Micro-lens image stack upsampling for hyperspectral light fields

Paul Hasenbusch

October 23, 2025

Contents

1 Models

1.1 Prelimanries

1.1.1 Neural Networks

Definition 1 (Fully Connected Layer) *Let $A \in \mathbb{R}^{n\times m}$ and $b \in \mathbb{R}^m$. We call the mapping*

$$F : \mathbb{R}^n \to \mathbb{R}^m \ , \quad F(x) = Ax + b \ ,$$

a fully connected layer.

In order to refer to the architecture, thats is a fully connected layer with input dimension $n \in \mathbb{N}$ and output dimension $m \in \mathbb{N}$, whose weights are not fixed but subject to optimization, we write $F(n, m)$.

Definition 2 (Paralelization) *Let X, Y be two sets, the parallelization operation P is defined by*

$$P : f(X,Y) \times f(X,Y) \to f(X,Y^2) \ , \quad P(f,g)(x) = \begin{pmatrix} f(x) \\ g(x) \end{pmatrix} \ .$$

1.1.2 Convolutional Layers

Definition 3 (Convolution over multiple feature maps) *Let $X \in \mathbb{R}^{C\times n_1\times\ldots\times n_d}$ and $k \in \mathbb{R}^{C\times d_1\times\ldots\times d_d}$. The convolution of X with the kernel k, denoted by $k * X$ is given by*

$$(k * X)(p_0) = \sum_{i=1}^{c} \sum_{p\in R} k(c,p)X(c, p_0 + p) \ ,$$

where $R = \prod_{i=1}^{d}[0, d_i] \cap \mathbb{N}$, for all $p_0 \in \prod_{i=1}^{d} \mathbb{N} \cap [1, n_i - d_i]$.

Definition 4 (Convolutional Layer) *Let $X \in \mathbb{R}^{C\times n_1\times\ldots\times n_d}$, $k_1, ..., k_{C'} \in \mathbb{R}^{C\times d_1\times\ldots\times d_d}$ and $b \in \mathbb{R}^{C'}$. We call the mapping*

$$C : \mathbb{R}^{C\times n_1\times\ldots\times n_d} \to \mathbb{R}^{C'\times n_1\times\ldots\times n_d} \ , \quad C(X) = [k_1 * X, ..., k_{C'} * X] + b \ ,$$

a convolutional layer.

In order to refer to the architecture, thats is a convolutional layer with input channel dimension $C \in \mathbb{N}$ and output channel dimension $C' \in \mathbb{N}$, kernels $k_1, ..., k_{C'} \in \mathbb{R}^{C\times d_1\times\ldots\times d_d}$ and $b \in \mathbb{R}^{C'}$, which are not fixed but subject to optimization, we write

$$C(n, m, \text{kernel-size} = (d_1, ..., d_d), \text{padding} = p, \text{padding-mode} = m) \ .$$

Definition 5 (Residual Connection) *Let X be some set, $F = \{f : X \to X | f \text{is a function}\}$ be the set of functions on X mapping back on X. The operation*

$$R : F \to F \ , \quad R(f)(x) = f(x) + x \ ,$$

for all $x \in X$, is called the residual mapping.

Pooling operations, as for example max-pooling, reducing the spatial dimension of feature maps by utilizing strides larger than 1 are widely known.

On the other side there are also variations of convolutional operations to increase the size of feature maps. Most notably transposed convolutions and sub-pixel convolutions.

1.1.3 Transformers

Transformers operate on sequences of data $(x_k)_{k=1}^n$, where $x_k \in \mathbb{R}^d$. In the literature the elements of the input sequence are commonly referred to as tokens. In the following we denote the set of sequence over a set A by $S(A)$. Central to Transformer models is the so-called attention mechanism. The tokens x_k are embedded into three different subspaces using linear mappings $Q, K, V \in \mathbb{R}^{d',d}$, to obtain queries $(q_k)_{k\in\mathbb{N}}$, keys $(k_k)_{k\in\mathbb{N}}$ and values $(v_k)_{k\in\mathbb{N}}$, where

$$q_k = Qx_k \ , \quad k_k = Kx_k \text{ and } v_k = Vx_k \ ,$$

for all $k = 1, ..., n$. The queries q_k and keys k_k are used to compute the attention scores among the tokens, measuring the level of relevance of their respective information for each other

