Implementation and evaluation of a transformer-based approach for super resolution with image restoration

Contents

1. Introduction

The code for the project will be available in the file sharing repository.
Image restoration techniques, including image super-resolution, denoising, and JPEG compression artifact reduction, aim to enhance the quality of degraded images by reconstructing high-quality versions. Convolutional neural networks (CNNs) have become the go-to approach for image restoration, offering significant improvements over traditional model-based methods. However, CNNs suffer from inherent limitations due to their content-independent convolutional operations and the inability to effectively model long-range dependencies. [5] [15]
In contrast, transformers[23], originally introduced for natural language processing, have demonstrated promising performance in capturing global interactions through self-attention mechanisms. While transformer-based approaches have been successful in various vision problems, applying them to image restoration tasks has presented challenges. Previous methods have divided input images into fixed-size patches and processed them independently, leading to limited interactions among neighboring pixels and introducing border artifacts.
A recent breakthrough, the Swin transformer [17], combines the advantages of CNNs and transformers. It leverages local attention mechanisms to process large-sized images while effectively modeling long-range dependencies using a shifted window scheme. Building upon this concept, we examine a novel image restoration model called SwinIR [15], which consists of shallow feature extraction, deep feature extraction, and high-quality image reconstruction modules.
The shallow feature extraction module captures low-frequency information using a convolution layer, while the deep feature extraction module utilizes residual Swin Transformer blocks for local attention and cross-window interactions. The integration of convolution layers and residual connections enhances the features and facilitates aggregation. Finally, the reconstruction module combines the shallow and deep features to achieve high-quality image reconstruction.
Compared to prevalent CNN-based image restoration models, SwinIR offers several advantages. It enables content-based interactions through attention weights, resembling spatially varying convolution. The shifted window mechanism enables the modeling of long-range dependencies. Additionally, SwinIR achieves superior performance with fewer parameters, as demonstrated by its improved peak signal-to-noise ratio (PSNR) [25] compared to existing image super-resolution methods.
In this paper, we explore the capabilities of SwinIR in image restoration and present experimental results showcasing its effectiveness. By harnessing the power of transformers and incorporating architectural innovations from the Swin transformer, we aim to push the boundaries of image restoration techniques and pave the way for improved image quality in various applications.

2. State of the art

2.1. Why do neural networks work?

A well-known achievement in neural network theory is the discovery that, with certain requirements on the activation function, the class of neural networks possesses great expressive power. This implies that any continuous function defined on a compact set can be approximated to a high degree of accuracy by a multilayer perceptron (MLP). The credit for this theorem goes to Hornik [12] and Cybenko [6], who were the first to demonstrate it.

2.2. What is the advantage of deep neural networks?

Deep neural networks (DNNs) possess the ability to approximate functions with comparable performance to traditional approximation methods. In many problem domains, deep neural networks outperform shallow neural networks. Based on these findings, it is evident that deep networks are preferable to shallow networks. However, it is not immediately clear why classical tools like B-splines cannot be used instead in practical applications. Interestingly, deep neural networks are highly effective in approximating HDF, showcasing their efficiency in this regard [18].

2.3. The importance of high-dimensional functions (HDF) in deep learning.

Increased Representational Capacity
HDF can represent complex and intricate relationships within data. Deep learning models, such as neural networks with multiple layers, can capture and model these intricate relationships, enabling them to learn more nuanced patterns and make more accurate predictions. The higher the dimensionality of the function, the greater its capacity to represent complex data distributions.[10]
Nonlinear Mapping
HDF provide a means to map data from low-dimensional spaces to higher-dimensional spaces. This mapping allows for the transformation of data into a more expressive and separable feature representation, making it easier for the model to learn complex decision boundaries and classify data accurately. Deep learning architectures, with their multiple layers and non-linear activation functions, excel at performing these mappings and leveraging high-dimensional representations.[10]
Feature Extraction and Abstraction
HDF enable deep learning models to automatically learn hierarchical representations of data. Each layer in a deep neural network extracts and learns relevant features at different levels of abstraction. These higher-dimensional representations often capture meaningful and discriminative features, allowing the model to make more informed decisions. By leveraging deep architectures, deep learning models can learn increasingly complex and abstract representations of data.[10]
Robustness to Noisy and Incomplete Data:
HDF can better handle noisy and incomplete data. In high-dimensional spaces, data points are typically more spread out, reducing the impact of individual noisy or missing data points. Deep learning models with HDF can learn to robustly generalize from incomplete or noisy inputs, improving their performance and robustness in real-world scenarios.[10]
Generalization and Transfer Learning
HDF facilitate generalization by capturing underlying patterns and relationships in the data. Deep learning models, with their high-dimensional representations, can learn more generalizable features that are transferable to new, unseen data. This ability to generalize well enables transfer learning, where pre-trained models on large datasets can be fine-tuned on smaller datasets for specific tasks, resulting in improved performance.[10]

