP-adaptive operation, and thus should not be viewed as an additional input. Indeed, except some ISP-specific parameters needed for color transformation and correction, this middle frame already contains all necessary information to produce the color reference frame.

As shown in Fig. 3.3, RawVSR has two branches: the texture restoration branch (upper branch) produces a provisional SR frame $\hat{X}_{sr}^{t} \in \mathbb{R}^{H \times W \times 3}$ while the color correction branch (lower branch) generates a spatial-specific color transformation T.

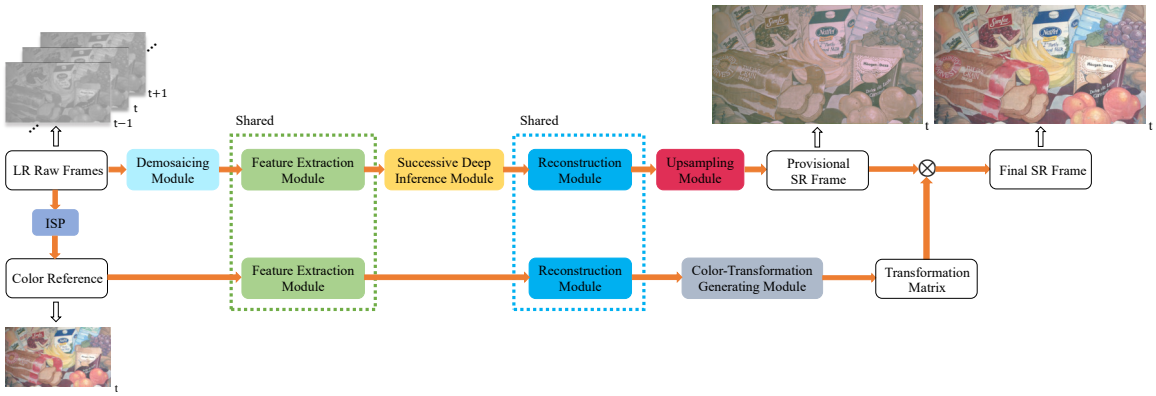


Figure 3.3: The architecture of the proposed RawVSR.

The final SR frame X_{sr}^t is obtained from $\hat{X}_{sr}^t$ through pixel-wise color correction specified by T. The orange arrow shows the image/feature flow direction, and the black arrow points to the illustration of input/color reference, provisional SR, and final SR frame respectively using the "Painting" video in RawVD.

In the texture restoration branch, the input LR raw frames are first processed by the demosaicing module, where each LR raw frame is converted to a 64-channel frame (with the spatial size unchanged) via packing, convolution, and pixel shuffling. We utilize the U-net [47] for frame-by-frame multi-scale feature extraction. The outputs of the feature extraction module induced by the given seven LR raw frames are jointly fed into the SDI module to perform feature alignment and fusion. The fused feature is then refined by the reconstruction module. The upsampling module consists of three convolutional layers and one pixel-shuffle operator. It enlarges the spatial size of the refined feature to generate a provisional SR frame. It can be seen that although the provisional SR frame suffers severe color distortion, the texture details are faithfully restored.

In the color correction branch, both the feature extraction module and the reconstruction module are shared from the texture restoration branch. To ensure dimension compatibility, the color reference frame is pre-processed by a single convolutional layer before fed into the feature extraction module. Different from its main role in the texture restoration branch (which is feature refinement), the reconstruction module is used in the color correction branch to learn the ISP information needed for generating a spatial-specific color transformation. The structure of the color-transformation generating module is the same as that of the upsampling module except for the last convolutional layer, which outputs a 9-channel feature map rather than a 3-channel one. The output feature map is reshaped from $H \times W \times 9$ to $H \times W \times 3 \times 3$, and the reshaped map is denoted by T. Note that T can be viewed as a collection of 3×3 matrices $\{T(i,j)\}$ with each (i,j) pair specifying a pixel position. These matrices are leveraged to perform a spatial-specific color transformation on the provisional SR frame $\hat{X}^t_{sr}$ to produce the final SR frame X^t_{sr} as follows:

$$X^t_{sr}(i,j) = T(i,j) \otimes \hat{X}^t_{sr}(i,j), \tag{3.5.1}$$

where $X^t_{sr}(i,j) \in \mathbb{R}^{3\times 1}$ ($\hat{X}^t_{sr}(i,j) \in \mathbb{R}^{3\times 1}$) is the RGB vector of the (i,j) pixel of X^t_{sr} ($\hat{X}^t_{sr}$), and $\otimes$ denotes matrix multiplication.

It is worth noting that due to the sharing of the feature extraction module and the reconstruction module as well as the lightweight design of the color-transformation generating module, the number of independent parameters that come from the color correction branch is negligible (accounting for about 5% of the model size of RawVSR). On the other hand, from the perspective of building an interpretable device-independent network, it is preferable to have a dual-branch architecture that disentangles the

super-resolution process from the color correction process, and one may argue that weight sharing does not suit this purpose. However, since the generated transformation matrix needs to be properly aligned with the provisional SR frame to perform pixel-wise color correction, it is essential to have a mechanism to facilitate the coordination of two processes. In this regard, weight sharing contributes to this coordination by helping maintain the latent consistency of intermediate features in these two branches. Moreover, it will be seen from the experimental results in Section 3.6 that weight sharing actually does not jeopardize the interpretability and device independence of the resulting design.

3.5.2 Successive Deep Inference Module

Although many existing VSR methods exploit temporal redundancy, the underlying alignment and fusion rules are often heuristic in nature. In contrast, we shall propose a systematic approach based on a canonical decomposition result for HMM inference.

First note that the VSR problem can be mathematically formulated as

$$\tilde{X}^t \triangleq \min_f \mathbb{E}[d(X^t - f(Y^{t-3}, \cdots, Y^{t+3}))], \tag{3.5.2}$$

where $(Y^{t-3}, \cdots, Y^{t+3}) \triangleq (Y^{t-3}_{raw}, \cdots, Y^{t+3}_{raw})$, $X^t \triangleq X^t_{rgb}$, $\tilde{X}^t \triangleq X^t_{sr}$, and $d(\cdot, \cdot)$ is the loss function. However, $\tilde{X}^t$ in general depends on the choice of $d(\cdot, \cdot)$. To gain insight into the fundamental architectural principle that is applicable under any loss function, it is instructive to consider the following alternative formulation with hard

reconstruction $\tilde{X}^t$ replaced by soft reconstruction

$$\tilde{\Theta}^t \triangleq p(X^t|Y^{t-3}, \cdots, Y^{t+3}), \tag{3.5.3}$$

where $p(X^t|Y^{t-3}, \cdots, Y^{t+3})$ denotes the conditional distribution of X^t given $(Y^{t-3}, \cdots, Y^{t+3})$. The SDI module is designed to solve (3.5.3), *i.e.*, to compute $\tilde{\Theta}^t$ based on $(Y^{t-3}, \cdots, Y^{t+3})$. Note that $\tilde{\Theta}^t$ is a minimal sufficient statistic of $(Y^{t-3}, \cdots, Y^{t+3})$ with respect to X^t in the sense that $(Y^{t-3}, \cdots, Y^{t+3})$ and X^t are conditionally independent given $\tilde{\Theta}^t$ and $\tilde{\Theta}^t$ is a function of any statistic of $(Y^{t-3}, \cdots, Y^{t+3})$ with this conditional independence property. As a consequence, it is possible to deduce $\tilde{X}^t$ from $\tilde{\Theta}^t$ once the loss function is specified; for example, $\tilde{X}^t$ is simply the mean of $\tilde{\Theta}^t$ (*i.e.*, $\tilde{X}^t = \mathbb{E}[X^t|Y^{t-3}, \cdots, Y^{t+3}]$) if the loss function is chosen to be Mean Squared Error (MSE). This deduction is basically accomplished by the reconstruction module to be described in Section 3.5.3. It is also worth noting that (3.5.3) is invariant under isomorphic representations of $(Y^{t-3}, \cdots, Y^{t+3})$. This is the reason why the features extracted from the LR raw frames can be used in lieu of the LR frames themselves as the input of the SDI module. Similarly, the output of the SDI module can be any equivalent representation of $\tilde{\Theta}^t$.

However, solving (3.5.3) through a data-driven approach is very difficult. For one thing, it requires learning joint patterns existent in the (features of) LR raw frames. Presumably the number of such patterns is already very large even for 2 frames and the number is likely to increase by several orders of magnitude with every additional frame. Therefore, it is unrealistic to expect that one can learn enough joint patterns needed to solve (3.5.3) reliably based on limited training data. Fortunately, it turns out that the learning complexity can be greatly reduced under a

proper fusion strategy. For ease of exposition, we assume that the HR raw frames $X^{t-3} \leftrightarrow \cdots \leftrightarrow X^{t+3}$ form a Markov chain in this order. Since each LR raw frame (or its feature) is generated from the corresponding HR raw frame through an independent degradation process, $(X^{t-3}, \cdots, X^{t+3})$ and $(Y^{t-3}, \cdots, Y^{t+3})$ jointly constitute an HMM. This HMM enables us to decompose (3.5.3) into a sequence of simpler problems. Note that $\Theta^{t-2} \triangleq p(X^{t-2}|Y^{t-3}, Y^{t-2})$ is a minimal sufficient statistic of (Y^{t-3}, Y^{t-2}) with respect to X^{t-2}. Therefore,

$$\tilde{\Theta}^t = p(X^t|\Theta^{t-2}, Y^{t-1}, \cdots, Y^{t+3}). \tag{3.5.4}$$

Moreover, removing X^{t-3} and replacing (Y^{t-3}, Y^{t-2}) with Θ^{t-2} resulting in a new HMM. One can iterate the above argument to show that

$$\tilde{\Theta}^t = p(X^t|\Theta^t, \Theta^{t+1}), \tag{3.5.5}$$

where $\Theta^{t+\ell} \triangleq p(X^{t+\ell}|\Theta^{t+\ell-1}, Y^{t+\ell})$ for $\ell = -2, -1, 0$, and $\Theta^{t+\ell} \triangleq p(X^{t+\ell}|Y^{t+\ell}, \Theta^{t+\ell+1})$ for $\ell = 1, 2$, with $\Theta^{t-3} \triangleq Y^{t-3}$ and $\Theta^{t+3} \triangleq Y^{t+3}$. Fig. 3.4 provides an intuitive illustration of this iterative argument using probabilistic graphical models. It is worth mentioning that this is similar to the reasoning underlying the well-known forward-backward algorithm for HMM inference. The main difference is that here we are mostly concerned with the fundamental architectural principle rather than the computational complexity aspect.

Our fusion strategy has several advantages over the existing ones [16, 48]. First, an important implication of (3.5.5) is that (3.5.3) can be solved by repeatedly performing pairwise fusion, which is much less demanding in terms of the amount of

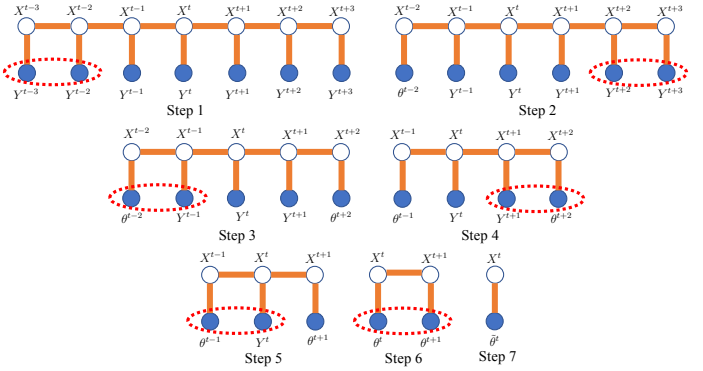


Figure 3.4: A graphical illustration of HMM inference via iterative pairwise fusion.

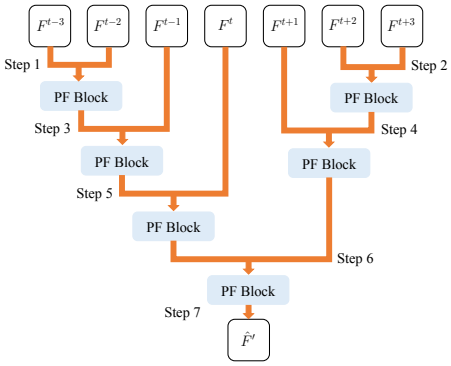


Figure 3.5: The architecture of the SDI module.

training data needed for reliable learning. Second, the two inputs are of comparable importance with respect to the target object and consequently are more likely to be thoroughly exploited. In contrast, for other fusion strategies where the inputs under consideration have a clear difference in importance (for example, Y^t is clearly more relevant than Y^{t-3} for the purpose of estimating X^t), the subtle details contained in the less important ones may easily get overlooked.

As shown in Fig. 3.5, the SDI module is designed according to the architectural principle suggested by (3.5.5) (note that the slight asymmetry in (3.5.5) also manifests in the structure of this module). It takes the features $F^{t-3}, \cdots, F^{t+3}$ extracted respectively from the LR raw frames $Y_{raw}^{t-3}, \cdots, Y_{raw}^{t+3}$ and outputs a fused feature $\hat{F}^t$. Each pairwise fusion (PF) step is realized by a PF block (see Fig. 3.6 for its structure). Specifically, the PF block first concatenates its two input features $\overline{F}$ and $\underline{F}$; it then uses two structurally-identical offset generators (without weight-sharing) to learn positional offsets $\overline{\Delta P} \triangleq (\overline{\Delta P}_1, \cdots, \overline{\Delta P}_K)$ and $\underline{\Delta P} \triangleq (\underline{\Delta P}_1, \cdots, \underline{\Delta P}_K)$, which are leveraged to align $\overline{F}$ and $\underline{F}$ using the modulated deformable convolution [49];

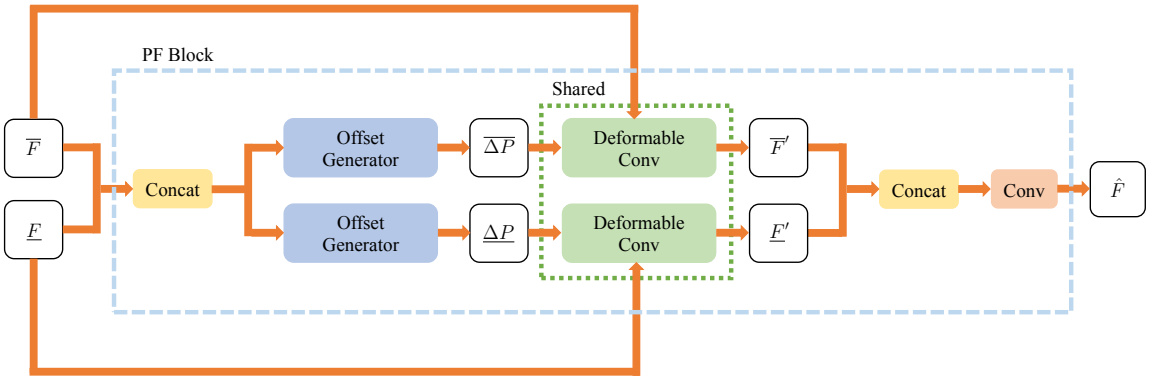


Figure 3.6: The structure of the PF block.

finally, the aligned features $\overline{F}'$ and $\underline{F}'$ are concatenated and passed through two convolutional layers to produce the fusion result $\hat{F}$. Note that the alignment procedure can be interpreted as implicit motion estimation and compensation in the feature space, and the relationship between pre- and post-alignment features at position P can be expressed as

$$\overline{F}'(P) = \sum_{i=1}^{K} \overline{w}_i \cdot \overline{F}(P + \overline{\Delta P}_i) \cdot \Delta \overline{S}_i, \tag{3.5.6}$$

$$\underline{F}'(P) = \sum_{i=1}^{K} \underline{w}_i \cdot \underline{F}(P + \underline{\Delta P}_i) \cdot \Delta \underline{S}_i, \tag{3.5.7}$$

where $\overline{w}_i$ ($\underline{w}_i$) and $\overline{\Delta S}_i$ ($\underline{\Delta S}_i$) denote the weight and the modulation scalar of the deformable convolution for $\overline{F}(P)$ ($\underline{F}(P)$) with respect to offset $\overline{\Delta P}_i$ ($\underline{\Delta P}_i$), $i = 1, \cdots, K$ (we set $K = 9$ in this work). It is worth noting that different from the flow-based method, the proposed SDI module does not require any pre-training and can be embedded into another network for end-to-end training. We would also like to point out that the first-order Markov assumption for HR raw frames is not required for the derivation of the architecture in Fig. 3.5. Indeed, a higher-order Markov chain (which

is more suitable to model scenarios with occlusion and disocclusion) can be reduced to a first-order one by grouping several consecutive states in a sliding-window fashion, after which our previous argument can be invoked with no essential change. In fact, the only constraining factor for the SDI module is the dimension of the output of each PF block in the sense that higher dimensional outputs are likely needed to accommodate more sophisticated Markov structures.

3.5.3 Reconstruction Module

The reconstruction module is elaborately designed with two tasks in mind: 1) refining the fused feature associated with the input LR raw frames and 2) learning the ISP-specific information needed for color transformation and correction from the color reference frame. It is built using ARDBs as shown in Fig. 3.7. Each ARDB consists of one convolutional layer and two residual dense blocks. We follow the default RDB settings in [42] except that the growth rate is set to 64. The generated intermediate feature maps are fused under the guidance of the channel-wise attention. To this end, we adopt the SENet in [50] to produce attention weights to enhance/suppress the contributions of the associated feature maps in accordance with their relevance. In addition to the local fusion in each ARDB, we also employ a global fusion with channel-wise attention to further strengthen the learning ability of the reconstruction module. To strike a balance between system performance and computational complexity, totally four ARDBs are used in our design.

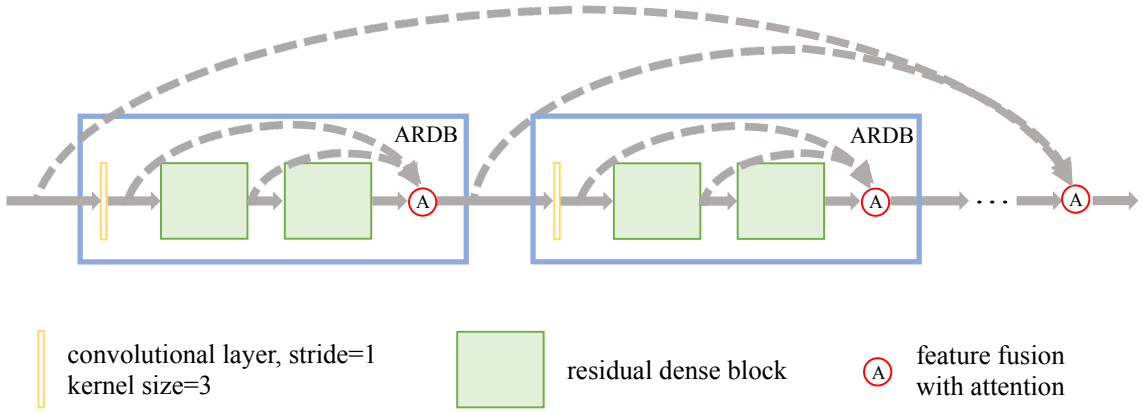


Figure 3.7: The architecture of the reconstruction module.

3.5.4 On the Practicability of Raw VSR

Although the present work mainly focuses on the technical feasibility of raw VSR, here we would like to briefly comment on its practical prospect. In particular, we shall address the following two questions: 1) the availability of camera raw data, and 2) the application scenarios of raw VSR. As raw data record untouched radiance information that can be fruitfully exploited for many different purposes, they receive considerable attention from professional photographers. To facilitate their use, the major camera manufacturers (*e.g.*, Canon and Nikon) have provided the official interface to access raw data. This removes a major hurdle for the wide adoption of raw VSR. Moreover, in most scenarios where conventional VSR is currently being considered, raw VSR can potentially be a more favorable choice to its superior performance and consistency with the real imaging pipeline. It is worth mentioning that some recent works [51, 52] attempt to generate raw-like data based on processed data. However, such raw-like data inevitably suffer from the information loss issue due to the non-invertibility of ISP operations, and great caution should be exercised when using them in lieu of authentic raw data such as those in our RawVD. More discussions can be found in

Section 3.6.4.

3.6 Experimental Results

Extensive experiments are conducted to demonstrate the competitive performance of the proposed RawVSR. They also provide solid justifications for our design (especially, the SDI module and the reconstruction module) and convincing evidence regarding the value of raw data.

