Universiteit Leiden

Master Computer Science

Res-Swin: Effective Combination of ResNet and Swin Transformer for LDCT Denoising

Name:	Donghang Lyu
Student ID:	s3084361
Date:	22/02/2023
Specialisation:	Data Science
1st supervisor:	Daan Pelt
2nd supervisor:	Lu Cao

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

With the rapid development of transformer modules, Vision Transformer (ViT) is replacing CNN models to become the main trend in the computer vision tasks. However, in the field of low-dose computed tomography (LDCT) denoising, CNN models still remain dominant and there are only a few works using transformer modules for LDCT denoising. The main reasons are related to the relatively large size of CT images and the high parameter volume of transformer models. Considering the situation that more and more efficient transformer models are proposed and the CNN module and transformer module can complement each other to a certain extent due to their working principles, I will introduce a new network model called Res-Swin in this work, which aims for combining the main characteristics of ResNet and Swin Transformer effectively to improve the denoising performance with the input siez of (512, 512). I design three kinds of structures to make experiments. Based on the results on three datasets, all of them can work better than the total CNN models under similar parameter volume. In addition, some of Res-Swin models can outperform over the TransUNet, which is composed of CNN module and transformer module as well. In general, the effectiveness of Res-Swin models can be reflected well in this work.

Contents

Chapter 1

Introduction

1.1 Problem Statement

Computed tomography (CT) refers to a computerized X-ray imaging procedure in which a narrow beam of X-rays is aimed at a patient and quickly rotated around the body, producing signals that are processed by the machine's computer to generate cross-sectional images, or "slices". When compared to standard X-rays, these slices, known as tomographic pictures, can provide more detailed information. The computer of the device will gather several successive slices and "stack" them to create a three-dimensional image of the patient. This will make it simpler to identify the patient's basic structures and some potential cancers.

Through the output of CT, radiologists and physicians will inspect the overall health of people's internal structures. Likewise, they can use these images to get relevant data surrounding your health, including the location, extent, and severity of any maladies you may have, like a blockage in your heart. Furthermore, CT scans can help improve the success and accuracy of health care, reduce the need for exploratory surgeries and the risk of complications in surgery, and so much more[1].

Although there are many bene ts to the application of CT scans, it does cause some concern for some patients when they are exposed to a radioactive environment for varying durations. The use of CT scans has increased the amount of radiation that patients receive during medical operations, a person might need to take about 3.3 years to get the same amount of background radiation that an abdominal CT delivers in less than a minute [1]. Therefore, reducing the dose of X-rays in CT scans would be a possible solution based on the above situation. However, it will result in certain clear issues including increased noise, decreased contrast in sharp features, and over-smoothing of images. Furthermore, CT scans can also meet the problem of artifacts, which can be caused by multiple factors like patient properties and scanners. So it is essential to implement a denoising technique on low-dose CT (LDCT) images to improve the image quality and help physicians to diagnose potential diseases better.

LDCT denoising is one kind of speci c image denoising and belongs to medical image processing. Due to the e ects of imaging equipment and the outside environment, various types of noise are generated throughout the digitization and transmission process, signi cantly lowering

[1] https://www.radiologyinfo.org/en/info/bodyct

the quality of the photos. Image denoising should therefore concentrate on removing as much noise as possible while preserving the original image's intricate structural details, such as edges and textures.

Arti cial intelligence has gained popularity in recent years and is now used extensively across many industries. It can assist people in nding e cient solutions to challenges. The two key phases in the evolution of neural network methods in the many branches of AI are convolutional neural network (CNN) models and transformer models. CNN uses the combination of convolution and nonlinear activation functions to perform feature extraction. Convolution is the inner product of the input data and the lter. A convolutional kernel is a set of neurons with xed weights in the matrix format. It extracts features through the convolution calculation, and the kernel continues shifting until walking through the whole image. In this way, local important features can be captured e ectively and the receptive eld can also be expanded gradually through the stacking of layers. Furthermore, the nonlinear activation function is introduced to increase the nonlinear tting capability of the model. Numerous well-established CNN models have been proposed, some representative ones, including AlexNet [2], VGGNet [3] and ResNet [4], can have a pretty good performance on image classi cation and serve as the foundation for many following models to carry out other related tasks like object detection and semantic segmentation.

Transformer models [5] are widely used in the natural language processing (NLP) eld. A transformer model is composed of multiple encoders and decoders, which implement attention mechanisms and feed-forward layers. Through this structure, a transformer model can capture the overall information very well [5]. After that, proposing of Vision Transformer (ViT) [6] provides a new way for computer vision tasks. One image is divided into several patches by ViT, and each patch is transformed into an embedding vector before being used as an input for the transformer model. The bene ts of the transformer model are introduced in this way, meaning that each patch through ViT can contain information from other patches due to the implementation of attention to cover the global information. According to the experiments, ViT can have a competitive performance on image classi cation compared to ResNet and even outperform over ResNet when data volume is large enough, which re ects the e ectiveness of ViT and broad prospects for applications in the computer vision eld.

As mentioned before, the working principle of CNN models is based on the shifting of the convolutional kernel and local features would be extracted through the convolution calculation. In this process, the parameter values of the convolutional kernel remain unchanged and they are used for feature extraction at all locations of the image, which means the parameter volume of the CNN model would be relatively small. But in the early stages of the CNN models, the receptive eld is limited due to the small size of the convolutional kernel, and the CNN models have to stack more and more layers to expand the overall receptive eld. So it is obvious that the front part of the CNN models always lack global information in this way. While for the transformer models, each layer of the transformer models contains the implementation of attention, which means each patch can contain the information from other patches with calculated attention weight, and the receptive eld of the transfomer models can be global in this way. However, lots of important local information might be neglected. Furthermore, big parameter volume is a clear drawback to the transformer models, because the attention map needs to be calculated by three matrixes, key, query, and value matrix. Their implementation

all rely on linear layers, which can lead to high parameter volume. Therefore, it is clear to see that CNN and transformer models can complement each other very well.

As a sub-task of computer vision, the development of image denoising also goes through the aforementioned procedures that involve switching from CNN models to transformer models. According to the above analysis of CNN models and transformer models, CNN models and transformer models can complement one another in terms of parameter volume and receptive field, which means the combination of them could be also a good way for image denoising. Although there are some proposing models consisting of transformer blocks and CNN blocks, such as SwinIR [7] and Swin-Conv-UNet [8], the distribution of transformer blocks and CNN blocks is uneven, which means the ratio of CNN blocks is much lower than transformer blocks. Therefore, these models will be kind of like total transformer models and CNN part might only have little impacts. Moreover, the size of CT images is ranging from 512 to 2000, which is much bigger than the size of normal images. Therefore, considering the large input size and relatively high parameter volume of the transformer block, the above models will lead to high GPU memory costs and are not suitable model structure for LDCT denoising.

Based on the above analysis, although transformer models have been widely used in many fields, also including normal image denoising, their development in LDCT denoising is still relatively slow. Some traditional CNN models like RED-CNN [9], WGAN-VGG [10] are still mainstream to be used for processing large CT images. Of course, there are also some models containing transformer modules being proposed for LDCT denoising [11, 12], but their experiments are based on the input size of 6464, which might not be suitable for the actual application. In conclusion, it is necessary to find a better way to combine the CNN module with the transformer module for improving the performance of LDCT denoising further at the same time of controlling resource costs.

1.2 Research Goal

Considering the multiple factors, including the size of LDCT images, computation resources, and characteristics of CNN models and transformer models, this thesis will concentrate in investigating the combination of CNN module and transformer module to improve the denoising performance of LDCT images. In actual use, the network model ought to be more denoising-efficient than previous neural networks and strike a fair compromise between GPU memory usage and denoising performance.