$$A_{ij} = \frac{\exp(k_i^T q_j)}{\sum_{k=1}^n \exp(k_k^T q_j)} \ . \tag{1}$$

The outputs are then computed for all $j = 1, ..., n$ by

$$y_j = \sum_{i=1}^n A_{ij} v_i \ . \tag{2}$$

Note that by construction for all $j = 1, ..., n$ holds

$$\sum_{i=1}^n A_{ij} = 1 \ .$$

Definition 6 (Self-Attention) *Let $Q, K, V \in \mathbb{R}^{d',d}$. The operation described in equations (1), (2), that is*

$$SA : S(\mathbb{R}^d) \to S(\mathbb{R}^d) \ , \quad SA(Q, K, V)\left((x_k)_{k=1}^n\right) = (y_k)_{k=1}^n \ ,$$

is called self-attention.

To increase the capacity of Transformer models, multiple self-attention operations, called heads in the literature, are used in parallel to process the input sequence.

Definition 7 (Multi Headed Self-Attention) *Let* $Q_h, K_h, V_h \in \mathbb{R}^{d',d}$ *for* $h = 1, ..., H$ *and let* $Q = (Q_1, ..., Q_H), K = (K_1, ..., K_H), V = (V_1, ..., V_H)$. *The operation*

$$MSA(Q, K, V)\left((x_k)_{k=1}^n\right) = \left[SA(Q_1, K_1, V_1)\left((x_k)_{k=1}^n\right), ..., SA(Q_H, K_H, V_H)\left((x_k)_{k=1}^n\right)\right],$$

is called multi headed self-attention.

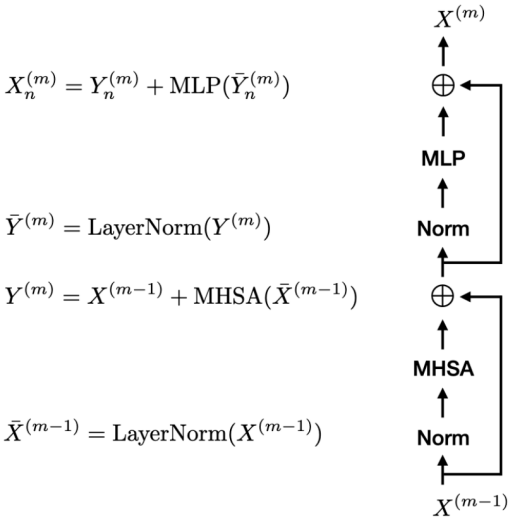

Figure 1: Image taken from [**?**], Transformer Block architecture.

In order to refer to the architecture, that is Multi Headed Self-Attention with dimension $d \in \mathbb{N}$ and number of heads $H \in \mathbb{N}$, whose weights are not fixed but subject to optimization, we write $\mathrm{MSA}(d, H)$.
Given a number of heads $H \in \mathbb{N}$ typically the embedding dimension of each head is chosen as $\frac{d}{H}$.
Another key ingredient for Transformer models is layer normalization.

Definition 8 (Layer Normalization) *Let* $\gamma \in \mathbb{R}$ *and* $\beta \in \mathbb{R}^d$. *The operation given by*

$$LN : \mathbb{R}^d \to \mathbb{R}^d \ , \quad LN(x) = \gamma \bar{x} + \beta \ \textit{where} \ \bar{x}_{ki} = \frac{1}{\sqrt{var(x_k)}}(x_{ki} - \sum_{j=1}^{d} x_{kj})$$

is called layer normalization.

Typically Multi Headed Self-Attention is used in transformer blocks, the architecture is outlined in figure 1. We describe this operation formally in the next definition.

Definition 9 (Transformer Block) *Let $d, H \in \mathbb{N}$ and $\Phi : \mathbb{R}^d \to \mathbb{R}^d$ be some mapping. The architecture*

$$T(d, H, \Phi) = R\big(\Phi \circ LN\big) \circ R\big(MSA(d, H) \circ LN\big)$$

is called a Transformer Block.

Typically the mapping Φ is some neural network, with only a few number of layers. The tokens are processed individually by the mapping Φ, so the sequence is treated as a batch.