3.6.1 Implementation

The proposed RawVSR is end-to-end trainable without any additional supervision or pre-training for sub-modules. We adopt RawVD for training and testing as it is the only available video dataset for raw-data-based VSR methods. The scale ratio S is set to 4; the size of the defocus blur is randomly chosen from $\{3, 5, 7, 9, 11\}$; the two parameters σ_1^2 and σ_2^2 of heteroscedastic Gaussian noise are randomly sampled from $[0, 0.1]$ and $[0, 0.02]$, respectively. We use patches of size 64×64 cropped from LR raw videos as the training inputs, and augment the training data through inverting the frame order, horizontal flips and rotations [53]. Following [54], batch normalization is not used in RawVSR. Without bells and whistles, we adopt a weighted version of Mean Square Error (MSE) and Structural SIMilarity index (SSIM) [55] as our loss function to measure the difference between the final reconstructed SR frame and the ground truth. Furthermore, we provide an additional supervision to the provisional SR frame to enforce the learning of texture detail using an extra SSIM loss. The total

loss function L can be formulated as:

$$L = L_F^{MSE} + \lambda_F L_F^{SSIM} + \lambda_P L_P^{SSIM}, \tag{3.6.1}$$

where L_F^{MSE}, L_F^{SSIM}, and L_P^{SSIM} are respectively the MSE loss, the SSIM loss from the final SR frame, and the SSIM loss from the provisional SR frame. The trade-off parameters λ_F and λ_P are both set to 0.005 to balance the value ranges of these sub-level losses. Since the SSIM loss is designed to measure the structural similarity between two images and is relatively insensitive to color distortions, two images photographed in the same scene under different light conditions might have a high SSIM value but a low PSNR value. This fact inspires us to employ SSIM for assessing the quality of the provisional SR frame. The textural consistency between the provisional SR frame and the ground truth then enables the spatial-specific transformation to focus on color correction. In this way, the texture restoration branch and the color correction branch are effectively functionally disentangled. It will be seen in Section 3.6.3 that the additional supervision on the provisional SR frame improves the interpretability of our network and makes it more compatible with the real imaging pipeline, where the super-resolution process is carried out before color correction. We would point out that the linear measurement X_{lin} provided in RawVD is not employed as the supervision reference of the provisional SR frame. This is because the proposed RawVSR is intended to be device-independent, whereas the use of X_{lin} might jeopardize its generalization ability due to the fact that X_{lin} is produced by a specific ISP (which is Rawpy in the current setting).

To accelerate network training, the Adam optimizer [56] is used with a batch size of 30 and the default parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the initial

learning rate to 2e-4, which is reduced by half every 20 epochs until a total of 100 epochs is reached. The training process is carried out on a PC with two NVIDIA GTX 1080Ti. All PSNR/SSIM values are evaluated using the test data in RawVD.

3.6.2 Quantitative and Qualitative Comparisons

We compare the proposed RawVSR with several existing processed-data-based VSR methods, including VSRNet [11], VESPCN [33], SPMC [34], DUF [14], RBPN [15], TDAN [39], and EDVR [16]. In particular, EDVR is the champion of the NTIRE 2019 challenge on video deblurring and super-resolution [40] and can be considered as the current state-of-the-art. For fair comparisons, we laboriously retrain all the aforementioned methods using the same strategy as described in Section 3.6.1 until convergence. These methods are trained with (Y_{rgb}, X_{rgb}) pairs instead of (Y_{raw}, X_{rgb}) pairs (see Fig. 3.2) since they are incapable of handling raw data. In contrast, the proposed RawVSR can leverage both raw data and processed data for training. To distinguish it from the original version of RawVSR for which the training is based on (Y_{raw}, X_{rgb}) pairs, the version trained with (Y_{rgb}, X_{rgb}) pairs is denoted as RawVSR†, where the color correction branch is removed since the color reference is not needed in this case.

Quantitative comparisons on the RawVD test data for 2× and 4× VSR are shown in Table 3.1 (with the only exception of RBPN for which we are unable to implement the same training strategy on our PC for 2× VSR due to its large model size). For qualitative comparisons, we only demonstrate in Fig. 3.8 4× VSR since it is more visually distinguishable than 2× VSR. It can be seen that RawVSR† outperforms the other processed-data-based methods in terms of PSNR and SSIM metrics, and

Table 3.1: Quantitative comparisons on the RawVD test data for 2× and 4× VSR. **Red** and Blue indicate the best and the second best performance, respectively.

RawVD 2×										
Name	Bicubic	VSRNet [11]	VESPCN [33]	SPMC [34]	DUF [14]	RBPN [15]	TDAN [39]	EDVR [16]	RawVSR†	RawVSR
Store	27.84/0.8299	28.99/0.8457	29.02/0.8494	29.41/0.8503	30.75/0.8710	-	31.27/0.8863	31.98/0.9060	33.04/0.9129	**34.30/0.9263**
Painting	28.80/0.8086	29.38/0.8225	29.45/0.8218	28.82/0.7998	31.22/0.8512	-	31.17/0.8556	32.55/0.8816	32.80/0.8773	**33.79/0.8943**
Train	28.11/0.7738	29.18/0.8081	29.32/0.8121	30.01/0.8298	31.25/0.8464	-	30.94/0.8396	31.42/0.8617	31.66/0.8522	**32.79/0.8729**
City	27.76/0.7527	28.99/0.8012	29.22/0.8047	29.44/0.8210	30.47/0.8190	-	30.26/0.8227	31.11/0.8517	31.41/0.8588	**32.11/0.8637**
Walk	26.43/0.7553	28.92/0.7871	28.78/0.7820	30.25/0.8439	30.13/0.8117	-	29.26/0.8207	30.58/0.8415	29.63/0.8282	**31.20/0.8516**
Average	27.79/0.7841	29.09/0.8129	29.16/0.8140	29.59/0.8290	30.76/0.8399	-	30.58/0.8450	31.53/0.8685	31.71/0.8659	**32.84/0.8818**

RawVD 4×										
Name	Bicubic	VSRNet [11]	VESPCN [33]	SPMC [34]	DUF [14]	RBPN [15]	TDAN [39]	EDVR [16]	RawVSR†	RawVSR
Store	22.59/0.6507	25.24/0.7135	25.46/0.7250	26.34/0.7351	27.84/0.8105	27.53/0.7647	26.67/0.7802	28.23/0.8202	28.29/0.8191	**29.04/0.8400**
Painting	24.78/0.6712	26.17/0.7119	26.02/0.7120	26.13/0.7199	27.71/0.7833	26.40/0.7209	26.75/0.7539	27.74/0.7908	28.51/0.7940	**29.02/0.8104**
Train	24.12/0.6166	25.80/0.6671	25.96/0.6724	26.70/0.6984	27.53/0.7377	27.64/0.7358	26.94/0.7205	27.81/0.7470	28.02/0.7443	**28.59/0.7625**
City	24.94/0.6330	26.59/0.6907	26.63/0.6927	27.03/0.6684	27.93/0.7508	28.39/0.7332	27.55/0.7336	28.24/0.7558	28.43/0.7640	**29.08/0.7843**
Walk	22.46/0.6229	25.52/0.6766	25.62/0.6726	25.30/0.6797	27.30/0.7512	26.46/0.7337	26.36/0.7229	27.02/0.7533	27.90/0.7607	**28.06/0.7724**
Average	23.78/0.6389	25.87/0.6919	25.94/0.6950	26.30/0.7002	27.66/0.7667	27.29/0.7377	26.85/0.7422	27.81/0.7734	28.23/0.7765	**28.76/0.7939**

Table 3.2: Quantitative comparisons on the iPhone data for 4× VSR. **Red** and Blue indicate the best and the second best performance, respectively.

iPhone 4×				
Name	DUF [14]	EDVR [16]	RawVSR†	RawVSR
Tree	27.15/0.6630	27.26/0.6504	28.66/0.7073	**29.54/0.7381**
Door	26.11/0.6334	26.05/0.6231	26.28/0.6694	**28.62/0.6957**
Flower	25.90/0.6225	25.74/0.6062	27.07/0.6771	**28.17/0.7307**
Average	26.38/0.6396	26.35/0.6266	27.34/0.6846	**28.78/0.7215**

visually restores sharper edges and finer details in most cases (see, *e.g.*, the word *Country* in video "Painting", and the balcony part of the building in video "City"), which provides solid justifications for our overall network design.

As compared to RawVSR†, the results of RawVSR are even more appealing, both quantitatively and qualitatively. This improvement provides convincing evidence regarding the benefits of raw data since it is clearly attributed to their richer information content. In addition, to validate that the proposed method is device-independent, three raw 4K videos (named "Tree", "Door", and "Flower") filmed by Adobe Lightroom software[3] on an iPhone 8p are used to complement our RawVD test data for 4× VSR. The quantitative and qualitative comparisons are illustrated in Table 3.2 and Fig. 3.9, respectively. The superior VSR results indicate that the proposed RawVSR, trained only on RawVD, continues to perform well on the data collected by other devices without fine-tuning, and thus possesses good generalization abilities. More evidence can be found in Section 3.6.5.

[3]https://www.adobe.com/ca/products/photoshop-lightroom.html

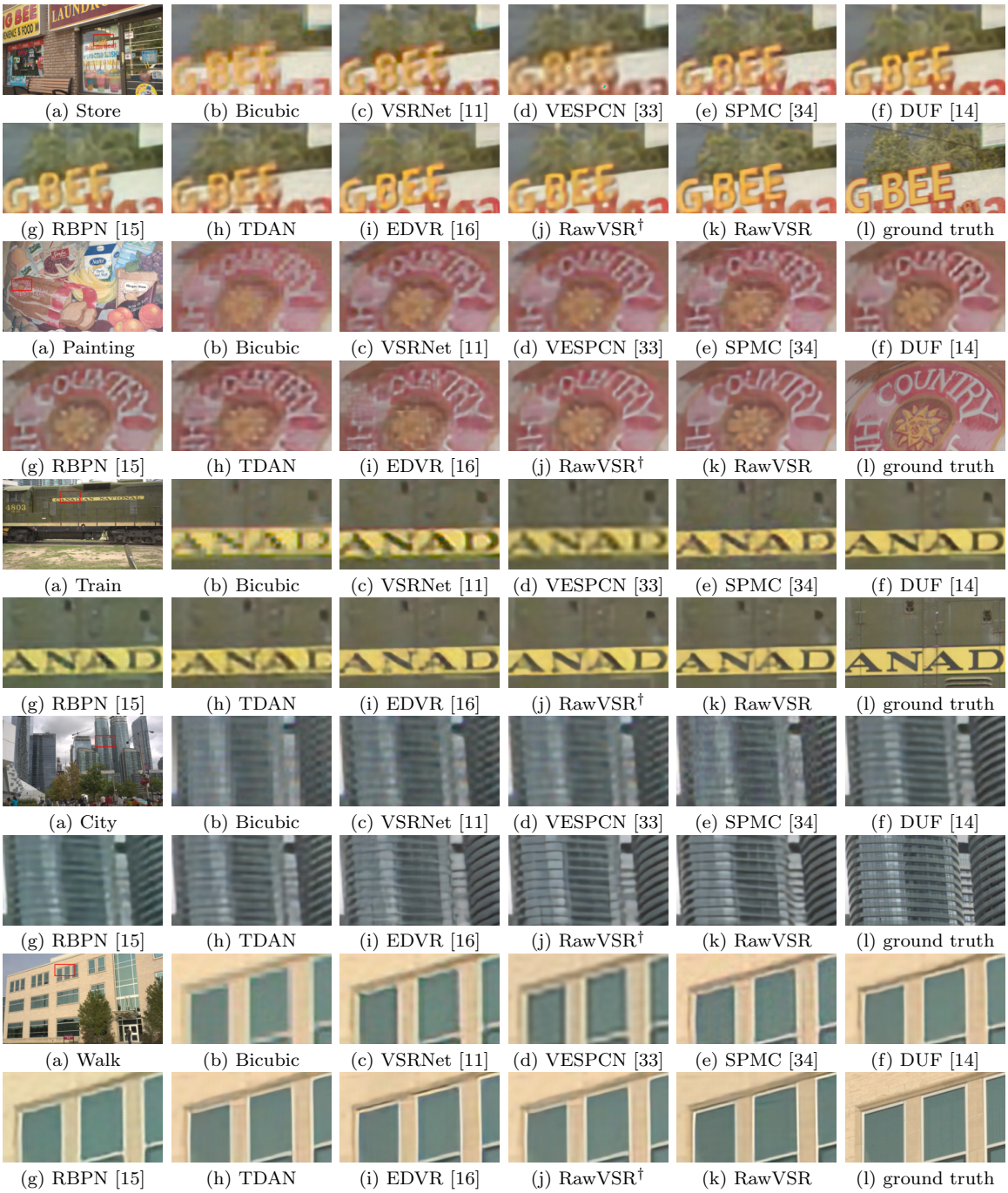


Figure 3.8: Qualitative comparisons on the RawVD test data for 4× VSR. Zoom in for better visualization.

(a) ground truth (b) DUF [14] (c) EDVR [16] (d) RawVSR† (e) RawVSR

Figure 3.9: Qualitative comparisons on the iPhone data for 4× VSR. The names of video sequences from the first row to the last row are "Tree", "Door", and "Flower", respectively. Zoom in for better visualization.

Figure 3.10: Visualization of the color correction process (zoom-in for details).

3.6.3 Visualization of the Color Correction Process

To show that the texture restoration branch and the color correction branch indeed fulfill their designated roles, Fig. 3.10 visualizes how a provisional SR frame $\hat{X}_{sr}^t$ undergoes the color correction process to produce the corresponding final SR frame X_{sr}^t. In particular, we highlight three representative pixels (*i.e.*, $\hat{X}_{sr}^t(800, 1650)$, $\hat{X}_{sr}^t(200, 1350)$, and $\hat{X}_{sr}^t(450, 350)$) extracted from a provisional SR frame in video "Painting" and their color-corrected counterparts (*i.e.*, $X_{sr}^t(800, 1650)$, $X_{sr}^t(200, 1350)$, and $X_{sr}^t(450, 350)$) together with the associated spatial-specific color transformation matrices. Clearly, although the provisional SR frame $\hat{X}_{sr}^t$ suffers severe color distortion, it exhibits correct details and is texturally consistent with the final SR frame X_{sr}^t. This validates the effectiveness of additional SSIM-based supervision in guiding the texture restoration branch to accomplish its desired purpose. The color-corrected pixels are produced by applying the spatial-specific transformation matrices on their respective provisional pixels according to Equ. (3.5.1). It is worth noting that even though the pixels extracted from $\hat{X}_{sr}^t$ are color-wise quite similar, their associated color transformation matrices are distinctively different, resulting in visually more distinguishable color-corrected pixels. This provides strong supporting evidence for the usefulness of the color correction branch.

3.6.4 Authentic Raw Data v.s. ISP-Inverted Data

Several recent works [51, 52] have attempted to invert the camera ISP to produce raw-like data (which will be referred to as ISP-inverted data) from processed data and use them in lieu of authentic raw data for training. However, the resemblance between ISP-inverted data and raw data should not be overestimated. Indeed, such

(a) w/ ISP-inverted data (b) w/ authentic raw data (c) ground truth

Figure 3.11: The proposed RawVSR with ISP-inverted data and authentic raw data respectively on the RawVD test dataset for 4× VSR.

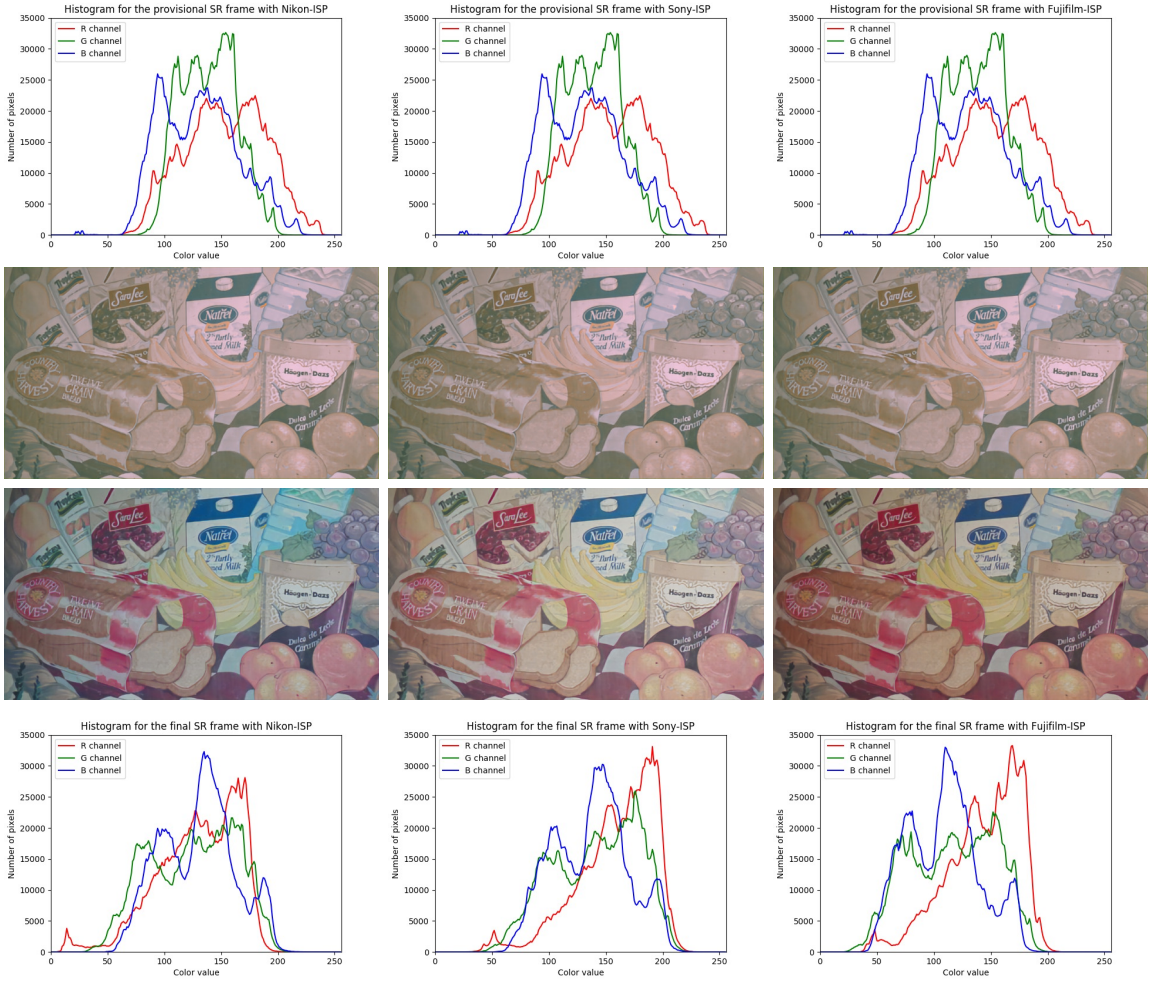
inversion can never be perfect due to the presence of non-invertible ISP components (*e.g.*, quantization); actually it may even cause additional information loss. To gain a concrete understanding, we follow the same procedure carried out in [51] to produce ISP-inverted data from processed data Y_{rgb} in RawVD, and input them to our network for testing. The results are then compared with those based on authentic raw data as shown in Fig. 3.11. It can be seen that ISP-inverted data incur two evident degradations: 1) significant color distortion (which is likely caused by the distribution mismatch between ISP-inverted data and raw data), 2) blurred texture detail (which is a consequence of information loss). Therefore, authentic raw data as those collected in RawVD remain indispensable for raw VSR.

3.6.5 Validation of Device Independence

Due to the separation of the super-resolution process and the color correction process, the proposed RawVSR can adapt easily to different camera-ISPs. We have demonstrated this desirable characteristic in Table 3.1 and Fig. 3.9 by blindly testing RawVSR, trained with data from simulated Canon-ISP (*i.e.*, Rawpy), on a new iPhone dataset. Here we shall provide more evidence. Due to the lack of Digital Single-Lens Reflex (DSLR) cameras (other than Canon 5D3) at our disposal, we choose to generate LR color references by simulating ISPs of three major camera manufacturers (Nikon, Sony, and Fujifilm) based on the analysis of their color styles and preferences. According to our observation, the video frames produced by Nikon-ISP exhibit the lowest color temperature as compared to the other two, which leads to an overall cold tone; in contrast, those by Fujifilm-ISP show the highest color temperature, yielding a warm tone; Sony-ISP is somewhere in between in terms of color temperature, yet produces the highest luminance. Based on these insights, we can closely approximate the effects of these ISPs by suitably adjusting the exposure degree, contrast, and color temperature of LR color references in RawVD.

Specifically, we increase the color temperature from 6000K to 8000K to approximate the warm tone of Fujifilm-ISP, and decrease it to 5200K to approximate the cold tone of Nikon-ISP. Since the Sony-ISP has a unique preference, simply adjusting one color setting does not yield good approximation. Instead, we increase the exposure degree from the default value 0 to 0.7, decrease the contrast from 0 to -8, and change the color temperature to 6500K in order to achieve the effect similar to that of Sony-ISP.