Similar to semantic segmentation, image denoising often uses an encoder-decoder structure. The encoder can explore the key features of an image and project them to the latent space, while the decoder will work to recover the original, noise-free image from the encoder's output. Then as a well-developed encoder-decoder structure, UNet [13] is frequently utilized in the fields of segmentation and denoising. As a result, it would be regarded as the model's general structure, and this thesis will concentrate on the encoder component by being substituted by the created network model. Furthermore, the model would be trained in a supervised way.

As mentioned before, CNN models and transformer models have developed very well and many models are proposed to optimize constantly. For the CNN module, I would like to

choose ResNet, which performs well in many tasks of computer vision through the introduction of residue operation and are used as backbone for many other models. While Swin Transformer [14] can be a good choice for the transformer part, because it can make a good balance between parameter volume and performance. Therefore, this thesis aims to propose a new network model called Res-Swin, which depends on the characteristics of ResNet and Swin Transformer to combine them e ectively. In the factual implementation, it will work as the encoder part of the overall structure. Furthermore, I mainly design three kinds of network structures to combine them, which can be shown as follows:

- Res-Swin-SG: Inspired by the overall structure of ResNet, it is a single branch structure. In each stage, a Swin Transformer module and a mixed CNN and DCNN module will be combined together.
- Res-Swin-DB: Considering the range of the receptive eld of CNN model and transformer model, the overall structure is composed of two branches. One branch is ResNet, while another branch is made up of Swin Transformer module and total dilated convolution [15] module . The output of two branches of each stage will combine to form the input of the corresponding decoder part.
- Res-Swin-DBM: This model structure builds on the structure of Res-Swin-DB. The overall structure still consists of two branches but adding a mixture block containing the attention mechanism at the end of each stage to make two branches complement and interact with each other better.

1.3 Thesis Overview

The rest of this thesis can be organized as follows: Chapter 2 focuses on some related works about original image denoising and LDCT denoising with CNN and transformer models, also, some brief descriptions of the used models are included. In Chapter 3, I will show more details about the structure of three Res-Swin models. After that, the pipeline of experiments, including the dataset, evaluation measures, some parameter settings, and getting results of models, as well as some discussions of the results, will be contained in Chapter 4. Finally, Chapter 5 is a conclusion to this thesis.

Chapter 2

Background & Related work

In this chapter, an overview will be given about the application of deep learning to image denoising, especially focusing on the LDCT denoising eld. Both of them can split models into two categories: the CNN models and the transformer models. After that, some models used in the Res-Swin models will be covered in more detail.

2.1 Related Work

2.1.1 Image Denoising

CNNs have been applied in the eld of image denoising [16] since 2009. However, due to the resource limitation and many related techniques still not being exposed, CNN models cannot achieve state-of-the-art denoising performance [16, 17]. With the rapid development of neural networks, denoising performance by CNN models is also improved gradually. DnCNN is proposed by Zhang et al.[18], this model is a simple (Convolution, Batch-Normaliztion, ReLU) cascade structure and the output is a residual image that represents the noise level, so the denoising image can be gotten by making subtraction between the original input image and output residue image. This model can outperform traditional denoising methods well and open up ideas for utilizing CNN structure to denoise. Mao et al [19] introduce skip connection to Fully Convolutional Network (FCN) [20] for doing denoising, this is also the rst method that implements the denoising task with di erent noise levels using a single model and achieves good performance. Santhanam et al. [21] propose a recursively branched deconvolutional network (RBDN), where pooling/unpooling operations are used to aggregate multi-context information. Pengju et al. [22] propose MVCNN, which introduces a discrete wavelet transformer (DWT) to replace pooling operation and get frequency information to save detailed texture. In this way, the model can also make a good balance between receptive eld and computation e ciency. In addition, Chunwei et al. [23] propose ADNet, which combines dilated convolution with common convolution to remove noise e ciently, and also introduces the attention mechanism for extracting noise from the hidden background.

As one of the most successful and widely applied models in the NLP eld, transformer [5] is also becoming a hot topic in the computer vision eld. ViT makes it feasible to be applied in the computer vision tasks, and it also achieves competitive performance on multiple sub-tasks of computer vision compared to some developed CNN models like ResNet. Due to some distinct drawbacks of ViT model, more and more transformer models like Pyramid Vision Transformer

(PVT) [24], Swin Transformer [14] and Data-e cient Image Transformer (DeiT) [25], are proposed for optimizing ViT further from di erent aspects, mainly about reducing parameter volume, optimizing the attention mechanism. In the eld of normal image denoising, there are also some transformer-based models being proposed recently. SUNet [26] combines Swin Transformer with CNN blocks based on the structure of UNet to do denoising. SwinIR [7] also utilizes both transformer blocks and CNN blocks, the shallow feature extraction is composed of convolutional layers, while deep feature extraction consists of RSTB (Residue Swin Transformer blocks) that uses Swin Transformer block and skip connection. Besides, Uformer [27] makes both the encoder part and decoder part composed of LeWin (Local-enhanced Window) Transformer block, which can reduce computation through a non-overlapping window attention mechanism. And it also uses a skip connection to deliver information from the encoder part to the decoder part.

2.1.2 LDCT Denoising

In the eld of LDCT denoising, there are many CNN models and can be good choices due to the good balance between recourse costs and denoising performance. First of all, Chen et al. [9] nd that a simple CNN can be used for suppressing the noise of LDCT images, which is a pioneering work. After that, models in [9, 28] show that the encoder-decoder network is an e cient structure for LDCT denoising. RED-CNN [9] introduces a shortcut of residue network which can get pretty good performance. WGAN-VGG [10] uses a generative adversarial network (GAN) with Wasserstein distance and perceptual similarity. This model transfer knowledge of visual perception to LDCT denoising, which saves critical image features at the same time of reducing noise level. Then EDCNN [29] designs an edge enhance module based on the novel trainable Sobel convolution, besides, the overall model structure is full of dense connections to fuse the extracted edge information.

The application of the transformer model on LDCT denoising is still in the development stage and the number of models is relatively small. Dayang et al. [11] propose TED-net, which is convolution-free and utilizes unfold operation to replace convolution and extract tokens from image patches. TED-net combines a typical transformer block with a Token-to-Token block to get noise level image. Eformer [12] is a kind of edge enhancement-based transformer model, it incorporates learnable Sobel-Felman operators to enhance edge features and concatenate them in the intermediate layers, which can also achieve good results. TransCT [30] decomposes LDCT image into two parts, low-frequency (LF) and high-frequency(HF), each of them goes through some CNN layers to get feature embedding, which contains some LF and HF information, respectively, and their outputs will be put into a typical transformer model to get the input of decoder part.

2.2 Models Description

Due to the close relationship with the planned network model, several models will be introduced here in more depth. The overall structure of Res-Swin uses the structure of UNet as a reference, which is the encoder-decoder structure. As mentioned before, Res-Swin will combine ResNet with the Swin Transformer module in general as the encoder part. In this way, it aims to make these two modules interact more and complement each other to improve together.

