connections in the CFA module to join together low-level and high-level features during the fusion of the structure and texture features to ensure robust prediction results.

Table 1. Details of the texture and structure generator architecture. PConv is defined as a partial convolutional layer with the specified filter size, stride, and padding. Concat indicates that structure features and texture features are connected by a skip connection.

Module Name	Filter Size	Channel	Stride	Padding	Nonlinearity
Texture/Structure (T/S) Encoder					
T/S Input		3/2			
T/S Encoder PConv1	7×7	64	2	3	ReLU
T/S Encoder PConv2	5×5	128	2	2	ReLU
T/S Encoder PConv3	5×5	256	2	2	ReLU
T/S Encoder PConv4	3×3	512	2	1	ReLU
T/S Encoder PConv5	3×3	512	2	1	ReLU
T/S Encoder PConv6	3×3	512	2	1	ReLU
T/S Encoder PConv7	3×3	512	2	1	ReLU
Texture Decoder					
S Encoder-PConv7		512	-	-	-
Concat (S Encoder-PConv7, T Encoder-PConv6)		512 + 512	-	-	-
T Decoder PConv8	3×3	512	1	1	LeakyReLU
Concat (T Decoder PConv8, T Encoder-PConv5)		512 + 512	-	-	-
T Decoder PConv9	3×3	512	1	1	LeakyReLU
Concat (T Decoder PConv9, T Encoder-PConv4)		512 + 512	-	-	-
T Decoder PConv10	3×3	512	1	1	LeakyReLU
Concat (T Decoder PConv10, T Encoder-PConv3)		512 + 256	-	-	-
T Decoder PConv11	3×3	256	1	1	LeakyReLU
Concat (T Decoder PConv11, T Encoder-PConv2)		256 + 128	-	-	-
T Decoder PConv12	3×3	128	1	1	LeakyReLU
Concat (T Decoder PConv12, T Encoder-PConv1)		128 + 64	-	-	-
T Decoder PConv13	3×3	64	1	1	LeakyReLU
Concat (T Decoder PConv13, T Input)		64 + 3	-	-	-
Texture Feature	3×3	64	1	1	LeakyReLU
Structure Decoder					
T Encoder-PConv7		512	-	-	-
Concat (T Encoder-PConv7, S Encoder-PConv6)		512 + 512	-	-	-
S Decoder PConv14	3×3	512	1	1	LeakyReLU
Concat (S Decoder PConv14, T Encoder-PConv5)		512 + 512	-	-	-
S Decoder PConv15	3×3	512	1	1	LeakyReLU
Concat (S Decoder PConv15, T Encoder-PConv4)		512 + 512	-	-	-
S Decoder PConv16	3×3	512	1	1	LeakyReLU
Concat (S Decoder PConv16, T Encoder-PConv3)		512 + 256	-	-	-
S Decoder PConv17	3×3	256	1	1	LeakyReLU
Concat (S Decoder PConv17, T Encoder-PConv2)		256 + 128	-	-	-
S Decoder PConv18	3×3	128	1	1	LeakyReLU
Concat (S Decoder PConv18, T Encoder-PConv1)		128 + 64	-	-	-
S Decoder PConv19	3×3	64	1	1	LeakyReLU
Concat (S Decoder PConv19, S Input)		64 + 2	-	-	-
Structure Feature	3×3	64	1	1	LeakyReLU

After the texture and structure generators have obtained their respective features, the Bi-GFF module is applied to fuse the structure and texture features to enhance their consistency, and then, the CFA module is applied to further refine the generated pseudo-optical image.

Bidirectional Gated Feature Fusion (Bi-GFF): This module follows the structure and texture generators, and implements information exchange between the structure and texture features, as shown in Figure 3. The texture features are denoted by f_t, the structure features are denoted by f_s, and the features after information exchange can be expressed as:

$$\acute{f}_s = f_s \oplus (W_s(Concat(f_t, f_s)) \otimes f_t) \tag{1}$$

$$\acute{f}_t = f_t \oplus (W_t(Concat(f_t, f_s)) \otimes f_s) \tag{2}$$

where $\oplus$ denotes elementwise addition, $\otimes$ denotes elementwise multiplication, and W_s and W_t denote the convolutional layer mapping functions with a convolutional kernel of 3.