Transformers operate on sequential data, so how to apply these models to images? In their 2020 paper *An image is worth 16 × 16 words: Transformers for image recognition at scale*, Dosovitskiy et al. [**?**] propose to partition image data $X \in \mathbb{R}^{C \times H \times W}$ into a sequence of patches, in order to bridge the gap between the two domains. An image is partitioned into patches sized $P \times P$, for some $P \in \mathbb{N}$, these are flattened and embedded using a linear mapping to obtain a sequence $(x_k)_{k=1}^N \in S(R^{C \cdot P^2})$, where $N = \frac{HW}{P^2}$. Let $w = \frac{W}{P}$, mathematically the partitioning can be described as follows

$$\hat{x}_k(i_c P^2 + i_h P + i_w) = X\left(i_c, \left\lfloor \frac{k}{w} \right\rfloor + i_h, k \mod w + i_w\right) , \tag{3}$$

for $k = 1, ..., N$, $i_c = 1, ..., C$ and $i_h, i_w = 1, ..., P$. The flattened patches are embedded to obtain the sequence of tokens $(x_k)_{k=1}^N$, that is

$$x_k = E\hat{x}_k ,$$

for some $E \in \mathbb{R}^{D \times C \cdot P^2}$. We implicitly assume that $H \mod P = W \mod P = 0$, i.e. both the height and the width are devisable by the patch size P. The approach is also visualized in figure 2.

Liu et al. [**?**] point out, that unlike in language, where a word naturally offers itself as the atomic unit, visual elements vary in scale, making the fixed patch sizes unsuitable for tasks requiring predictions at pixel level, as for example semantic segmentation. Simply treating each individual pixel as a token would solve the problem, but at the same time introduce immense computational complexity. For a full HD image of size 1920×1080 this leads to a sequence length of $2.0736 \cdot 10^6$. Thus to reduce computational complexity, but at the same time maintain a global receptive field, Liu et al. [**?**] propose Hierarchical Shifted Window Transformers (SWinT).

As before the image is partitioned into non-overlapping patches, as described in equation (3), Liu et al. [**?**] opt for a patch size $P = 4$, to obtain a sequence of tokens $(x_k)_{k=1}^N$. The tokens are then partitioned into subsequences

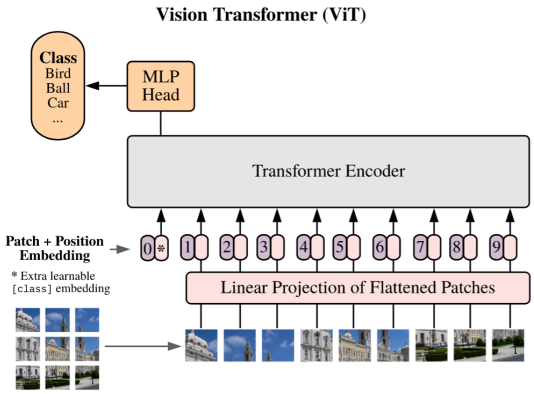

Figure 2: Image taken from [**?**], Vision Transformer.

$$\left(x_{k_l^{(1)}}\right)_{l=1}^{M^2}, ..., \left(x_{k_l^{(N')}}\right)_{l=1}^{M^2}$$

where $N' = \frac{HW}{4^2M^2}$. The subsequences $(k_l^{(p)})_{l=1}^{M^2}$ are chosen so that neighboring patches form a super patch of size $M \times M$, for some $M \in \mathbb{N}$, formally that is

$$k_l^{(p)} = \underbrace{M^2 \cdot \left\lfloor \frac{p}{\left(\frac{W}{4M}\right)} \right\rfloor}_{\text{inter patch row}} + \underbrace{M \cdot \left(p \mod \frac{W}{4M}\right)}_{\text{inter patch column}} + \underbrace{\left\lfloor \frac{l}{M} \right\rfloor \cdot \left(\frac{W}{4M} - 1\right)}_{\text{intra patch row}} + \underbrace{l}_{\text{intra patch column}}, \tag{4}$$

for all $p = 1, ..., N'$ and $l = 1, ..., M^2$, we are enumerating the super patches from left to right, top to bottom, note that $\frac{W}{4P}$ returns the number of super patches along the horizontal axis. Self-attention is then performed locally inside of each super patch

$$(y_{k_l^{(p)}})_{l=1}^{M^2} = \text{MSA}(Q, K, V)\left((x_{k_l^{(p)}})_{l=1}^{M^2}\right), \tag{5}$$

for $p = 1, ..., N'$, for some $Q, K, V \in \mathbb{R}^{H \times \frac{d}{H} \times d}$.

