SAIN: SIMILARITY-AWARE VIDEO FRAME INTERPOLATION

Yue Lv[1] Wenming Yang[1] Wangmeng Zuo[2] Qingmin Liao[1] Rui Zhu[3]

[1]Shenzhen International Graduate School,
Tsinghua University, Shenzhen 518055, China
[2]School of Computer Science and Technology,
Harbin Institute of Technology, Harbin 150001, China
[3]Faculty of Actuarial Science and Insurance,
City, University of London, London EC1Y 8TZ, UK

ABSTRACT

Video frame interpolation (VFI) aims to synthesize an intermediate frame between two consecutive original frames. Most existing methods simply linearly combine the warped frames, leading to a loss of image texture. Since moving objects usually have similarities in consecutive frames, we propose a similarity-aware video frame interpolation method (SAIN) that searches patches with similar texture in the embedding space from input frames to extract features and capture image details. To gather the frame details and restore image texture, SAIN incorporates an implicit neural representation learning from similar patches to enrich image details and refine outputs in frame synthesis networks. Experiments demonstrate that SAIN preserves image texture and enhances interpolated image quality significantly.

Index Terms— VFI, Implicit Neural Representation, Similar Patches Aggregation, Restore Image Texture

1. INTRODUCTION

Generally, low frame rate video displays fewer images per second and requires less storage space and bandwidth, but it is more likely to cause aliasing and yields abrupt motion artifacts. On the contrary, high frame rate videos effectively alleviate visual artifacts such as flicker, motion blur, and discontinuousness and provide a better visual experience. Due to the increase in temporal resolution by synthesizing intermediate frames between existing consecutive frames, video frame interpolation (VFI) has shown enormous potential in visual quality enhancement. Unfortunately, it is hard to accurately estimate intermediate frames because of diverse factors, such as occlusions, tiny objects with large or nonlinear motions, and lighting conditions.

Flow-based methods utilize optical flow to warp the input frame to obtain the intermediate frame. These models can be linear or nonlinear according to whether considering the object's acceleration [16]. One common approach for flow-based methods is to estimate the optical flow $F_{0\to1}$ and $F_{1\to0}$ between two input frames I_0 and I_1 and then calculate $F_{t\to0}$ and $F_{t\to1}$ from the perspective of the synthesizing target frame I_t. The interpolation result can then be obtained by backward warping I_0 and I_1 according to $F_{t\to0}$ and $F_{t\to1}$. Recently, Huang et al. [6] proposed a method that directly generates intermediate optical flows to improve efficiency and achieves real-time speed.

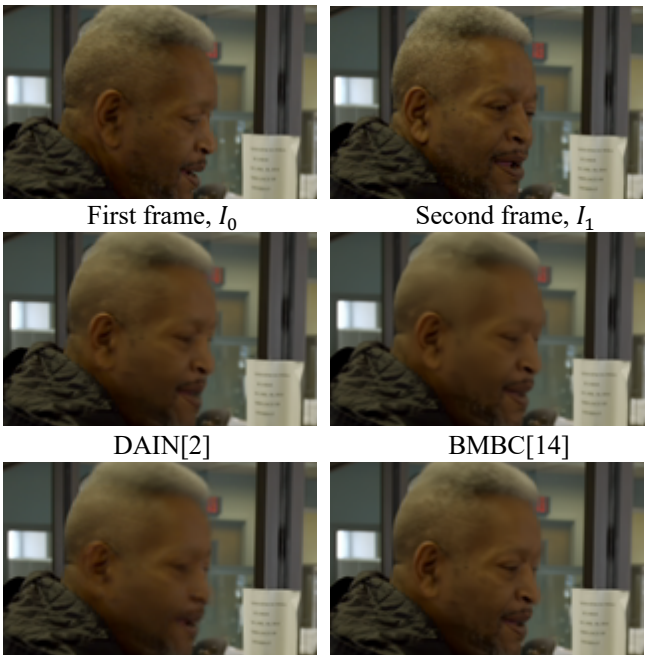


Fig. 1. A challenging example of VFI. Compared to the existing frame interpolation methods, our approach can restore more image texture details on the man's hair and face and produce a high-quality result.