2.4. CNNs for Superresolution

Modern algorithms try to tackle the issue of data processing inequality by training a convolutional neural network (CNN) with a set of corresponding low- and high-resolution images, which may contain additional information to recognize and restore details of an input image [21]. The CNN consists of

multiple layers of convolutional, activation, and pooling operations. Skip connections, which simply add the values of tensors of the same shape together, are added to the CNN architecture to enable the flow of information from early layers to later layers. These connections allow the network to capture both low-level details and high-level semantic information. Skip connections help in preserving low-frequency information during the upscaling process. After the image passes through the network, the output goes through a reconstruction process. This can involve reshaping, interpolation, or other techniques to match the desired high-resolution dimensions. The result is an upscaled image that has enhanced visual details compared to the original low-resolution input, as illustrated in Figure 1.
Modern algorithms address the issue of data processing inequality by employing convolutional neural networks (CNNs) trained on corresponding low- and high-resolution images. This training approach allows the CNN to recognize and restore details in an input image [21]. The CNN comprises multiple layers of convolutional, activation, and pooling operations. To mitigate the vanishing gradient problem and facilitate efficient information flow, skip connections are incorporated into the CNN architecture. These connections involve the addition of tensors of the same shape and enable the network to capture both local features and global contextual information. Notably, skip connections play a crucial role in preserving low-frequency information during the upscaling process, retaining important low-frequency details that might otherwise be lost. Once the image passes through the network, the output undergoes a reconstruction process, which may involve reshaping, interpolation, or other techniques to match the desired high-resolution dimensions. Consequently, the resulting upscaled image exhibits enhanced visual details compared to the original low-resolution input, as illustrated in Figure Figure 1.

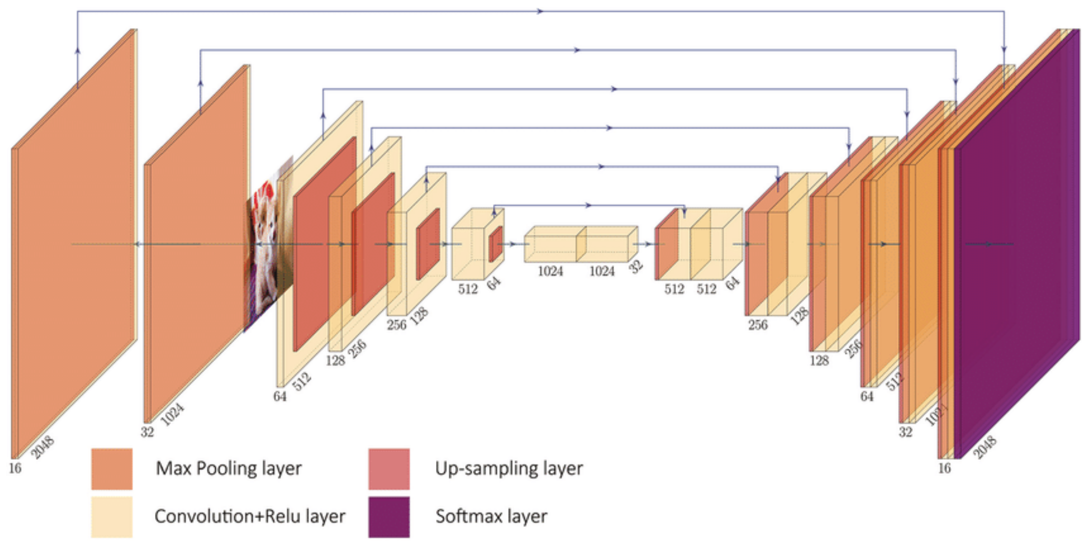


Figure 1: Example of a super-resolution CNN using multiple layers and skip connections.

Source: https://arxiv.org/pdf/1911.09428.pdf

2.4.1. Difficulties of CNNs. Difficulties of CNNs arise in two main aspects, as pointed out by the authors of the SwinIR architecture. Firstly, the interactions between images and convolution kernels in CNNs are content-agnostic. This means that the same convolution kernel is used to restore different image regions, which may not be ideal. Adjusting the weights and biases of a small kernel for the entire image can limit the ability to capture details and nuances in different regions.

Secondly, CNNs rely on the principle of local processing through convolution, which hinders their efficiency in modeling long-range dependencies. Convolutional operations have a limited receptive field, meaning that each output feature is influenced by a small local neighborhood in the input. As a result, capturing long-range dependencies across the entire image can be challenging for traditional CNNs.
The SwinIR architecture addresses these difficulties by introducing novel mechanisms that improve the restoration performance. It incorporates hierarchical patch partitioning and shifted windows to enable capturing fine-grained details in different image regions. Additionally, SwinIR employs a shifted window mechanism that allows information to flow across the image and model long-range dependencies effectively [15].

2.5. Transformers in language processing

To better explain the functionality of vision transformers and the SWIN-approach, this chapter first explains the general idea behind transformers in the field of text processing, in which they were first developed.
The key idea behind transformers is self-attention, which allows the model to weigh the importance of different words, which are encoded into vectors called *word embeddings*, in a sentence when making predictions or generating output. This attention mechanism enables the model to focus on relevant words and understand the context in which they appear. [23]
Transformers process words in parallel rather than sequentially like in RNNs (recurrent neural networks), which allows them to capture long-range dependencies efficiently. They also add *positional encoding* to the *word embeddings* to consider the order or position of words within a sequence. [23]
Transformers in language processing usually also contain encoder- and decoder-blocks. The latter won't be explained further, because the SWIN transformer only uses so called *feed forward transformers* which only consists of encoders. [23]

2.6. Multi-head self attention in language processing

Multi-head attention is a critical component of transformers that enhances their modeling capabilities. It helps the model capture different types of relationships and dependencies within the input sequence. [23]
In simple terms, multi-head attention can be thought of as multiple attention mechanisms working together. Each attention head attends to different parts of the input sequence independently and learns different patterns or relationships. [23]
By using multiple heads, the model can capture various aspects of the input and learn diverse representations. Each head focuses on different parts of the sequence and contributes its unique perspective. The outputs from each head are then combined or concatenated to create a comprehensive representation of the input. [23]

2.7. Vision Transformers (ViT)

[7] ViTs can be used for image reconstruction by leveraging their ability to capture both local and global dependencies in an image. While ViTs were initially designed for tasks like image classification, they can be adapted for image reconstruction tasks such as in painting, super-resolution, or denoising as discussed in [1]. The architecture of ViT consists of the following parts.