2.2.1 UNet

UNet is proposed for medical image segmentation and is composed of two parts, encoder and decoder. The encoder consists of four stages, with each stage having two 3×3 unpadded convolutions as the first step, followed by a rectified linear unit (ReLU), a 2×2 max pooling layer with stride 2, while the channel number will be doubled at the same time. There are four stages in the decoder section that correspond to those in the encoder part. Each stage of the expanding path passes via an upsampling layer, which cuts the channel number in half, before being concatenated with the appropriate encoder output in the channel direction. Through this way, more features will be fused to reduce the possibility of important image features being lost. Finally, two 3×3 convolutions are followed by a ReLU activation function for each. A 1×1 convolution is applied as the last layer to map to the appropriate output channel number. In the task of LDCT denoising, the output channel number is just the channel number of the original image, which is always 1. Furthermore, in the actual implementation, choices of encoder part of UNet can be diverse, including simple CNN model, or some mature models like VGG and ResNet.

2.2.2 ResNet

Before the proposing of ResNet, VGG had previously run into the issue where performance would suffer from stacking too many layers, limiting the maximum number of layers that could be used in a CNN model. ResNet then addresses this issue with the introduction of shortcut operation. It implies that there would be direct connections that omit intermediate levels. In general, the output term can be represented as follows

$$H(x) = f(wx + b) \tag{2.1}$$

where f means activation function, w is weight, b is bias, x means input feature map and $H(x)$ is the output feature map. With the introduction of shortcut operation, $H(x)$ can be as follows:

$$H(x) = f(wx + b) + x \tag{2.2}$$

When the dimensions of the input are different from that of the output, some 1×1 convolutional layers can be added into the shortcut to solve it. In such a case, the representation of $H(x)$ can be changed as follows:

$$H(x) = f(wx + b) + f_1(w_1 x + b_1) \tag{2.3}$$

where f_1 part represents the 1×1 convolution operation. In this way, the gradient vanishing issue can be effectively resolved by the shortcut operation since it ensures that the higher layer will perform at least as well as the lower layer, and more layers can be stacked without any problems.

Without the limitation of layer number, there are multiple variants of ResNet based on different layer numbers and parameter settings, mainly including ResNet18, ResNet34, ResNet50, and ResNet101. The number in the name denotes the total number of layers. In fact, ResNet could theoretically have a lot more layers, but that will also require much more GPU resources. Then Table 2.1 shows some details about the composition of ResNet variants when the input size of the image is (224, 224).

Table 2.1: Structures of various ResNet models

stages	output size	18-layer	34-layer	50-layer	101-layer
stage-1	112x112	7x7, 64, stride 2	7x7, 64, stride 2	7x7, 64, stride 2	7x7, 64, stride 2
stage-2	56x56	2x2 max pool $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}$ x2	2x2 max pool $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}$ x3	2x2 max pool $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$ x3	2x2 max pool $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$ x3
stage-3	28x28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}$ x2	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}$ x4	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}$ x4	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}$ x4
stage-4	14x14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}$ x2	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}$ x6	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}$ x6	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}$ x23
stage-5	7x7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}$ x2	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}$ x3	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}$ x3	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}$ x3

2.2.3 Swin Transformer

Before giving a detailed description of Swin Transformer, there are some recaps about ViT. First of all, ViT's proposal makes it feasible to implement transformer model for the computer vision tasks, and ViT can get competitive performance compared to the well-known CNN model ResNet. ViT also lays the groundwork for the proposal of numerous alternative transformer-based models to process vision tasks. However, ViT still has some drawbacks. For example, memory cost is high, because ViT's computational complexity scales quadratically with image size. Furthermore, ViT needs a larger data volume to train, because induction bias of self-attention is weaker than CNN, which means CNN can make some assumptions for data it doesn't encounter, while self-attention cannot.

According to the above drawbacks of ViT, Swin Transformer is proposed for optimizing the performance of ViT further. Swin Transformer is built around two key ideas: shifted window attention and hierarchical feature maps. Besides, similar to ResNet, Swin Transformer also includes numerous variations based on the various channel numbers, layer numbers, and head numbers of attention. More information is shown in Table 2.2 and the model size is larger from up to down.

Table 2.2: Details about variants of Swin Transformer

	channel numbers	layer numbers	head numbers
Swin-T	(96, 192, 384, 768)	(2, 2, 6, 2)	(3, 6, 12, 24)
Swin-S	(96, 192, 384, 768)	(2, 2, 18, 2)	(3, 6, 12, 24)
Swin-B	(128, 256, 512, 1024)	(2, 2, 18, 2)	(4, 8, 16, 32)
Swin-L	(192, 384, 768, 1536)	(2, 2, 18, 2)	(6, 12, 24, 48)

The overall design of Swin Transformer can be illustrated in Figure 2.1. The "Patch Merging" block and the "Swin Transformer" block make up the majority of each stage block. For the patch merging block, it is a kind of convolution-free downsampling technique. When a feature map is downsampled by a factor of n, n neighboring patches would concatenate in the chan-

Figure 2.1: The overall structure of Swin-T and output shape change of each stage with the input size of (224, 224). It is composed of 4 stages, and the overall structure is kind of hierarchical, which means the output shape of each step would gradually be cut in half as the network grows deeper.

nel direction, which means the shape of $H \times W \times C$ would turn into $(H/n) \times (W/n) \times (n^2 \times C)$, where H, W and C refer to the height, width and channel of the feature map, respectively.

In the Swin Transformer block, it is composed of a Window MSA (W-MSA) module and a Shifted Window MSA (SW-MSA) module, MSA is short for multi-head self-attention. In the original ViT, each patch needs to participate in the calculation for the standard MSA, which can result in heavy computation and quadratic complexity for the number of patches. For the W-MSA of Swin Transformer, it can highly reduce the computation and complexity by calculating attention within each window. A window means a collection of patches. However, there is still a distinct restriction for W-MSA, which means no interaction between windows, and the modeling power of the network would be limited. Therefore, the following SW-MSA aims to alleviate this restriction. To introduce cross-window connections, SW-MSA shifts the windows to the bottom right corner by a factor of $M/2$, M is the size of the window. But after moving, the original belonging relationship between the patches and the window is disrupted, leaving some patches orphaned. According to this issue, SW-MSA applies the "Cyclic Shift" technique to make orphaned patches move to the incomplete windows to make up a new complete window. After that, it is obvious that the new complete window might consist of patches that are not adjacent to the original feature map, so some certain masks need to be added to prevent this and pose no issues for the calculation of attention.

Chapter 3

Models

3.1 Single Branch Structure - Res-Swin-SG

As previously stated, the purpose of this thesis is to explore the e ective combination of the CNN module and transformer module to get better denoising performance with appropriate resource costs. At the beginning stage of the model, I aim to introduce the global features gotten from the transformer part to promote the CNN part, while local important features from the CNN part can also enrich the information of the transformer part at the same time of balancing the overall parameter volume. For the CNN part, I choose to use the ResNet structure, which is an established and successful structure and also has been validated in many applications. Furthermore, it is served as an inspiration for the proposing of the model rstly, which means trying replacing some modules of ResNet. Based on the observation of Table 2.1, we can see that there is a xed mode that input would go through a downsampling block rstly to halve the shape, and some stacking CNN blocks attempt to extend the feature map's receptive eld without changing the shape. Therefore, the rst thought is to keep the overall structure of ResNet and just replace the above modules by Swin Transformer module and a mixed CNN and DCNN module that is short for MCNN module, respectively.

