Lightweight Transformer Super-Resolution

Dokyun Kim, Gyeongseon Eo, Sooyoun Park, Soyeon Park
Seoul National University
dokyunkim, eks104, soopark0221, spark415 @snu.ac.kr

Abstract

Deep neural networks (DNNs) have shown great success in image super-resolution (SR) task, which is generating a high-resolution (HR) image from a low-resolution (LR) image. However, most SR networks are computationally challenging, requiring substantial computation and memory consumption.

Although there exist many research applying model compression techniques to a classification task, applying them to image restoration has not been studied extensively. This is mainly because an image restoration model needs to maintain the details of the image, and compressing may lose some of the features of the image. To address this problem, SwinIR was proposed, which is lightweight restoration model based on Shifted Window Transformer (Swin Transformer). In this work, we aim to compress SwinIR network further to achieve even lower computational cost. Our main approaches are knowledge distillation (KD) and network pruning. Our experimental results show that our models achieve performance comparable to that of the original SwinIR network with much fewer channels and parameters. There are some challenges remaining, and we would like to share them in this paper.

1. Introduction

Image super-resolution (SR) aims at recovering a high-resolution image from its low-resolution counterpart. Deep neural networks (DNNs) have brought a lot of improvement in image SR as much as other computer vision tasks [8]. Various network architecture designs and training strategies have continuously improved SR performance. Many recent works showed that increasing the model depth helps to improve the reconstruction quality. Kim et al. [15, 16] proved in their paper that the deeper the model gets, the better the performance gets in the SR task. GAN-based methods have attracted a lot of attention showing outstanding performance [17, 31].

Recently, Transformer-based methods have been introduced in image restoration tasks and showed competitive results compared to other advanced SR models [2, 4, 20, 32]. Despite of the outstanding performance, most state-of-the-art SR networks are very deep and complex and thus not efficient for general applications such as using on mobile devices due to high computational cost and memory consumption. Liang et al. [20] proposed SwinIR which applies Swin Transformer [23] to image restoration tasks including SR and image denoising. Swin Transformer, originated from Vision Transformer (ViT) [7], applies bounded self-attention to reduce computational complexity. SwinIR uses residual Swin Transformer blocks (RSTB) for feature extraction and achieves good performance with fewer parameters.

In this paper, we utilize SwinIR as our baseline model and apply model compression techniques for Transformer networks. In particular, we focus on knowledge distillation [10, 28, 29, 35] and network pruning [22, 25, 34, 38]. Knowledge distillation is a process of transferring knowledge from a large (teacher) model to a small (student) model. The student model is trained to learn the exact behavior of the teacher network by trying to replicate it's outputs. We train the original SwinIR with 6 Residual Swin Transformer Blocks (RSTB) as a teacher model, and that with 4 RSTBs as a student model. A loss function is updated to consider the loss from comparing the teacher and the student output and also the difference between ground truth and student output together. Network pruning is removing parameters from a trained model. It cuts off redundant parameters and reduce the size of the network. We apply pruning on the multi-layer perceptron (MLP) and multi-head self-attention (MSA) layers in transformer blocks of SwinIR.

DIV2K dataset is used to train our model and performed test on Urban100 and Manga109 dataset. The KD student model taught by the teacher network along with ground truth outperforms the plain student model that only learns the ground truth by 0.25 dB PSNR and 0.01 SSIM. Despite the optimal ratio of distillation loss leads to better outcomes, weight more on it decreases the PSNR and SSIM. We observed that the KD is more effective with a set of optimized hyperparameters. The pruned model shows comparable results with the original SwinIR. When pruning 25% of MLP

blocks, the average PSNR only drops by 0.22 dB (averaging Urban100 and Manga109). When pruning 25% of both and MLP and MSA blocks, PSNR drops by 0.37 dB. This is a reasonable result as more channels are reduced. It is a trade-off between the performance and computational load, but the experiments demonstrate that the model can preserve significant information even with large reduction in parameters.