If equation (5) would be used repeatedly to update features, information is restricted to flow only inside individual super patches. To establish information flow amongst super patches, Liu et al. [**?**] introduce the shifted window mechanism. Consecutive operations of self-attention use different partitionings of the sequence $(x_k)_{k=1}^N$. We consider the partitioning described in equation (**??**) as the unshifted variant, for the shifted partitioning the borders of the patches are moved down and to the right by $s = \lfloor \frac{P}{2} \rfloor$ units. In order to achieve the shift, while keeping the same partitioning, a cyclic shift is applied to the feature map,

it is visualized in figure 3. Formally, this can be described by processing an auxilliary feature map $X' \in \mathbb{R}^{C\times H\times W}$, defined by

$$X'(c,i,j) = X\big(c,(i+s) \mod H,(j+s) \mod W\big)\ , \tag{6}$$

for $c = 1,...,C$, $i = 1,...,H$ and $j = 1,...,W$. Instead also padding techniques could be applied, but Liu et al. [**?**] report achieving better results, whilst saving computational complexity by employing the cyclic shift.

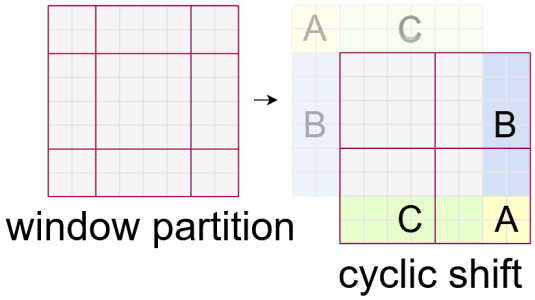

Figure 3: Image taken from [**?**], visualizing the cyclic shift. Here the pixels in the image are rearranged, and then the unshifted partitioning is used, to obtain the cyclic shift.

Definition 10 (Cyclic shift) *We denote the function implementing equation (6) by*

$$CS : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{C\times H\times W}\ ,\quad CS(X) = X'\ .$$

To enable a global receptive field, Liu et al. [**?**] propose to merge neighboring patches, after the inputs are processed by a certain number of SWin TransformerBlocks. To this end patches $P_1,...,P_4 \in \mathbb{R}^{C\times P\times P}$, inside a neighborhood of size 2×2 are stacked along the channel dimension, to form a super patch $\hat{P} = [P_1,...,P_4] \in \mathbb{R}^{4\cdot C\times P\times P}$. To fuse the features a convolutional layer is applied, halving the channel dimension of the super patch

$$P = C(4\cdot C, 2\cdot C, \text{kernel-size} = 3, \text{padding} = 1)(\hat{P})\ .$$

This process is repeated until the entire feature map is of size $P\times P$. We implicitly assume that $H = 2^{n_1}P$ and $W = 2^{n_2}P$ for some $n_1, n_2 \in \mathbb{N}$. The overall architecture of the SWin Transformer is shown in figure 4.

Definition 11 (Shifted window Transformer Block) *Let $d, H \in \mathbb{N}$ and $\Phi_1, \Phi_2 : \mathbb{R}^d \to \mathbb{R}^d$. The architecture given by*

$$SWinT(d,H,\Phi_1,\Phi_2) = T(d,H,\Phi_2)\circ CS\circ T(d,H,\Phi_1)\ .$$

is called Shifted Window Transformer Block.

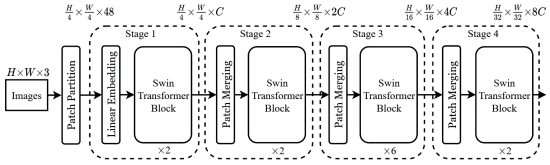

Figure 4: Image taken from [**?**], SWin Transformer.