Based on the above thought, I de ne this single branch model as the rst variant of Res-Swin called Res-Swin-SG. Figure 3.1 shows that the overall structure of Res-Swin-SG, which is composed of one stem stage and four following stages that all have consistent arrangement. Input rst passes through a stem component made up of several convolutional layers so that shape can be halved and some key features can be extracted. Then in the following stages, each one includes two main parts, one of which uses Swin Transformer module to capture global information at the beginning of the stage and halve the output shape of the feature map. And there is another branch composed of several convolutional layers as a kind of shortcut operation. Besides, as mentioned before, Swin Transformer comes in four sizes that range from tiny to large by increasing the layer number and some parameters. With the increase in model size, the performance can become better, but resource costs also become higher. Therefore, considering both performance and resource costs, I choose to use the tiny version of Swin Transformer, which can make a good balance between them. Another part is MCNN block, its structure is inspired by ADNet [23], which means having an interleaved arrangement of convolution layers and dilated convolution layers with a rising dilation rate. Through this way, the receptive eld can be expanded further at the same time of keeping getting local important features by convolutional layers. Additionally, the setting layer number of the MCNN block

Figure 3.1: The overall structure of Res-Swin-SG. For the input size of (512, 512), it goes through a stem part composed of several CNN layers rstly to halve the size and capture some important features. Then the output will go through four stages. Each stage is composed of Swin Transformer module and MCNN module. After each stage, the output size is halved and the nal output size of the encoder should be (16, 16).

is based on the original ResNet34, when downsampling layer in each stage is not taken into account, the layer number of the MCNN block is around half that of ResNet34, according to Table 2.1. Besides, considering the situation that reduction of CNN layers can have some bad impacts to the overall performance, more shortcut operations are introduced in the MCNN block to alleviate the bad impacts to a certain extent and keep the performance.

3.2 Double Branches Structure - Res-Swin-DB

According to the above description of Res-Swin-SG, we can infer that Res-Swin-SG mixes the CNN module with the Swin Transformer module in each stage to encourage interaction and complement each other. However, in this way, the number of CNN layers is highly reduced, which alleviates the ability to explore local important information, so the single branch struc-

ture might be not the best way to combine the transformer module and CNN module, which cannot maximize the overall e ectiveness of the model. For keeping the number of convolutional layers and exploring the ability of the Swin Transformer as much as possible, I consider the double-branches structure, one of which aims to investigate local information with all of the CNN layers and the ResNet34 model can be a good choice, because it is already a mature model to be applied in various elds. The other of which can use the Res-Swin-SG model but making some little changes to focus on the large receptive eld totally, all the convolutional layers in each stage will be replaced by dilated convolutional layers. This variant is called Res-Swin-DB and the overall structure is like what Figure 3.2 shows.

Here, we can notice some di erences and similarities in structure between Res-Swin-SG and Res-Swin-DB. Firstly, the overall structure is still composed of 5 stages, the rst stage is convolutional layers, and the other 4 stages are composed of CNN module and Swin Transformer module. Likewise, the output size of each stage will be halved. While the di erence is the number of branches, for the Res-Swin-DB, the output of the stem stage enters two branches, one branch is just traditional ResNet34 without any changes, and another branch has the same structure with Res-Swin-SG except replacing all the convolutional layers by dilated convolutional layers to expand the receptive eld further. Therefore, these two branches have di erent main focuses. The ResNet branch concentrates in the local information with a limited receptive eld to get more local details, while the Res-Swin-SG branch can have a larger receptive eld and get the whole picture. For covering more features to make the decoder part restore the original image better, the output of two branches needs to combine through the way of concatenation. And $1 \quad 1$ convolutional layer can be utilized to change the channel number to combine features and reduce the calculation volume of the decoder part.

3.3 Double Branches Structure with a Mixture Block - Res-Swin-DBM

The above proposing models both have some distinct drawbacks in their structures. Res-Swin-SG mixes the Swin Transformer module with the CNN module into a single branch, but it alleviates the ability of the CNN part due to the decreasing layer number. For the Res-Swin-DB, considering the drawback of Res-Swin-SG, it is composed of two branches and each branch focuses on the di erent aspect. In this way, the original structure of ResNet can be saved and the number of CNN layers can be promised to explore more important local details, likewise, another branch follows the overall structure of Res-Swin-SG and tries to expand receptive eld further to capture global information better. However, one distinct drawback is there is no interaction between two branches in the encoder part and they just combine together at the end of each stage to be input of decoder. If these two branches can interact and complement each other, the combined output will be better and it will be also bene cial for the learning of next stage. Then Res-Swin-DBM introduces a mixture block at the end of each stage to make two branches complement one another, which can address the potential shortcomings of Res-Swin-DB. The mixture block allows the ResNet branch to obtain some useful global information from the other branch, at the same time, the Res-Swin-SG branch can also obtain some crucial local details from the ResNet branch before moving on to the next stage. Furthermore, the mixture block can also be helpful to delete some harmful features of each branch. Figure 3.3 depicts the overall structure of Res-Swin-DBM.

From Figure 3.2 and Figure 3.3, it is clear to see that the main difference between Res-Swin-DB and Res-Swin-DBM is the introduction of mixture block. The mixture block can be divided into three phases. First of all, an attention map is produced by combining features from the two branches, which can be represented by the following formulas:

$$M_q = Cat_2(f(w_{lq}; h_l); f(w_{gq}; h_g)) \tag{3.1}$$

$$M_k = Cat_1(f(w_{lk}; h_l); f(w_{gk}; h_g)) \tag{3.2}$$

$$= softmax(M_q \quad M_k) \tag{3.3}$$

where f is the convolution function, w means the weights of the convolutional kernel, h is the input feature map of the model. For the l and g, they are short for local and global, respectively. Therefore, it is clear that h_l is the local feature map from the ResNet branch, while h_g is the global feature map from the Res-Swin-SG branch. For q and k, they are short for query and key, query matrix and key matrix are two essential matrixes in the calculation process of attention map. Then Cat is short for the concatenation operation, for the two inputs with the shape of $(C_1; H; W)$ and $(C_2; H; W)$, the shape of output after concatenation in the channel direction should be $(C_1 + C_2; H; W)$. Furthermore, subscript of Cat means the dimension to be concatenated, so Cat_1 means concatenating in the second dimension, which is height of feature map and Cat_2 is to concatenate in the third dimenstion, which is the width of feature map. In this way, the shape of M_q and M_k can be suitable to do matrix multiplication. Therefore, Equation 3.1 and Equation 3.2 can represent the process of generating query matrix M_q and key matrix M_k through concatenating the outputs from two branches after convolution operation. Then attention map can be obtained by applying $softmax$ function to the multiplication result of M_q and M_k.

After getting the attention map, the attention map will multiply with the original local and global features map in turn. Additionally, there is a learnable parameter involved in the multiplication, as the following equations illustrate:

$$a_l = \quad h_l \quad r_l \tag{3.4}$$

$$a_g = \quad h_g \quad r_g \tag{3.5}$$

where a_l and a_g are the attention feature maps of the ResNet branch and Res-Swin-SG branch. Since harmful features like noise are included in the feature map, r can be either negative for reducing harmful features or positive for reinforcing key features. And learnable parameters of the local feature map and global feature map are denoted by r_l and r_g. Furthermore, one thing to emphasize is that the value range of r is limited between -1 and 1 by the following equation:

$$r = sigmoid(p) \quad 2 \quad 1 \tag{3.6}$$

where p is the original learnable parameter whose initial value is 0 and it will be updated gradually with the learning of models. As we know, the output range of $sigmoid$ function is between 0 and 1, so the output range of the above equation will be between -1 and 1.

Then for making the branches complement each other, the final step is to add the obtained

feature maps with attention information of one branch to the original features of the other branch, which can be represented as follow:

$$h_{fl} = h_l + f(w_1; a_g) \tag{3.7}$$

$$h_{fg} = h_g + f(w_2; a_l) \tag{3.8}$$

where h_{fl} and h_{fg} represent the final output of the ResNet branch and Res-Swin-SG branch. In this way, it is obvious that these two branches will contain information about the other party. Likewise, some important information can be reinforced, while harmful information can be alleviated in this process. And the mixture block is also beneficial for the learning of two branches in the following stage, as the model grows deeper, these positive impacts will be continually enhanced.