Patch-based Representation
ViTs divide an image into smaller fixed patches and treats them as input tokens (Patch embedding). Each patch is then embedded into a high-dimensional vector representation similar to word embeddings. This patch-based representation allows ViT to process images in a sequence-to-sequence manner, where the order of patches encodes spatial information [7].
Positional Embeddings
To provide spatial information to the ViT model, positional embeddings are added to the input patches. These embedding informs the model about the relative positions of the patches, allowing it to understand the image's underlying structure similar to positional embeddings in text processing. [1]
Transformer Encoder
The Transformer encoder is responsible for processing and encoding the input sequence. input patches and extracts high-level features, capturing both local and global information.
Self-attention
The core of the Transformer encoder is the self-attention mechanism, which allows the model to capture dependencies between different elements in the sequence. Self-attention computes weighted sums of the input embeddings, where the weights are determined by the relevance or importance of each element to others in the sequence.[23]
Training with Reconstruction Loss
ViT are trained using a combination of supervised learning and reconstruction loss e.g Charbonnier loss [**liang2021swinir**]. During training, the model is provided with pairs of original and corrupted images. The model's objective is to minimize the difference between the reconstructed image and the original image, thereby learning to restore missing or corrupted parts.

2.7.1. Difficulties of ViT. The biggest drawbacks of Transformer-based architectures when compared to CNN-based methods is the computation complexity. ViTs are computationally more expensive than CNNs. The self-attention mechanism used in ViTs requires computing pairwise interactions between all tokens in the sequence, resulting in a quadratic complexity with respect to the sequence length. In contrast, CNNs leverage shared weight filters and localized receptive fields, resulting in more efficient computations. Furthermore, when it comes to large images, ViT struggles to outperform the CNN-based methods due to memory limitations. When processing high-resolution images, the number of patches increases, leading to higher computational requirements and memory consumption. In contrast, CNNs can handle larger images more efficiently by downsampling and processing them in a hierarchical manner. [7] As an example, having a 256x256px input image with a patch size of 16x16px (resulting in 16x16 non-overlapping patches) would result in $(16x16)^2 = 256^2 = 65536$ attention weights. Increasing the image by only 16px in both dimensions would result in $(17x17)^2 = 289^2 = 83521$ attention weights.

3. The architecture of the SWIN Transformer

To understand what transformers actually do, a deep dive is needed to understand the ways the architecture yields results. Unfortunately, not all the mechanisms are known. We will provide a comprehensive overview of the available literature.
Notation:
$LinearLayer$: responsible for computing linear transformation $y = xA^T + b$
X: local feature (input)
Q: Vector(Linear layer output) related with what we encode(output, it can be output of encoder layer or decoder layer)
K: Vector(Linear layer output) related with what we use as input to output.
V: Learned vector(Linear layer output) as a result of calculations, related with input
P_Q, P_K, P_V are projection matrices are learnable parameters, and they learned by the standard back-propagation algorithm.
The Q, K, V are calculated as followed:$Q = X \cdot P_Q, K = X \cdot P_K, V = X \cdot P_V$
B: is the learnable relative positional encoding

3.1. Scaled Dot Product Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. The input for Scaled Dot Product Attention consists of queries (Q) and keys(K) of dimension d_k, and values of dimension d_v . With the following function, the attention is calculated:

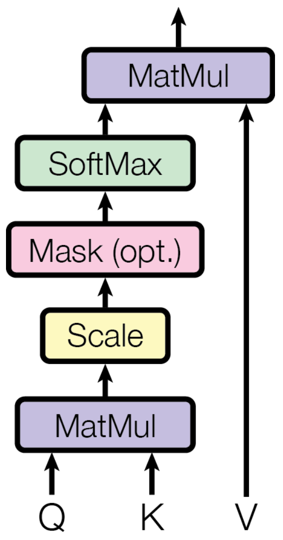
$$Attention(Q, K, V) = softmax(\frac{(Q \cdot K^T)}{\sqrt{d}}) \cdot V$$

3.2. Multi-Head Self-Attention

The concept behind Multi-Head Self-Attention (MSA) is to combine multiple attention functions. Instead of applying a single attention function with keys, values, and queries of d_{model}-dimension, it has been discovered that it is advantageous to linearly project the queries, keys, and values number h times using different learned linear projections, each with dimensions d_k, d_k, and d_v, respectively. Parallel attention functions are then performed on each of these projected versions, resulting in output values of size d_v. These values are concatenated and subjected to another projection to obtain the final values. This process is illustrated in Figure 2.
Multi-head attention allows the model to attend to information from various representation subspaces and different positions simultaneously. Averaging, which occurs with a single attention head, inhibits this joint attention. [23]

3.3. Swin Transformer Layer (STL) Architecture

In general, it consists of a Multi-Head Self-Attention(MSA), Multilayer Perceptron (MLP), Layer Normalization and a few residual connections. This layer extracts intermediate features $F_{i1}, F_{i2}, ..., F_{iL}$. In our



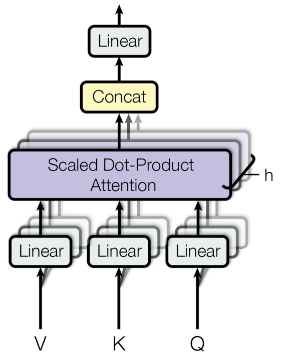


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel

use case for image processing, the output of the encoder is achieved by prepend a learnable embedding to the sequence of embedded patches, whose state at the output of the Transformer encoder serves as the image representation y.