2. Related Work

2.1. Lightweight Super-Resolution

Hui et al. [13] proposed a lightweight information multi-distillation network (IMDN) which outperforms state-of-the-art models [1, 6, 15]. IMDN splits the output channel of convolutional layer before information multi-distillation block (IMDB) into 'refined' and 'coarse' features, to reduce the number of parameters drastically. Utilizing adaptive cropping strategy to process images of any arbitrary size diminishes the computational cost, memory occupation and, inference time as well. Lattice Net [24] proposed another approach of lightweight SR. The major idea of the network is to have Lattice Block (LB) with two butterfly structures, each of which comprises a residual block(RB). Lattice Net is beneficial to attain to efficiency of its structure that linear combination patterns of two RBs. Lattice Net reduces the parameters by up to half against state-of-the-art models without compromising its performance.

2.2. Knowledge Distillation for CNN-based SR

Knowledge distillation is an efficient network training idea that a small student model imitates a pre-trained heavy model. [10]. There are only a few attempts to implant KD onto CNN-based SR. A feature affinity-based KD (FAKD) for image SR [9] used the correlation within a feature map to transfer structural knowledge. Lee et al. [18] used ground truth HR images as privileged information and feature distillation to train compact SR network. Both models successfully improved performance.

2.3. Vision Transformers

Vision Transformers. Recently, various studies have been conducted using Transformer [30] in the field of computer vision problems such as image classification [7, 19, 23, 26, 33], object detection [3, 21, 23, 29], and segmentation [23, 33, 37]. Ramachandran et al. [26] proposed ResNet architecture consists of self-attention to capture long-range dependencies. Dosovitskiy et al. [7] interpreted the image as a sequence of patches and devised a model, ViT. ViT showed similar performance to state-of-the-art CNN based models while reducing computational resources to train. Since ViT model relies on huge training datasets, Touvron et al. [29] proposed Data-efficient image Transformer (DeiT) by applying knowledge distillation and data augmentation to ViT. Liu et al. [23] proposed a Swin Transformer that can be used as a backbone for various vision tasks. By applying shifted window approach and hierarchical feature maps, the parameters were reduced while successfully handling vision problems.

Image Restoration Architecture. Several recent works proved that image restorations using transformers can yield remarkable performance [2, 4, 20, 32]. Chen et al. [4] developed a new pre-trained backbone model Image Processing Transformer (IPT) for computer vision tasks based on standard Transformer. Cao et al. [2] proposed a VSR-Transformer for video super-resolution. However, both IPT and VSR-Transformer employ patch-wise attention, which may introduce border artifacts around each patch in the restored image. To address this problem, Wang et al. [32] presented Uformer, which is Swin Transformer based U-shaped architecture for image restoration. Liang et al. [20] proposed the SwinIR model and with feature extraction modules, image reconstruction module, and several residual connections, SwinIR achieves better Peak Signal-to-Noise Ratio (PSNR) with fewer parameters compared to prevalent CNN-based image restoration models.

2.4. Compressing Transformer-Based Model

Transformer-based models showed remarkable performance in many tasks. However, these models are very heavy and cause high latency. A model compression is an active area of research these days, and there are some main methods: Knowledge Distillation and Pruning.

Knowledge Distillation (KD). Sun et al. [28] proposed MobileBERT for compressing and accelerating the popular BERT model. They use knowledge distillation and quantization methods to lighten it. Jiao et al. [14] proposed TinyBERT and Sanhet et al. [27] proposed DistilBERT which also use distill strategies.

Pruning. Zhu et al. [38] showed a dimension-wise pruning for Vision Transformers. They applied pruning operations on the MSA and MLP blocks. Yang et al. [34] proposed NVIT, which applies global structural pruning with latency-aware regularization on all parameters of the ViT model. Mao et al. [25] and Hou et al. [12] presented pruning with KD methods for BERT.

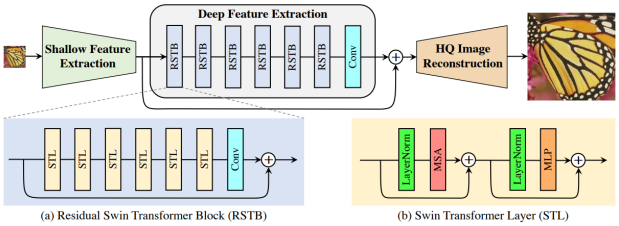


Figure 1. The architecture of the SwinIR (adopted from [20]).