Figure 3.2: The overall structure of Res-Swin-DB. Input still goes through the stem stage and the following four stages. But there are two branches instead of a single branch. One is ResNet34, and another one is Res-Swin-SG but with totally dilated convolutional layers. The output of two branches will concatenate together in the channel direction and combine further through a 1 1 convolutional layer to be the input of the corresponding part of the decoder.

Figure 3.3: The overall structure of Res-Swin-DBM. The overall structure is still hierarchincal and almost same with the Res-Swin-DB. The main di erence is to add a mixture block at the end of each stage. In the mixture block, the output from two branches will interact with each other and each branch can get some useful information from the other side.

Chapter 4

Experiments

4.1 Dataset

In this section, some descriptions to three datasets that I use in this thesis project will be given. One is the 2016 NIH-AAPM-Mayo Clinic Low-Dose CT Grand Challenge[1] [31] that is short for the AAPM dataset, the other two ones are generated from foam-like phantoms dataset[2] [32] by only introducing noise and introducing both noise and some artifact features, respectively.

4.1.1 AAPM Dataset

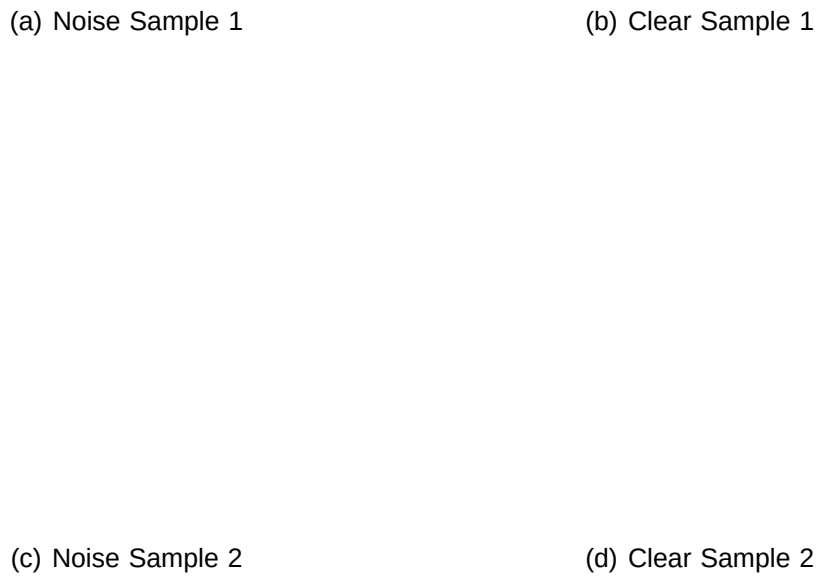
The AAPM dataset aims for reducing image noise with either image-domain or projection-domain techniques, which can maximize the ability to detection for liver lesions. Poisson noise is inserted into the projection data for each case to reach a noise level that corresponded to the 25% of the full dose. The overall dataset is composed of four folders: 1mmB 30, 1mmD 45, 3mmB 30, and 3mmD 45, which means different thick images and choices between medium and sharp reconstruction kernels. Likewise, there are two sub-folders in each folder, one starts with full stores clear images, while another one starts with quarter stores noisy images. Moreover, both of these two folders consist of 10 folders and each folder means different patient. Considering the high data volume of each beginning folder in the AAPM dataset and limited computing resources, I choose 1mmB 30 folder for training and testing in this thesis. Table 4.1 shows the number of images in each sub-folder of 1mmB 30 and the total number of images.

Table 4.1: Number of images in each sub-folder of 1mmB 30

	L067	L096	L109	L143	L192	L286	L291	L310	L333	L506	Sum
Num	560	823	318	585	600	525	856	533	610	526	5936

For the splitting of the training set and testing set, through referring to some related experiments of this dataset by other papers [11, 12], I only pick L506 folder as the testing set, and the other folders will form the training set. Figure 4.1 shows some examples of the AAPM dataset after transforming into the HU format.

[1] https://www.aapm.org/grandchallenge/lowdosect/#trainingData
[2] https://github.com/dmpelt/foam_ct_phantom



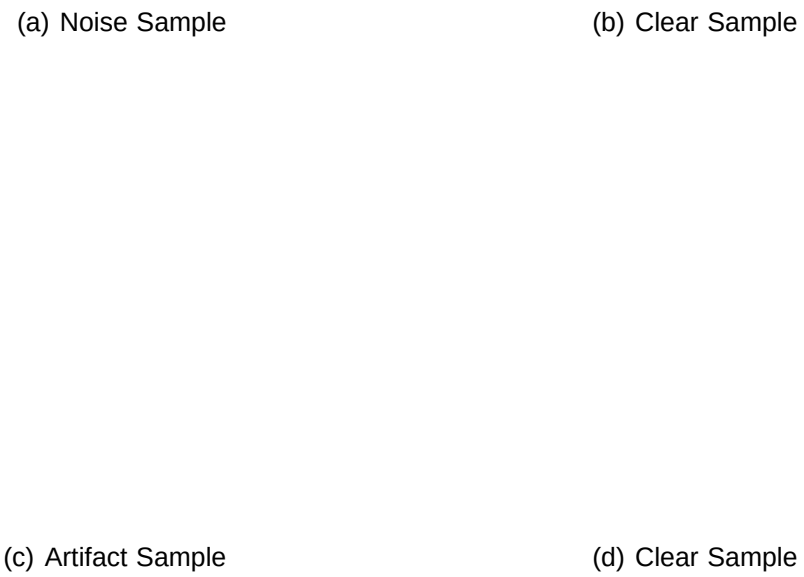
(a) Noise Sample 1 (b) Clear Sample 1

(c) Noise Sample 2 (d) Clear Sample 2

Figure 4.1: Some examples of noise image and corresponding clear image of AAPM dataset. The left side is images with noise, and the right side represents the corresponding target clear images.

4.1.2 Phantom Dataset

Considering the situation that current benchmark tomographic datasets are often limited by the acquisition mode and number of samples, they cannot t all the ideal requirements very well. Therefore, the phantom dataset is proposed to satisfy all the requirements. It is very challenging by introducing foam-like structures that have more than 100000 features. In general, the phantom dataset can be e cient to measure the overall performance of models due to the diversity of features. Therefore, the left two datasets both build on the phantom dataset. The second dataset is generated by adding the high-level noise, while the third dataset introduces both high-level noise and some artifact features, which is more di cult for the model to denoise and recover. Besides, both of them have 512 training data and 512 testing data. From Figure 4.2, a sample pair of each dataset is represented, and we can see that there are many small circles contained in the image of the phantom dataset, which will be hard to restore, especially after adding some interference features.



(a) Noise Sample (b) Clear Sample

(c) Artifact Sample (d) Clear Sample

Figure 4.2: Some examples of the second dataset and the third dataset. The upper part represents a noise image and a corresponding clean image of the second dataset. While the bottom part shows the same thing about the third dataset, but it is clear to see that some circle artifact features are added.

4.2 Evaluation Metrics

To measure the denoising performance of models, I choose to use two widely used metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [33]. PSNR is an engineering term indicating the ratio of the maximum possible power of a signal to the destructive noise power that a ects its representation accuracy. Its calculation is based on the maximal possible pixel value of the image and the Mean Square Error Loss (MSE) value between the noise-free image and its denoising approximation. MSE is widely used in the eld of image reconstruction and image compression to calculate the di erence between the output image and the original image, which can represented by Equation 4.1.