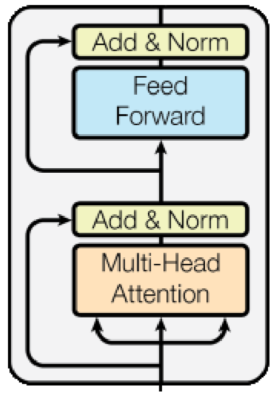


Figure 3: Swin Transformer Layer

3.3.1. Layer Normalization. Layer Normalization (LN) [2] was motivated by Batch normalization (BN) [13]. BN is transposed into layer normalization by computing the mean and variance used for normalization from all of the summed inputs to the neurons in a layer on a single training case. Like BN, LN also gives each neuron its own adaptive bias and gain, which are applied after the normalization but before the non-linearity. Unlike batch normalization, layer normalization performs exactly the same computation at training and test times. Layer normalization is very effective at stabilizing the hidden state dynamics in recurrent networks. Empirically, we show that layer normalization can substantially reduce the training time compared with previously published techniques. The popular belief is that this effectiveness stems from controlling the change of the layers' input distributions during training to reduce the so-called “internal covariate shift”. Current research indicates that it makes the optimization landscape significantly smoother. This smoothness induces a more predictive and stable behavior of the gradients, allowing for faster training [22]. The effect of BN, is that it parametrizes the underlying optimization problem to make it more stable, mitigating avoiding gradient explosion/vanishing [19].

3.3.2. Residual connection. Two residual connections are used in the STB. The residual connections are used to mitigate the degradation problem, also known as the degradation phenomenon or degradation effect, refers to the observation that increasing the depth of a neural network can lead to a decrease in training accuracy.[11]. The author did not provide any theoretical explanation of why residual connections work, he just empirically proves that they work well. The authors of the SwinIR paper speculate that residual connection provide an identity-based connection from different blocks to the reconstruction module, allowing the aggregation of different levels of features.[15]

3.3.3. Feed Forward Multilayer Perceptron (MLP). It is used to help with output alignment. It is thought that the MLP layer helps mitigate the alignment problem known from NLP problems. This problem is very well explained in [14]. The idea for using a dense layer to mitigate this problem comes from [3]. The ongoing effort to analyze the function of the real impact of this layer are inconclusive as of to date. Following are some attempts at understanding MLP [9] and [8].

3.3.4. Remark. The MLP and residual connections make up about $2/3$ of the whole model parameters. Surprisingly, without those components, the model would not work and could not be trained [8].

3.4. Shifted Windows (SW) in the SWIN Transformer

Since the quadratic complexity of ViTs is the result of every single patch calculating an attention for every other patch in the image, the SW-mechanism of the original paper introduces the idea of *patches* and *windows*. The image gets divided into 4x4 pixel blocks called *patches*. Each *window* contains 4x4 patches. The multi-headed self attention is then only calculated between the patches in each window, which reduces the quadratic complexity of ViTs to a linear one. The idea behind the attention patches is to introduce long-range dependencies like a deep CNN, but with less parameters. [15]

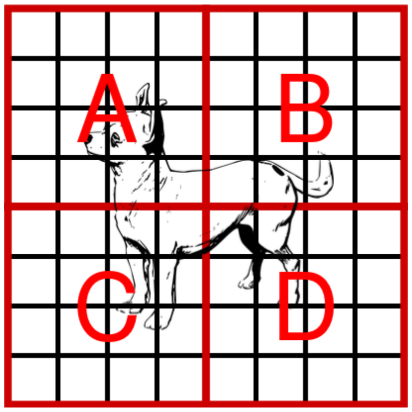


Figure 4: Visualization of a problem of the window-mechanism. A figure lies between the borders of windows.

Since elements in an image can lay between borders as shown in Figure 4 the windows are shifted through the image for each additional layer of the network by half of the patch size. When a part of a patch crosses over the border of the image while being shifted, it *rolls* to the corresponding other side of it, which is called *cyclic shifting*. This method introduces a new problem: some patches that previously didn't border each other now do after a cyclic shift, as shown in Figure 5. The attention between these so-called *orphan patches* should be as tiny as possible, since they contain no information which would be useful regarding the principle of locality.