Traditional flow-based video interpolation methods are incapable of capturing large motions between frames and prone to average or linearly combined warped frames, leading to a loss of image texture and blur artifacts at motion boundary. To improve the quality of the VFI result, we note that self-similarity is a vital property of video frames, which can be utilized to capture the detailed texture from consecu-

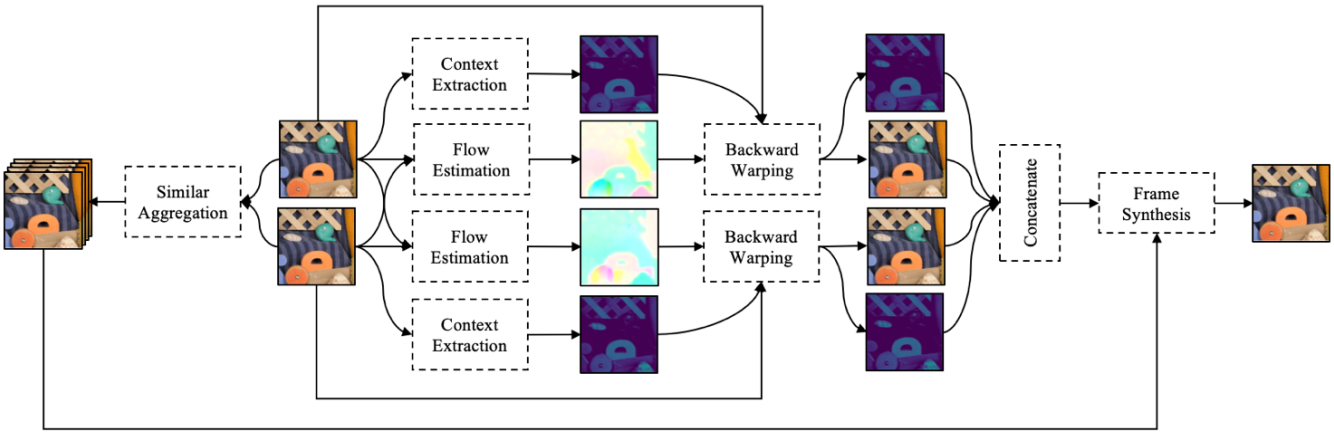


Fig. 2. The proposed SAIN diagram. Given two consecutive input frames I_0 and I_1, SAIN integrates IFNet [6] to estimate the intermediate flow $F_{t\to0}$ and $F_{t\to1}$ directly and extracts pyramid contextual features from raw inputs separately. Then it warps the input frames and corresponding context feature maps. Additionally, we find k similar patches and aggregate them together to assist the fusion and refine step. Finally, the warped input frames and feature pyramids are fed into the frame synthesis network to generate the interpolation result $\hat{I}_t$.

tive frames, in order to produce more precise intermediate frames. To be more specific, we search similar patches from input frames to extract features, capture similar image details, and integrate the implicit function to generate intermediate frames.

To sum up, SAIN exploits self-similarity in the embedding space to extract features, employs a frame synthesis network to enrich image texture, and fine-tune the interpolation results based on an implicit neural representation. Different from other traditional methods, SAIN establishes ties between the same object at different areas in two video frames to enrich the image local texture. The visual comparison shows that SAIN can restore image texture and enhance interpolated image quality significantly.

2. METHODOLOGY

In this section, we first provide an algorithm overview of our frame interpolation algorithm, SAIN. Then we elaborate on the similar patch aggregation method that can be utilized to capture the detailed features from consecutive frames. Next, we describe the frame synthesis network, which incorporates the implicit neural representation to generate interpolated frames as a critical component to refine output and generate a high-quality result. Finally, we provide the implementation details of the proposed model.

2.1. Overview

Given two input frames I_0 and I_1, our goal is to synthesize an intermediate frame I_t at time t. Traditional frame interpolation methods commonly warp the input frames and average or linearly combine two warped frames, which usually causes smoothness in local image details. Considering that objects in consecutive frames may have the same image texture, we aggregate similar patches of the frames, extract the features, and then generate interpolated frames to enrich the detailed information. We adopt IFNet [6] to estimate the intermediate optical flow directly while extracting contextual features and aggregating similar patches from raw inputs. Subsequently, we apply the backward warping to sample the input frames and contextual features, which are then concatenated and fed into the frame synthesis network with the implicit neural representation to generate the interpolated frames. The overall pipeline of SAIN is shown in Fig. 2.

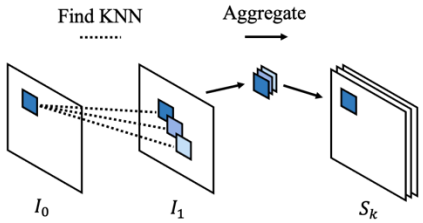


Fig. 3. An illustration of the similar patches aggregation. For each patch of one input frame, we search k similar patches from another frame and aggregate them together to and generate similar pictures S_k.