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i;j) - K(i;j)]^2 \tag{4.1}$$

In the situation of denoising, we can consider I as a given noise-free image, whose size is $m \quad n$, and K is its denoising approximation. The equation of PSNR can be de ned as:

$$PSNR = 10 \quad \log_{10}(\frac{MAX_I^2}{MSE}) \tag{4.2}$$

where MAX_I means the maximal possible pixel value of I.

SSIM measures the distance between two images with more emphasis on the structural similarity instead of calculating the di erence between the two images pixel by pixel, which can be more re ective of the human visual system's judgment of the similarity of the two images. SSIM compares the similarity of two images from three aspects: luminance (l), contrast (c), and structure (s). Therefore, the overall representation of SSIM is composed of three functions and these three functions can be de ned as the following equations:

$$l(x;y) = \frac{2 \quad _x \quad _y + C_1}{^2_x + \quad ^2_y + C_1} \tag{4.3}$$

$$c(x;y) = \frac{2 \quad _x \quad _y + C_2}{^2_x + \quad ^2_y + C_2} \tag{4.4}$$

$$s(x;y) = \frac{_{xy} + C_3}{_x \quad _y + C_3} \tag{4.5}$$

$$SSIM(x;y) = [l(x;y)] \quad [c(x;y)] \quad [s(x;y)] \tag{4.6}$$

where $$ is the mean of variable, ^{2} is the variance of variable, $_{xy}$ means covariance of x and y, C is constant. In general, $C_1 = (k_1L)^2$, $C_2 = (k_2L)^2$ and $C_3 = \frac{C_2}{2}$. L is the value range of the pixel, usually from 0 to 255. Then the default value of k_1 and k_2 is 0.01 and 0.03, respectively. When setting $;$ as 1, Equation 4.6 will be as follows:

$$SSIM(x;y) = \frac{(2 \quad _x \quad _y + C_1)(2 \quad _{xy} + C_2)}{(\ ^2_x + \quad ^2_y + C_1)(\ ^2_x + \quad ^2_y + C_2)} \tag{4.7}$$

In the factual implementation of this thesis, the above default values of variables will be used and brought into the Equation 4.7 to calculate SSIM. In addition, each calculation takes a window from the image, then keeps sliding the window for calculation and uses the average value as the global SSIM.

4.3 Experiments

4.3.1 Models

For measuring the performance of variants of Res-Swin, it is necessary to make some comparisons between them and some proposed models under the situation of similar parameter volume. Moreover, for validating the impacts of the Swin Transformer module and mixed CNN and DCNN module in the Res-Swin-SG, I also create some related models to make ablation experiments. Therefore, I will use 9 models to make experiments to validate the performance of Res-Swin models. And one thing to mention is that this thesis mainly focuses on the encoder part, while the decoder part of models just keeps the original decoder of UNet. Then there are some descriptions of the models used in this thesis:

- RED-CNN : A proposed e ective CNN backbone for LDCT denoising. It follows the residue encoder-decoder structure. The overall structure is composed of multiple CNN layers both in the encoder and decoder parts. Besides, there is a residue connection between the output of the encoder and its corresponding part of the decoder.

- UNet34 : UNet utilizes an encoder-decoder structure and makes a good performance in the eld of medical image segmentation, which can also be applied for denoising tasks. In the factual application, the encoder structure of UNet can also be replaced by the other e ective CNN backbones like VGG and ResNet. Considering ResNet is an important factor in this thesis, I choose to take ResNet34 as the encoder of UNet, called UNet34.

- UNet34-Swin : It is created for validating the impacts of the Swin Transformer module in the Res-Swin-SG. Replace the rst part with the corresponding part of the Swin Transformer in each stage and keep the original second part of each stage of ResNet34.

- UNet34-MCNN : It is created for validating the impacts of MCNN module in the Res-Swin-SG. Keep the original rst part of each stage of ResNet34 and replace the second part with the MCNN module.

- Res-Swin-SG: The rst Res-Swin model that was mentioned before. It is inspired by the overall structure of ResNet34 that each stage is composed of two parts. Then the rst part for down-sampling is replaced by the Swin Transformer module to expand the receptive eld, while another part is replaced by the MCNN module for reducing the parameter volume without highly degrading the overall performance.

- UNet50 : The overall structure still follows UNet but taking the ResNet50 as the encoder of UNet. In this way, it will have similar parameter volumes with the other two Res-Swin models and TransUNet, which can be better to validate the performance of Res-Swin models that have high parameter volumes.

- Res-Swin-DB: The second Res-Swin model. It is composed of two branches, one branch is the original ResNet34 to focus on the local features, while the other branch builds on the structure of Res-Swin-SG but replaces all the CNN layers by DCNN layers to expand the receptive eld further and concentrate in the global perspective. At the end of each stage, the outputs of the two branches will be combined as the corresponding input of the decoder.

- TransUNet : TransUNet [34] is proposed for improving the performance of medical image segmentation rstly, and it uses the overall structure of UNet as a reference but also re-designs the structure of the encoder part. The encoder of TransUNet has two compositions, the CNN module and the ViT module. The image will rst be processed by the CNN module to extract local information, after which it will be divided into multiple patches and fed into the ViT module. After going through the ViT module, the output will be put into the decoder part directly. Decoder part of TransUNet still uses the original decoder part of UNet.

- Res-Swin-DBM : The overall model structure is the same as the Res-Swin-DB except for introducing a mixture block at the end of each stage. In the mixture block, the attention mechanism is implemented for the output of two branches to generate the corresponding attention feature map, and the attention feature map of one branch will be added to the feature map of another branch so that they can interact and complement each other to promote the overall performance further in this way.

Table 4.2 records the parameter volume of each model so that the models can be split into groups to compare more reasonably, which means the models with similar parameter volumes will be divided into the same group to compare together.

Table 4.2: Parameter Volume (MB) of each model

	RED-CNN	UNet34	UNet34-Swin	UNet34-MCNN	Res-Swin-SG	UNet50	Res-Swin-DB	TransUNet	Res-Swin-DBM
Val	1.85	24.17	38.07	15.48	29.85	61.51	57.92	67.85	59.32

Based on the above results, UNet34, UNet34-Swin, UNet34-MCNN, and Res-Swin-SG can be considered as one group for comparison, while UNet50, Res-Swin-DB, TransUNet, and Res-Swin-DBM form another group.

4.3.2 Scheduler

As is well known, the learning rate plays an important role in the training of deep learning models. If it is set too high, optimization will diverge and cause over- tting easily, while training will take a long time and produce a sub-optimal outcome when the learning rate is set too low. It is challenging to achieve the optimal results in the original training since the learning rate is xed and back propagation is done using an optimizer depending on the xed learning rate setting. Then scheduler can make the learning rate dynamic by setting it explicitly at each step. There are many di erent types of schedulers, such as factor scheduler that decays the learning rate with a set rate beyond a reasonable lower bound, multi-factor scheduler that creates numerous stages and maintains the learning rate in each stage before reducing the learning rate at the end of each stage, or a heuristic method proposed by Loshchilov and Hutter [35] called cosine scheduler, the learning rate alters according to a cosine-like function, which means learning rate does not fall too quickly before gradually reducing to a minimum value and then re ne the solution through the setting of minimal learning rate.