3. Method

3.1. Preliminaries

3.1.1 SwinIR Network Architecture

The SwinIR [20] architecture is presented in figure 1. L1 pixel loss is employed as a loss function.

Shallow and Deep Feature Extraction. A 3×3 convolutional layer is used in Shallow feature extraction module. Then SwinIR extract deep feature F_{DF} by deep feature extraction module H_{DF}. The module contains sequentially stacked K RSTBs and a 3×3 convolutional layer at the end of the module. Through this RSTB, intermediate features $F_1, F_2, \ldots, F_k$ are created, and finally the output deep feature F_{DF} are extracted by applying the last convolutional layer to the last intermediate feature F_k as

$$F_i = H_{RSTB_i}(F_{i-1}), \quad i = 1, 2, ...K, \tag{1}$$

$$F_{DF} = H_{CONV}(F_K), \tag{2}$$

where H_{RSTB_i} denotes the i-th RSTB and H_{CONV} is the last convolutional layer.

Image Reconstruction. SwinIR reconstructs the high-quality image I_{RHQ} by aggregating shallow and deep features as

$$I_{RHQ} = H_{REC}(F_0 + F_{DF}), \tag{3}$$

where H_{REC} is the function of the reconstruction module. With the skip connection, deep feature extraction module can focus on high-frequency information and stabilize training.

Residual Swin Transformer Block and Swin Transformer layer. The RSTB is a residual block consists of L Swin Transformer Layers (STLs) and a convolutional layer. When the input feature is the i-th RSTB, $F_{i,0}$, the intermediate features, $\{F_{i,1}, F_{i,2}, ..., F_{i,l}\}$, are extracted by j-th Swin Transformer Layers of the i-th RSTB $H_{STL_{i,j}}$ as

$$F_{i,j} = H_{STL_{i,j}}(F_{i,j-1}) \tag{4}$$

Swin Transformer layer (STL) is adopted from the Swin Transformer architecture in [23].

3.2. Model Compression

3.2.1 Knowledge Distillation

Knowledge distillation (KD) is a process of transferring knowledge from a large model (teacher model) to a small model (student model). This is a special technique that does not explicitly compress the model from any dimension of the network [10]. To utilize the KD method, we should define a teacher model and a student model first. 3

In TinyBERT [14], which has fewer layers than the originalBERT, the authors used the original BERT as a teacher model and TinyBERT as a student model. So that they could make a lightweight network by cutting some layers from the original network while minimizing the performance loss. We applied this idea to the SwinIR [20]. Original SwinIR has 6 RSTBs and each RSTB consists of 6 STLs. We use the original SwinIR as teacher model and lightweight SwinIR which has 4 RSTBs and each of them consists of 4 STLs as a student model. Since the authors of the SwinIR provided a pre-trained classical model which has achieved SOTA, we used that model as a teacher to solve limited computing resources problem.

We also adapt the idea of distillation token from the DeiT model [29]. The structure is shown in 2. We add a distillation token along with the patch tokens, in the process of patch embeddings, so that the token interacts with other embeddings through self-attention. The distillation embedding makes the student model learn from the teacher's output. The distillation loss is reduced through back propagation.

In case of the SR task, the loss function of KD can be

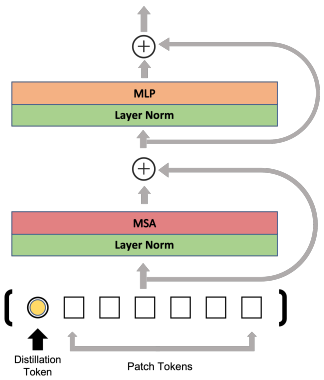


Figure 2. The diagram of inducing a distillation token. The distillation token which is input of the STL learns by back-propagation(adapted from [29]).