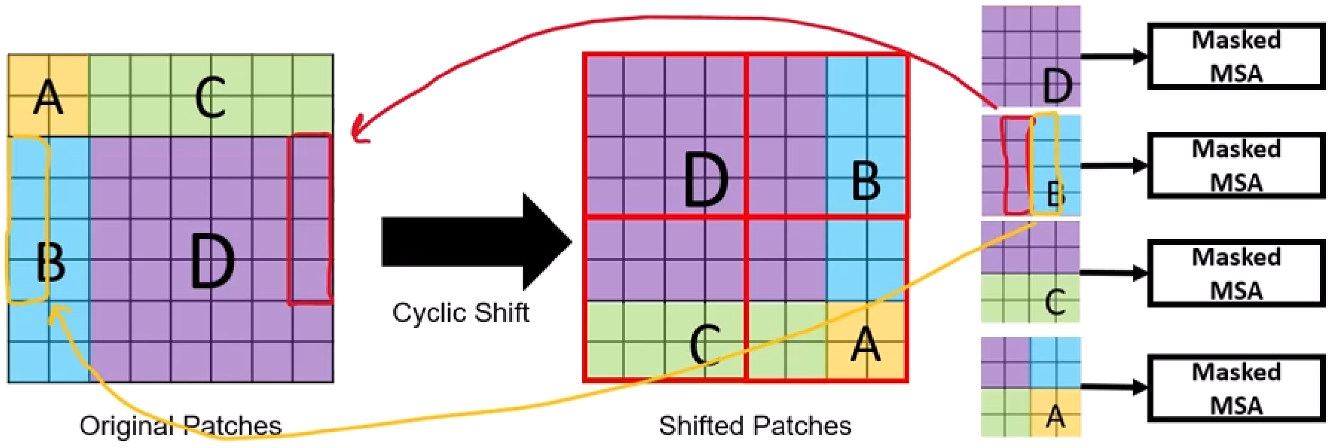


Figure 5: Visualization of the SW-algorithm where the windows are kept in place to better show how the patches change inside the windows. A problem arises when patches that previously didn't border each other now do after a shift-cycle, as illustrated with the yellow and red arrow in window.

Source: https://www.youtube.com/watch?v=qUSPbHE3OeU

3.5. Masked MSA in Shifted Windows

To not calculate attention between orphan-patches in the same window, a so-called attention mask is calculated, which adds an attention mask value to each patch. If the mask-value between two patches is different, its value is set to $mask\_val = -100.0$. If it's the same, it's set to $mask\_val = 0$. The attention mask is a matrix which contains a mask value for every The calculated attention value between two patches. The attention matrix is calculated with the parameter d as a normalization value which is dependent on the number of channels in a tensor, b is the positional encoding and the respective *Query*, *Key* and *Value*-matrices which are all wrapped in a *softmax* function to decrease the influence of small and negative values:

$$Attention(Q, K, V) = softmax(\frac{(Q * K^T)}{\sqrt[2]{d}} + B + attn\_matrix) * V$$

As we see, if 2 patches are not from the same area, the attention value gets decreased drastically due to the negative shift of the attention mask matrix inside the softmax function. [15] The softmax function is defined as:

$$softmax(v)_i = \frac{e^{v_i}}{\sum_{j=1}^{K} e^{v_j}}, \text{for } i = 1, ..., K \text{ and } v = (v_1, ..., v_K) \in \mathbb{R}^K \text{ where } K = dim(v)$$

This operation is only defined for vectors. Pytorch computes softmax row-wise. For higher dimensional tensors, there is an option to decide along what axis the softmax should be computed. Since we apply the function to a matrix, softmax only gets applied row-wise in one dimension. The sum of all scalars of a row is 1. [24]

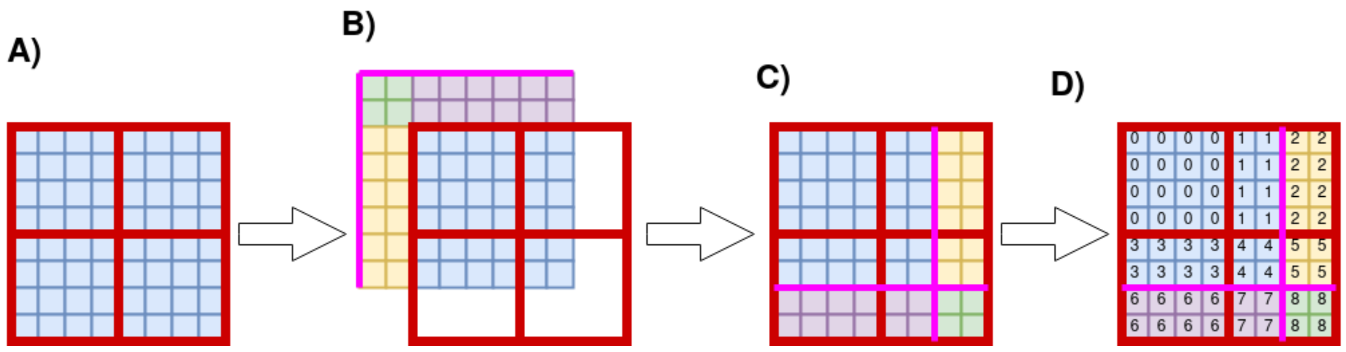


Figure 6: Visualization the calculation of the SW-MSA Mask for 4 Windows with 4x4 Patches from a fixed window perspective. The green, yellow and blue area's are colored to better show their position after the cyclic shift. The pink line represents the *critical border* between patches that did not border previously.

The cyclic shift of the windows by half of their own size always results in 3 critical areas, as illustrated with yellow, green and purple colors in Figure 6, which now have new borders. An algorithm is used to assign one specific value for each area that borders with either a window or the critical border.