2.2. Similar patches aggregation

Synthesize intermediate frames from the warped frames directly will cause blur and a lack of image texture. Since moving objects usually have similarities in consecutive frames, aggregating image texture from adjacent frames to enrich frame details is an instinctive idea. Inspired by Internal Graph Neural Network (IGNN) [18], we consider self-similarity as an important property to assist VFI. Specifically, raw images are cropped to a specific size to integrate similar patches. For each patch of one input frame, we search k similar patches from the other frame and aggregate them together for mining available valuable details.

To speed up the calculation, we adopt the PyINN library implemented by CuPy to convert each pixel block into column vectors. The embedding features are extracted by the pre-trained first three layers of VGG19 [19]. The k similar patches are obtained based on the Euclidean distance calculated in the embedding space:

$$\mathbf{y}^* = \arg\min \mathrm{d}(\mathrm{ENC}(I_0(\mathbf{x})), \mathrm{ENC}(I_1(\mathbf{y}))), \quad (1)$$

where $\boldsymbol{x}$ is the coordinate of I_0 and $\boldsymbol{y}$ is that of I_1, ENC encodes patches to the embedding space, and d is the Euclidean distance between embedding features. Finally, the closest k pixel blocks within a given window size are aggregated as S_k for each pixel block:

$$S_k(x) = \{I_1(\mathbf{y}^*)\}. \quad (2)$$

2.3. The structure of the frame synthesis network

The implicit neural representation takes an image coordinate and the 2D deep features as inputs to predict the RGB value at a given coordinate as an output. To utilize the implicit neural representation in our frame synthesis network, we first generate a coarse interpolation frame, then concatenate the input frames, backward warping frames, and coarse interpolation output, and then feed them into an encoder to generate a feature map. After that, the extracted features are fed into the implicit neural representation to predict the RGB value according to the given coordinates. The network for generating coarse immediate frames consists of four transpose convolution layers, which adopts sigmoid as an activation function. After obtaining the coarse results, we input them into an encoder consisting of convolution layers and eight residual blocks. Finally, the RGB value is queried from the extracted features by implicit function, consisting of a 5-layer MLP, with an activation function of ReLU and 256 hidden dimensions. The detailed architecture of the frame synthesis network is illustrated in Fig. 4.

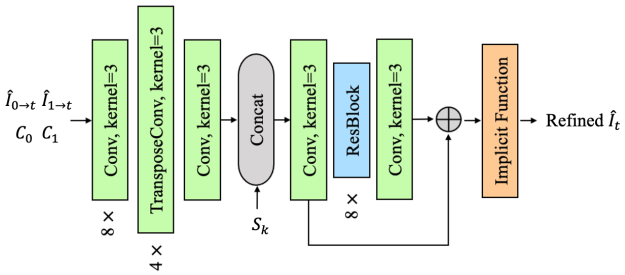


Fig. 4. The structure of our frame synthesis network.

2.4. Implementation details

The loss function to train the neural network consists of the TripLoss to suppress the residual artifacts, the classic reconstruction loss [20], and the leakage loss from the IFNet [6].

TripLoss The target frame retains the residual of the moving object in I_0 and I_1, and the position of the moving object in I_t has an unclear boundary in many traditional methods. We consider that the result may retain a part of the original image I_0 and I_1. We propose the TripLoss to stimulate what needs to be retained in the frames and suppress unnecessary residuals:

$$\mathcal{L}_{tri} = \max\left(2 \cdot d\left(\hat{I}_t, I_t\right) - d\left(\hat{I}_t, I_0\right) - d\left(\hat{I}_t, I_1\right) + \text{margin}, 0\right), \quad (3)$$

Reconstruction Loss [20] The reconstruction loss function is commonly used in video interpolation to model the reconstruction quality:

$$\mathcal{L}_{\text{rec}} = \sum_{\mathbf{x}} \rho\left(\hat{I}_t(\mathbf{x}) - I_t^{GT}(\mathbf{x})\right), \quad (4)$$

where $\rho(x) = \sqrt{x^2 + \epsilon^2}$ is the Charbonnier penalty function. We set the constant ϵ to 10^{-6}.