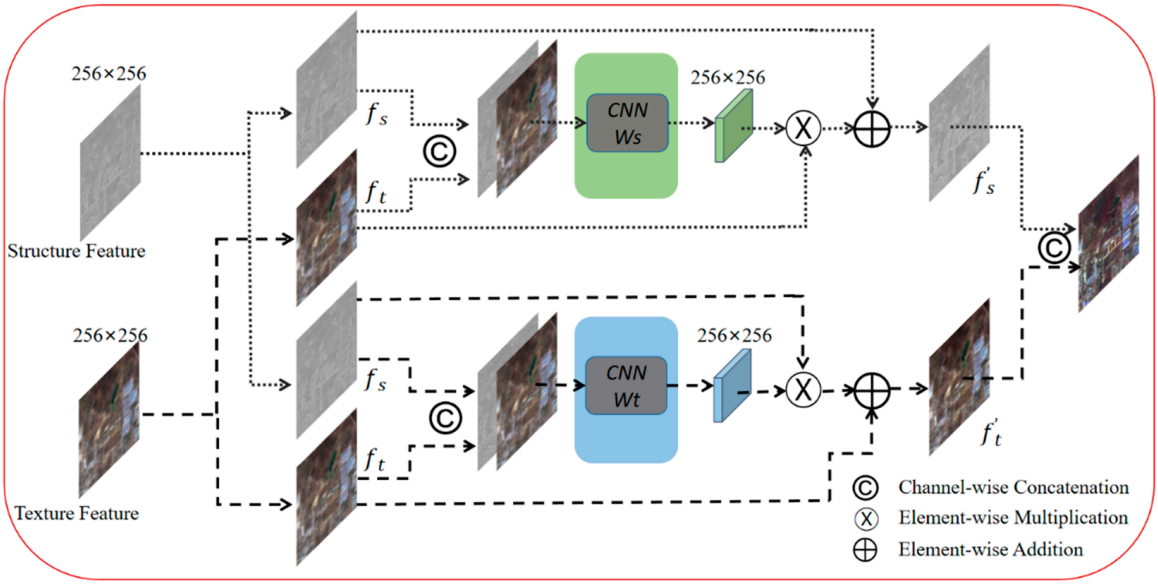

Figure 3. Structural diagram of the Bi-GFF module, in which deep fusion of texture and structure features is performed.

Finally, $\acute{f}_s$ and $\acute{f}_t$ are fused at the channel level to obtain the fused features:

$$f = Concat\left(\acute{f}_s, \acute{f}_t\right) \tag{3}$$

Contextual Feature Aggregation (CFA): As shown in Figure 4, the CFA module is introduced to determine which information in the SAR image contributes to SAR-to-optical translation, thereby enhancing the correlation between image features, and ensuring the overall consistency of the image.