Chen et al. [**?**] introduce overlapping Cross Attention (OCA), a modification of SWin Transformers. Whilst in equation (4) the patches are constructed to partition the feature map, OCA establishes cross patch connections, by constructing the patches so that they overlap. Formally, let $\gamma \in (0,1)$ be the factor of overlap and $p = 1, ..., N$ index the patches. Let $M_o = \lfloor (1+2\gamma)M \rfloor$, the subsequences associated to patch p is given by

$$k_l^{(p)} = \underbrace{M^2 \cdot \left\lfloor \frac{p}{\left(\frac{W}{4M}\right)} \right\rfloor}_{\text{inter patch row}} + \underbrace{M \cdot \left(p \mod \frac{W}{4M} \right)}_{\text{inter patch column}} + \underbrace{\left\lfloor \frac{l}{M_o} \right\rfloor \cdot \left(\frac{W}{4M_o} - 1 \right)}_{\text{intra patch row}} + \underbrace{l - \gamma M}_{\text{intra patch column}} + \underbrace{\gamma M}_{\text{padding}} , \tag{7}$$

for $l = 1, ..., M_o^2$, note padding was added and only the terms responsible for intra patch indexing were manipulated, this way the overlap is guaranteed. Super patches sharing an edge have an overlap of $\lfloor \gamma M \rfloor M$ pixels, whereas super patches sharing only a corner have an overlap of $\lfloor \gamma M \rfloor^2$ pixels. For query, key and value matrices $Q, K, V \in \mathbb{R}^{d',d}$, for a patch $p = 1, ..., P$, the attention scores are computed in the following way

$$A_{ij} = \frac{\exp({x_i^{(p)}}^T K^T Q x_j^{(p)})}{\sum_{k=1}^{n} \exp({x_k^{(p)}}^T K^T Q x_j^{(p)})} , \tag{8}$$

for $i = 0, ..., M_o$ and $j = \lfloor \gamma M \rfloor, ..., M + \lfloor \delta M \rfloor$. The outputs are then computed for all $j = \lfloor \gamma M \rfloor, ..., M + \lfloor \gamma M \rfloor$ by

$$y_j = \sum_{i=1}^{n} A_{ij} V x_i . \tag{9}$$

Thereby every token is updated exactly once, while capturing information from tokens belonging to other partitions. Note that equation (1) and (8) differ only by the indices attained by j, leading to the queries only coming from the part of the super patch, which is not shared by others. The operation is visualized in figure 5.

We conclude this chapter by introducing definitions for overlapping cross-attention, multi headed overlapping cross-attention and overlapping cross-attention block analogously to the definitions 10, 6, 7 and 9.

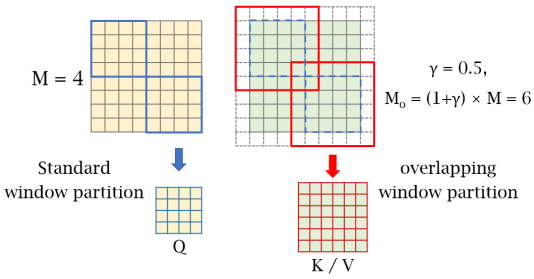

Figure 5: Image taken from [**?**], Overlapping Cross-Attention Block.

Definition 12 (Overlapping Cross-Attention) *Let* $Q, K, V \in \mathbb{R}^{d',d}$. *The operation described in equations (8), (9), that is*

$$OCA : S(\mathbb{R}^d) \to S(\mathbb{R}^d) \ , \quad OCA(Q,K,V)\big((x_k)_{k=1}^N\big) = (y_k)_{k=1}^N \ ,$$

is called overlapping cross-attention.

Analogously to standard self-attention, we also introduce multi headed overlapping cross-attention.

Definition 13 (Multi Headed Overlapping Cross-Attention) *Let* $Q_h, K_h, V_h \in \mathbb{R}^{d',d}$ *for* $h = 1, ..., H$ *and let* $Q = (Q_1, ..., Q_H), K = (K_1, ..., K_H), V = (V_1, ..., V_H)$. *The operation*

$$MOCA(Q,K,V)\left((x_k)_{k=1}^n\right) = \left[OCA(Q_1,K_1,V_1)\left((x_k)_{k=1}^n\right), ..., OCA(Q_H,K_H,V_H)\left((x_k)_{k=1}^n\right)\right] ,$$

is called multi headed overlapping cross-attention.