Leakage Loss [6] First, we use pre-trained LiteFlownet [7] to generate immediate flow $F_{t\to 0}^{Leak}$ and $F_{t\to 1}^{Leak}$as labels. Then we train the distilled optical estimation networks by shortening the L1 distance between the estimated and labels.

$$\mathcal{L}_{\text{dis}} = \left\|\mathbf{F}_{\text{t}\to 0} - \mathbf{F}_{\text{t}\to 0}^{\text{Leak}}\right\|_1 + \left\|\mathbf{F}_{\text{t}\to 1} - \mathbf{F}_{\text{t}\to 1}^{\text{Leak}}\right\|_1. \quad (5)$$

The total loss to train the neural network is the weighted sum of the three losses:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_1 \mathcal{L}_{\text{tri}} + \lambda_2 \mathcal{L}_{\text{dis}}, \quad (6)$$

where we set $\lambda_1 = 0.01$ and $\lambda_2 = 0.01$.

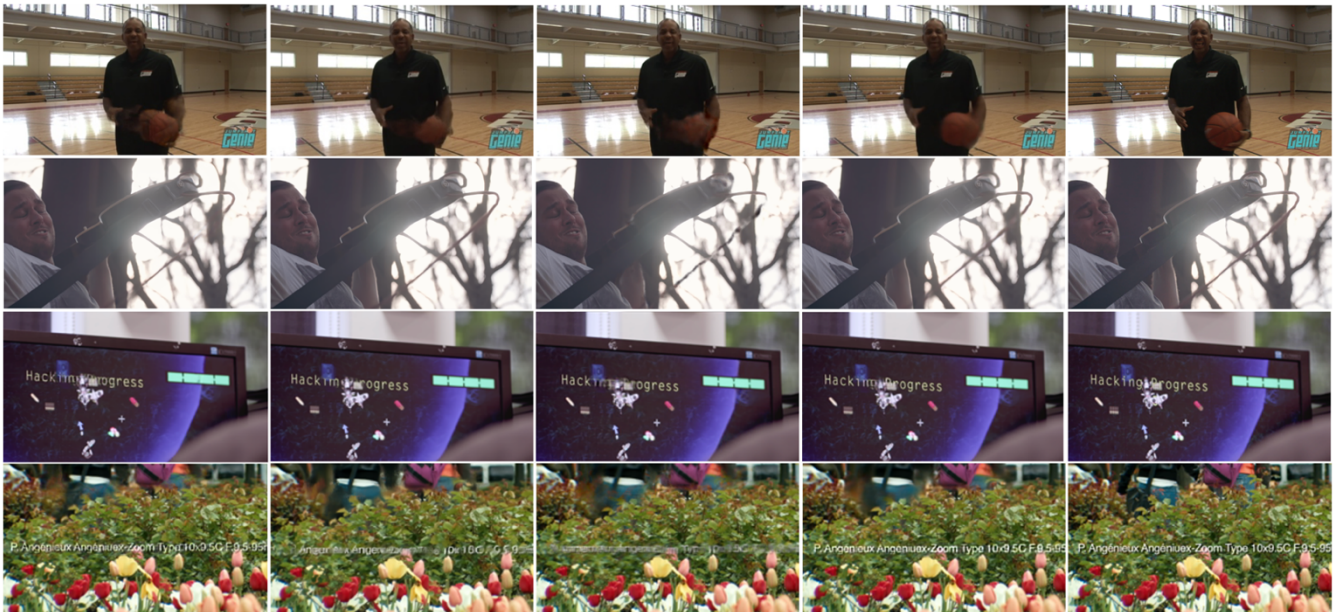
Training Dataset Vimeo-90K dataset [17] is a large-scale and high-quality video dataset provided by the Massachusetts Institute of Technology. It contains 89,800 video clips in different scenarios with various actions from vimeo.com, which can be used for video interpolation, denoising, deblocking, and super-resolution tasks. There are 51313 triples in the training set and 3782 triples in the validation set. The image resolution in this dataset is 448 × 256. Besides, we adopt LiteFlownet [7] to generate immediate flow as labels to train the distilled optical estimation networks.

Training Strategy We implement SAIN in PyTorch, iteratively train it on the Vimeo90K dataset for 200 epochs, and set weight decay to 10^{-3}. For data augmentation, frames are randomly cropped, flipped horizontally or vertically, and inverted by the temporal order. We use a batch size of 18 and train our model on two Nvidia 2080Ti GPUs. It takes about eight days to converge.