3.6. Patch Merging

To enable the model to progressively capture features at larger receptive fields. Multiple SW-MSA blocks are chained together, merging the adjacent patches in each iteration. [17]

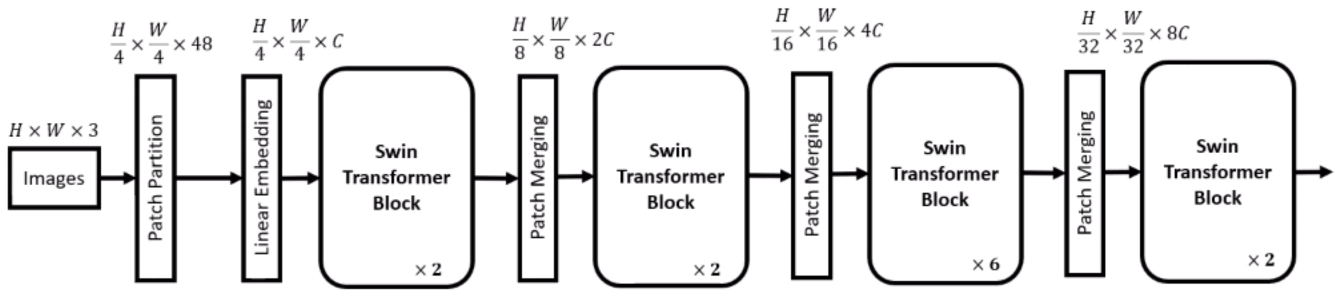

$H \times W \times 3$
Images
$\frac{H}{4} \times \frac{W}{4} \times 48$
Patch Partition
$\frac{H}{4} \times \frac{W}{4} \times C$
Linear Embedding
Swin Transformer Block $\times 2$
$\frac{H}{8} \times \frac{W}{8} \times 2C$
Patch Merging
Swin Transformer Block $\times 2$
$\frac{H}{16} \times \frac{W}{16} \times 4C$
Patch Merging
Swin Transformer Block $\times 6$
$\frac{H}{32} \times \frac{W}{32} \times 8C$
Patch Merging
Swin Transformer Block $\times 2$

Figure 7: Visualisation of the Patch Merging blocks between each SW-MSA Swin Transformer Block. Since an increased Window Size by a factor of 2 in 2 dimensions $(H * W)$ would increase the channel cize by a factor of 4 due to a tensor reshape. A linear embedding is introduces which maps $4C$ to $2C$. [17]

4. Image Restoration using the SwinIR Transformer

The SwinIR transformer supports compressed image super resolution and restoration and consists of three parts: the shallow feature extraction, deep feature extraction and high-quality image reconstruction module as shown in Figure 8. In the next chapter, we aim to explain the idea behind each of these modules [15].

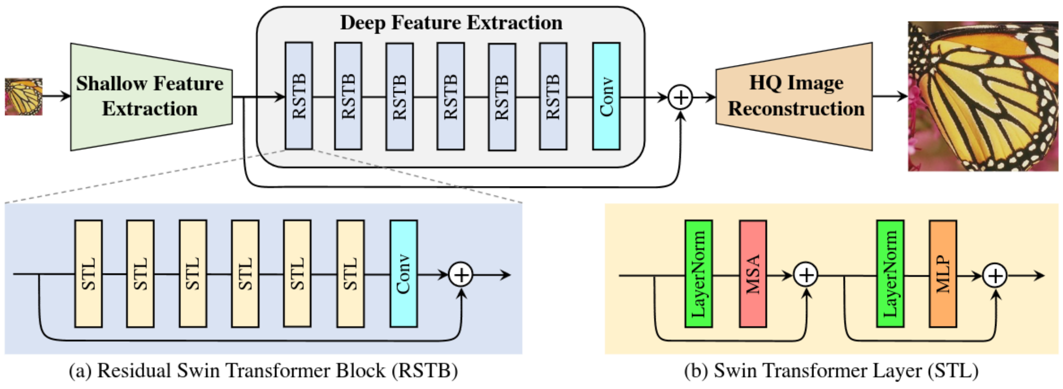


Figure 8: The architecture of the SwinIR transformer

4.1. Image restoration

Image restoration generally consists of two steps, Shallow Feature Extraction and Deep feature Extraction, which involves recovering and enchanting of the features using Swin transformer architecture.

4.1.1. Shallow Feature Extraction. The shallow feature extraction uses a 2D convolutional layer with a $3x3$ kernel, stride of 1 and padding of 1 with input dimension of 3 and output dimension being equal to Patch embedding dimension which is 96 by default. This operation is described as shallow feature extraction to extract the low-frequency information and as described serves as reshaping to the desired RSTB embedding input [15].

4.1.2. Deep feature Extraction.
Residual Swin Transformer Block. The deep feature extraction part chains together so-called Residual Transformer Blocks Figure 8 which contain recurrent STL blocks, which are explained in subsection 3.3. Each RSTB block ends with a convolutional layer. While the Transformer can be seen as a particular example of spatially varying convolution, the use of convolutional layers with spatially invariant filters in SwinIR improves the translational equivariance[15].
By incorporating a convolutional layer at the end of the feature extraction process, the Transformer-based network can benefit from the inductive bias inherent in convolution operations. This inclusion establishes a stronger groundwork for effectively combining shallow and deep features during subsequent aggregation stages [15].
In general, the shape RSTB and STL block do not modify the embedding dimensions. The dimension stay fixed trough out the forwarding. The output of the RSTB can be seen as a representation of the image. The process of forwarding through many STLs is designed to restore the image details. As pointed out in the subsection 3.3 the architecture iteratively modify the features from the previous layer.