Fig. 3.12 shows the final SR frames with simulated Nikon, Sony, and Fujifilm-ISPs,



(a) w/ Nikon-ISP (b) w/ Sony-ISP (c) w/ Fujifilm-ISP

Figure 3.12: Visualization of the provisional frames (in the 2nd row), the final SR frames (in the 3rd row), and their associated color histograms (in the 1st and 4th rows) with simulated Nikon, Sony and Fujifilm-ISPs, to validate the device-independence characteristic of RawVSR.

respectively, and the corresponding provisional SR frames as well as their associated color histograms for a representative frame in video "Painting". Here, the input to the texture restoration branch is fixed, and different color references (based on simulated ISPs) are fed into the color correction branch without additional training. Note that the choice of simulated ISP has no impact on the provisional frame (see the 2nd row in Fig. 3.12) and its associated color histogram (see the 1st row in Fig. 3.12). As such, the texture restoration branch and the color correction branch are indeed functionally disentangled, which is beneficial to the interpretability of overall network design. On the other hand, the color of the final SR frame changes in accordance with the provided color reference (see the 3rd row in Fig. 3.12), which is also reflected in the color histograms (see the 4th row in Fig. 3.12). This shows convincingly that the proposed RawVSR can automatically adapt to the selected ISP, thus is device-independent.

3.6.6 Validation of Temporal Consistency

A lack of temporal consistency may lead to flickering artifacts (*e.g.*, in the form of jagged edges) that can degrade the quality of reconstructed SR frames [35]. To compare the proposed RawVSR with the existing methods from this perspective, we extract some co-located columns from the reconstructed "City" sequence, rotate and then stack them vertically to produce a temporal profile [13]; we also generate another temporal profile based on co-located rows from the reconstructed "Walk" sequence. It can be seen from Fig. 3.13 that RawVSR yields the most consistent temporal profiles in terms of texture clarity, edge sharpness, and detail accuracy. For example, in the temporal profile of the "Walk" sequence, the proposed RawVSR successfully recovers

most line patterns whereas the other methods fail to distinguish dashed lines from solid ones.

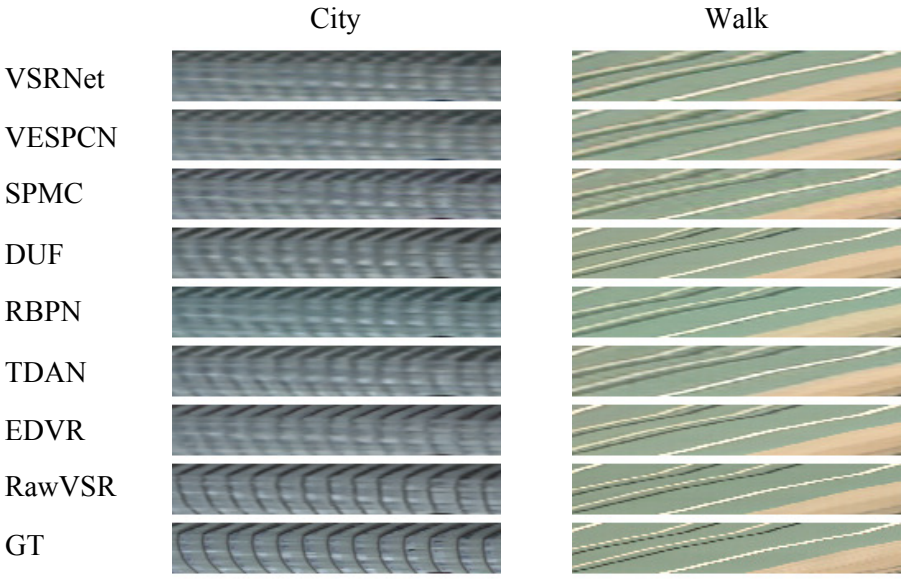


Figure 3.13: Visualization of temporal profiles produced by the proposed RawVSR and some existing methods.

3.6.7 Ablation Studies

To gain a better understanding of key constituents of the proposed network as well as the influence of the input sequence length, we conduct elaborately designed ablation studies by examining various alternative implementations. All such implementations are trained from scratch with the same training strategy described in Section 3.6.1. The proposed RawVSR (trained on raw data) will be referred to as the baseline.

3.6.7.1 Justification of the SDI Module

Most existing VSR methods adopt early fusion, slow fusion, or 3D convolutional fusion to merge input frames [33]. For early fusion, all input frames are concatenated at the very beginning and fed jointly into the network. As an example, VSRNet [11] follows this fusion strategy. In contrast, the slow fusion strategy merges input frames progressively according to a prescribed rule. The scheme of the proposed SDI module falls into this general category; one of our main contributions is, in a certain sense, an optimal fusion rule deduced via HMM analysis. 3D convolutional fusion performs convolution on input frames in both temporal and spatial domains, and tends to be computationally expensive. To justify the design of the SDI module, three variants are considered, each associated with a different fusion scheme. The first variant, named w/ EF, simply follows the early fusion strategy. The second variant, named w/ improved EF, is similar to the first one except that all other frames are first aligned to the middle frame. The last one, named w/ SF, adopts a slow fusion scheme, where all other frames are first aligned and fused with the middle frame in a pairwise manner before joint fusion is performed. For fair comparisons, these alternative fusion modules are designed to have roughly the same size as that of the SDI module (no structural change is made to the rest of the network). Note that 3D convolutional fusion can not be implemented under this size constraint as it requires significantly more parameters. The comparison results are shown in Table 3.3. It can be seen that the baseline performs favorably in terms of PSNR and SSIM metrics, which provides strong evidence in support of our design of the SDI module.

3.6.7.2 Justification of the Reconstruction Module

To justify the design of the reconstruction module, two variants are considered for comparison. The first one (w/o attention) has all channel-wise feature fusion removed in order to show the effectiveness of the attention mechanism, and the second one (w/ RDB) replaces the reconstruction module with a module of similar size — a cascade of 8 RDBs to verify the strength of the reconstruction module in terms of dual-task performance. Quantitative comparisons in Table 3.3 provide strong evidence in favor of our original design.

3.6.7.3 Influence of the Input Sequence Length

Here we investigate how the overall VSR performance depends on the number of input frames. To this end, we consider the cases with 3 consecutive frames as inputs (w/ 3 frames), 5 consecutive frames as inputs (w/ 5 frames), and 7 consecutive frames as inputs (baseline). The corresponding quantitative results are shown in Table 3.3. It is clear that the VSR performance improves as the number of input frames increases, which is expected since one can more likely find the missing information when there are more frames available for exploitation. It is also worth mentioning that due to the effective design of the SDI module, the proposed RawVSR can flexibly handle any input length without adjusting the network size (although increasing the number of input frames typically leads to longer processing time).

3.6.8 Runtime and Model Size Comparisons

Fig. 3.14 compares different VSR methods in terms of runtime and model size as well as PSNR values. Here runtime refers to the average time needed for producing one

Table 3.3: ablation studies for different variants for 4× VSR. **Bold** indicates the best performance.

ablation	SDI			Reconstruction module		Input frame number		Baseline
Variant	w/ EF	w/ improved EF	w/ SF	w/o attention	w/ RDB [42]	w/ 3 frames	w/ 5 frames	RawVSR
Store	28.00/0.8226	28.61/0.8304	28.82/0.8351	28.52/0.8303	28.48/0.8277	28.30/0.8176	28.75/0.8324	**29.04/0.8400**
Painting	28.05/0.7910	28.71/0.8007	28.91/0.8034	28.16/0.8006	28.00/0.7981	28.36/0.7927	28.59/0.8057	**29.02/0.8104**
Train	27.44/0.7481	28.08/0.7565	28.29/0.7556	28.07/0.7568	28.08/0.7555	28.01/0.7481	28.27/0.7554	**28.59/0.7625**
City	28.44/0.7699	28.84/0.7769	28.94/0.7775	28.73/0.7750	28.74/0.7743	28.64/0.7684	28.96/0.7776	**29.08/0.7843**
Walk	26.26/0.7525	27.78/0.7649	27.87/0.7628	26.69/0.7559	27.10/0.7567	27.95/0.7558	28.04/0.7695	**28.06/0.7724**
Average	27.64/0.7768	28.40/0.7859	28.57/0.7869	28.03/0.7837	28.08/0.7825	28.25/0.7765	28.52/0.7881	**28.76/0.7939**

1080p frame (without taking into account the data loading phase) for 4× VSR. It can be seen that the proposed RawVSR is able to achieve the highest PSNR value with moderate model size and near-real-time processing.

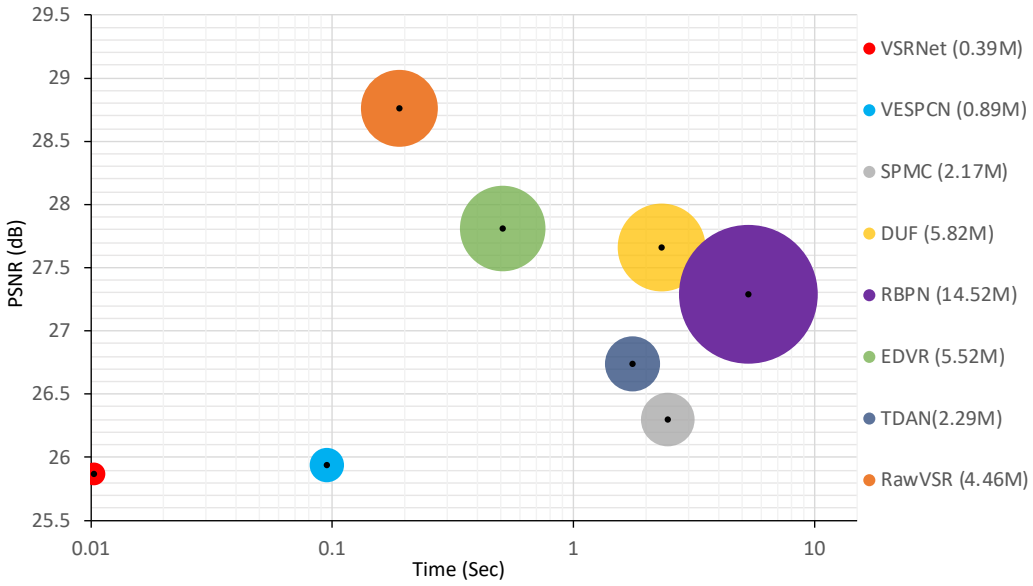


Figure 3.14: Runtime and model size comparisons of different VSR methods for 4× VSR.

3.7 Conclusion

We have proposed a new deep-learning-based method that can effectively exploit raw data for VSR. The associated neural network has a dual-branch structure that disentangles the super-resolution process from the color correction process; as a consequence, it better fits the real imaging pipeline and has the desirable property of being device-independent. Its SDI module is designed according to the architectural principle obtained via the analysis of a suitable probabilistic graphical model; this design philosophy is likely applicable to a wide range of problems, especially those

involving data fusion. Its reconstruction module employs an attention mechanism to enhance the learning ability. Moreover, weight sharing is adopted to reduce the model size and to ensure the latent consistency of intermediate features in the two branches. Finally, it is also expected that the new raw video dataset collected in this paper can potentially benefit other video tasks including, among others, video interpolation and visual enhancement via exploiting raw data.

Bibliography

[1] H. Greenspan, "Super-resolution in medical imaging," *The Computer Journal*, vol. 52, no. 1, pp. 43–63, 2008.

[2] D. H. Trinh, M. Luong, F. Dibos, J.-M. Rocchisani, C. D. Pham, and T. Q. Nguyen, "Novel example-based method for super-resolution and denoising of medical images," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1882–1895, 2014.

[3] H. Demirel and G. Anbarjafari, "Discrete wavelet transform-based satellite image resolution enhancement," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 1997–2004, 2011.

[4] F. Li, X. Jia, D. Fraser, and A. Lambert, "Super resolution for remote sensing images based on a universal hidden markov tree model," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 3, pp. 1270–1278, 2010.

[5] K. Jia and S. Gong, "Generalized face super-resolution," *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 873–886, 2008.

[6] L. Zhang, H. Zhang, H. Shen, and P. Li, "A super-resolution reconstruction algorithm for surveillance images," *Signal Processing*, vol. 90, no. 3, pp. 848–859, 2010.

[7] S. Hao, W. Wang, Y. Ye, E. Li, and L. Bruzzone, "A deep network architecture for super-resolution-aided hyperspectral image classification with classwise loss," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 8, pp. 4650–4663, 2018.

[8] P. H. Hennings-Yeomans, S. Baker, and B. V. Kumar, "Simultaneous super-resolution and feature extraction for recognition of low-resolution faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[9] B. Na and G. C. Fox, "Object detection by a super-resolution method and a convolutional neural networks," in *Proceedings of the IEEE International Conference on Big Data*. IEEE, 2018, pp. 2263–2269.

[10] R. Liao, X. Tao, R. Li, Z. Ma, and J. Jia, "Video super-resolution via deep draft-ensemble learning," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 531–539.

[11] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos, "Video super-resolution with convolutional neural networks," *IEEE Transactions on Computational Imaging*, vol. 2, no. 2, pp. 109–122, 2016.

[12] O. Shahar, A. Faktor, and M. Irani, "Space-time super-resolution from a single video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[13] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6626–6634.

[14] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*

(CVPR), 2018, pp. 3224–3232.

[15] M. Haris, G. Shakhnarovich, and N. Ukita, "Recurrent back-projection network for video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3897–3906.

[16] X. Wang, K. C. Chan, K. Yu, C. Dong, and C. Change Loy, "Edvr: Video restoration with enhanced deformable convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[17] H. Jiang, Q. Tian, J. Farrell, and B. A. Wandell, "Learning the image processing pipeline," *IEEE Transactions on Image Processing*, vol. 26, no. 10, pp. 5032–5042, 2017.

[18] E. Schwartz, R. Giryes, and A. M. Bronstein, "Deepisp: Toward learning an end-to-end image processing pipeline," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 912–923, 2018.

[19] X. Xu, Y. Ma, and W. Sun, "Towards real scene super-resolution with raw images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1723–1731.

[20] V. Bychkovsky, S. Paris, E. Chan, and F. Durand, "Learning photographic global tonal adjustment with a database of input/output image pairs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 97–104.

[21] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multiframe super resolution," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1327–1344, 2004.

[22] C. Liu and D. Sun, "On bayesian adaptive video super resolution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 2, pp. 346–360, 2014.

[23] Z. Ma, R. Liao, X. Tao, L. Xu, J. Jia, and E. Wu, "Handling motion blur in multiframe super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5224–5232.

[24] T. Köhler, X. Huang, F. Schebesch, A. Aichert, A. Maier, and J. Hornegger, "Robust multiframe super-resolution employing iteratively re-weighted minimization," *IEEE Transactions on Computational Imaging*, vol. 2, no. 1, pp. 42–58, 2016.

[25] X. Liu and J. Zhao, "Robust multi-frame super-resolution with adaptive norm choice and difference curvature based btv regularization," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2017, pp. 388–392.

[26] X. Liu, L. Chen, W. Wang, and J. Zhao, "Robust multi-frame super-resolution based on spatially weighted half-quadratic estimation and adaptive btv regularization," *IEEE Transactions on Image Processing*, 2018.

[27] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to see in the dark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3291–3300.

[28] X. Liu, Y. Ma, Z. Shi, and J. Chen, "Griddehazenet: Attention-based multi-scale network for image dehazing," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 7314–7323.

[29] Y. Shao, L. Li, W. Ren, C. Gao, and N. Sang, "Domain adaptation for image dehazing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2808–2817.

[30] X. Xu, L. Siyao, W. Sun, Q. Yin, and M.-H. Yang, "Quadratic video interpolation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 1647–1656.

[31] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 184–199.

[32] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 802–810.

[33] J. Caballero, C. Ledig, A. P. Aitken, A. Acosta, J. Totz, Z. Wang, and W. Shi, "Real-time video super-resolution with spatio-temporal networks and motion

compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 7.

[34] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia, "Detail-revealing deep video super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 22–29.

[35] X. Liu, L. Kong, Y. Zhou, J. Zhao, and J. Chen, "End-to-end trainable video super-resolution based on a new mechanism for implicit motion estimation and compensation," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 2416–2425.

[36] M. Drulea and S. Nedevschi, "Total variation regularization of local-global optical flow," in *14th IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2011, pp. 318–323.

[37] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1874–1883.

[38] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1664–1673.

[39] Y. Tian, Y. Zhang, Y. Fu, and C. Xu, "Tdan: Temporally-deformable alignment network for video super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 3360–3369.

[40] S. Nah, S. Baik, S. Hong, G. Moon, S. Son, R. Timofte, and K. Mu Lee, "Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[41] A. Ignatov, L. Van Gool, and R. Timofte, "Replacing mobile camera isp with a single deep learning model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 536–537.

[42] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[43] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp. 1106–1125, 2019.

[44] K. Hirakawa and T. W. Parks, "Adaptive homogeneity-directed demosaicing algorithm," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 360–369, 2005.

[45] T. Plotz and S. Roth, "Benchmarking denoising algorithms with real photographs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1586–1595.

[46] J. Cai, H. Zeng, H. Yong, Z. Cao, and L. Zhang, "Toward real-world single image super-resolution: A new benchmark and a new model," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 3086–3095.

[47] 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.

[48] W. Bao, W.-S. Lai, X. Zhang, Z. Gao, and M.-H. Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[49] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9308–9316.

[50] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.

[51] R. Zhou, R. Achanta, and S. Süsstrunk, "Deep residual network for joint demosaicing and super-resolution," in *Color and Imaging Conference*, vol. 2018, no. 1. Society for Imaging Science and Technology, 2018, pp. 75–80.

[52] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11 036–11 045.

[53] R. Timofte, R. Rothe, and L. Van Gool, "Seven ways to improve example-based single image super resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1865–1873.

[54] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 136–144.

[55] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

The following chapter is reproduced from a paper on arXiv:

Xiaohong Liu, Yaojie Liu, Xiaoming Liu, and Jun Chen. "PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization.", arXiv preprint arXiv:2103.10596, 2021.

Contribution Declaration: Xiaohong Liu (the author of this thesis) is the first author and the main contributor of this paper (more than 95%). He proposed the algorithm, synthesized all data, wrote the corresponding program, conduct all experiments, and composed the paper. Yaojie Liu participated in the discussions and helped polishing the manuscript; Prof. Xiaoming Liu is the Ph.D. supervisor of Yaojie Liu and the supervisor of Xiaohong Liu when he was a visiting scholar at Michigan State University; Prof. Jun Chen is the Ph.D. supervisor of Xiaohong Liu.

Chapter 4

PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization

4.1 Abstract

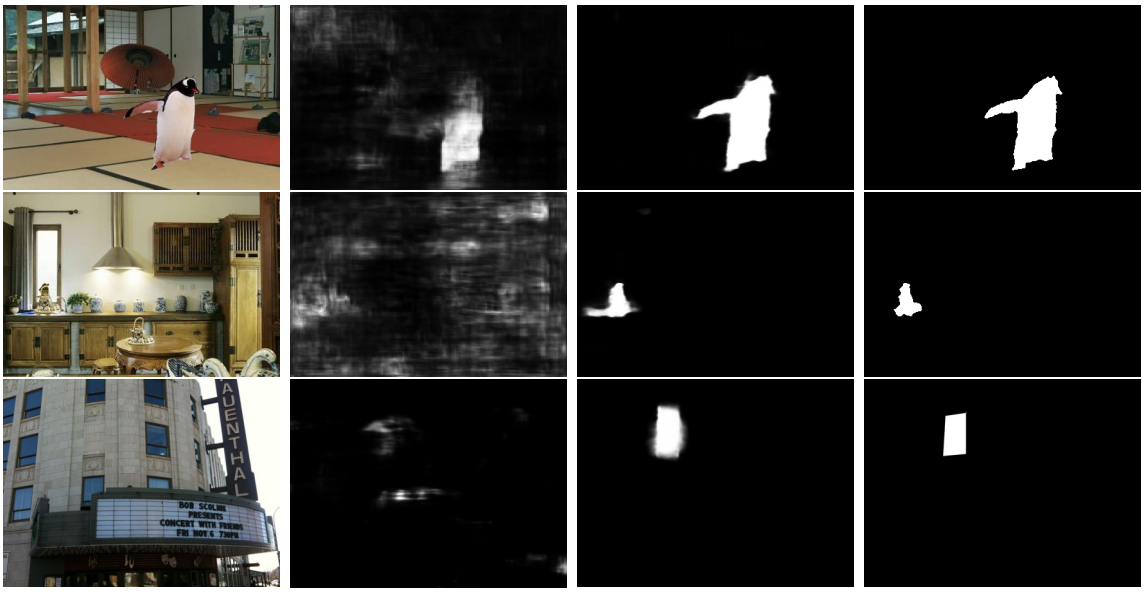
To defend against manipulation of image content, such as splicing, copy-move, and removal, we develop a Progressive Spatio-Channel Correlation Network (PSCC-Net) to detect and localize image manipulations. PSCC-Net processes the image in a two-path procedure: a top-down path that extracts local and global features and a bottom-up path that detects whether the input image is manipulated, and estimates

its manipulation masks at 4 scales, where each mask is conditioned on the previous one. Different from the conventional encoder-decoder and no-pooling structures, PSCC-Net leverages features at different scales with dense cross-connections to produce manipulation masks in a coarse-to-fine fashion. Moreover, a Spatio-Channel Correlation Module (SCCM) captures both spatial and channel-wise correlations in the bottom-up path, which endows features with holistic cues, enabling the network to cope with a wide range of manipulation attacks. Thanks to the light-weight backbone and progressive mechanism, PSCC-Net can process $1,080$P images at 50+ FPS. Extensive experiments demonstrate the superiority of PSCC-Net over the state-of-the-art methods on both detection and localization.