3. EXPERIMENTS

3.1. Comparison with state-of-the-arts

We compare SAIN with several state-of-the-art VFI methods, including DAIN [2], BMBC [14], and RIFE [6]. The evaluation results in terms of PSNR and SSIM are given in Table 1. It is obvious that our method is superior to DAIN, BMBC, and RIFE on both measurements. The visual comparison of SAIN and state-of-the-art methods is shown in Fig. 5. SAIN can align the content well and show excellent performance for objects with large motions. It is also clear that the results of SAIN are free of flicking of subtitles.



DAIN[2] BMBC[14] RIFE[6] SAIN (Ours) GT

Fig. 5. A visual comparison of SAIN and state-of-the-art methods with zoomed-in images for better visualization.

Table 1. Quantitative comparison (PSNR (dB) and SSIM) on Vimeo90K. The best results are shown in red.

Metrics	DAIN[2]	BMBC[14]	RIFE[6]	SAIN (Ours)
PSNR	34.64	35.01	35.35	35.74
SSIM	0.9753	0.976	0.9772	0.9787

3.2. Ablation study

Here we aim to demonstrate the effectiveness of utilizing the following three important parts in the frame synthesis network: synthesizing with the coarse results (R), incorporating the implicit neural representation (INF), and synthesizing with aggregated similar patches (SA). The results of using none of them in the network, only INF, R+INF, and R+SA+INF are shown in Table 2.

Table 2. The analysis of the frame synthesis network structure. The best results are shown in red, and the second-best ones are shown in blue with underlines.

Method	Vimeo90K[17]		UCF101[15]		M.B.[1]
	PSNR	SSIM	PSNR	SSIM	IE
None	35.35	0.9772	35.17	0.9688	2.0813
INF	35.20	0.9752	34.41	0.9612	2.0278
R+INF	35.58	0.9784	35.17	0.9689	2.0864
R+SA+INF	35.74	0.9787	35.20	0.9691	2.0835

Without generating coarse results, the model does not perform well on the Vimeo90K and UCF101 datasets. Incorporating implicit neural representation in the frame synthesis network facilitates the performance of SAIN. It is also obvious that aggregating similar patches contribute to a substantial improvement on the Vimeo90K dataset.

(a) original frame (b) w/o TripLoss (c) w/ TripLoss (d) GT

Fig. 6. A visual comparison to demonstrate the effectiveness of the TripLoss.

4. CONCLUSION

In this work, we propose SAIN, a VFI method searching similar patches from input frames to capture image details. We take self-similarity property into consideration and aggregate image texture from adjacent frames. In the future, one possible direction is to achieve time-space video super-resolution. SAIN employs an implicit neural representation, which takes an image coordinate and the 2D deep features as inputs to predict the RGB value at a given coordinate as output and can generate any higher resolution image easily.

5. ACKNOWLEDGEMENTS

This work was partly supported by the Natural Science Foundation of China (No.62171251), the Natural Science Foundation of Guangdong Province (No.2020A1515010711), the Special Foundation for the Development of Strategic Emerging Industries of Shenzhen (No. JCYJ20200109143010272) and Oversea Cooperation Foundation of Tsinghua Shenzhen International Graduate School.

6. REFERENCES

[1] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International journal of computer vision*, 92(1):1–31, 2011.

[2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019.

[3] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8628–8638, 2021.

[4] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10663–10671, 2020.

[5] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[6] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Rife: Real-time intermediate flow estimation for video frame interpolation. *arXiv preprint arXiv:2011.06294*, 2020.

[7] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989, 2018.

[8] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[9] Li Siyao, Shiyu Zhao, Weijiang Yu, Wenxiu Sun, Dimitris Metaxas, Chen Change Loy, and Ziwei Liu. Deep animation video interpolation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6587–6595, 2021.

[10] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. Xvfi: Extreme video frame interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14489–14498, 2021.

[11] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446, 2020.

[12] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 670–679, 2017.

[13] Mantang Guo, Jing Jin, Hui Liu, and Junhui Hou. Learning dynamic interpolation for extremely sparse light fields with wide baselines. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2450–2459, 2021.

[14] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 109–125. Springer, 2020.

[15] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[16] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. *arXiv preprint arXiv:1911.00627*, 2019.

[17] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019.

[18] Shangchen Zhou,Jiawei Zhang,Wangmeng Zuo,and Chen Change Loy. Cross-scale internal graph neural network for image super-resolution. *arXiv preprint arXiv:2006. 16673*, 2020.

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[20] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.