4.1.3. Combining Deep features and shallow features. In this step the shallow and deep features are combined using addition to a single tensor of shape

4.2. Super resolution/ High-quality image reconstruction

The combined shallow and deep features are upscaled using the pixel shuffle method. The SwinIR work uses the so-called sub-pixel convolutional neural network to scale up the restored image. The sub-pixel CNN consists of multiple layers that learn the mapping between low-resolution input patches and high-resolution output patches. [20]
The key component of the proposed method is the sub-pixel convolutional layer. This layer is responsible for the crucial step of upscaling the low-resolution feature maps into high-resolution ones. It achieves this by rearranging the pixel values effectively. By learning from the training data, the network understands the relationships between low-resolution and high-resolution image patches and utilizes this knowledge to generate detailed and sharper representations.[20]
During the training phase, the model is trained on a dataset consisting of pairs of low-resolution and corresponding high-resolution images or videos. The network learns to minimize the discrepancy between its predicted high-resolution outputs and the ground truth high-resolution images.[20]
Once trained, the model can be applied to new, unseen low-resolution images or videos. By passing the low-resolution inputs through the sub-pixel CNN, the network generates high-resolution outputs that exhibit enhanced levels of detail and quality. This process is computationally efficient, enabling real-time super-resolution.[20] using [20]
For tasks that do not need upsampling, such as image denoising and JPEG compression artifact reduction, a single convolution layer is used for reconstruction [15].

4.3. Training and loss function

The SwinIR model implements several loss functions which are use-case dependent. In the most cases, a L1-pixel-loss $L = ||I_{RHQ} - I_{HQ}||$ is used. The 2 exceptions are the real-world-SR use-case, which uses a combination of pixel loss, GAN loss and perceptual loss to improve visual quality and the denoising and JPEG compression artifact reduction use-case, which uses the Charbonnier loss [4] $L = ||I_{RHQ} - I_{HQ}||^2 + \epsilon^2$ where ϵ is a constant that is empirically set to 10^{-3} [15]. The ϵ is used so that the Charbonnier penalty (the loss function) does not become zero. The window size is set to 7 specifically for JPEG compression artifact reduction because there is a substantial decrease in performance when using a window size of 8. This drop in performance is likely attributed to the fact that JPEG encoding employs an 8 × 8 image partitioning scheme. The used network model was trained on DIV2K and Flickr2K datasets. [15]

4.4. Later improvements with the SwinV2 Transformer

The SwinV2 architecture enhances the shifted window self-attention module to improve the model's capacity and window resolution. By employing post normalization instead of pre-normalization, it reduces the average feature variance in deeper layers and enhances numerical stability during training. As a result, the SwinV2 Transformer can be scaled up to 3 billion parameters without encountering training instabilities. Additionally, by utilizing scaled cosine attention rather than the dot product between queries and keys, it mitigates the dominance of certain attention heads for specific pixel pairs. In certain tasks, the Swin2SR model achieved comparable outcomes to SwinIR while undergoing 33% fewer training iterations. Lastly, the utilization of log-spaced continuous relative position bias enables generalization to higher input resolution during inference.[5]

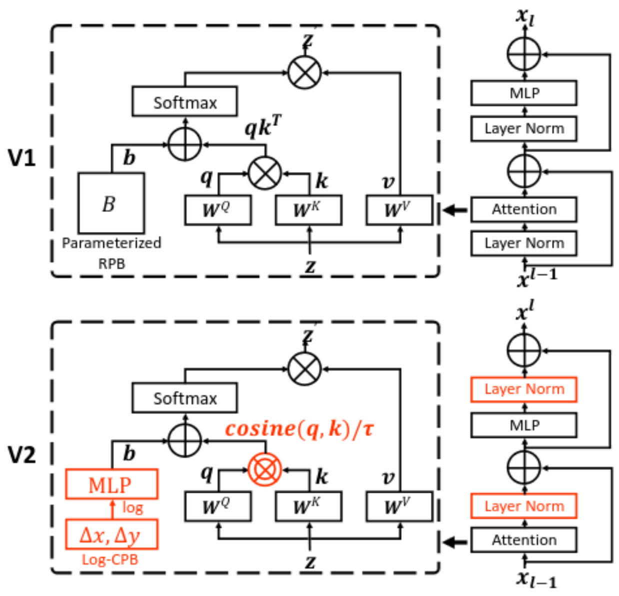


Figure 9: Improvements of the SwinV2 attention mechanism [16] compared to the original Swin transformer [17]

5. Results

Big part of this work was explaining the way transformer works. After researching the used components and the architecture of the transformer that majority of parameters in the model, $> 2/3$ mainly in the MLP layers and residual connections are still not explained. The funny thing is that the model performs better with those parts in-place.
Additionally, the applied model to video quality restoration and super resolution has shown great success. We adapted the code and the model to work with video instead of the image.
To evaluate the performance and applicability of the proposed SwinIR model, we conducted comprehensive testing on videos rather than individual frames. This approach allowed us to assess the model's ability to handle temporal information and maintain consistency throughout the video sequences.
During the testing phase, we applied the SwinIR model to a sample video that was a short anime movie. Our primary focus was to examine the effectiveness of the model in restoring high-quality images from low-resolution compressed jpeg frames while preserving temporal coherence and ensuring the absence of noticeable jitters.
Remarkably, the results obtained from our experiments demonstrate the model's exceptional performance on video restoration. The SwinIR model successfully reconstructed high-quality frames across the entire video sequences, maintaining consistency and temporal coherence throughout. Notably, we observed no visible jitters or artifacts that would compromise the overall visual experience.
By leveraging the strengths of the Swin Transformer architecture and incorporating multi-head self-attention mechanisms, SwinIR effectively captures and models temporal dependencies within videos. The ability to handle video inputs further highlights the robustness and versatility of the SwinIR model in real-world scenarios. Thus, the approach is as good as state-of-the-art CNN-based image restoration techniques.