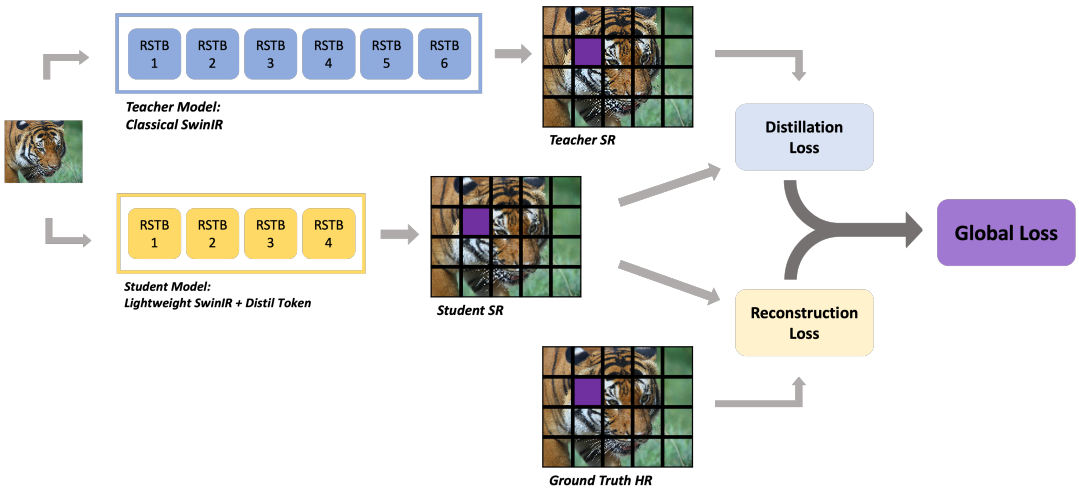


Figure 3. Framework of the proposed knowledge distillation in SR task. The student model is trained with distillation loss and reconstruction loss.

applied by the following function [36],

$$L_{distill} = E_{x \in p_x(x)}[||T(x) - S(x)||_1], \tag{5}$$

where x is the training sample and $p_x(x)$ is the distribution of the original dataset. T is the teacher network and S is the student network.

We define the global loss function as an affine combination of the reconstruction loss L_{recon} and distillation loss $L_{distill}$ as below.

$$L_{global} = (1-\lambda)L_{recon} + \lambda L_{distill},\ \lambda \in [0,1] \tag{6}$$

$$L_{global} = (1-\lambda)||I_{RHQ} - I_{HQ}||_1 + \lambda||I^T_{RHQ} - I_{RHQ}||_1, \tag{7}$$

For the reconstruction loss, the loss function of original SwinIR has been selected, where I_{RHQ} is the reconstructed high-quality image from the student model and I_{HQ} is the high-quality ground truth image. Distillation loss is calculated as reconstruction loss does with I^T_{RHQ}, reconstructed image by the teacher model. λ is the only hyperparameter of this loss function.

3.2.2 Network Pruning

The network pruning is removing parameters from an existing network. The pruning is categorized into two: unstructured pruning and structured pruning. The unstructured pruning is pruning individual weights and the structured pruning is pruning channels. Generally, the pruning consists of three parts. 1) training, 2) pruning, 3) fine-tuning [22,38]. an Zhu et al. [38] proposed a vision transformer pruning. They applied channel pruning on ViT, and demonstrated comparable performance with the original model despite of large reduction of parameters. NVIT paper [34] presented latency-aware global structural pruning method for making lightweight ViT model [7]. We adapted the idea to SwinIR.

As Swin Transformer [23] is originated ViT, we utilize the channel pruning method. The channel reduction is done on MSA and MLP in the STLs. We first train the importance score for each channel of linear projections in the STLs. This is done along with the training of the network. To push the importance score to zero, ℓ_1 regularization is applied to the importance score as in $\gamma\|a\|$, where γ is the sparsity hyperparameter and a is the importance score. [22, 38]. This is added to the objective function in training. We multiply the channels with the learnt importance scores to get pruned channels: $X^* = X\text{diag}(a)$, where X is channels. The importance scores for all channels are ranked from low to high. After aligning channels along with the ranked importance scores, we cut the channels starting from the ones having low importance score. The amount to pruning is determined by the pruning rate. We apply pruning on all MLP and MSA blocks.