In this thesis, the cosine scheduler with warmup will be used for better performance. The working principle of the cosine scheduler can be explained by the following equation:

$$_t = \ _{min} + \frac{1}{2}(\ _{max} \quad \ _{min})(1 + \cos(\frac{T_{cur}}{T_{max}})) \tag{4.8}$$

where $_t$ is the learning rate of the current step, $_{min}$ is the setting minimal learning rate, while $_{max}$ is the initial learning rate, also the maximal learning rate. T_{cur} is the current epoch and T_{max} is the half-cycle length of the cosine function. It is obvious that $\frac{1}{2}(1 + \cos(\frac{T_{cur}}{T_{max}})$ ranges from 0 to 1 and $_t$ has a range between $_{min}$ and $_{max}$. Likewise, when T_{cur} is approaching to T_{max} gradually, the current learning rate $_t$ will also be close to the minimal value gradually. For the warmup, it means setting up a warmup period during which the learning rate increases to its maximal value, then cool down to the minimal value by the cosine scheduler within the setting period. For simplicity, a linear function is often used in the warmup period, in addition, the warmup period should not be set too large. Figure 4.3 depicts the changes in learning rate based on the cosine scheduler with warmup to explain it further.

4.3.3 Settings

In this section, more details about the implementation of experiments will be given. First of all, for the parameter settings of each model, most of them just follow the original settings except

Figure 4.3: In this example, the initial learning rate is 0.3, the minimum learning rate is 0.01, the warmup period is 5, and T_{max} of the scheduler is 30. The learning rate increases linearly in the warmup period and reaches the maximal value at the end, then the learning rate decreases gradually by following the cosine-like curve. After T_{max}, the learning rate is at its lowest and keeps this value for the left optimization process.

RED-CNN and Swin Transformer module. For the RED-CNN, the original structure sets the output channel dim of each layer as 96, but it will cost too much time and GPU memory to train, so I decrease its channel value to 64 without highly impacting the performance. As mentioned before, I choose the tiny version of the Swin Transformer as the composition of Res-Swin in this thesis. In its original parameter settings, the window size is 7 due to the input size of (224, 224), in this way, input size can be divided by the window size. However, the size of the input image is (512, 512) in this project, which can not be divided by 7, so I set the window size as 8. Besides, the original down-scaling factors are (4, 2, 2, 2), but from Figure 3.3, we can see that the stem part has already done down-sampling with a factor of 2. Therefore, the down-scaling factor of stage 1 should be 2 and the overall down-scaling factors will be (2, 2, 2, 2). Moreover, the original hidden dim is 96, but for adapting to the output dim of the ResNet34 model and reducing GPU memory costs, I set the hidden dim as 64 in the experiments.

Some settings of the overall training pipeline are shown as follows:

- Random Seed: 0
- Batch Size: 8
- Interval for Testing: 50
- Loss Function: MSE Loss
- Input Image Size: (512, 512, 1)
- Implementation Framework: Pytorch

- Data Augmentation: HorizontalFlip, VerticalFlip, and ShiftScaleRotate
- Device: 2 NVIDIA GeForce RTX 3090 (24GB memory)
- Training Epoch: 500 for AAPM dataset, 11000 for the two Phantom datasets
- Optimizer: AdamW [36] with the learning rate of 1e-4 and weight decay of 5e-5
- Scheduler: Cosine scheduler with warmup. The warmup period is 20. The episodes of the scheduler are di erent for each dataset, 500 for the AAPM dataset, 500 for the phantom dataset with noise, and 2000 for the phantom dataset with noise and artifacts. Besides, the minimum learning rate of the scheduler is 1e-6.

For the above settings, I cannot make sure they are the best combination, because some settings like learning rate, weight decay, loss function, optimizer and episodes of scheduler can also have other choices, but they can make the model work well on these three datasets. Furthermore, I want to specify the reasons for some settings. First of all, batch size plays an important role in the training of the neural network, because it can speed up the convergence of the model and improve the overall performance. Moreover, the batch size is often set as an exponential multiple of 2, so considering the balance between memory cost and overall performance, I set the batch size as 8. Of course, if GPU resource is enough, batch size can be set larger, like 16. Data augmentation is also an important trick in the training process of neural network models, especially in the eld of computer vision. It can increase the number of samples for training and improve the generalization and robustness of the model. Here, the three methods I choose are widely used in the eld of computer vision and they do not have some negative impacts on the denoising task. Then the settings of the training epoch are mainly based on the observation in the training process. The AAPM dataset can be trained very well within the setting epochs of 500 due to its low-level noise and relatively simple features. For the two generated phantom datasets, I set training epochs as 15000 rstly, but PSNR of Res-Swin-DBM begins to decrease after about 11000 epochs in the testing process like what Figure 4.4 shows, which is kind of over- tting. To make sure all the models are under the same situation and avoid over- tting that lowers the performance and wastes some time, I set the training epoch as 11000 nally. In addition, as one of the state-of-the-art optimizers, AdamW can improve the generalization of Adam through the introduction of weight decay, which can replace L2 Regularisation and punish the large gradient e ectively. As described in Section 4.3.2, the scheduler can dynamic the learning rate by decreasing it with the increasing of training epochs so that the overall performance of the model can be improved further. In the settings of the scheduler, warmup episodes and minimum learning rate are mainly based on some experience, while the episodes of the scheduler are based on some attempts in the experiments. For example, when I set 500 for the phantom dataset with noise and artifacts, most of the models cannot work very well, but after having a try to a higher value like 2000, all the models can work and have relatively good performance. Therefore, there might be some potential rules for the setting of episodes. The value should be larger when the image is hard to process, like an image with some artifact features, while it can be lower when the image is only with noise, because a higher value of episodes can slow down the decreasing of the learning rate to make the model have more epochs to learn well, which is essential for the complex input.

Figure 4.4: The PSNR values of test data of the phantom dataset with noise by the Res-Swin-DBM model. The x-axis means the interval of 50. It is clear to see that the PSNR line has a high increasing tendency in the rst 2500 epochs. After that, it becomes at and there is a small decreasing tendency after about 11000 epochs.

4.3.4 Implementation

More details about the training of models will be represented in this section. First of all, there is a data pre-processing part only for the AAPM dataset by looking through some related materials[3]. The unit of radiation measurement in CT scans is the Houns eld Unit (HU), and CT scanners are carefully calibrated to accurately measure it. In general, the value of HU ranges from -1024 to 3072. But after loading the original dataset, the pixel values are beyond that range, which means the original dataset is not in the HU format. Therefore, I transform all images into the HU format by multiplying with the rescale slope and adding the intercept. In addition, the HU format might also have some positive impacts on the overall denoising performance [37].

After that, the overall training pipeline of the three datasets is all the same, which is shown as follows:

1. Build the dataset for training and testing and introduce the setting data augmentation methods in this process at the same time. Then shu e the training set and make batch data by data loader.

2. Train the model based on the above settings of each dataset. Furthermore, the PSNR and SSIM of the test dataset will be calculated in the training process with an interval of 50 to measure the performance of the current model.

[3]https://www.kaggle.com/code/gzuidhof/full-preprocessing-tutorial/notebook , https://github.com/SSinyu/RED-CNN

One thing to emphasize is the recording of PSNR and SSIM values in the testing process. First of all, the maximal and minimal pixel values of the overall test dataset need to be recorded so that the value range of pixels of the output image from models can be clipped, which means making sure that all the pixel values should be between the recorded minimal value and maximal value and performance can be improved a little in this way. Then calculate the PSNR and SSIM of each image by setting the data range as the subtraction between maximal and minimal value. The average value of PSNR and SSIM of the whole testing set are considered as the overall performance. If both PSNR and SSIM are larger than the current best PSNR and SSIM, replace the best PSNR and SSIM with the current values. The generating images and weight les will also be saved at the same time for the following analysis. Finally, the best PSNR and SSIM values will be used for measuring the performance of models.