Definition 14 (Overlapping Cross-Attention Block) *Let* $\delta \in (0,1)$, $d, H \in \mathbb{N}$ *and* $\Phi : \mathbb{R}^d \to \mathbb{R}^d$ *be some mapping. The architecture*

$$OCAB(d,H,\Phi) = R\big(\Phi \circ LN\big) \circ R\big(MOCA(d,H) \circ LN\big) \circ P(\delta)$$

is called a Overlapping Cross-Attention Block.

1.2 General Model Architecture

Independent of the domain of application, a general architectural choice that can be observed in all super resolution models, is that the architecture is made up of three components. A shallow feature extraction module H_S, a deep feature extraction module H_D and an image reconstruction module H_{IR}. Typically the model architecture is conceptualized as follows

$$H = H_{IR} \circ R(H_D) \circ H_S \ . \tag{10}$$

The shallow feature extraction module H_S scales the channel dimension of the input to a higher dimension, which is used throughout the majority of the network. Additionally it extracts low frequency features. The module is usually composed of only one or few convolutional layers.
The deep feature extraction module H_D forms the main part of the model. It is supposed to recover high frequency information. Here is where different architectures proposed in the literature vary the most, convolutional layers, transformer models and various combinations thereof have been tried out.
Note the residual connection in equation (10), the rational behind this being that this way the low frequency features extracted by H_S can bypass the deep feature extraction module H_D. The image reconstruction module H_{IR} maps the input back to the original channel dimension and scales the spatial dimension to the desired size. It has been experimentally confirmed that better results are achieved when scaling is done at the end, rather than processing the already spatially upsampled input. To this end usually transposed convolutional layers or pixel sshuflling layers are employed.

1.3 Single Image Super Resolution

1.3.1 Deep Residual Channel Attention Network

The Deep Residual Channel Attention Network (DRCAN) proposed by Zhang et al. [**?**], the channel attention mechanism is introduced to single image super resolution. Channel Attention enables the network to dynamically assess which feature maps / channels are more important or need more refinement. This is achieved by processing the globally pooled average of the feature maps using a lightweight network and then reweighing the feature maps based thereon.

The overall model architecture is depicted in figure 6. The input image X is first processed via an initial convolutional layer

$$F_0 = C(3, 64, \text{kernel-size} = 3, \text{padding} = 1)(X) \ .$$

The following convolutional layers used in the architecture of the DRCAN are of the form

$$C = C(64, 64, \text{kernel-size} = 3, \text{padding} = 1, \text{padding-mode} = \text{zero}) \ ,$$

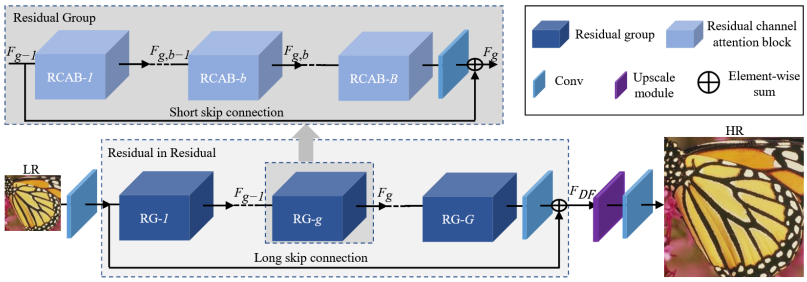

Figure 6: Image taken from [**?**], DRCAN model architecture.

that is 64 ingoing feature maps processed by 64 quadratic kernels of size 3×3 with zero-padding of size 1, so that feature map sizes are conserved throughout the model. The inial features F_0 are then further processed by a network with a residual in residual architecture

$$F_1 = H_D(F_0) \ .$$

For low-frequency features to bypass the deep feature extraction, a residual connection is used before the upsampling is performed

$$F_2 = F_0 + F_1 \ .$$

The final features F_2 are then upsampled using transposed convolutional layers.
The H_{RIR} network is composed of 10 Residual Groups followed by a final convolutional layer, that is

$$H_D = C \circ H_{RG} \circ ... \circ H_{RG} \ .$$

The Residual Groups (RG) are again composed of 20 Residual Channel Attention Blocks followed as well by a convolutional layer, the structure is encapsuled in a residual connection

$$H_{RG} = R\big(C \circ H_{RCAB} \circ ... \circ H_{RCAB}\big) \ .$$

The Residual Channel Attention Block (RCAB) depicted in figure 7, is made up of two convolutional layer, with a ReLU activation function in between, followed by a channel attention module, the output is then added back to the input again via a residual connection

$$H_{RCAB} = R\big(H_{CA} \circ C \circ \mathrm{ReLU} \circ C\big) \ . \tag{11}$$