4. Experiment

4.1. Datasets and Metrics

We train SwinIR (Classical, Lightweight) and our models on DIV2K dataset. DIV2K dataset contains 900 2K resolution images. DIV2K is divided into 800 train images and 100 test images. Due to the limitations on resources, we use only 384 train images for training and resize the height of weight of images into $1/8$ from the originals. Then, we test our models on Urban100 and Manga109 datasets. Per-

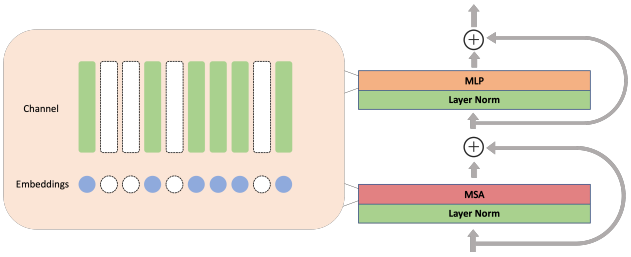


Figure 4. The channel pruning architecture. Channels with low importance scores are pruned.

formance measures (PSNR and SSIM) and model compression(reduction of number of parameters and channels) is estimated as quality metrics.

4.2. Implementation Details

All experiments are performed with a scaling factor of x2 between LR and HR images. The training HR images are resized to 255 x 169 and LR images are resized to 127 x 84. We set the batch size to 32, window size to 8 x 8 and the patch size to 48 x 48. For training model, we utilize GeForce RTX 3090 TI GPUs in GSDS cluster. Therefore, we attempted distributed data parallel programming and succeeded using multiple GPUs for training.

However, since we are sharing GPUs with other colleagues and students in the college, it was limited to train numerous versions of the model with different hyperparameters. For the reliable and consistent result comparison in time, it was forced to use single GPU experiment with maximum GPU access time to 12 hours.

4.3. Results

As image quality evaluation methods for the SR task, we used PSNR for objective method and SSIM for subjective method which are well described in the paper [11]. A small PSNR value implies high numerical differences between images. The SSIM is a quality metric used to consider the quality perception of the human visual system. As a combination of three factors that are loss of correlation, luminance distortion, and contrast distortion, SSIM is designed to catch image quality loss.

Table 1 shows the test result of SwinIR on two datasets. The classical SwinIR model has an average PSNR value of 27.46 dB for the Urban100, and 28.41 dB for the Manga109. SSIM values of Urban100 and Manga109 are 0.9135 and 0.9088, respectively. All results are based on scale factor x2. Visual comparisons are shown in Figure 5,6 .

4.3.1 Knowledge Distillation

The global loss function of the knowledge distilled lightweight SwinIR which is well represented in equation 6 and 7 is a sum of reconstruction and distillation loss terms, balanced by the hyperparameter λ. We test with three λ value: 0.25, 0.5, 0.75, and each result is shown in table 1. The larger the lambda value, the more weight is given to reduce the loss between the teacher and the student, and the smaller the lambda value, the more weight is given to reduce the loss between the student and the ground truth.

The parameters of the lightweight model was nearly reduced by a factor of 13 (from 11750k to 910.15k), compared to the classical model, but the model performance was also decreased. For example, the average PSNR value of the Manga109 was decreased by 0.51 dB (from 28.41 dB to 27.90 dB), and that of Urban100 was decreased as well by 0.42 dB (from 27.46 dB to 27.04 dB).

When knowledge distillation was applied, the number of parameters was 910.45k, which was similar to that of the lightweight model as 910.15k. However, when the hyperparameter λ is 0.25, PSNR in the Manga109 was 28.26dB, which increased by 0.36 dB compared to lightweight without KD, and in the Urban100, it increased by 0.13 dB to 27.17 dB. That is, when KD was applied, the number of parameters could be dramatically reduced, and the model performance drop could be minimized at the same time. Visual comparisons in Fig 5, 6 are also reflect the qualitative comparison results as above.