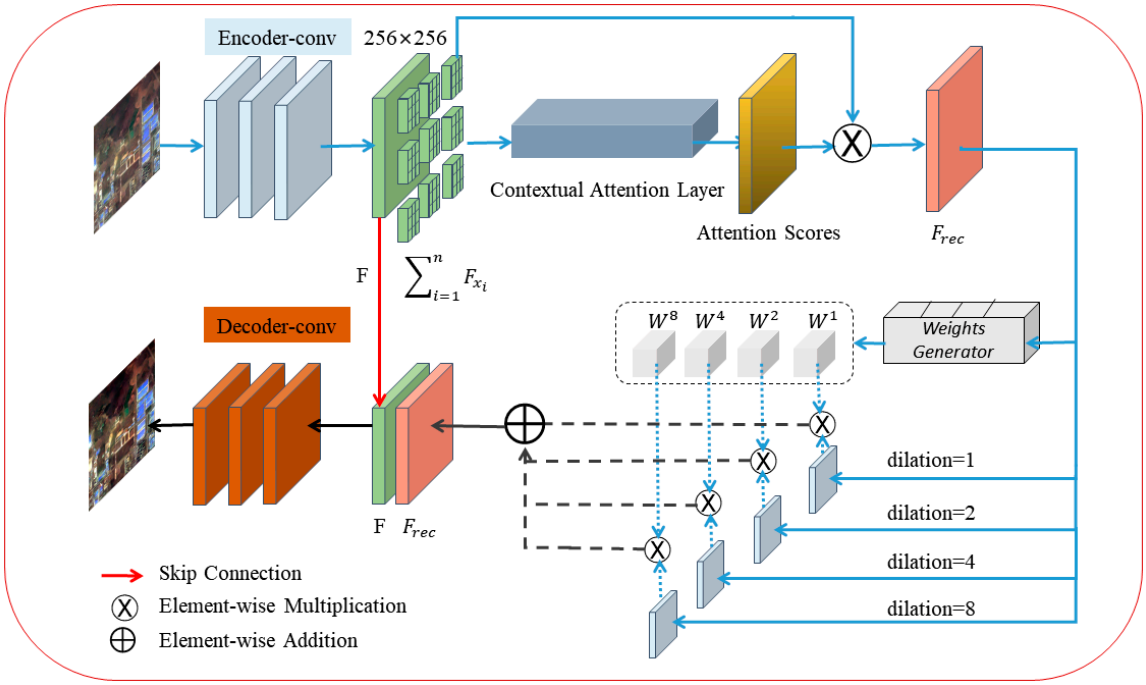

Figure 4. Structural diagram of the CFA module, which effectively models long-term spatial dependence through multiscale information.

First, the feature map F is divided into $\sum_{i=1}^{n} F_{x_i}$ 3×3 patches after being encoded in convolutional layers, and attention scores are obtained by a contextual attention layer, which calculates the cosine similarity between each pair of patches and applies the softmax function to this similarity to obtain the corresponding attention score. Then, the attention scores are multiplied by the 3×3 patches to obtain a reconstructed feature map. The contextual attention layer is defined as:

$$F = EncoderConv(F) \tag{4}$$

$$s_{Attention} = \frac{exp(\left\langle \frac{F_{x_i}}{\|F_{x_i}\|_2}, \frac{F_{x_j}}{\|F_{x_j}\|_2} \right\rangle)}{\sum_{i=1}^{n} exp(\left\langle \frac{F_{x_i}}{\|F_{x_i}\|_2}, \frac{F_{x_j}}{\|F_{x_j}\|_2} \right\rangle)} \tag{5}$$

$$F_{rec} = \sum_{i=1}^{n} F_{x_i} * s_{Attention} \tag{6}$$

where F_{x_i} and F_{x_j} denote the i-th and j-th patches, respectively; $s_{Attention}$ denotes the attention scores; and F_{rec} denotes the reconstructed feature map.

Then, multiscale semantic features are captured from the reconstructed feature map by using different dilation rates:

$$f_{rec}^{k} = Conv_k(F_{rec}) \tag{7}$$

where $Conv_k(\,.\,)$ denotes the k-th dilated convolution layer, $k \in \{1, 2, 4, 8\}$.

A weight generator module is defined to produce pixel-level prediction maps, which are split into four weight modules:

$$W^1, W^2, W^4, W^8 = Slice(W) \tag{8}$$

$$F_{rec} = (F_{rec}^1 \otimes W^1) \oplus (F_{rec}^2 \otimes W^2) \oplus (F_{rec}^4 \otimes W^4) \oplus (F_{rec}^8 \otimes W^8) \tag{9}$$

Finally, we use skip connections to splice F and $f1$ to prevent semantic information from being lost.

$$\acute{F} = DecoderConv(Concat(F_{rec}, F)) \tag{10}$$

2.2. Discriminator

The discriminator [38] distinguishes pseudo-optical images from real optical images by means of two branches: a texture branch and a structure branch. The last layer of the discriminator uses the sigmoid nonlinear activation function, and the structure branch has the same architecture as the texture branch. In the structure branch, the mapping for edge detection is first obtained by a residual network module [43] and a convolutional layer with a kernel size of 1. Then, the structure features are obtained by splicing with greyscale features. In the texture branch, the pseudo-optical image is directly mapped to obtain texture features, and finally, it is stitched to compute the adversarial loss. In addition, we apply spectral normalization [44] in the network to effectively solve the instability problem during network training.