The authors of the SwinIR paper observed that SwinIR consistently outperformed the other methods, exhibiting average PSNR gains of at least 0.11dB and 0.07dB on benchmark datasets [15].
Notably, SwinIR achieved these superior results while maintaining a significantly lower parameter count. Compared to the previous best-performing model, DRUNet, SwinIR demonstrated a remarkable reduction in the number of parameters, with only 11.5M parameters compared to DRUNet's 32.7M parameters.
Overall, the testing results validate the capability of our SwinIR model to effectively restore videos by leveraging the power of deep learning and the advanced features of the Swin Transformer architecture. The absence of noticeable jitters and the consistent visual quality throughout the video sequences reinforce the potential of SwinIR in real-time video restoration applications.

6. Conclusion

In this work, we have explored the implementation and evaluation of a deep learning-based approach for image restoration, specifically focusing on super-resolution with image restoration. Through our discussions, we have covered various aspects related to the state-of-the-art techniques, including the advantages of neural networks, the importance of high-dimensional functions in deep learning, the use of CNNs and Transformers in language processing, and the emergence of Multi-Head Self-Attention (MSA).
We have discussed the architecture of the SWIN Transformer, highlighting key components such as Scaled Dot Product Attention, Multi-Head Self-Attention, Swin Transformer Layer (STL) Architecture, Shifted Windows (SW), and Masked MSA in Shifted Windows. Additionally, we have delved into image restoration using the SWIN2SR Transformer, covering topics such as improvements in SwinV2 Transformer, the overall architecture, and the image restoration process involving shallow feature extraction, deep feature extraction with Swin, combining deep features and shallow features, and super-resolution.
Through our analysis and evaluations, we have observed the potential of Swin Transformer as an effective solution for image restoration tasks. The integration of CNN and Transformer in the Swin Transformer architecture combines the advantages of both approaches, enabling local attention for large-sized images and effective modeling of long-range dependencies with the shifted window scheme.
Looking ahead, we anticipate further advancements in image restoration by leveraging the strengths of SwinIR and exploring its applicability to other restoration tasks such as image deblurring and deraining. The ongoing development and refinement of deep learning-based approaches hold great promise for enhancing image quality and addressing real-world challenges in diverse scenarios.

References

[1] Anas M Ali, Bilel Benjdira, Anis Koubaa, Walid El-Shafai, Zahid Khan, and Wadii Boulila. "Vision transformers in image restoration: A survey". In: *Sensors* 23.5 (2023), p. 2385.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization". In: *arXiv preprint arXiv:1607.06450* (2016).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[4] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. "Two deterministic half-quadratic regularization algorithms for computed imaging". In: *Proceedings of 1st international conference on image processing*. Vol. 2. IEEE. 1994, pp. 168–172.

[5] Marcos V Conde, Maxime Burchi Ui-Jin Choi, and Radu Timofte. "Swin2SR: SwinV2 Transformer for Compressed Image Super-Resolution and Restoration". In: 2022.

[6] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[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". In: *arXiv preprint arXiv:2010.11929* (2020).

[8] Nelson Elhage. *A Mathematical Framework for Transformer Circuits*. 2021. URL: https://transformer-circuits.pub/2021/framework/index.html (visited on 06/22/2023).

[9] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. "Transformer feed-forward layers are key-value memories". In: *arXiv preprint arXiv:2012.14913* (2020).

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[12] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural networks* 2.5 (1989), pp. 359–366.

[13] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

[14] Philipp Koehn. "IBM Model 1 and the EM Algortihm". In: *Machine Translation, dated* (2017), pp. 1–46.

[15] 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*. 2021, pp. 1833–1844.

[16] Ze Liu, Yutong Lin Han Hu, Zhenda Xie Zhuliang Yao, Jia Ning Yixuan Wei, Zheng Zhang Yue Cao, Furu Wei Li Dong, and Baining Guo. "Swin Transformer V2: Scaling Up Capacity and Resolution". In: 2021.

[17] Ze Liu, Yue Cao Yutong Lin, Yixuan Wei Han Hu, Stephen Lin Zheng Zhang, and Baining Guo. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: 2021.

[18] Philipp Christian Petersen. "Neural network theory". In: *University of Vienna* (2020).

[19] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. "How does batch normalization help optimization?" In: *Advances in neural information processing systems* 31 (2018).

[20] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan 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*. 2016, pp. 1874–1883.

[21] Aarti Singh. “Lecture 2: Gibb’s, Data Processing and Fano’s Inequalities”. In: 2012, pp. 1–6.

[22] Harshveer Singh. *The real reason why BatchNorm works*. 2020. URL: https://towardsdatascience.com/why-batchnorm-works-518bb004bc58 (visited on 06/22/2023).

[23] 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* 30 (2017).

[24] *Wikipedia - Softmax function*. https://en.wikipedia.org/wiki/Softmax_function. Accessed: 2023-06-27.

[25] Stefan Winkler and Praveen Mohandas. “The evolution of video quality measurement: From PSNR to hybrid metrics”. In: *IEEE transactions on Broadcasting* 54.3 (2008), pp. 660–668.