However, when applying KD, the principle concern is setting the appropriate hyperparameter λ. In our training environment, the restoration performance was superior than the lightweight model when the λ is 0.25. This implies that model performance loss due to the lightweight-ization could be lessened when the student model focused more on reducing the loss with the ground truth than on reducing the loss with the teacher.

4.3.2 Network Pruning

The original SwinIR network consists of 6 RSTBs and each RSTB contains 6 STLs. We have 36 blocks of MLP and 36 blocks of MSA in total. Each STL has 6 heads. Each MLP block has 360 channels and MSA has 180 channels. The hyperparameters are:

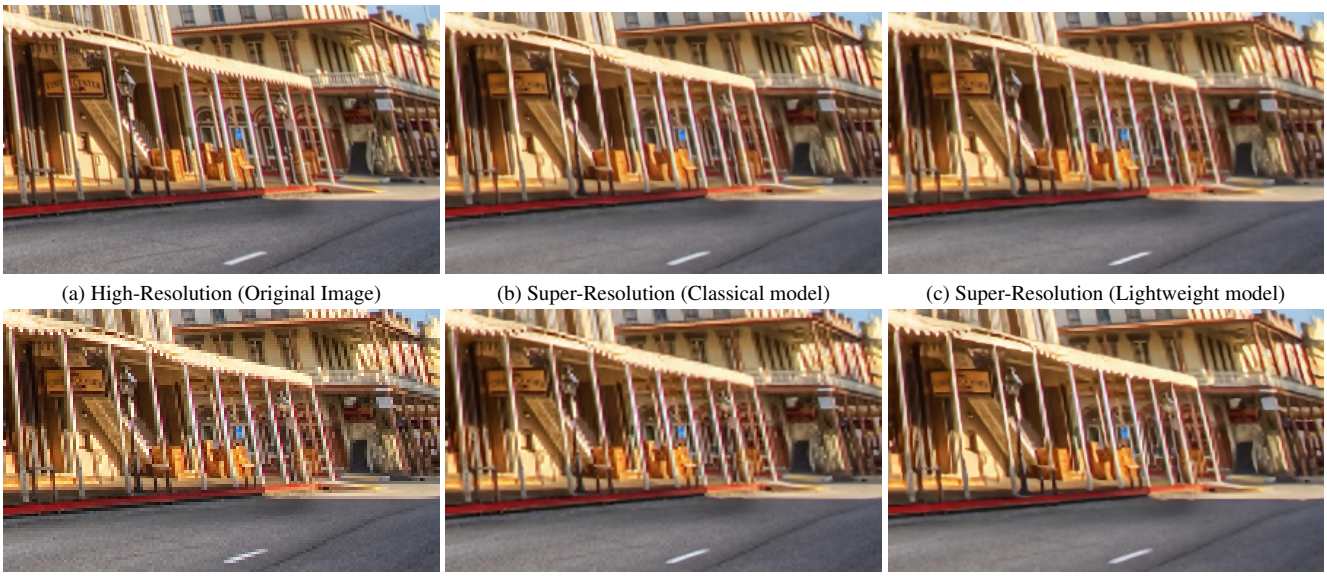
- Pruning rates : 0.25, 0.5, 0.75
- Sparsity γ : $1e^{-4}$, $1e^{-5}$

Pruning (MLP) When pruning 25% of channels (75% remaining), the average PSNR of Urban100 is 27.32 dB, 28.20 dB for Manga109. The average SSIM is 0.9201 and 0.8655 for Urban100 and Manga109, respectively. When

(a) High-Resolution (Original Image) (b) Super-Resolution (Classical model) (c) Super-Resolution (Lightweight model)

(d) Low-Resolution (Downsample, 1/2 H x W) (e) Super-Resolution (Knowledge Distillation) (f) Super-Resolution (Pruning)

Figure 5. Cropped partial images for the purpose of better visual comparison. (a) Ground truth when training. (b) SR image by classical SwinIR. (c) SR image by lightweight SwinIR. (d) LR input, half the width and height of the original image. (e) SR image by KD model. (f) SR image by pruned model.