The channel attention mechanism depicted in 8. The information of a feature map is first condesated into a single value by using global pooling

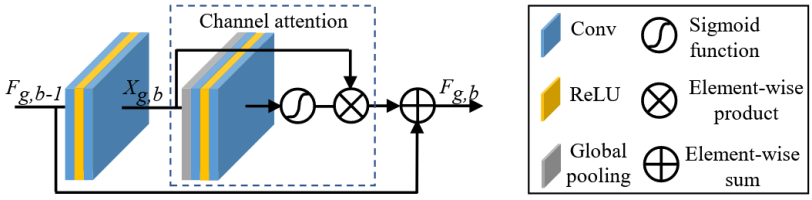

Figure 7: Image taken from [**?**], architecture of RCAB module.

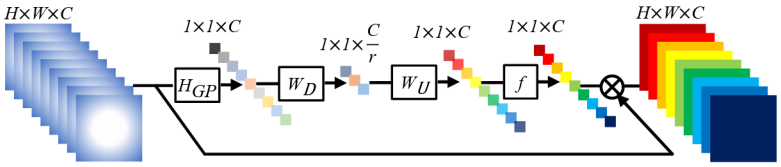

Figure 8: Image taken from [**?**], Channel Attention machanism.

$$z_c = H_{GP}(x_c) = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} x_c(i,j) \ ,$$

with the input $X = [x_1, ..., x_C] \in \mathbb{R}^{C\times H\times W}$. The vector $z \in \mathbb{R}^C$ is then processed by a two-layer neural network

$$\Phi = \sigma \circ F(4, 64) \circ \text{ReLU} \circ F(64, 4) \ ,$$

the sigmoid activation function is applied at last, in order to squash the attention scores into the interval $[0, 1]$. Channel attention the weights the inputs according to the attention scores

$$H_{CA}(X) = \Phi \circ H_{GP}(X) \cdot X \ . \tag{12}$$

1.3.2 Shifted Window Transfomer Image Restoration

The SWinIR model proposed by Liang et al. [**?**], makes use of the shifted window transfomer architecture introduced by Liu et al. [**?**]. While the model does not employ the hierarchical structure of the original architecture, it makes extensive use of the shifted window mechanism. The model architecture is depicted in figure 9.

The broader architectural design follows that described in section 1.2. Given inputs $X \in \mathbb{R}^{3\times H\times W}$, the shallow feature extraction is performed via a single convolutional layers

$$F_0 = C(3, 180, \text{kernel-size} = 3, \text{padding} = 1)(X) \ .$$

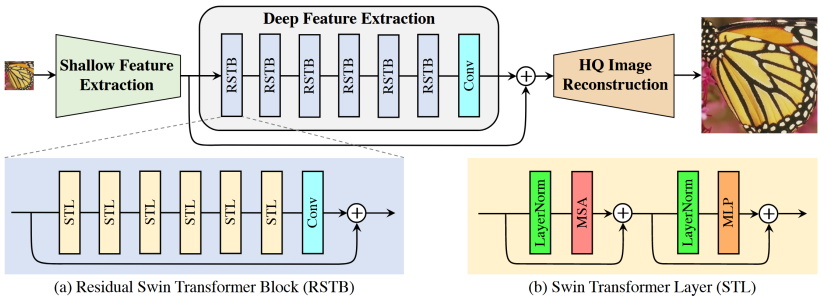

Figure 9: Image taken from [**?**], architecture of SWinIR model.

The features are then further processed by the deep feature extraction module

$$F_1 = H_D(F_0) + F_0 \ ,$$

before being upsampled using the image reconstruction module $I_{SR} = H_{IR}(F_1)$. To this end the authors employ a sub-pixel convolutional layer.