4.2 Introduction

Seeing is believing?

Not anymore. Recent advances on image manipulation techniques [1–4] enable easy editing of raw images, such as removing unwanted objects [5–8], face swapping [2], attribute changing [9], *etc.* Although such techniques are neutral, malicious attackers may utilize them to create deceitful content to propagate false information, *e.g.*, fake news [10], insurance fraud [11], and Deepfake [12, 13]. Thus, concerns of the adverse impact on social media and even real-world systems have been raised [14, 15]. To alleviate the concerns, it is crucial to develop reliable models to expose the manipulated images. While being used in machine and systems, the model is required to, at a minimal, distinguish manipulated images from pristine ones, where the objective is to *detect*. While being used for human's viewing, the model is further required to



(a) Manipulated (b) SPAN [16] (c) PSCC-Net (d) GT

Figure 4.1: Examples of image manipulation localization. Three examples are splicing, copy-move, and removal manipulations respectively. With novel designs of progressive mechanism and correlation module, our method demonstrates robust and accurate estimation at different scales and types.

estimate tampered areas in forged images, where the objective is to *localize*.

Generally, image manipulation consists of the content-dependent process and content-independent process. The former includes splicing, copy-move, and removal, as shown in Fig. 4.1. Both splicing and copy-move are content-copying forgeries, where the splicing content is from a different donor image while the copy-move content is from the target image *per se*. Removal takes out certain objects from the target image and performs refilling via inpainting. Often, the content-dependent process follows the semantic arrangement in the target image, *e.g.*, placing a car on the road and replacing one face with another, which makes the resulting image visually "authentic" and indistinguishable from the pristine one. However, based on image/camera trace analysis [17, 18], subtle patterns can still be revealed to indicate

the manipulation. On the other hand, the content-independent process includes universal modifications such as brightness/contrast change, blurring, noising and image compression. They barely create any disinformation, but their resultant noise may undermine the analysis of image/camera traces and potentially hide the discrepancy between the manipulated and pristine areas.

To defend against manipulations, many Image Manipulation Detection and Localization (IMDL) methods have been proposed in the past. In the early stages, methods are designed to handle a single type of manipulation. In recent years, works [11, 13, 16, 19–22] are proposed to build generic IMDL models for *multiple* manipulation types. However, there are still 3 major unsolved problems for IMDL:

1. **Scale variation:** The forged area varies in sizes. Most prior works neglect the importance of scale variations and encounter difficulty when detecting forged areas of different sizes. Both the conventional encoder-decoder [21, 22] and no-pooling [11,16] structures have difficulties in leveraging local and global features jointly, thus can only handle a limited scale variation.

2. **Image correlation:** Manipulated regions can best be determined while comparing to pristine regions, especially for splicing attacks. A naive learning of mapping from the manipulated image to manipulation mask may lead to an overfitting to the specific attack type in training. In contrast, considering the image spatial correlation can lead to a more generalized localization solution. Yet, such correlation is mostly neglected in prior works.

3. **Detection:** In principle, manipulation detection and localization are highly relevant tasks, where the detection score can be simply derived from the response of the predicted manipulation mask, *i.e.*, at least one part of the forged image

has high response while no part of the pristine one does. However, most prior works assume the *existence of manipulation* in all input images. As a result, this could cause many false alarms on pristine images and make the detection unreliable.

To address the above issues, we propose a novel Progressive Spatio-Channel Correlation Network (PSCC-Net), as in Fig. 4.2. PSCC-Net consists of a top-down path and a bottom-up path. In the top-down path, a backbone encoder first extracts the local and global features from an input image. We adopt the network structure of [23] as our encoder, whose dense connections among different scales facilitate information exchange. In the bottom-up path, we leverage the learned features to estimate 4 manipulation masks from small scales to large ones, where each mask serves as a prior in the next-scale estimation. Thanks to such a design, the final mask is estimated in a coarse-to-fine fashion, harvesting both the local and global information. Moreover, this design enables a potential speed-up by terminating the bottom-up mask estimation, if the intermediate mask is satisfactory. Moreover, rather than investigating the response of predicted manipulation masks, we feed the learned features into a detection head to produce the score for binary classification.

To cope with the image correlation, we propose a Spatio-Channel Correlation Module (SCCM) that grasps both spatial and channel-wise correlations at each bottom-up step. The spatial correlation aggregates the global context among local features. The channel-wise correlation computes the similarity among feature maps to enhance the representation in interest areas. Given the light-weight design of the encoder, PSCC-Net can process $1,080$P at 50+ FPS. Our proposed approach demonstrates a superior manipulation localization on several benchmarks. In addition, we show that

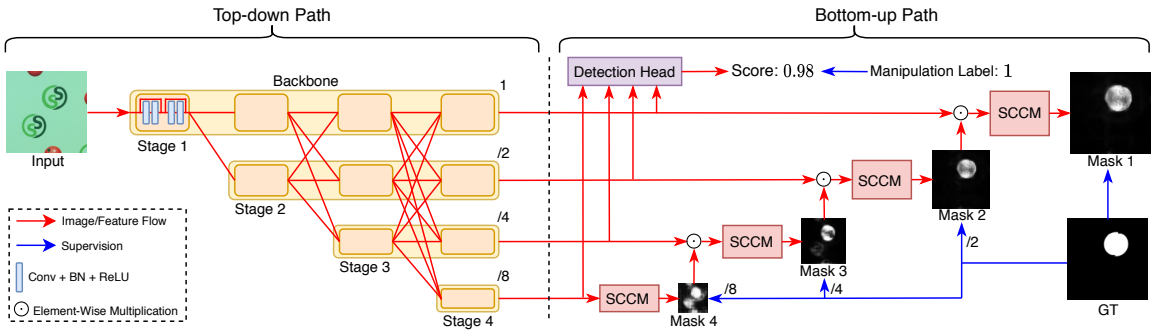


Figure 4.2: The architecture of the proposed PSCC-Net. The detection score predicted by the detection head indicates if the input is manipulated or not. The accuracy of manipulation localization from *Mask 4* to *Mask 1* is gradually improved, *e.g.*, the prediction of *Mask 4* confuses the pasted (forged) region with the pristine (copied) one, while *Mask 1* effectively fixes it.

the recent IMDL methods encounter difficulty in distinguishing manipulated images from pristine ones. By explicitly introducing a detection head, our method achieves the State Of The Art (SOTA) on manipulation detection.

We summarize the contributions of this work as follows:

1. We propose a new PSCC-Net that performs favorably on manipulation detection and enables progressive improvement of manipulation localization in a coarse-to-fine fashion;

2. We design a novel SCCM module to capture the spatial and channel-wise correlations for better generalization. SCCM avoids the use of massive annotated data to pre-train our feature extractor;

3. We achieve the SOTA results for both image manipulation detection and localization.

4.3 Related Work

4.3.1 Image Manipulation Detection

Image manipulation detection aims to distinguish manipulated images from pristine ones via image-level binary classification. There are two major approach for this detection: the implicit manner [10,24] and the explicit manner [25]. The former obtains the detection score by the statistics (*e.g.*, average [10] or maximum [24] value) of the predicted manipulation mask, and the latter explicitly outputs the score from a dedicated classification module. Recent works [11, 16] focus on pixel-level manipulation localization but neglect the importance of image-level detection. Instead, this work leverages both manipulated and pristine images in training and jointly considers detection and localization of image manipulation.

4.3.2 Image Manipulation Localization

Early works propose to localize the manipulation of one specific type, *e.g.,* splicing [10, 17, 26–31], copy-move [24, 25, 32–34], removal [35], and the content-preserved process [22, 36]. Although most methods perform well on detecting that specific forgery type, they fall short in handling real-world cases, where usually the forgery type is unknown in advance and various types of forgery might be utilized in manipulation. In the related problem of face anti-spoofing, researchers also study how to localize the facial pixels covered with various spoof mediums [37].

Recent works attempt to tackle multiple forgeries in one model. J-LSTM [19] and H-LSTM [22] integrate the LSTM and CNN to capture the boundary-discriminative features. However, due to the patch-based design, both methods are time-consuming,

and the size of detectable regions is limited by the preset patch size. RGB-N [21] adopts the steganalysis rich model [38] and Faster R-CNN [39], but it can only provide bounding boxes instead of segmentation masks. Later, ManTra-Net [11] learns features to distinguish 385 known manipulation types and treats the problem as anomaly detection. To learn the distinguishable features, auxiliary labeled data, such as camera sensors, are used. SPAN [16] extends ManTra-Net to further model the spatial correlation via local self-attention blocks and pyramid propagation. However, as the correlation is only considered in the local region, ManTra-Net and SPAN fail to take full advantage of the spatial correlation and consequently have limited generalizability. In this work, our PSCC-Net utilizes a progressive mechanism to improve the multi-scale feature representation and SSCM modules to better explore spatial and channel-wise correlations.

4.3.3 Progressive Mechanism

Progressive mechanism tackles a challenging task in a coarse-to-fine fashion. It has been widely adopted in many low-level and high-level vision tasks, such as dehazing/deraining [40, 41], inpainting [5], super-resolution [42, 43], and object detection [44–47]. The pyramid structure is commonly utilized to build multi-scale features. In this work, we propose a densely connected pyramid structure that progressively refines the manipulation mask from small scales to large ones, where each predicted mask serves as a prior for the next-scale estimation.

4.3.4 Attention Mechanism

The pioneer work [48] proposes an attention mechanism to improve the feature representation with relatively low cost, which has been widely employed in various vision tasks [13, 25, 49–53]. According to the applied domain, the attention mechanism can be divided into two types: spatial attention [50] and channel-wise attention [49]. Recent works [54–56] take the benefit of both types to further improve the representation capability of DNN. These methods adopt separate schemes to explore the spatial and channel-wise attentions and thus require additional efforts to fuse them. In this work, a uniform SCCM jointly explores the image correlation and discrepancy in both spatial domain and feature channels, leading to better information sharing and faster inference.

4.4 PSCC-Net

Our PSCC-Net enables the detection and localization of various types of manipulations. As compared to the image-level detection, the pixel-level localization is more difficult. Therefore, PSCC-Net pays special attention to tackling the localization problem. Indeed, since the features for detection and localization are jointly learned, improving the localization performance will naturally benefit detection.

4.4.1 Network Architecture

4.4.1.1 Top-Down Path

Most prior works use the conventional encoder-decoder [21, 22] and no-pooling structures [11, 16] to extract features. Since forged areas have various sizes, it is important to fuse local and global features to handle the scale variation. However, both structures extract features in a sequential pipeline and neglect feature fusion among different scales, and thus can only handle a limited scale variation. To address this issue, we adopt a light-weight backbone in [23], named HRNetV2p-W18. Following its default setting, the stage down-scaling ratio s is set to 2, and there are totally 4 stages.

Compared to encoder-decoder and no-pooling structures, the benefits of our backbone are two-fold. First, features from different scales are computed in parallel. Hence, dense connections among different scales enable effective information exchange, which is beneficial for handling scale variations. Second, since the local and global feature fusion is performed for every scale, each feature contains sufficient information to predict a manipulation mask at the corresponding scale. Therefore, this backbone is in line with our progressive mechanism, where the prediction of each mask should rely on all local and global features to improve its accuracy. Indeed, except the predicted mask on the last scale, the others serve as a prior for the next-scale mask prediction. After the top-down path, the manipulated features on 4 scales are extracted. Then, we use the bottom-up path to perform manipulation detection and localization.

4.4.1.2 Bottom-Up Path

The bottom-up path in PSCC-Net estimates the detection score and the manipulation mask. Specifically, the detection score is predicted based on the extracted features from the top-down-path via a detection head [23], then the manipulation mask is generated through a progressive mechanism with full supervision. In particular, the coarse-to-fine progressive mechanism mimics how human tackles complicated problems in daily life.

We denote the input image as $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$. The extracted features at 4 scales are $\mathbf{F}_1 \in \mathbb{R}^{H\times W\times C}$, $\mathbf{F}_2 \in \mathbb{R}^{H/s\times W/s\times sC}$, $\mathbf{F}_3 \in \mathbb{R}^{H/s^2\times W/s^2\times s^2C}$ and $\mathbf{F}_4 \in \mathbb{R}^{H/s^3\times W/s^3\times s^3C}$, and their corresponding masks are denoted as $\mathbf{M}_1 \in \mathbb{R}^{H\times W}$, $\mathbf{M}_2 \in \mathbb{R}^{H/s\times W/s}$, $\mathbf{M}_3 \in \mathbb{R}^{H/s^2\times W/s^2}$ and $\mathbf{M}_4 \in \mathbb{R}^{H/s^3\times W/s^3}$. Here H, W, and C are the height, width, and channel number of the image/feature respectively. Formally, we have

$$\mathbf{M}_{n-1} = f_{n-1}(\tau(\mathbf{M}_n)\cdot \mathbf{F}_{n-1}), \quad n = 2, 3, 4, \tag{4.4.1}$$

where f_n denotes the SCCM on the nth scale, and τ is the upsampling operation (*e.g.*, the bilinear interpolation). Since $\mathbf{M}_4$ is the mask on the last scale, it can be directly expressed as $\mathbf{M}_4 = f_4(\mathbf{F}_4)$. For Scales 1-3, the feature on the current scale is associated with the upsampled mask from the previous scale for feature modulation. Then, the modulated feature is fed into SCCM to produce a manipulation mask.

To reduce the prediction difficulty, the proposed progressive mechanism avoids generating the mask at the finest scale directly. Instead, the mask on the coarsest scale is first predicted to locate the regions that are potentially forged based on current available information. The subsequent prediction on the finer scale can leverage

the previous mask and pay more attention to those selected regions. This process continues until the generation of the manipulation mask at the finest scale, which serves as the final prediction. However, without explicit supervision on each scale, the intermediate masks might not follow the coarse-to-fine order. Therefore, full supervisions are applied on all scales to guide the mask estimation.

4.4.2 Spatio-Channel Correlation Module

Attention mechanisms are commonly used to modulate learned features according to their relative significance. As the final manipulation mask is binary, the localization can be considered as a pixel-level binary classification. Ideally, we expect the learned features on forged regions are similar to each other but distinct from those in pristine regions. In this case, a simple clustering method may suffice to produce an effective mask. Therefore, to better tackle manipulation localization, we propose a SCCM that employs the spatial attention to aggregate the pixel-level features based on their contextual correlations, and the channel-wise attention to consolidate the feature maps based on their channel correlations.

We illustrate the detailed structure of SCCM in Fig. 4.3, where the input feature $\mathbf{X}$ is of size $H \times W \times C$. Note that even though $\mathbf{X}$ is small (256×256), the size of its spatial correlation can be enormous ($65,536 \times 65,536$), easily exceeding the memory limit. Therefore, inspired by [42], we use function h to reshape the input $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ to $\mathbf{X}' \in \mathbb{R}^{HW/r^2 \times Cr^2}$, where each feature map is flattened to form a vector based on SCCM down-scaling ratio r. This operation preserves all feature information and avoids modeling the spatial correlation of potentially large size $HW \times HW$.

To build the spatial and channel-wise correlations, one may directly leverage $\mathbf{X}'$.

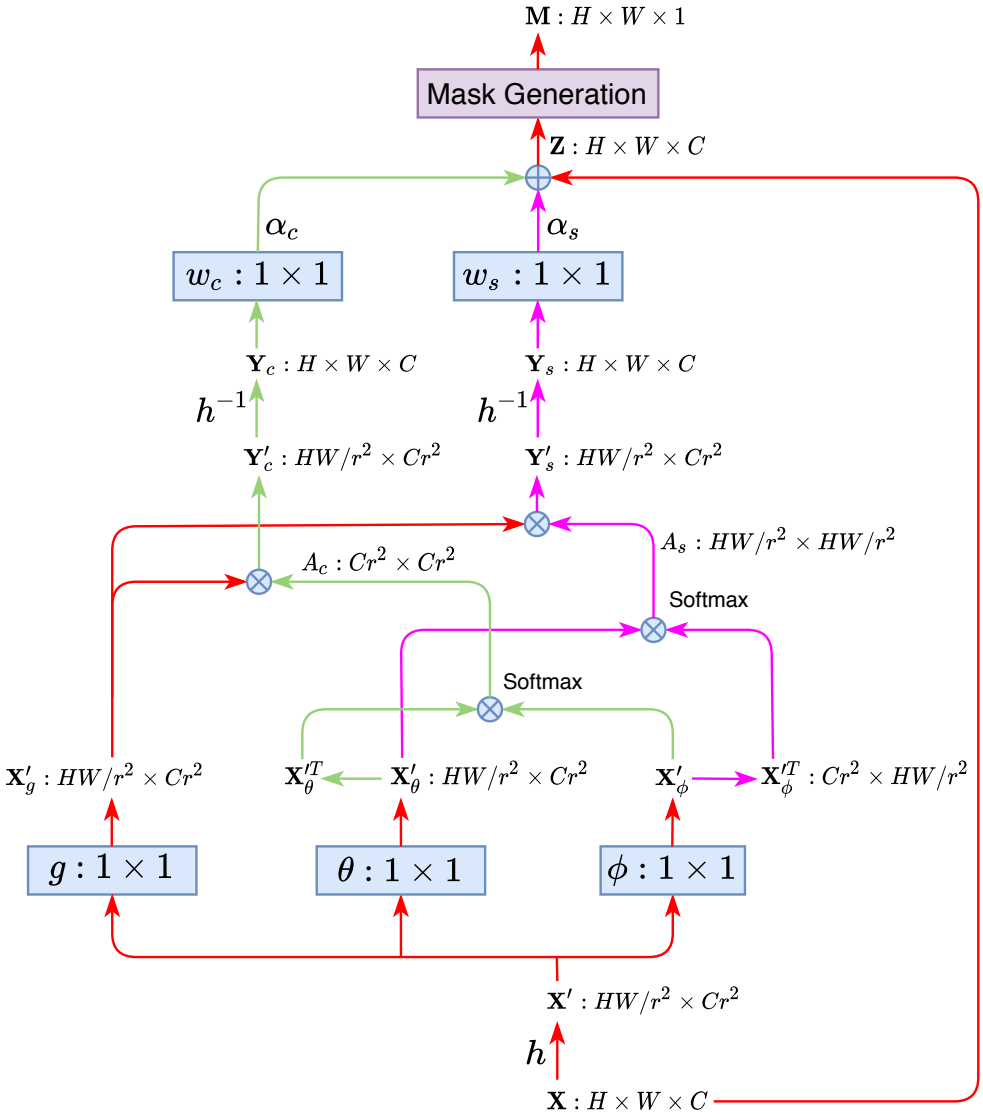


Figure 4.3: The structure of SCCM. Here $\otimes$ represents the matrix multiplication and $\oplus$ the element-wise addition; the red arrow shows the common feature flows; the pink and green arrows show the feature flows of spatial and channel-wise attentions respectively.

However, additional flexibility could be achieved by introducing the embedded Gaussian function [50]. Therefore, we use the 1×1 convolution to build different functions g, θ, and ϕ to transform $\mathbf{X}'$ into new linear embeddings as $\mathbf{X}'_g = g(\mathbf{X}')$, $\mathbf{X}'_\theta = \theta(\mathbf{X}')$, and $\mathbf{X}'_\phi = \phi(\mathbf{X}')$, all with the same size as $\mathbf{X}'$. Subsequently, the spatial and channel-wise correlations (denoted as $\mathbf{A}_s \in \mathbb{R}^{HW/r^2 \times HW/r^2}$ and $\mathbf{A}_c \in \mathbb{R}^{Cr^2 \times Cr^2}$) of embedded features $\mathbf{X}'_\theta$ and $\mathbf{X}'_\phi$ are computed, and the Gaussian operation is implemented by Softmax function. In the end, the spatial and channel-wise attentions are realized by performing matrix multiplications $\mathbf{A}_s\mathbf{X}'_g$ and $\mathbf{X}'_g\mathbf{A}_c$, respectively. Unlike prior methods [54–56] that employ two attentions on *different* features, we apply both on the *same* linear embedding for better information sharing and faster inference. Indeed, applying attentions in this way reduces the difficulty of subsequent fusion process, and also saves computational operations in SCCM. Specifically, the spatial attention can be formulated as:

$$\mathbf{Y}'_s = \mathbf{A}_s\mathbf{X}'_g = \text{softmax}(\mathbf{X}'_\theta\mathbf{X}'^T_\phi)\mathbf{X}'_g, \tag{4.4.2}$$

where $\mathbf{Y}'_s \in \mathbb{R}^{HW/r^2 \times Cr^2}$ is the feature resulting from the application of spatial attention, and softmax$(\cdot)$ denotes the Softmax function. The element (i, j) in $\mathbf{A}_s$ indicates the similarity between the feature vectors in the ith row of $\mathbf{X}'_\theta$ and jth row of $\mathbf{X}'_\phi$. The more similar they are, the higher correlation they have. This helps the network to learn feature representations for distinguishing forged regions from pristine ones and avoid overfitting to a specific attack type in training. Similarly, the channel-wise attention is expressed as:

$$\mathbf{Y}'_c = \mathbf{X}'_g\mathbf{A}_c = \mathbf{X}'_g\text{softmax}(\mathbf{X}'^T_\theta\mathbf{X}'_\phi), \tag{4.4.3}$$

where $\mathbf{Y}'_c \in \mathbb{R}^{HW/r^2 \times Cr^2}$ is the feature resulting from the application of channel-wise attention. The element (i, j) in $\mathbf{A}_c$ measures the similarity between the channel maps in the ith column of $\mathbf{X}'_\theta$ and jth column of $\mathbf{X}'_\phi$. Since the response from different channels might be associated with the same class, *e.g.*, manipulated or pristine, the channel-wise correlation aggregates feature maps based on their similarities to enhance the representation in forged regions.