2.3. Loss Functions

The algorithm proposed in this paper includes two generators, a texture generator G_t and a structure generator G_s, and a discriminator to learn the translation from the SAR image domain $\{x_i\}_{i=1}^{n} \in X$ to the optical image domain $\{y_i\}_{i=1}^{n} \in Y$. The original SAR image x_i^{SAR}, the greyscale image x_i^{gray} of the SAR image, and the edge image x_i^{Edge} of the SAR image are passed to the generators to generate the texture features f_t through the texture generator and the structure features f_s through the structure generator. The texture and structure features are then fused by the Bi-GFF module β_{Bi-GFF} and the CFA module σ_{CFA} to obtain the pseudo-optical image Y_{pseudo}. The discriminator D is similarly divided into two branches, i.e., a structure branch D_s and a texture branch D_t. The edge structure image Y_{pseudo}^{Edge} obtained through edge detection convolution is input into the structure branch of the discriminator, the generated pseudo-optical image is input into the texture branch of the discriminator, and finally, the features from the two branches are concatenated in the channel dimension to distinguish a real optical image Y_{real} from a generated pseudo-optical image Y_{pseudo}:

The generator is defined as:

$$Y_{pseudo} = \sigma_{CFA}\left(\beta_{Bi-GFF}\left(\left\{f_t, f_s = \left(G_t\left(x_i^{SAR}\right), G_s\left(x_i^{\text{gray}}, x_i^{\text{Edge}}\right)\right)\right\}\right)\right) \tag{11}$$

The discriminator is defined as:

$$Real/Fake = D\left(\left\{D_t\left(Y_{pseudo}\right),\ D_s\left(Y_{pseudo}^{Edge}, Y_{pseudo}^{Gray}\right)\right\}\right) \tag{12}$$

where $(\ .)$ denotes the projection function implemented by the convolutional layer and $\{\ .\}$ denotes concatenation in the channel dimension.

Reconstruction Loss: We define the reconstruction loss in terms of the differences between a real optical image and the corresponding pseudo-optical image obtained after the Bi-GFF and CFA modules have fused the structure and texture features.

(1) The mean square error (*MSE*) loss function is adopted to reduce the difference in the spatial domain between the pseudo-optical and real optical images at the pixel level. This loss function has the following form:

$$l_{MSE}^{rec} = \mathbb{E}\left[\|Y_{pseudo} - Y_{real}\|_2\right] \tag{13}$$

(2) The focal frequency loss (*FFL*) function is adopted to reduce the difference between the pseudo-optical and real optical images in the frequency domain, and to reduce the artefacts in the pseudo-optical image. The *FFL* function was proposed in [41]. We first use the 2D discrete Fourier transform (DTF) to separately adjust the frequency representations

of the pseudo-optical image and the real optical image, dividing each frequency value by $\sqrt{HW}$ for standard orthogonalization to obtain a smooth gradient, and adjusting the spatial frequency weights of each image by means of a dynamic spectral weight matrix $w(u,v)$. Then, the *FFL* function can be expressed as:

$$F(u,v)^{Y}_{pseudo} = \sum_{x=0}^{H-1}\sum_{y=0}^{W-1} f(x,y)\cdot e^{-i2\pi(\frac{ux}{H}+\frac{vy}{W})} \tag{14}$$

$$F(u,v)^{Y}_{real} = \sum_{x1=0}^{H-1}\sum_{y1=0}^{W-1} f(x_1,y_1)\cdot e^{-i2\pi(\frac{ux}{H}+\frac{vy}{W})} \tag{15}$$

$$l^{rec}_{FFL} = \frac{1}{HW}\sum_{u=0}^{H-1}\sum_{v=0}^{W-1} w(u,v)\left|F(u,v)^{Y}_{pseudo} - F(u,v)^{Y}_{real}\right|^2 \tag{16}$$

where $F(u,v)^{Y}_{pseudo}$ denotes a frequency value in the pseudo-optical image, $F(u,v)^{Y}_{real}$ denotes the corresponding frequency value in the real optical image, (u, v) represents the coordinates of a spatial frequency in the frequency spectrum, H × W denotes the size of the image, (x, y) denotes the coordinates of an image pixel in the spatial domain, $f(x, y)$ is the corresponding pixel value, and $w(u, v)$ denotes the dynamic spectral weight matrix.