The deep feature extraction module is composed of 6 Residual Swin Transformer Blocks (RSTBs), followed by a last convolutional layer

$$H_D = C(180, 180) \circ H_{RSTB} \circ ... \circ H_{RSTB} \ .$$

Each RSTB is consists of 6 Transformer layers where every second makes use of the shifted window mechanism, or using the notation introduced in section 1.1.3 3 Shifted Window Transformer Blocks SWinT, these are suceeded by a final convolutional layer

$$H_{RSTB} = C(180, 180) \circ \text{SWinT}(180, 6, \Phi, \Phi) \circ ... \circ \text{SWinT}(180, 6, \Phi, \Phi) \ .$$

The network $\Phi : \mathbb{R}^{180} \to \mathbb{R}^{180}$ is given by

$$\Phi = F(360, 180) \circ \text{GELU} \circ F(180, 360) \ . \tag{13}$$

1.3.3 Hybrid Attention Transfomer

The Hybrid Attention Transformer proposed by Chen et al. [**?**], combines the shifted window mechanism as used in SWinIR by Liang et al. [**?**] and Channel Attention [**?**], in a Hybrid Attention Block where both operations are performed in parallel. On top of that they introduce Overlapping Cross-Attention, which we already discussed in section 1.1.3.

The overall architecture is shown in figure 10. The architecture follows the general scheme described in section 1.2, initial feature extraction is performed

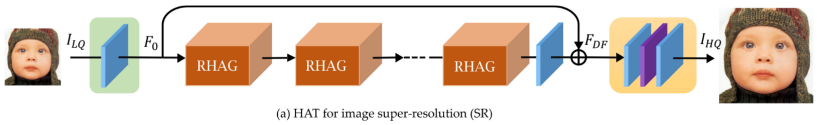

Figure 10: Image taken from [**?**], architecture of HAT model.

via a single convolutional layer, spatial upsampling is achieved using subpixel convolution. The deep feature extraction module proposed by Chen et al. [**?**] is composed of 6 cascaded Residual Hybrid Attention Groups (RHAG)

$$H_D = H_{RHAG} \circ ... \circ H_{RHAG} \ .$$

A RHAG is made up of 6 Hybrid Attention Blocks (HABs) followed by one OCA-Block as defined in 14 and a convolutional layer

$$H_{RHAG} = C(180, 180) \circ \text{OCAB}(180, 6, \Phi) \circ H_{HAB} ... \circ H_{HAB} \ .$$

A HAB is a modification of a Shifted Window Transformer Block, where in parallel to the Shifted Window Multi Headed Attention, Channel Attention Block (CAB) H_{CAB} is employed

$$H_{CAB} = H_{CA} \circ C \circ \text{ReLU} \circ C \ ,$$

with H_{CA} defined as in equation 12. Let $s : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d, s(x, y) = x + y$ be the element-wise summation

$$H_{HAB} = R(\Phi \circ \text{LN}) \circ R(s \circ P(H_{CAB}, \text{SWinMSA}) \circ \text{LN}) \ .$$

The exact architecture of the network Φ is not specified by Chen et al. [**?**]. A construction as in equation 13 can be used. The modules H_{RHAG}, H_{HAB}, OCAB and H_{CAB} are visualized in figure 11.

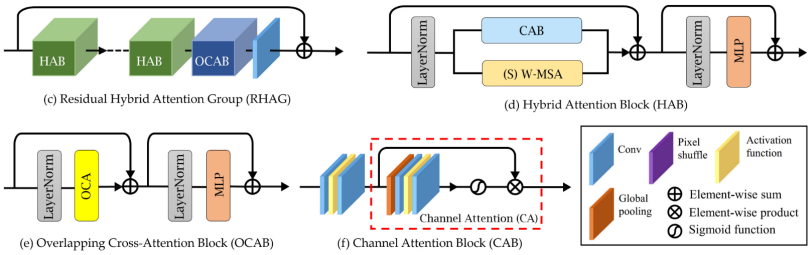

Figure 11: Image taken from [**?**], Residual Hybrid Attention Group.

2 Training

2.1 Preprocessing the Data

2.2 Training Methods

2.2.1 Single Image Super Resolution

2.2.2 Light Field-, Hyperspectral Image- and Spectral Super Resolution Methods

2.2.3 Diffusion Models