We use h^{-1} to reshape $\mathbf{Y}'_s$ and $\mathbf{Y}'_c$ respectively back to $\mathbf{Y}_s$ and $\mathbf{Y}_c$ of size $H \times W \times C$. Further, two functions ω_s and ω_c are built by 1×1 convolution to improve their feature representations. The output features from ω_s and ω_c are complement to each other. As it is non-trivial to determine their relative significance, two learnable parameters α_s and α_c, both initialized as 1, are used for trade-off. We also adopt the residual learning [57] to express the feature $\mathbf{Z}$ as:

$$\mathbf{Z} = \mathbf{X} + \alpha_s \cdot \omega_s(\mathbf{Y}_s) + \alpha_c \cdot \omega_c(\mathbf{Y}_c). \tag{4.4.4}$$

The final output of SCCM is a predicted mask with only one channel. To reduce the channel number in $\mathbf{Z}$, we employ a mask generation block with the sequential order of *Conv-ReLU-Conv-Sigmoid*, where *Conv* is a 3×3 convolution.

4.4.3 Loss Function

To train the PSCC-Net, we adopt the binary cross-entropy loss (L_{bce}) for both detection and localization tasks. The predicted detection score (s_d) is supervised by the Ground-Truth (GT) label (l_d) with 0 standing for pristine image and 1 for forged image. Moreover, full supervisions are applied on each predicted mask by downsampling the GT mask $\mathbf{G}_1$ to $\mathbf{G}_2$, $\mathbf{G}_3$, and $\mathbf{G}_4$ according to their corresponding sizes, with 0

standing for pristine pixel and 1 for forged pixel. The masks predicted through the progressive mechanism at different scales are considered to be of equal importance. Therefore, our final loss function $\hat{L}$ can be expressed as:

$$\hat{L} = L_{bce}(s_d, l_d) + \frac{1}{4}\sum_{m=1}^{4} L_{bce}(\mathbf{M}_m, \mathbf{G}_m). \tag{4.4.5}$$

4.4.4 Training Data Synthesis

Since there is no standard IMDL dataset for training, a synthetic dataset is built to train and validate our PSCC-Net. This dataset includes four categories 1) splicing, 2) copy-move, 3) removal, and 4) pristine classes. For splicing, inspired by [30, 58], we use the MS COCO [59] to generate spliced images, where one annotated region is randomly selected per image, and pasted into a different image after several transformations. We adopt the same transformation as [30] including the scale, rotation, shift and luminance changes. Since the spliced region is not necessarily an object, we use the Bezier curve [60] to generate random contours, then fill them to produce splicing masks. We follow the same processes above but randomly select donor and target images in KCMI [61], VISION [62], and Dresden [63] that are commonly used to identify camera source [18], to generate additional spliced images as supplementary. For copy-move, the dataset from [24] is adopted. For removal, we adopt the SOTA inpainting method [6] to fill one annotated region that is randomly removed from each chosen MS COCO image. As to the pristine class, we simply select images from the original datasets mentioned above.

In summary, we have ∼100k images per class, thus 400k in total. As it is inefficient to train all manipulated images in one epoch, we uniformly sample 25k images per

class to form a 100k dataset on-the-fly for training in each epoch. In addition, we also build a validation set that contains 4×100 images. The size of synthetic images are all set to 256×256.

4.5 Experiments

4.5.1 Experimental Setup

4.5.1.1 Test Data

We evaluate the manipulation localization on 4 standard test datasets: Columbia [64], Coverage [33], CASIA [65] and NIST16 [66], and 1 real-world dataset: IMD20 [67]. To finetune PSCC-Net, we follow the same training/testing split on Coverage, CASIA, and NIST16 as in [16, 21] for fair comparisons. Specifically, Columbia [64] is a splicing dataset of 180 images. Coverage [33] is a copy-move dataset of 100 images; for fine-tuning, it is split into 75/25 for training and testing. CASIA [65] (v1.0 + v2.0) includes both splicing and copy-move; for fine-tuning, $5,123$ images from v2.0 is adopted for training, and 921 images from v1.0 is for testing. NIST16 [66] has 564 images, involving all three manipulations; for fine-tuning, 404 images are used for training and 160 for testing. IMD20 [67] consists of $2,010$ real-life manipulated images collected from Internet.

As the manipulation detection is not considered by prior works, there is no standard dataset for benchmarking. To address this issue, we include both forged and pristine images in CASIA dataset and define a evaluation protocol for detection. This dataset is named CASIA-D and consists of $1,842$ images with 50% forged and 50% pristine.

4.5.1.2 Metrics

To quantify the localization performance, following previous works [11, 16], we use pixel-level Area Under Curve (AUC) and F1 score on manipulation masks. To evaluate the detection performance, we use image-level AUC and F1 score, Equal Error Rate (EER), and True Positive Rate at 1% false positive rate ($\text{TPR}_{1\%}$). Since binary masks and detection scores are required to compute F1 scores, we adopt the EER threshold to binarize them.

4.5.1.3 Implementation Details

PSCC-Net is end-to-end trainable and light-weighted. Its top-down path and bottom-up path have 2.0 and 1.6 Million (M) parameters. In the bottom-up path, the detection head has 0.9 M and the rest part (for localization) has only 0.7 M parameters. In comparison, the ManTra-Net [11] and SPAN [16] have 3.8 and 3.7 M parameters, respectively. Implemented by PyTorch, our model is trained with two NVIDIA GeForce GTX 1080Ti, but only one is used in testing. We initialize our backbone with ImageNet pre-trained weights, and optimize the whole model by Adam [68] with a batch size of 10 and an initial learning rate of $2e$-4. The learning rate is halved every 5 epochs and the total training period is 25 epochs.

Our network can take arbitrary-size images as input. To avoid performance degradation caused by size mismatch between training (*e.g.*, 256×256) and testing data (*e.g.*, $4,000 \times 3,000$), at the end of top-down path, we resample the extracted features from the first to the last scales respectively into fixed sizes 256×256, 128×128, 64×64, and 32×32, where the ratio r in SCCM is set to 4, 2, 2, and 1 respectively to reduce the computational burden. The produced masks are resampled back to the

Table 4.1: Localization AUC (%) of pre-trained models.

Method	Columbia	Coverage	CASIA	NIST16	IMD20
ManTra-Net [11]	82.4	81.9	81.7	79.5	74.8
SPAN [16]	93.6	**92.2**	79.7	84.0	75.0
PSCC-Net	**98.2**	84.7	**82.9**	**85.5**	**80.6**

same size as the input image for localization evaluation.

4.5.2 Comparisons on Localization

Our baseline IMDL methods include ELA [69], NOI1 [70], CFA1 [71], J-LSTM [19], H-LSTM [22], RGB-N [21], ManTra-Net [11], and SPAN [16] where SPAN has reported the SOTA performance on localization. Following the evaluation protocol defined in SPAN [16], we compare the localization performance using two models: 1) the pre-trained model is trained on the synthetic dataset and evaluated on the *full* test datasets, and 2) the fine-tuned model is the pre-trained model further fine-tuned on the training split of test datasets and evaluated on their *test split*.

4.5.2.1 Pre-Trained Model

We choose the best pre-trained model based on the performance on our validation set. Tab. 4.1 shows the localization performance of pre-trained models for different methods on 4 standard datasets and 1 real-world dataset under pixel-level AUC. The pre-trained PSCC-Net achieves the best localization performance on Columbia, CASIA, NIST16, and IMD20, and ranks the second on Coverage. The most significant performance gain is achieved while tackling real-life manipulated images (5.6% ↑). This validates that the PSCC-Net has the best generalization ability as compared to

Table 4.2: Evaluation of the fine-tuned models. Localization AUC/F1s are reported (in %). Type U denotes an unsupervised model, and type F denotes a fine-tuned model. ManTra-Net is not shown here as it has only developed the pre-trained model.

Method	Type	Coverage	CASIA	NIST16
ELA [69]	U	58.3 / 22.2	61.3 / 21.4	42.9 / 23.6
NOI1 [70]	U	58.7 / 26.9	61.2 / 26.3	48.7 / 28.5
CFA1 [71]	U	48.5 / 19.0	52.2 / 20.7	50.1 / 17.4
J-LSTM [19]	F	61.4 / -	- / -	76.4 / -
H-LSTM [22]	F	71.2 / -	- / -	79.4 / -
RGB-N [21]	F	81.7 / 43.7	79.5 / 40.8	93.7 / 72.2
SPAN [16]	F	93.7 / 55.8	83.8 / 38.2	96.1 / 58.2
PSCC-Net	F	**94.1 / 72.3**	**87.5 / 55.4**	**99.6 / 81.9**

the others. We fail to achieve the best performance on Coverage, despite surpassing ManTra-Net 2.8% under AUC. The reason might be the imperfection of our training data for the case, where the copied object is intentionally moved to cover a pristine object with similar appearance. Indeed, by fine-tuning the pre-trained model on Coverage, PSCC-Net achieves the 0.4% gain over SPAN under AUC (Tab. 4.2).

4.5.2.2 Fine-Tuned Model

We further fine-tune the pre-trained model on specific datasets using our training strategy. The cross validation on training data helps to select the best fine-tuned models on each test dataset. We compare the fine-tuned models in Tab. 4.2. For AUC, PSCC-Net surpasses baselines in all cases (over 2.5% to SPAN on average). As for F1 score, our model outperforms them with a large margin (over 19% to SPAN on average).

4.5.3 Comparisons on Detection

Since ManTra-Net and SPAN are the best performing baselines in the localization evaluation, and ManTra-Net does not develop the fine-tuned model, we choose to use the pre-trained model for detection evaluation, in order to make comparisons to both of them. Although these two baselines make no direct attempt to perform detection, their estimated manipulation masks can be leveraged for this purpose. As such, we simply regard the average of the mask as their scores. For fair comparisons, we build a variant that adopts the same averaging strategy to calculate this score, denoted as PSCC-Net†. In Tab. 4.3, owing to our well-predicted manipulation masks, the PSCC-Net† achieves the best detection performance on all used metrics. It is evident that the detection performance can be dramatically improved by introducing a tailored head. With a favorable detection, the IMDL methods can be more efficient. That is, detection is performed before localization, and only the detected forgery is passed for localization. Our network design is compatible with this efficiency consideration as the detection head is placed at the beginning of the bottom-up path.

Table 4.3: Detection evaluation on CASIA-D, all reported in %.

Method	AUC ↑	F1 ↑	EER ↓	$TPR_{1\%}$ ↑
ManTra-Net [11]	59.94	56.69	43.21	5.43
SPAN [16]	67.33	63.48	36.47	5.54
PSCC-Net†	74.40	66.88	33.21	28.37
PSCC-Net	**99.65**	**97.12**	**2.83**	**95.65**

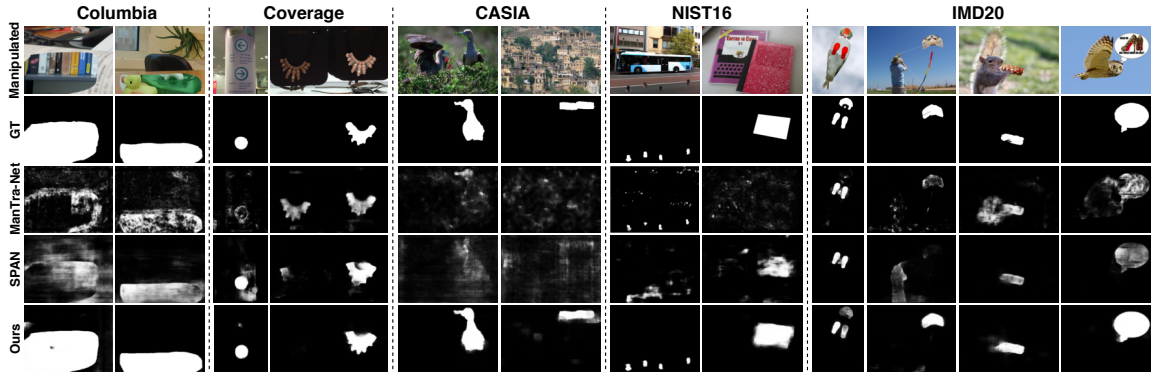
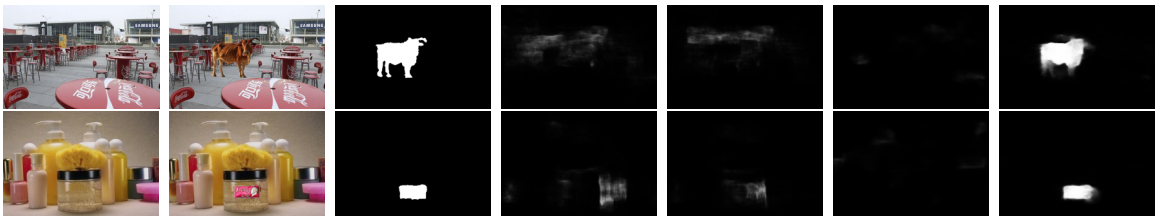


Figure 4.4: Qualitative localization evaluations on 5 datasets. From top to bottom, we show manipulated images, GT manipulation masks, predictions of ManTra-Net, SPAN, and ours. Best models are used to produce masks. Zoom in for details. See Sec. 4.7 for more results.



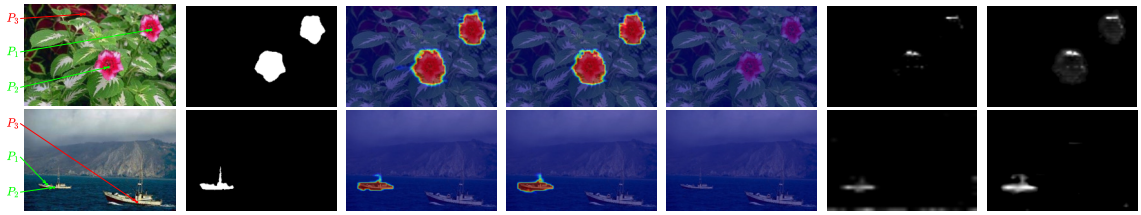
(a) Pristine image (b) Manip. image (c) GT Manip. mask (d) Pristine mask [16] (e) Manip. mask [16] (f) Our pristine mask (g) Our manip. mask

Figure 4.5: Qualitative detection evaluations on CASIA-D. Since GT pristine masks are blank, they are not shown here.

4.5.4 Visualization, Ablation and Analysis

4.5.4.1 Qualitative Results

We provide qualitative evaluations of manipulation localization and detection in Figs. 4.4, 4.5. PSCC-Net predicts more accurate and sharper manipulation masks while maintaining low false alarms on pristine regions, especially for *small* manipulations.



(a) Manipulated (b) GT (c) P_1 response (d) P_2 response (e) P_3 response (f) 1st channel in $\mathbf{X}$ (g) 1st channel in $\mathbf{Y}_c$

Figure 4.6: Visualization of spatial and channel-wise attentions in SCCM. For each row, we show a manipulated image, its GT mask, 3 spatial response maps (one for each selected pixel), and the 1st channel map in $\mathbf{X}$ and $\mathbf{Y}_c$. Zoom in for details.

4.5.4.2 Visualization of SCCM

To provide insights into SCCM, we visualize the spatial response map for forged and pristine pixels in $\mathbf{M}_3$, by examining its spatial correlation represented in $\mathbf{A}_s$. After interpolation, each row of $\mathbf{A}_s$ is associated with one pixel (*e.g.*, P_1) in the manipulated image, and its grayscale spatial response map can be obtained by reshaping this row vector from $1 \times HW$ to $H \times W$ (*e.g.*, P_1 response). In Fig. 4.6 (a), 2 examples of splicing and copy-move manipulations from CASIA are shown in the 1st and 2nd rows. We select 3 representative pixels for each image and annotate as P_1, P_2, and P_3, where P_1 and P_2 are from forged regions, and P_3 is from pristine regions. We project their grayscale spatial response maps into *Jet* color space and overlay them on the manipulated image as in Figs. 4.6 (c-e). It is evident that the spatial response maps of P_1 and P_2 have high values in forged regions and low values in pristine regions, but the map of P_3 retains low values in all regions including the one providing the copied content (*e.g.,* the P_3 response in the 2nd row of Fig. 4.6 (e)). This visualization indicates that the features in forged regions are clustered together, thus justifies the effectiveness of spatial attention in SCCM.

Table 4.4: Ablation study of PSCC-Net. Average AUC/F1s are reported (in %). The run time (in proportion) is relative to that of PSCC-Net. Our full model takes $0.019s$ to process one $1,080$P image, whereas ManTra-Net and SPAN take $0.208s$ and $0.161s$, respectively. Terminating the prediction earlier on *Mask 4* can shorten the run time to $0.012s$, *i.e.*, $\sim 37\%$ additional saving.

Variants	Columbia	Coverage	CASIA	NIST16	Time
Mask 4	93.34 / 79.22	82.99 / 44.23	81.49 / 31.69	84.15 / 30.55	0.63
Mask 3	98.08 / 92.41	83.48 / 47.29	82.55 / 34.64	85.25 / 33.55	0.75
Mask 2	98.18 / 93.32	84.44 / 49.08	82.78 / 35.59	85.38 / 34.94	0.88
w/o CA+SA	85.78 / 70.32	79.95 / 43.27	79.26 / 31.06	79.58 / 31.73	0.84
w/o SA	90.70 / 75.68	80.56 / 43.50	79.51 / 31.08	83.49 / 32.34	0.92
w/o CA	94.50 / 85.34	82.16 / 45.04	82.63 / 35.97	84.65 / 33.42	0.92
PSCC-Net	**98.19** / **93.45**	**84.65** / **49.78**	**82.93** / **36.27**	**85.47** / **35.73**	1.00

For channel-wise correlation $\mathbf{A}_c$, it is hard to provide a comprehensible visualization. Instead, we choose to visualize one channel of $\mathbf{Y}_c$ and compare it to the same channel of $\mathbf{X}$ to see if any region is enhanced. We visualize the 1st channel of $\mathbf{X}$ and $\mathbf{Y}_c$ in Figs. 4.6 (f,g). Indeed, the forged region in $\mathbf{Y}_c$ is consolidated compared to the one in $\mathbf{X}$, which proves the effectiveness of channel-wise attention in SCCM.

4.5.4.3 Ablation Study

To justify our network design, we test several variants of PSCC-Net to show the effectiveness of progressive mechanism and SCCM in Tab. 4.4, where all variants are pre-trained on our dataset. *Mask 4*, *Mask 3*, and *Mask 2* are the variants that truncate the original model after generating manipulation masks on the 4th, 3rd, and 2nd scales, respectively. The comparisons of *Mask 4*, *Mask 3*, *Mask 2*, and the original PSCC-Net demonstrate the gradual improvement in performance, which is a clear manifestation of our progressive mechanism. Since *Mask 4* performs well under

Table 4.5: Robustness comparison with respect to various distortions. AUCs are reported (in %).

Distortion	Columbia			NIST16		
	[11]	[16]	Ours	[11]	[16]	Ours
Resize (0.78×)	71.66	89.99	**93.40**	77.43	83.24	**85.29**
Resize (0.25×)	68.64	69.08	**78.41**	75.52	80.32	**85.01**
GaussianBlur ($k = 3$)	67.72	78.97	**84.18**	77.46	83.10	**85.38**
GaussianBlur ($k = 15$)	62.88	67.70	**73.24**	74.55	79.15	**79.93**
GaussianNoise ($\sigma = 3$)	68.22	75.11	**82.64**	67.41	75.17	**78.42**
GaussianNoise ($\sigma = 15$)	54.97	65.80	**74.35**	58.55	67.28	**76.65**
JPEGCompress ($q = 100$)	75.00	93.32	**97.97**	77.91	83.59	**85.40**
JPEGCompress ($q = 50$)	59.37	74.62	**89.11**	74.38	80.68	**85.37**
w/o distortion	77.95	93.60	**98.19**	78.05	83.95	**85.47**

AUC and F1 scores, the mask prediction can be terminated earlier to save time. The comparisons among the variants without spatial and channel-wise attentions (*w/o SA+CA*), without spatial attention (*w/o SA*), without channel-wise attention (*w/o CA*), and original PSCC-Net illustrate that both SA and CA outperform the baseline (*w/o SA+CA*), where the performance gain acquired from SA is more than that from CA. Owing to SCCM, the original PSCC-Net achieves the best performance as compared to its attention variants.