(a) High-Resolution (Original Image) (b) Super-Resolution (Classical model) (c) Super-Resolution (Lightweight model)

(d) Low-Resolution (Downsample, 1/2 H x W) (e) Super-Resolution (KD model) (f) Super-Resolution (Pruning model)

Figure 6. Cropped partial images for the purpose of better visual comparison. (a) Ground truth when training. (b) SR image by classical SwinIR. (c) SR image by lightweight SwinIR. (d) LR input, half the width and height of the original image. (e) SR image by KD model. (f) SR image by pruned model.

50% are pruned, the average PSNR drops to 27.33 dB (Urban100) and 27.94 dB (Manga109). As can be seen in the experiment, pruning half of MLP channels barely affects the performance. The results are based on $\gamma = 1e^{-5}$.

Pruning (MLP+MSA) When pruning 25% of channels, the average PSNR of Urban100 is 27.52 dB, 27.62 dB for Manga109. Manga109 shows even higher PSNR than the classical model. The average SSIM is 0.9139 and 0.8691 for Urban100 and Manga109, respectively. With 50% pruning, the PSNR drops to 26.24 dB (Urban100) and 26.56 dB (Manga109).

$\gamma = 1e^{-5}$ shows better performance than $\gamma = 1e^{-4}$.

		Base		KD			Pruning(MLP)			Pruning(MLP+Attention)		
		C[1)]	L[2)]	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
Manga	PSNR	28.41	27.90	28.26	27.37	26.40	28.20	27.94	27.33	27.62	26.56	23.00
	SSIM	0.9088	0.9152	0.9208	0.9072	0.8908	0.9201	0.9149	0.8620	0.9139	0.8877	0.7461
Urban	PSNR	27.46	27.04	27.17	26.13	24.61	27.32	27.33	27.36	27.52	26.24	20.97
	SSIM	0.9135	0.8583	0.8729	0.8567	0.8252	0.8655	0.8620	0.8540	0.8691	0.8310	0.5694

Table 1. Experimental results of all models. Best and second best performance are in red and blue colors, respectively. 1) C means 'Classic'. 2) L means 'Lightweight'. KD's parameter indicates λ and Pruning's parameter indicates γ

γ	Urban100	Manga109
$\gamma = 1e^{-5}$	27.52/0.8691	27.62/0.9139
$\gamma = 1e^{-4}$	25.06/0.8256	26.69/0.9053

Table 2. Average PNSR/SSIM results of two sparsity regularization (25% MLP+MSA pruning)

The comparison is shown in Table 2. With $\gamma = 1e^{-4}$, the pruning mostly occurs on the front layers. Whereas with $\gamma = 1e^{-5}$, the pruning happens across the entire channels. The Fig 5, 6 show the image after pruning. It is hard to distinguish the original model and the pruned model perceptually.

5. Conclusions

5.1. Summary

In this paper, we propose a lightweight SR model using KD and the network pruning. We use SwinIR as our baseline model. The KD and the pruning is applied to the STLs of SwinIR, the part where the deep feature extractions of images occurs. The experiments show that the model compression could reduce computational costs and number of parameters without losing the performance. The KD with appropriate hyperparameter λ reduces the number of parameters by a factor of 13 (11750k to 910.15k) compared to the classical model. At the same time, the model performance drop improves by nearly 70% (0.51 dB to 0.15 dB) on Manga109 and 30% (0.42 dB to 0.29 dB) on Urban100 compared to the model without teacher guidance on a PSNR basis. Pruning 25% of channels from linear projections in STLs reduces PSNR level by 0.37 dB and -0.020 SSIM level (averaging Urban100 and Manga109) as compared to the original SwinIR model. We demonstrate the effectiveness of model compression in Transformer based SR.

5.2. Limitations and Future Work

In our experiments, we had limited resources in terms of the number of GPUs and training time. Under the limited computing resources, we could not reproduce the models as the original SwinIR experiment setting. With an optimum computational power, we will be able to establish a better baseline for verifying the efficiency of model compression methods we have applied in this paper.