4.4 Results

In this part, some results of three datasets by all the models will be represented. The best overall PSNR and SSIM values for each model on the testing set are displayed in each table, with the best result denoted in bold. Two gures will focus on the overall performance and some detailed information, respectively. In the rst gure, there are 27 sub- gures that arrange in 9 rows and 3 columns. From left to right, the columns denote the noise input, the clear target, and the denoising result of the model, while each row shows the outcome of one model. Besides, in the bottom right corner of each sub- gure, there is a square area, which is clipped from the center of sub- gure with the length of 120 pixels. The second gure enlarges the above clipped square area so that more details can be shown clearly to validate the performance of models. Finally, some discussions will be given to analyze the potential reasons for the getting results.

4.4.1 AAPM Dataset

Table 4.3: Performance of each model on the AAPM dataset

	RED-CNN	UNet34	UNet34-Swin	UNet34-MCNN	Res-Swin-SG	UNet50	Res-Swin-DB	TransUNet	Res-Swin-DBM
PSNR	46.985	46.775	46.812	46.787	46.861	46.757	46.917	47.100	47.159
SSIM	0.9806	0.9799	0.9802	0.9800	0.9804	0.9798	0.9805	0.9810	0.9812

4.4.2 Phantom Dataset with Noise

Table 4.4: Performance of each model on the phantom dataset with noise

	RED-CNN	UNet34	UNet34-Swin	UNet34-MCNN	Res-Swin-SG	UNet50	Res-Swin-DB	TransUNet	Res-Swin-DBM
PSNR	20.739	21.609	21.684	21.599	21.757	21.583	21.576	21.679	21.846
SSIM	0.6814	0.7411	0.7389	0.7393	0.7479	0.7402	0.7424	0.7596	0.7564

4.4.3 Phantom Dataset with Noise and Artifact

Table 4.5: Performance of each model on the phantom dataset with noise and artifact

	RED-CNN	UNet34	UNet34-Swin	UNet34-MCNN	Res-Swin-SG	UNet50	Res-Swin-DB	TransUNet	Res-Swin-DBM
PSNR	19.693	20.452	20.664	20.612	20.728	20.607	20.764	20.478	20.982
SSIM	0.6306	0.7137	0.7254	0.7050	0.7207	0.7159	0.7228	0.7205	0.7316

4.4.4 Discussion

RED-CNN performs worst in the two datasets generated from the phantom dataset, while it has relatively good performance on the AAPM dataset. In addition, the denoising performance of AAPM dataset of all the models are much better than that of the two phantom datasets. The potential reasons should be mainly from three aspects: one is the di erent noise levels. The noise level of the AAPM dataset is much lower than that of the two phantom datasets. The low noise level will reduce interference with the model capture of key features and the models can delete noise features more easily. On the other hand, images from the AAPM dataset are mainly composed of organs, which belong to large object features and can be extracted by CNN models very well, while images in the phantom dataset contain many small circles and diverse features, which are hard for models to recover, especially in the situation of having some harmful features. Finally, the number of samples in the dataset is also an important factor. In the AAPM dataset, there are 5410 images as the training set and 526 images as the testing set, while the number of the training set and testing set are both 512 in the two phantom datasets. As is well known, more data volume can motivate the learning of models to extract important features better and improve the generalization of models. Therefore, it makes sense that RED-CNN performs quite well on the AAPM dataset but has the worst performance on the two phantom datasets. In general, RED-CNN is a simple CNN model based on its structure. Phantom datasets re ect the limitations of its capabilities very well and it is not a good choice in this situation.

UNet34, UNet34-Swin, UNet34-MCNN, and Res-Swin-SG For the comparison between UNet34 and UNet34-Swin, we can see that replacing the down-sampling part of each stage with the Swin Transform module can improve the overall performance to a certain extent. Then for the comparison between UNet34 and UNet34-MCNN, it seems that MCNN module neither highly improve nor decrease the performance, but it can decrease the parameter volume well based on Table 4.2. Therefore, Res-Swin-SG inherits the above advantages and performs much better than UNet34 on all the datasets with a similar parameter volume. Likewise, from the above gures, it is obvious that the denoising performance of UNet34-Swin and Res-Swin-SG is better than that of UNet34-MCNN and UNet34, especially for the two phantom datasets. From the square clipped area of outputting results of UNet34-MCNN and UNet34, we can see that they perform worse in the aspect of recovering small objects, which can be re ected by lighter color and more blurred shape of some small circles. Besides, artifact features have greater negative impacts on the denoising of CNN models due to some clear left-over circle features of artifacts. Therefore, the e ectiveness of Res-Swin-SG can be approved. For the potential reasons, the model structure should be the main one. Res-Swin-SG introduces the Swin Transformer module at the beginning of each stage, in this way, the receptive eld can be expanded to the global perspective, which is bene cial for the reconstruction of objects and deleting some harmful features because a larger receptive eld can determine the location and level of noise better. After that, the following MCNN module is also good at extracting some local details. Moreover, the overall structure of Res-Swin-SG can make sure the enough interaction between the transformer module and CNN module, because CNN part and transformer part can pass information to each other within a stage and between stages.

UNet50, Res-Swin-DB, TransUNet, and Res-Swin-DBM Among them, UNet50 performs worst, while other models with the transformer module work better, which represents

the similar conclusion of the last comparison and approves the e ectiveness of Res-Swin models further. It seems that there is no major di erence in performance for the models with the transformer module, but Res-Swin-DBM has the best overall performance. For the TransUNet, we can see that its denoising performance is good but there are still some distinct shortcomings. One is darker color in all the datasets, which means the getting results have lower pixel values, and PSNR value will be lowered in this situation. On the other hand, it can be impacted by the artifact features more heavily, there is an outstanding part for the big center circle in the Figure 4.10, which should be caused by artifact features. For the artifact features, they are di erent from noise. Noise can obscure features in an image, but artifacts can become features of an image as false structures, so artifact is hard to be processed and can cause some negative impacts to the denoising and recovering of object features, which can also be re ected by lower PSNR and SSIM values in the Table 4.5. Furthermore, the introduced artifact features are kind of similar to the object features of the phantom dataset. Therefore, TransUNet might consider the artifact features as object features and extract them to recover the original circle objects. For the two Res-Swin models, the above problems are not as distinct as TransUNet. Although there are still some existing blurred artifact features, artifact features seem to have less interference to them. Likewise, the overall picture tone is basically the same as the original one. Of course, the main reason should still depend on the structure. The two separate branches can focus on local features and global features, respectively, and the combination at the end of each stage will cover more information to help decoder denoise and recover. What's more important is the introduction of the mixture block, it can give key features with higher weights based on the attention mechanism and promise the enough interaction between two branches. In addition, the value of learnable parameter can make sure the saving of key features and deletion of some harmful features.

Figure 4.5: The overall denoising performance of 9 models on the AAPM dataset. It seems that all the models can work well without too much di erence and getting results are even better than the target.

(a) RED-CNN (b) UNet34 (c) UNet34-Swin

(d) UNet34-MCNN (e) Res-Swin-SG (f) UNet50

(g) Res-Swin-DB (h) TransUNet (i) Res-Swin-DBM

Figure 4.6: The clipped area for each model on a sample of AAPM dataset. All the results look similar but it seems that models with the transformer module can have darker color compared to the total CNN models, which looks a little bit more clear.

Figure 4.7: The overall denoising performance of 9 models on the phantom dataset with noise. RED-CNN performs worst, while the other models have similar performance. Based on the observation of square areas, big circles can be recovered well, while most small circles can be blurred and have some damage to the shape. But models with transformer modules can recover more details compared to UNet models, which corresponds with the results in the table.