4.5.4.4 Robustness Analysis

To analyze the robustness of PSCC-Net for localization, we follow the distortion settings in [16] to degrade the raw manipulated images from Columbia and NIST16. These distortions include resizing images to a different scale, applying Gaussian blur

with kernel size k, adding Gaussian noise with standard deviation σ, and performing compression with quality factor q. Table 4.5 shows the robustness analysis under pixel-level AUC with pre-trained models. The PSCC-Net is more robust than ManTra-Net and SPAN under all distortions. It is worth noting that resizing is commonly performed when uploading images to social media. Indeed, benefiting from the operation that resamples the manipulation features into the fixed sizes, the impact of resizing to PSCC-Net is the least as compared to the others.

4.6 Conclusion

In this work, a novel PSCC-Net is proposed to meet the challenge of advanced image manipulation techniques. We employ a progressive mechanism to predict the manipulation mask on all backbone scales, where each mask serves as a prior to help predict the next-scale mask. Moreover, a SCCM is designed to perform spatial and channel-wise attentions on extracted features, which provides holistic information to make our model more generalized to manipulation attacks. Extensive experiments demonstrate that our PSCC-Net outperforms the SOTA methods on both detection and localization. For future work, we will develop techniques for estimating the uncertainty of predicted manipulation masks.

4.7 Supplementary

4.7.1 Network Details

The proposed PSCC-Net consists of a top-down path and a bottom-up path. We illustrate how the dense cross-connections in our backbone aggregate the information from different scales, and provide the detailed structure of our backbone in the top-down path (based on the HRNetV2p-W18 [23] with several modifications) and the detailed structure of our detection head in the bottom-up path. In addition, we demonstrate the learned values of parameters α_s and α_c in Equ. 4.4.4 for SCCMs at four scales of our pre-trained and fine-tuned models.

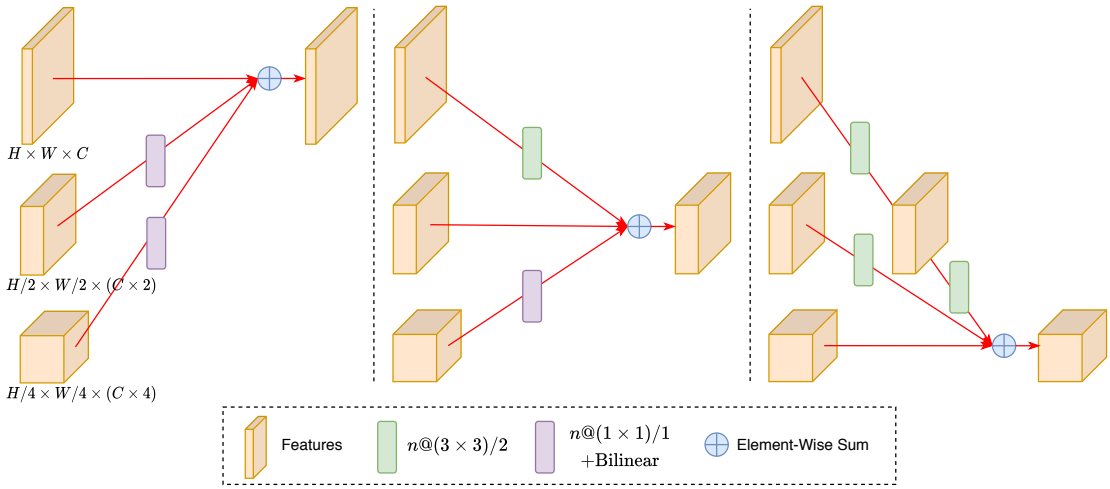


Figure 4.7: An example of dense cross-connections for the first 3 scales in backbone. From left to right, the dense cross-connections to *Scale* 1, 2, and 3 are illustrated. $n@(k \times k)/s$ denotes a $k \times k$ convolution with stride s for n filters, followed by a batch normalization and a ReLU activation function. *Bilinear* stands for bilinear interpolation. H, W, and C represent the height, width, and channel number of features, respectively.

Table 4.6: Size transitions among different feature scales for multi-scale fusions, where the size of *Source* scale is converted according to the size of *Target* scale. $n@(k \times k)/s$ denotes a $k \times k$ convolution with stride s for n filters, followed by a batch normalization and a ReLU activation function. *Bilinear* stands for bilinear interpolation.

Source \ Target	Scale 1	Scale 2	Scale 3	Scale 4
Scale 1	-	$36@(3 \times 3)/2$	$18@(3 \times 3)/2$ $72@(3 \times 3)/2$	$18@(3 \times 3)/2$ $18@(3 \times 3)/2$ $144@(3 \times 3)/2$
Scale 2	$18@(1 \times 1)/1$ Bilinear	-	$72@(3 \times 3)/2$	$36@(3 \times 3)/2$ $144@(3 \times 3)/2$
Scale 3	$18@(1 \times 1)/1$ Bilinear	$36@(1 \times 1)/1$ Bilinear	-	$144@(3 \times 3)/2$
Scale 4	$18@(1 \times 1)/1$ Bilinear	$36@(1 \times 1)/1$ Bilinear	$72@(1 \times 1)/1$ Bilinear	-

4.7.1.1 Dense Cross-Connections

An example of dense cross-connections at the first 3 scales in backbone is demonstrated in Fig. 4.7. From left to right, the multi-scale fusion to *Scale* 1, 2, and 3 are given respectively. A 3×3 convolution with stride 2 is adopted for feature dowmsampling, and a 1×1 convolution with stride 1 followed by a bilinear interpolation is adopted for feature upsampling, which unifies the feature size from different scales. Finally, the sum operation is utilized to perform multi-scale fusions.

In addition, for the sake of completeness, we show all size transitions among different feature scales in Tab. 4.6. Note that for the feature downsampling among nonadjacent scales, the downsampling operation consists of several convolutions with stride 2 to avoid the large stride factor (*e.g.,* 8) that might potentially degrade multi-scale fusions.

Table 4.7: The structure of our backbone. There are 4 stages corresponding to 4 different scales. $n@(k \times k)/s$ denotes a $k \times k$ convolution with stride s for n filters, followed by a batch normalization and a ReLU activation function. $[\cdot]$ represents the residual unit, and $\times 2$ indicates the repeat times of the corresponding residual unit. For clarity, dense cross-connections after *Transition* 2, *Transition* 3, and *Block* 4 are not demonstrated in this table as already shown in Fig. 4.7 and Tab. 4.6.

The structure of our backbone										
Stage	Size	Stem	Block 1	Transition 1	Block 2	Transition 2	Block 3	Transition 3	Block 4	
Stage 1	1	$64@(3\times3)/1$ $64@(3\times3)/1$	$\begin{bmatrix} 64@(1\times1)/1 \\ 64@(3\times3)/1 \\ 256@(1\times1)/1 \end{bmatrix} \times 2$	$18@(3\times3)/1$	$\begin{bmatrix} 18@(3\times3)/1 \\ 18@(3\times3)/1 \end{bmatrix} \times 2$	-	$\begin{bmatrix} 18@(3\times3)/1 \\ 18@(3\times3)/1 \end{bmatrix} \times 2$	-	$\begin{bmatrix} 18@(3\times3)/1 \\ 18@(3\times3)/1 \end{bmatrix} \times 2$	
Stage 2	1/2			$36@(3\times3)/2$	$\begin{bmatrix} 36@(3\times3)/1 \\ 36@(3\times3)/1 \end{bmatrix} \times 2$	-	$\begin{bmatrix} 36@(3\times3)/1 \\ 36@(3\times3)/1 \end{bmatrix} \times 2$	-	$\begin{bmatrix} 36@(3\times3)/1 \\ 36@(3\times3)/1 \end{bmatrix} \times 2$	
Stage 3	1/4					$72@(3\times3)/2$	$\begin{bmatrix} 72@(3\times3)/1 \\ 72@(3\times3)/1 \end{bmatrix} \times 2$	-	$\begin{bmatrix} 72@(3\times3)/1 \\ 72@(3\times3)/1 \end{bmatrix} \times 2$	
Stage 4	1/8							$144@(3\times3)/2$	$\begin{bmatrix} 144@(3\times3)/1 \\ 144@(3\times3)/1 \end{bmatrix} \times 2$	

4.7.1.2 Detailed Structure of Our Backbone

The detailed structure of our backbone is shown in Tab. 4.7. In summary, this backbone contains 1 stem, 4 blocks, 3 transitions, and 3 dense cross-connections. Note that the last dense cross-connection after the *Block* 4 is not shown in Fig. 4.2 for clarity.

In comparison to the original design in [23], instead of setting the stride to 2 for both convolutions in stem, we avoid dowmsampling input images in the beginning by setting them to 1. For all blocks, transitions, and dense cross-connections, we follow the same settings as in [23].

4.7.1.3 Detailed Structure of Our Detection Head

We modify the classification head in [23] as our detection head. Fig. 4.8 shows the overall architecture of our detection head. More details can be found in Tab. 4.8. The output of our detection head is the detection score that predicts whether the input image is manipulated.

4.7.1.4 Values of learned α_s and α_c

In Tab. 4.9, we show the values of learned α_s and α_c for the spatio-channel trade-off in SCCMs of our models. It can be seen that α_s and α_c are insensitive to training data since the values are quite similar among different models.

4.7.2 Limitations

Our PSCC-Net enables to detect and localize various types of manipulations. As compared to the image-level detection, the pixel-level localization is more challenging,

Table 4.8: The structure of our detection head. $n@(k \times k)/s$ denotes a $k \times k$ convolution with stride s for n filters, followed by a batch normalization and a ReLU activation function. $[\cdot]$ represents the residual unit. "FC layers" stands for the Fully Connected (FC) layers that produce the logit for classification. "Softmax" denotes the Softmax function.

The structure of our detection head									
Stage	Size	Block 1	Downsampling 1	Block 2	Downsampling 2	Block 3	Downsampling 3	Block 4	Classifier
Stage 1	1	$\begin{bmatrix} 18@(1\times1)/1 \\ 18@(3\times3)/1 \\ 36@(1\times1)/1 \end{bmatrix}$	$72@(3\times3)/2$	-	-	-	-	-	-
Stage 2	1/2			$\begin{bmatrix} 36@(1\times1)/1 \\ 36@(3\times3)/1 \\ 72@(1\times1)/1 \end{bmatrix}$	$144@(3\times3)/2$	-	-	-	-
Stage 3	1/4					$\begin{bmatrix} 72@(1\times1)/1 \\ 72@(3\times3)/1 \\ 144@(1\times1)/1 \end{bmatrix}$	$288@(3\times3)/2$	-	-
Stage 4	1/8							$\begin{bmatrix} 144@(1\times1)/1 \\ 144@(3\times3)/1 \\ 288@(1\times1)/1 \end{bmatrix}$	$128@(1\times1)/1$ FC layers Softmax

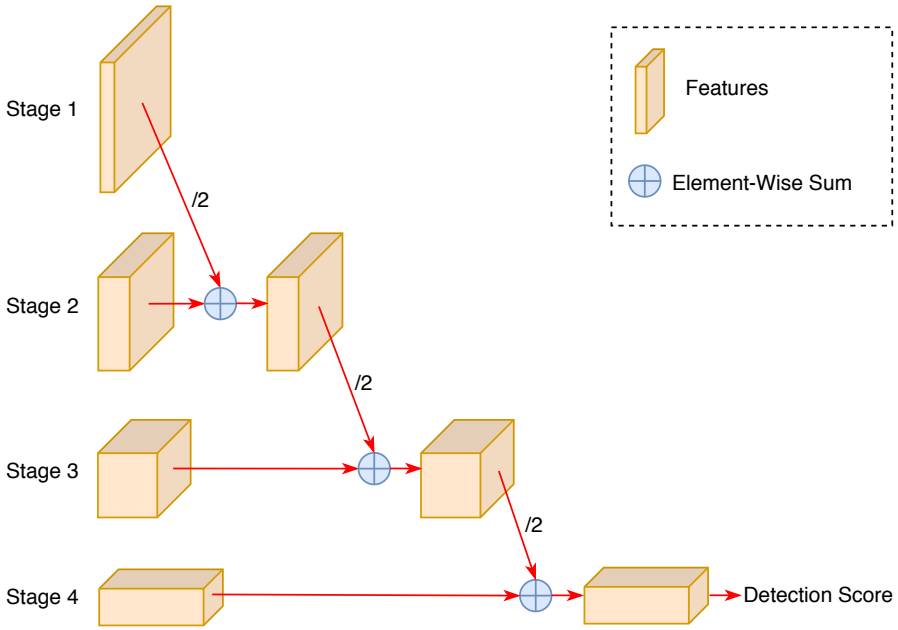


Figure 4.8: The overall architecture of our detection head.

Table 4.9: The values of learned α_s and α_c for SCCMs on four scales of our models. *Dataset* illustrates the training data for different models, where *Syn* stands for our synthetic data.

Model	Dataset	SCCM 4		SCCM 3		SCCM 2		SCCM 1	
		α_s	α_c	α_s	α_c	α_s	α_c	α_s	α_c
Pre-trained	Syn	0.6516	0.7218	0.6690	0.7962	0.9080	0.9558	0.9020	0.9523
Fine-tuned	Coverage	0.6496	0.7355	0.6710	0.8029	0.9117	0.9587	0.9056	0.9551
	CASIA	0.6404	0.6913	0.6666	0.7929	0.9010	0.9579	0.8953	0.9576
	NIST16	0.6654	0.7234	0.6782	0.8001	0.9091	0.9589	0.9033	0.9550

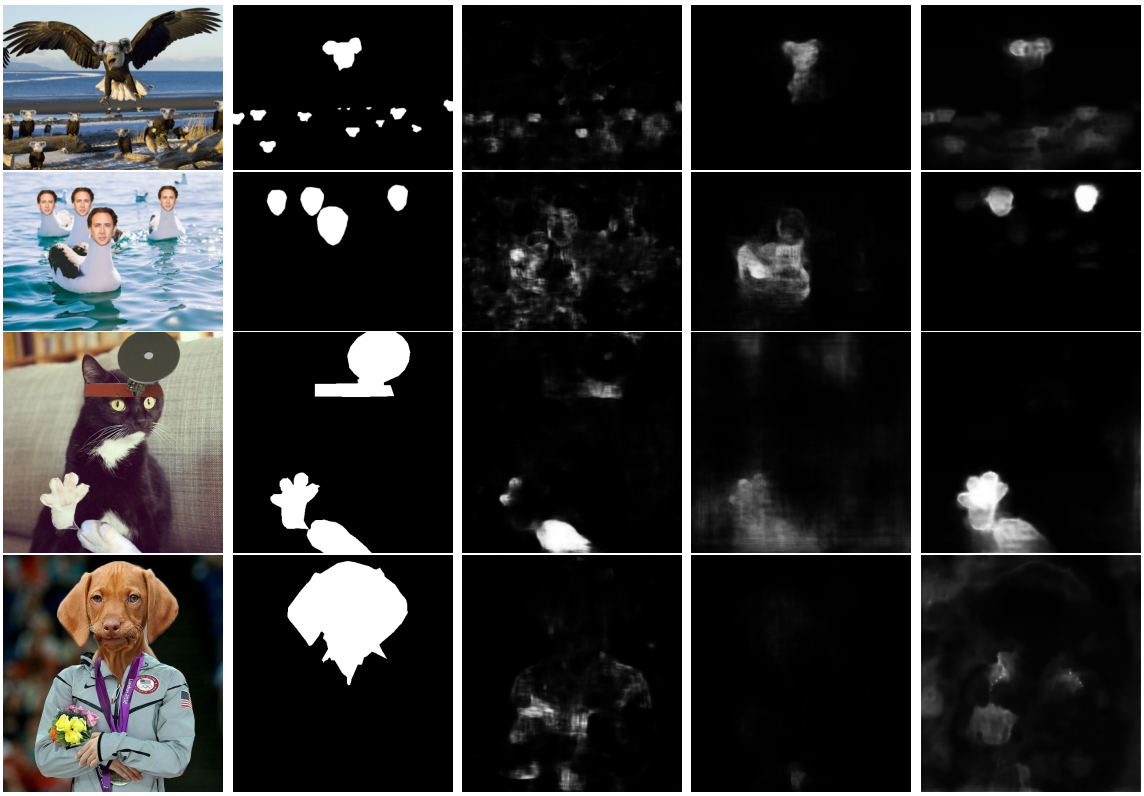
especially while dealing with real-life manipulated images. Here we demonstrate some failure cases on the real-world dataset IMD20 [67] to show limitations of our work.

In Fig. 4.9, it is clear that for real-life manipulated images, the forged regions have various sizes and shapes. In the first two rows, we show a specific case where the same pattern is copied several times but with different scales. Despite our method fails to localize all forged regions, it is less sensitive to scale variation as compared to ManTra-Net [11] and SPAN [16], owing to our tailored network design. In addition, our method may fail to localize the whole forged regions or only localize part of them in some cases (*e.g.,* see the last two rows). One possible reason is that some manipulation traces are elaborately removed by fabricators. Indeed, the compared IMDL methods also have difficulty to tackle these manipulated images. Note that even in these cases, our PSCC-Net still performs relatively better than the SOTAs [11, 16] for image manipulation localization (*e.g.,*, see the 3rd row).

4.7.3 Visualization of Synthetic Data

Since there is no standard IMDL dataset for training, we synthesize our own data to train and validate the proposed PSCC-Net, which contains four categories including 1) splicing, 2) copy-move, 3) removal, and 4) pristine images. For splicing class, in addition to synthesize the content-aware splicing manipulation, we also produce spliced images based on randomly-generated masks from [60] since the manipulated region is not necessarily an object.

In Fig. 4.10, we demonstrate several examples of content-aware splicing and random-mask based splicing. It is clear that their GT masks are quite different in terms of size and shape, thus are complement to each other. We use both splicing data to



(a) Manipulated (b) GT mask (c) ManTra-Net [11] (d) SPAN [16] (e) Ours

Figure 4.9: Failure cases. Zoom in for details.

train the PSCC-Net to achieve better generalization ability. For copy-move class, the data from [24] is adopted and we do not synthesize any data for this class. For removal class, the SOTA inpainting method [6] is leveraged. Some synthesized removal examples are shown in Fig. 4.11. As for pristine class, we simply select the original images before any manipulation, some of which are shown in the 1st row of Fig. 4.11.

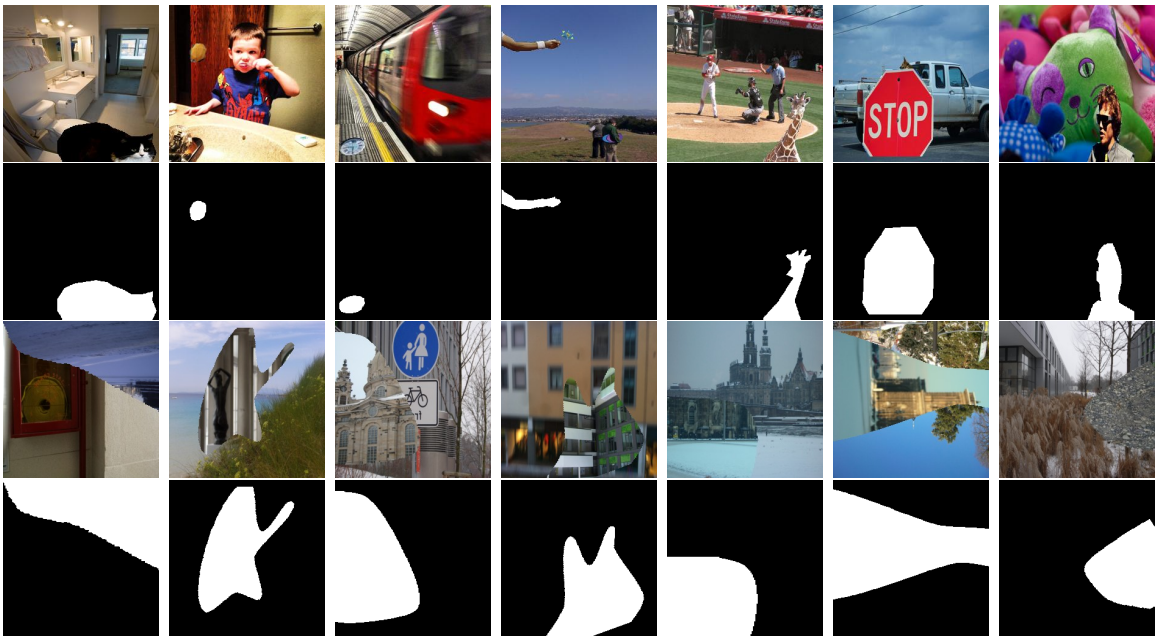


Figure 4.10: Examples of synthesized splicing data. The 1st row shows content-aware splicing, and the 3rd row shows the random-mask based splicing. We show their corresponding GT masks in the 2nd and 4th rows respectively. Zoom in for details.

4.7.4 Visualization of Predicted Manipulation Masks on Different Scales

The proposed PSCC-Net utilizes a progressive mechanism to reduce the prediction difficulty by avoiding generating the mask from the finest scale directly. Instead, the mask on the coarsest scale is first predicted to locate the regions that are potentially forged based on the current available information. The subsequent prediction on the finer scale can leverage the previous mask and pay more attention to those selected regions. This process repeatedly performs till generating the manipulation mask at the finest scale as our final prediction.