As future works, different model compression methods can be attempted. Network quantization could be another good compression methods for Transformer. It compresses the original network by reducing the number of bits required to represent each weight [5]. The computational costs of vision transformer heavily depend on the large matrix multiplication in MSA and MLP module, thus the quantization can be applied on Transformer layers as well.

References

[1] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network, 2018. 2

[2] Jiezhang Cao, Yawei Li, Kai Zhang, and Luc Van Gool. Video super-resolution transformer. *arXiv preprint arXiv:2106.06847*, 2021. 1, 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 2

[4] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021. 1, 2

[5] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks, 2020. 7

[6] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks, 2015. 2

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 4

[8] Kai Zhang et al. Ntire 2020 challenge on perceptual extreme super-resolution: Methods and results, 2020. 1

[9] Zibin He, Tao Dai, Jian Lu, Yong Jiang, and Shu-Tao Xia. Fakd: Feature-affinity based knowledge distillation for efficient image super-resolution. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 518–522, 2020. 2

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 1, 2, 3

[11] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. 5

[12] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. Dynabert: Dynamic bert with adaptive width and depth, 2020. 2

[13] Zheng Hui, Xinbo Gao, Yunchu Yang, and Xiumei Wang. Lightweight image super-resolution with information multi-distillation network. *Proceedings of the 27th ACM International Conference on Multimedia*, Oct 2019. 2

[14] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding, 2020. 2, 3

[15] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks, 2016. 1, 2

[16] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution, 2016. 1

[17] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2017. 1

[18] Wonkyung Lee, Junghyup Lee, Dohyung Kim, and Bumsub Ham. Learning with privileged information for efficient image super-resolution. In *ECCV*, 2020. 2

[19] Yawei Li, Kai Zhang, Jiezhang Cao, Radu Timofte, and Luc Van Gool. Localvit: Bringing locality to vision transformers. *arXiv preprint arXiv:2104.05707*, 2021. 2

[20] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021. 1, 2, 3

[21] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020. 2

[22] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming, 2017. 1, 4

[23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 1, 2, 3, 4

[24] Xiaotong Luo, Yuan Xie, Yulun Zhang, Yanyun Qu, Cuihua Li, and Yun Fu. Latticenet: Towards lightweight image super-resolution with lattice block. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 272–289, Cham, 2020. Springer International Publishing. 2

[25] Yihuan Mao, Yujing Wang, Chufan Wu, Chen Zhang, Yang Wang, Yaming Yang, Quanlu Zhang, Yunhai Tong, and Jing Bai. Ladabert: Lightweight adaptation of bert through hybrid model compression, 2020. 1, 2

[26] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019. 2

[27] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. 2

[28] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices, 2020. 1, 2

[29] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 1, 2, 3

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[31] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. Esrgan: Enhanced super-resolution generative adversarial networks, 2018. 1

[32] Zhendong Wang, Xiaodong Cun, Jianmin Bao, and Jianzhuang Liu. Uformer: A general u-shaped transformer for image restoration. *arXiv preprint arXiv:2106.03106*, 2021. 1, 2

[33] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020. 2

[34] Huanrui Yang, Hongxu Yin, Pavlo Molchanov, Hai Li, and Jan Kautz. Nvit: Vision transformer compression and parameter redistribution, 2021. 1, 2, 4

[35] Yiman Zhang, Hanting Chen, Xinghao Chen, Yiping Deng, Chunjing Xu, and Yunhe Wang. Data-free knowledge distillation for image super-resolution. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7848–7857, 2021. 1

[36] Yiman Zhang, Hanting Chen, Xinghao Chen, Yiping Deng, Chunjing Xu, and Yunhe Wang. Data-free knowledge distillation for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7852–7861, 2021. 4

[37] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao

Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021. 2

[38] Mingjian Zhu, Yehui Tang, and Kai Han. Vision transformer pruning, 2021. 1, 2, 4