Since we only visualize one example of the coarse-to-fine mask prediction in

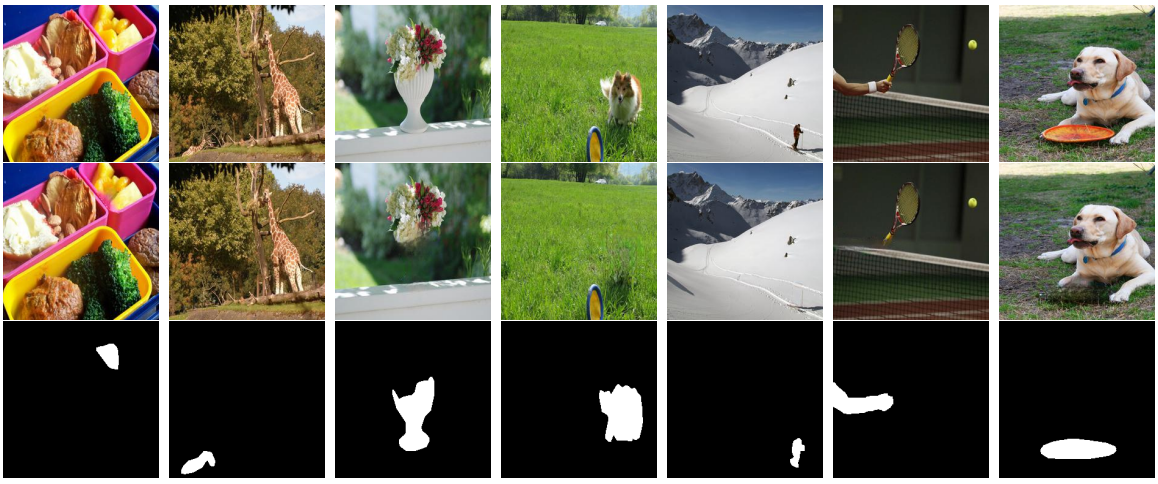
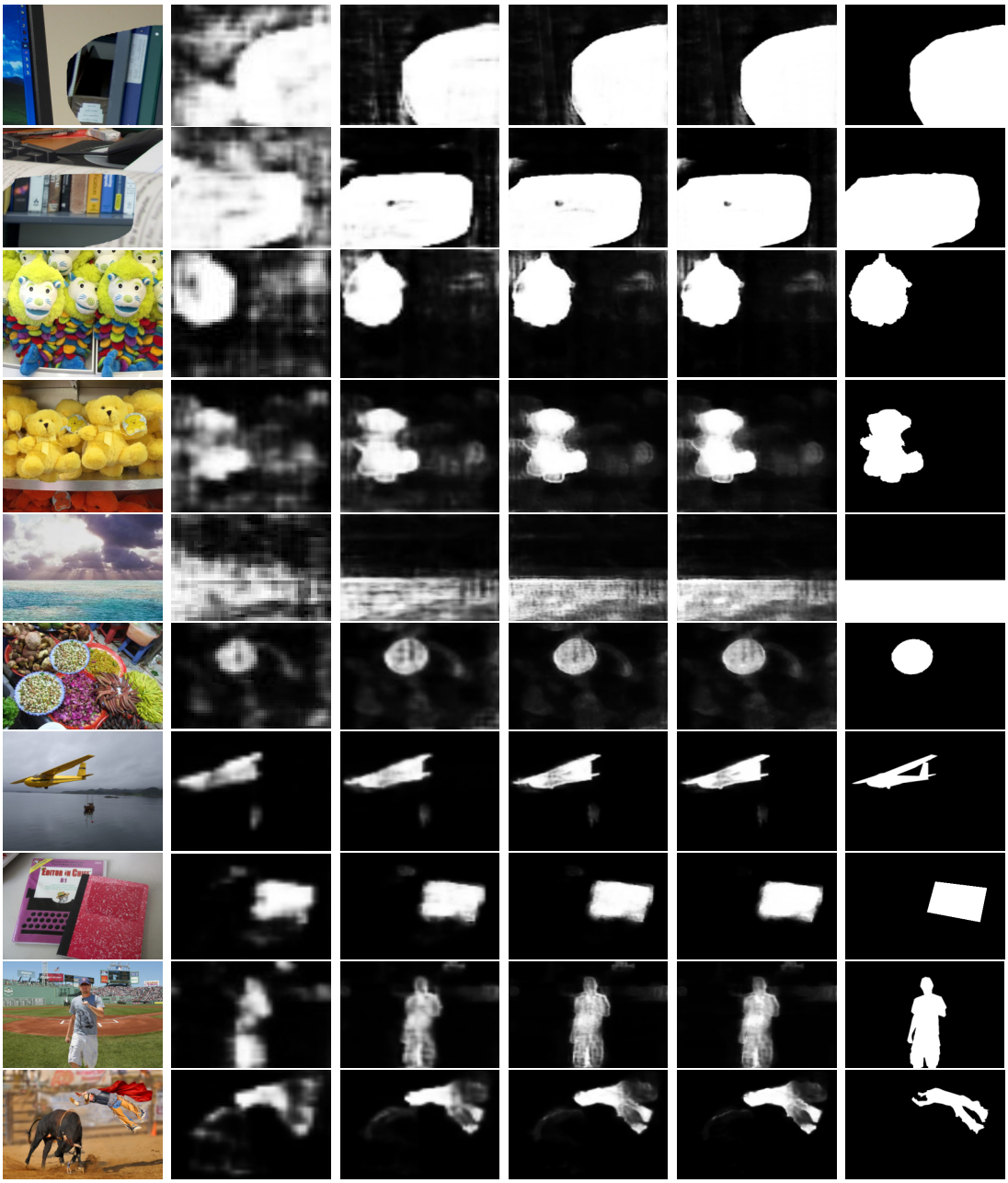


Figure 4.11: Examples of removal manipulation. The 1st row shows original images, the 2rd row shows the corresponding images after removal, and the last row shows the GT masks. Zoom in for details.

Fig. 4.2, we give more results to visualize the performance improvement of manipulation localization from the *Scale* 4 to *Scale* 1. In Fig. 4.12, it can be seen that benefiting to the proposed progressive mechanism, the localization performance is gradually improved from the *Mask* 4 to *Mask* 1 in terms of lower false alarms and clearer boundaries. Moreover, the results in Fig. 4.12 are in line with our ablation study that demonstrates the mask prediction can be terminated earlier to save runtime while the intermediate mask, *e.g.*, *Mask* 3, already performs well enough.

4.7.5 More Manipulation Localization and Detection Results.

Here, additional qualitative comparisons are performed to further evaluate the IMDL performance of PSCC-Net against the SOTAs [11, 16]. To qualitatively evaluate the manipulation localization, we compare the PSCC-Net with ManTra-Net [11] and SPAN [16] on 4 standard datasets and 1 real-world dataset. The corresponding results on Columbia, Coverage, CASIA, NIST16, and IMD20 are shown in Fig. 4.13,

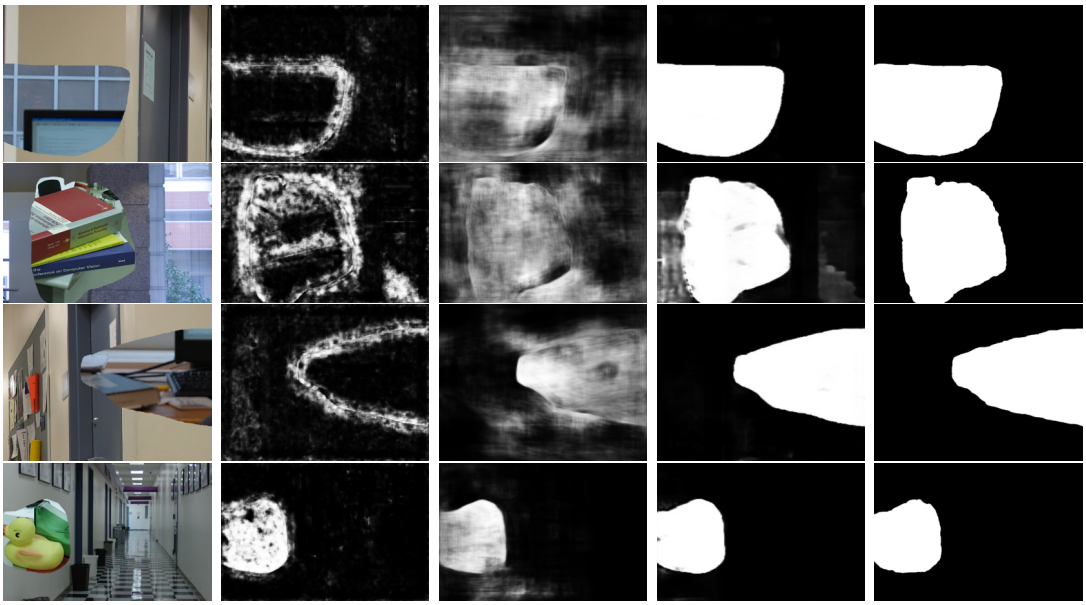


(a) Manipulated (b) Mask 4 (c) Mask 3 (d) Mask 2 (e) Mask 1 (f) GT

Figure 4.12: Visualization of predicted manipulation masks from the *Scale* 4 to *Scale* 1. We employ our pre-trained model on 4 standard test datasets and 1 real-world dataset to generate these masks. From top to bottom, manipulated images are from Columbia, Coverage, CASIA, NIST16, and IMD20, each with 2 images respectively. It is evident that the performance of manipulation localization is gradually improved from the *Mask* 4 to *Mask* 1 in terms of lower false alarms and clearer boundaries. Zoom in for details.

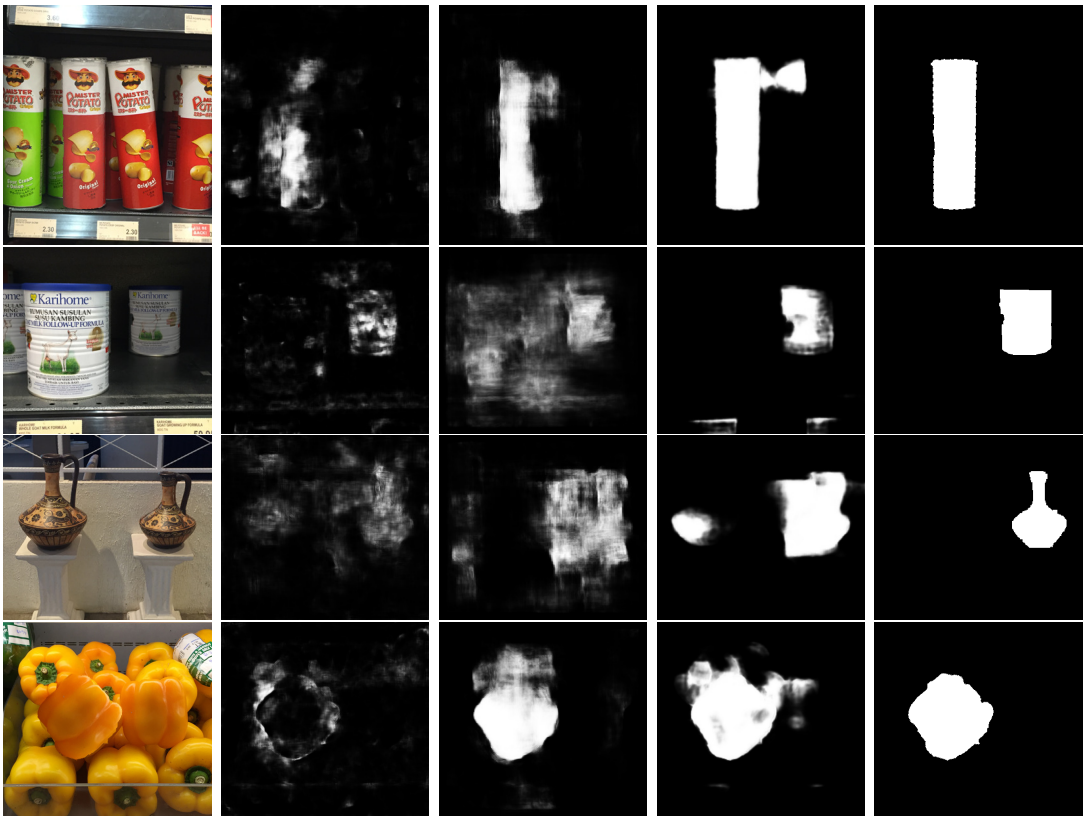
Fig. 4.14, Fig. 4.15, Fig. 4.16, and Fig. 4.17, respectively. It it clear that our PSCC-Net outperforms the compared methods by producing more accurate manipulation masks with less false alarms.

Our PSCC-Net can perform manipulation detection in two different strategies. The first one produces the detection score by simply averaging the predicted manipulation mask, and the second one resorts to the detection head to produce this score. In the paper, we justify that the detection performance can be dramatically improved by introducing a tailored head. Note that qualitatively evaluating the manipulation detection is difficult. However, since manipulation detection and localization are highly relevant tasks, we can evaluate detection by showing the corresponding predicted manipulation masks. Indeed, if the predicted mask retains blank for a pristine image and indicates the forged regions for a manipulated image, this mask (or the features to generate this mask) naturally helps the detection. Therefore, we compare our predicted manipulation masks to SPAN [16] on CASIA-D to qualitatively evaluate the detection. In Fig. 4.18, it can be seen that by leveraging both manipulated and pristine images in training, the PSCC-Net enables to accurately predict their corresponding masks, yet the SPAN [16] may cause false alarms on pristine ones since they assume the existence of manipulation in all input images. This is one of the reasons why we achieve the SOTA result for image manipulation detection.



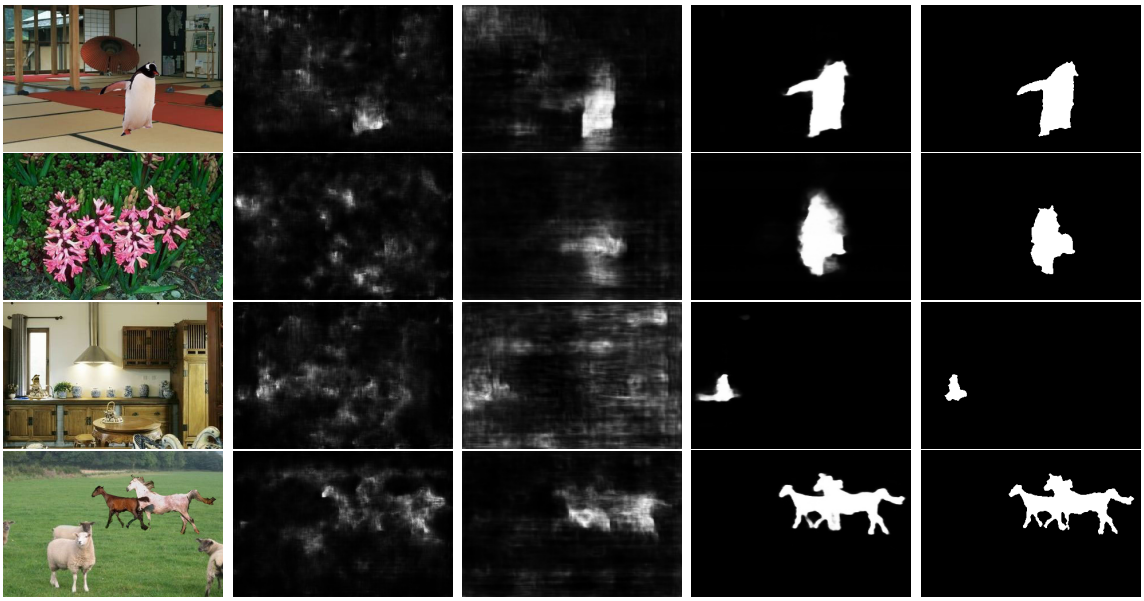
(a) Manipulated (b) ManTra-Net [11] (c) SPAN [16] (d) Ours (e) GT

Figure 4.13: Qualitative localization evaluations on Columbia.



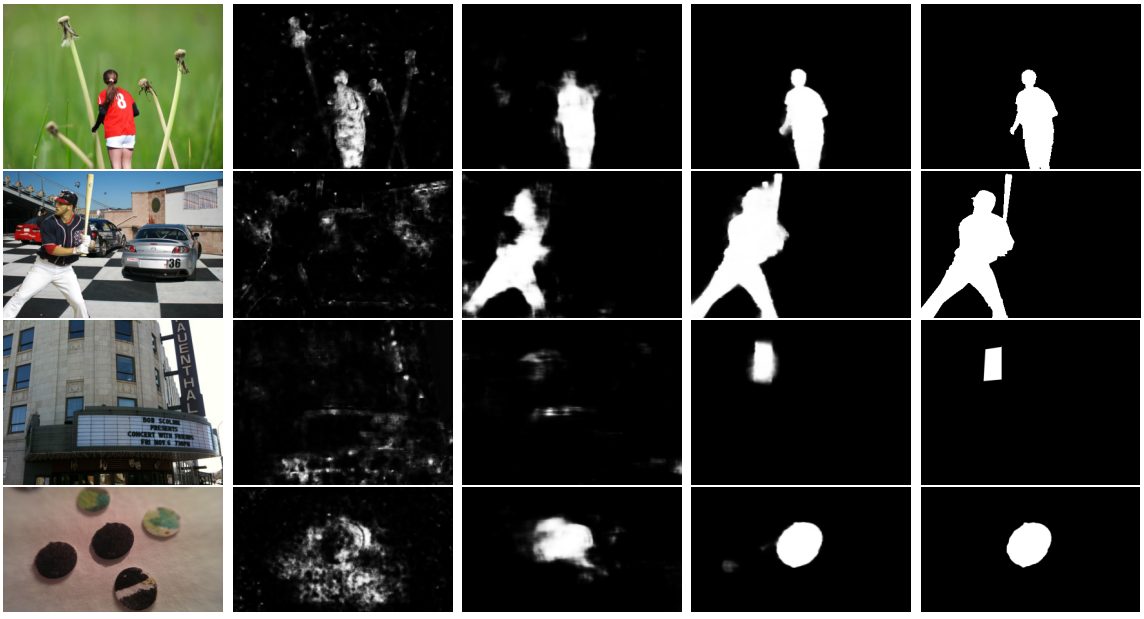
(a) Manipulated (b) ManTra-Net [11] (c) SPAN [16] (d) Ours (e) GT

Figure 4.14: Qualitative localization evaluations on Coverage.



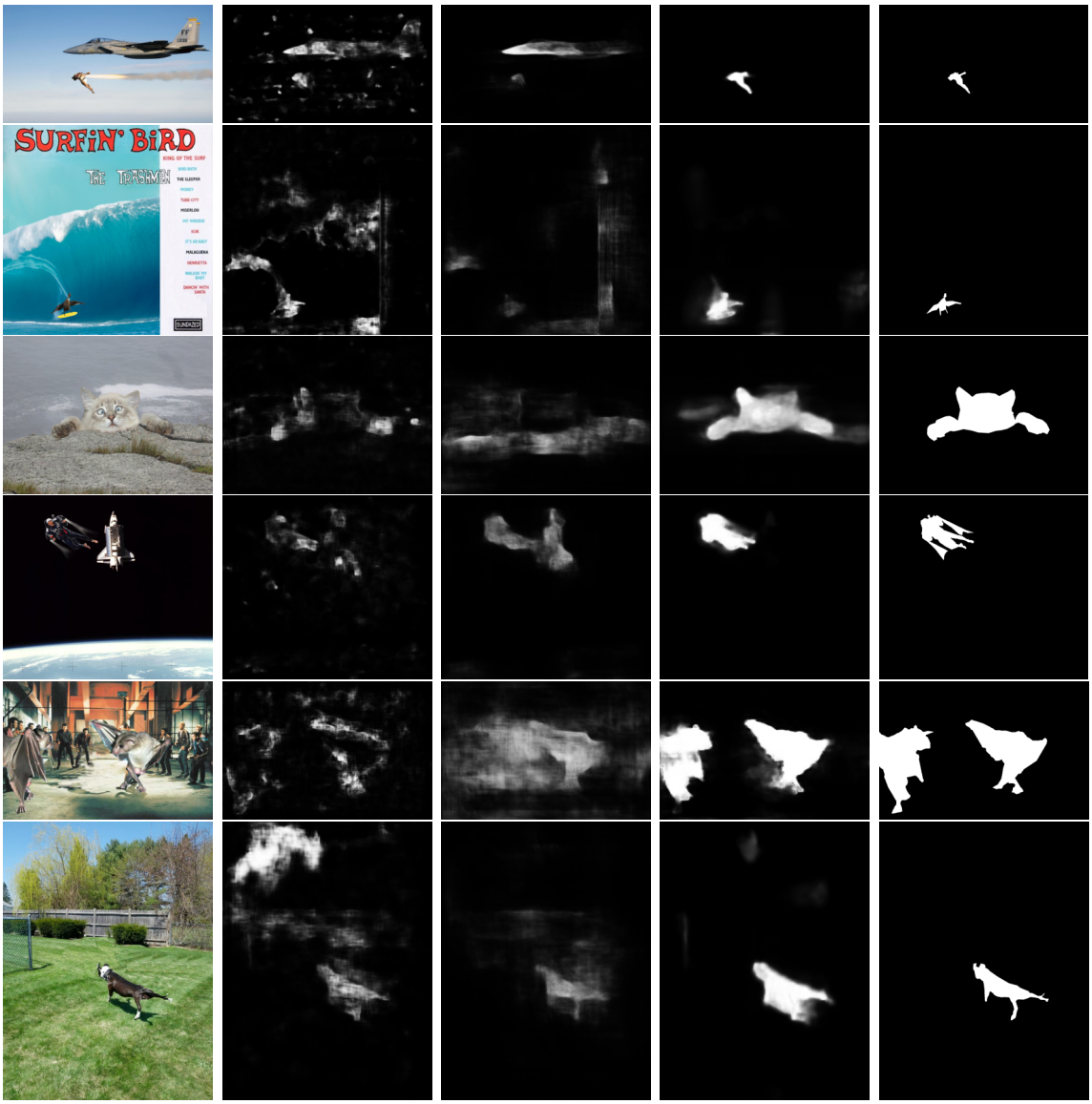
(a) Manipulated (b) ManTra-Net [11] (c) SPAN [16] (d) Ours (e) GT

Figure 4.15: Qualitative localization evaluations on CASIA.



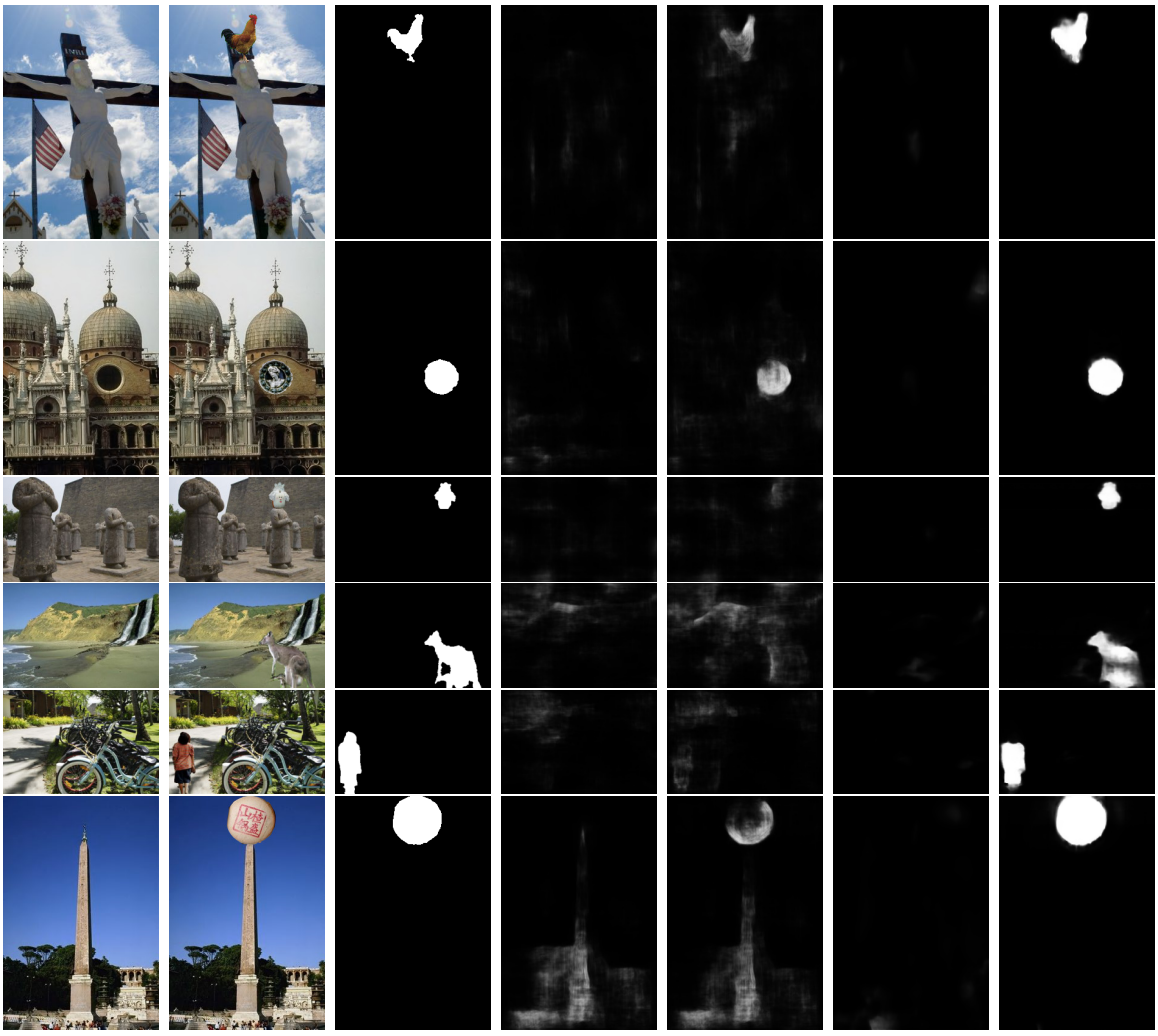
(a) Manipulated (b) ManTra-Net [11] (c) SPAN [16] (d) Ours (e) GT

Figure 4.16: Qualitative localization evaluations on NIST16.



(a) Manipulated (b) ManTra-Net [11] (c) SPAN [16] (d) Ours (e) GT

Figure 4.17: Qualitative localization evaluations on IMD20.



(a) Pristine (b) Manipulated (c) GT Manip. mask (d) Pristine mask [16] (e) Manip. mask [16] (f) Our pristine mask (g) Our manip. mask

Figure 4.18: Qualitative detection evaluations on CASIA-D. From left to right, we show pristine and manipulated images, GT manipulation masks, predicted pristine and manipulation masks from [16], our predicted pristine and manipulation masks. Since GT pristine masks are all blank, they are not shown here for clarity. Zoom in for details.

Bibliography

[1] B. Li, X. Qi, T. Lukasiewicz, and P. H. Torr, "ManiGAN: Text-guided image manipulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[2] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "MaskGAN: Towards diverse and interactive facial image manipulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] H. Dhamo, A. Farshad, I. Laina, N. Navab, G. D. Hager, F. Tombari, and C. Rupprecht, "Semantic image manipulation using scene graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[4] X. Liu, Z. Lin, J. Zhang, H. Zhao, Q. Tran, X. Wang, and H. Li, "Open-Edit: Open-domain image manipulation with open-vocabulary instructions," in *European Conference on Computer Vision (ECCV)*, 2020.

[5] J. Li, F. He, L. Zhang, B. Du, and D. Tao, "Progressive reconstruction of visual structure for image inpainting," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[6] J. Li, N. Wang, L. Zhang, B. Du, and D. Tao, "Recurrent feature reasoning for image inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[7] A. Lahiri, A. K. Jain, S. Agrawal, P. Mitra, and P. K. Biswas, "Prior guided GAN based semantic inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[8] Y. Zeng, Z. Lin, J. Yang, J. Zhang, E. Shechtman, and H. Lu, "High-resolution image inpainting with iterative confidence feedback and guided upsampling," in *European Conference on Computer Vision (ECCV)*, 2020.

[9] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of GANs for semantic face editing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[10] M. Huh, A. Liu, A. Owens, and A. A. Efros, "Fighting fake news: Image splice detection via learned self-consistency," in *European Conference on Computer Vision (ECCV)*, 2018.

[11] Y. Wu, W. AbdAlmageed, and P. Natarajan, "ManTra-Net: Manipulation tracing network for detection and localization of image forgeries with anomalous features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[12] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, 2020.

[13] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain, "On the detection of digital face manipulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[14] T. Thomson, D. Angus, and P. Dootson, "Seeing no longer means believing," *In Daily*. [Online]. Available: https://indaily.com.au/opinion/2020/11/04/seeing-should-not-mean-believing/

[15] A. Willingham, "Is that video real?" *CNN*. [Online]. Available: https://www.cnn.com/interactive/2020/10/us/manipulated-media-tech-fake-news-trnd/

[16] X. Hu, Z. Zhang, Z. Jiang, S. Chaudhuri, Z. Yang, and R. Nevatia, "SPAN: Spatial pyramid attention network for image manipulation localization," in *European Conference on Computer Vision (ECCV)*, 2020.

[17] D. Cozzolino and L. Verdoliva, "Noiseprint: a CNN-based camera model fingerprint," *IEEE Transactions on Information Forensics and Security*, 2019.

[18] C. Chen, Z. Xiong, X. Liu, and F. Wu, "Camera trace erasing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[19] J. H. Bappy, A. K. Roy-Chowdhury, J. Bunk, L. Nataraj, and B. Manjunath, "Exploiting spatial structure for localizing manipulated image regions," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[20] R. Salloum, Y. Ren, and C.-C. J. Kuo, "Image splicing localization using a multi-task fully convolutional network (MFCN)," *Journal of Visual Communication and Image Representation*, 2018.

[21] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Learning rich features for image manipulation detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] J. H. Bappy, C. Simons, L. Nataraj, B. Manjunath, and A. K. Roy-Chowdhury, "Hybrid LSTM and encoder-decoder architecture for detection of image forgeries," *IEEE Transactions on Image Processing*, 2019.

[23] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[24] Y. Wu, W. Abd-Almageed, and P. Natarajan, "BusterNet: Detecting copy-move image forgery with source/target localization," in *European Conference on Computer Vision (ECCV)*, 2018.

[25] A. Islam, C. Long, A. Basharat, and A. Hoogs, "DOA-GAN: Dual-order attentive generative adversarial network for image copy-move forgery detection and localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[26] S. Lyu, X. Pan, and X. Zhang, "Exposing region splicing forgeries with blind local noise estimation," *International Journal of Computer Vision*, 2014.

[27] D. Cozzolino, G. Poggi, and L. Verdoliva, "Splicebuster: A new blind image splicing detector," in *International Workshop on Information Forensics and Security (WIFS)*, 2015.

[28] I. Amerini, T. Uricchio, L. Ballan, and R. Caldelli, "Localization of JPEG double compression through multi-domain convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[29] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "Tampering detection and localization through clustering of camera-based CNN features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[30] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Deep matching and validation network: An end-to-end solution to constrained image splicing localization and detection," in *ACM International Conference on Multimedia (ACMMM)*, 2017.

[31] V. V. Kniaz, V. Knyaz, and F. Remondino, "The point where reality meets fantasy: Mixed adversarial generators for image splice detection," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[32] D. Cozzolino, G. Poggi, and L. Verdoliva, "Efficient dense-field copy–move forgery detection," *IEEE Transactions on Information Forensics and Security*, 2015.

[33] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler, "COVERAGE–A novel database for copy-move forgery detection," in *IEEE International Conference on Image Processing (ICIP)*, 2016.

[34] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Image copy-move forgery detection via an end-to-end deep neural network," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[35] X. Zhu, Y. Qian, X. Zhao, B. Sun, and Y. Sun, "A deep learning approach to patch-based image inpainting forensics," *Signal Processing: Image Communication*, 2018.

[36] R. M. Joseph and A. Chithra, "Literature survey on image manipulation detection," *International Research Journal of Engineering and Technology (IRJET)*, 2015.

[37] Y. Liu, J. Stehouwer, and X. Liu, "On disentangling spoof traces for generic face anti-spoofing," in *European Conference on Computer Vision (ECCV)*, 2020.

[38] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, 2012.

[39] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[40] W. Ren, L. Ma, J. Zhang, J. Pan, X. Cao, W. Liu, and M.-H. Yang, "Gated fusion network for single image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[41] K. Jiang, Z. Wang, P. Yi, C. Chen, B. Huang, Y. Luo, J. Ma, and J. Jiang, "Multi-scale progressive fusion network for single image deraining," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[42] P. Yi, Z. Wang, K. Jiang, J. Jiang, and J. Ma, "Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[43] Y. Chen, Y. Tai, X. Liu, C. Shen, and J. Yang, "FSRNet: End-to-end learning face super-resolution with facial priors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[44] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang, "Progressive attention guided recurrent network for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[45] J. Zhu, D. Li, T. Han, L. Tian, and Y. Shan, "ProgressFace: Scale-aware progressive learning for face detection," in *European Conference on Computer Vision (ECCV)*, 2020.

[46] X. Song, K. Zhao, W.-S. C. H. Zhang, and J. Guo, "Progressive refinement network for occluded pedestrian detection," in *European Conference on Computer Vision (ECCV)*, 2020.

[47] G. Brazil and X. Liu, "Pedestrian detection with autoregressive network phases," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[49] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[50] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[51] X. Liu, Y. Ma, Z. Shi, and J. Chen, "GridDehazeNet: Attention-based multi-scale network for image dehazing," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[52] T. Isobe, S. Li, X. Jia, S. Yuan, G. Slabaugh, C. Xu, Y.-L. Li, S. Wang, and Q. Tian, "Video super-resolution with temporal group attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[53] S. Gong, X. Liu, and A. Jain, "Mitigating face recognition bias via group adaptive classifier," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[54] J. Park, S. Woo, J.-Y. Lee, and I. S. Kweon, "BAM: Bottleneck attention module," in *British Machine Vision Conference (BMVC)*, 2018.

[55] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "CBAM: Convolutional block attention module," in *European Conference on Computer Vision (ECCV)*, 2018.

[56] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[58] Y. Liu, X. Zhu, X. Zhao, and Y. Cao, "Adversarial learning for constrained image splicing detection and localization based on atrous convolution," *IEEE Transactions on Information Forensics and Security*, 2019.

[59] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[60] M. E. Mortenson, *Mathematics for computer graphics applications*. Industrial Press Inc., 1999.

[61] I. S. P. Society, "Camera model identification," https://www.kaggle.com/c/sp-society-camera-model-identification.

[62] D. Shullani, M. Fontani, M. Iuliani, O. Al Shaya, and A. Piva, "VISION: a video and image dataset for source identification," *EURASIP Journal on Information Security*, 2017.

[63] T. Gloe and R. Böhme, "The 'dresden image database' for benchmarking digital image forensics," in *ACM Symposium on Applied Computing*, 2010.

[64] T.-T. Ng, J. Hsu, and S.-F. Chang, "Columbia image splicing detection evaluation dataset," *DVMM lab. Columbia Univ CalPhotos Digit Libr*, 2009.

[65] J. Dong, W. Wang, and T. Tan, "Casia image tampering detection evaluation database," in *China Summit and International Conference on Signal and Information Processing*, 2013.

[66] "NIST: Nist nimble 2016 datasets," https://www.nist.gov/itl/iad/mig/, 2016.

[67] A. Novozamsky, B. Mahdian, and S. Saic, "IMD2020: A large-scale annotated dataset tailored for detecting manipulated images," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2020.

[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[69] N. Krawetz and H. F. Solutions, "A pictures worth," *Hacker Factor Solutions*, 2007.

[70] B. Mahdian and S. Saic, "Using noise inconsistencies for blind image forensics," *Image and Vision Computing*, 2009.

[71] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, "Image forgery localization via fine-grained analysis of CFA artifacts," *IEEE Transactions on Information Forensics and Security*, 2012.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Deep learning based visual content processing and analysis has been researched in this thesis, especially in the fields of video super-resolution and image manipulation detection and localization. Three algorithms have been proposed in this thesis to address some common issues in these fields.

More specifically, in the first work, a new approach to VSR using local dynamic filters via locally connected (LC) layers for implicit motion compensation is proposed, named LCVSR. The effectiveness of this new approach can be attributed to three major factors: 1) The overall system is end-to-end trainable and does not require any pre-training; the accuracy of motion estimation improves progressively through the training process; 2) Local motion estimation and compensation is performed implicitly by a novel Dynamic Local Filter Network (DLFN) with LC layers. There are at least two benefits of using DLFN. Firstly, the implicit motion estimation, realized

by sample-specific and position-specific dynamic local filters generated on-the-fly according to the target pixels, can deal with complicated local transformations in video frames such as regional blurring, irregular local movement and photometric changes. Secondly, the simultaneous action of dynamic local filters on all input LR frames via LC layers helps to maintain the temporal consistency; 3) The spatial consistency of super-resolved outputs is enforced by a novel Global Refinement Network (GRN) constructed using ResBlock and autoencoder structures. Since the implicit motion estimation performed by the DLFN is spatially localized, it may cause inconsistencies across neighboring areas. As such, the GRN plays a critical role of restoring the spatial consistency. Moreover, the GRN has the capability of exploiting non-local correlations due to its constituent autoencoder structure, which makes up for the lack of global motion estimation in DLFN. Extensive experiments validate the strength of this approach in terms of local transformation handling, temporal consistency as well as edge sharpness.

In the second work, to avoid information loss and better fit the imaging pipeline, a new Raw VSR method, named RawVSR, is proposed that can directly exploit camera sensor data. The RawVSR has a dual-branch structure that disentangles the super-resolution process from the color correction process, where weight sharing is adopted to reduce the model size and to ensure the latent consistency of intermediate features in the two branches. As a result, it has the desirable property of being device-independent. In RawVSR, the Successive Deep Inference (SDI) module is designed according to the architectural principle obtained via the analysis of a suitable probabilistic graphical model. This design philosophy is likely applicable to a wide

range of problems, especially those involving data fusion. Moreover, an Attention-based Residual Dense Block (ARDB) is elaborately designed to simultaneously refine the fused feature and generate a spatial-specific color transformation for favorable color correction. Since there is no such raw video dataset released in public, the first Raw Video Dataset (RawVD) is laboriously built to train, validate, and test the RawVSR. It is also expected that the collected RawVD can potentially benefit other video tasks such as video interpolation, video enhancement, etc. The experimental results validates the superiority of employing camera raw data for training, as well as the effectiveness of overall network design.

In the third work, a generic Image Manipulation Detection and Localization (IMDL) method, named Progressive Spatio-Channel Correlation Network (PSCC-Net), is built to tackle multiple manipulations types. In summary, there are three major issues in IMDL task. The first one is the scale variation, in which most prior works neglect its importance, thus encounter difficulty when detecting forged areas of different sizes. To address this issue, a backbone network with dense connections is adopted as the feature extractor to effectively exchange information among different scales. The second one is the neglect of image correlation. Since manipulated regions can best be determined while comparing to pristine regions, a naive learning of mapping from the manipulated image to manipulation mask may lead to an overfitting to the specific attack type in training. In contrast, considering the image spatial correlation can lead to a more generalized localization solution. Based on this insight, a novel Spatio-Channel Correlation Module (SCCM) is proposed to capture the spatial and channel-wise correlations for better generalization, which also avoids the use of massive annotated data to pre-train the feature extractor. The third one is the neglect of

manipulation detection. Since most prior works assume the existence of manipulation in all input images, they might cause many false alarms on pristine images and make the detection unreliable. Instead, a detection head is designed to distinguish the pristine image from the manipulated image before localizing any potential manipulations, which is in line with practical applications. Extensive experiments demonstrate that the PSCC-Net outperforms the SOTA methods on both detection and localization. More importantly, owing to the light-weighted design, it can be potentially deployed online to defend the propagation of deceitful manipulation contents on social media.

5.2 Future Work

There are still some future works that can be conducted to further improve the effectiveness and efficiency of the proposed three methods (*i.e.*, LCVSR, RawVSR, and PSCC-Net).

In LCVSR and RawVSR, to train these two methods, the LR-HR pairs are collected for supervised learning, where the LR frames are produced by downsampling the HR counterparts with several degradations (*e.g.*, add blur effect or noises). However, this generation process is just an *approximation* of the relation between LR and HR frames in real scenarios, in which blur effect and camera imaging noises might be too complicated to be accurately modelled by a simple Gaussian or Laplacian function. In other words, domain discrepancy exists between synthesized LR frames and real-world ones. To address this issue, one possible solution is to leverage the GAN technique, where LR and HR frames are unpaired and directly collected from the real world for training. As such, the domain discrepancy between network training and testing is alleviated.

Although the number of parameters of these two methods (*i.e.*, 5.81M for LCVSR and 4.46M for RawVSR) is moderate as compared to others, it is still difficult to deploy them on edge devices (*e.g.*, phone and surveillance system). Therefore, model compression techniques such as pruning, quantization, and knowledge distillation can be leveraged to further decrease their complexity.

In PSCC-Net, three typical manipulations (*i.e.*, splicing, copy-move, and removal) can be effectively detected and localized in image. However, there are two additional manipulation types that are necessarily considered in the future. One is entire synthesis, where the image is fully generated by image synthesis methods. The other is image enhancement, where the image is enhanced to be more appealing in vision without modifying any content. Different from three considered manipulations, entire synthesis and image enhancement tamper with the image holistically, rendering the clue of exploring image correlation invalid. Therefore, it is necessary to discover new clues to cope with them.

Moreover, the uncertainty of predicted manipulation masks is still a concern. There is a great need for developing a method that can evaluate the accuracy of these masks.