(3) The VGG loss is used for the perceptual loss of the pseudo-optical and real optical images in terms of high-level semantic information. The pseudo-optical and real optical images are input into the VGG model pretrained on ImageNet [45] to obtain their high-level semantic information. The VGG loss can then be expressed as:

$$l^{rec}_{VGG} = \mathbb{E}\left[\sum_{i}^{3}\|\varphi^{i}_{VGG}\left(Y_{pseudo}\right) - \varphi^{i}_{VGG}(Y_{real})\|_1\right] \tag{17}$$

where $\varphi^{i}_{VGG}(.)$ denotes the projection function of the i-th pooling layer of the pretrained VGG network model.

(4) A style loss is used to ensure that SAR images are translated into pseudo-optical images with the same style as real optical images. The style loss can be expressed as:

$$l^{rec}_{Style} = \mathbb{E}\left[\sum_{i}^{3}\|\mu^{i}_{VGG}\left(Y_{pseudo}\right) - \mu^{i}_{VGG}(Y_{real})\|_1\right] \tag{18}$$

where $\mu^{i}_{VGG}(.) = \varphi^{i}_{VGG}(.)^{T}$, with $\varphi^{i}_{VGG}(.)$ denoting the Gram matrix constructed from the activation map φ^{i}_{VGG}. We choose to use the style loss [46] as demonstrated by Sajjadi et al. [47], based on its effectiveness in eliminating checkerboard artefacts.

Adversarial Loss: We define the adversarial loss in terms of a criterion for similarity evaluation between the pseudo-optical image and the real image.

(1) The GAN loss function is adopted to ensure that the generated pseudo-optical image is as close as possible to a real optical image. The pseudo-optical image and the corresponding real optical image are passed into the structure and texture branches, respectively, of the discriminator to ensure the consistency of the structure and texture. The GAN loss can be expressed as:

$$l_{GAN} = min_G max_D \mathbb{E}\left[\log D\left(Y_{real}, Y^{Edge}_{real}\right)\right] + \mathbb{E}\left[\log 1 - D\left(Y_{pseudo}, Y^{Edge}_{pseudo}\right)\right] \tag{19}$$

Structure Loss: We define the structure loss by comparing the structure features generated by the structure generator with the structure features of the real optical image.

(1) The *MSE* loss function is adopted to ensure that the structure features generated by the structure generator are close to those of a real optical image. The texture *MSE* loss can be expressed as:

$$l^{Struture}_{MSE} = \mathbb{E}\left[\|f_s - Y^{Edge}_{pseudo}\|_2\right] \tag{20}$$

Texture Loss: Distinct from the reconstruction loss, we define the texture loss by comparing the texture features generated by the texture generator with the texture features of the real optical image.

(1) The *MSE* loss function is adopted to ensure that the texture features generated by the texture generator are close to those of a real optical image. The texture *MSE* loss can be expressed as:

$$l_{MSE}^{Texture} = \mathbb{E}\left[\|f_t - Y_{pseudo}\|_2\right] \tag{21}$$

(2) The *FFL* function is adopted to reduce the differences between the texture features generated by the texture generator and those of a real optical image in the frequency domain, and to reduce the artefacts in the texture features:

$$F(u,v)_{f_t}^{Y} = \sum_{x=0}^{H-1}\sum_{y=0}^{W-1} f(x,y)\cdot e^{-i2\pi(\frac{ux}{H}+\frac{vy}{W})} \tag{22}$$

$$F(u,v)_{\text{real}}^{Y} = \sum_{x1=0}^{H-1}\sum_{y1=0}^{W-1} f(x_1,y_1)\cdot e^{-i2\pi(\frac{ux}{H}+\frac{vy}{W})} \tag{23}$$

$$l_{FFL}^{Texture} = \frac{1}{HW}\sum_{u=0}^{H-1}\sum_{v=0}^{W-1} w(u,v)\left|F(u,v)_{f_t}^{Y} - F(u,v)_{real}^{Y}\right|^2 \tag{24}$$

where $F(u,v)_{f_t}^{Y}$ denotes the frequency value of the texture features generated by the texture generator, $F(u,v)_{real}^{Y}$ denotes the corresponding frequency value of the real optical image, (u, v) represents the coordinates of the spatial frequency in the frequency spectrum, H × W denotes the size of the image, (x, y) denotes the coordinates of an image pixel in the spatial domain, $f(x, y)$ is the corresponding pixel value, and $w(u, v)$ denotes the dynamic spectral weight matrix.

In summary, the total loss is written as:

$$\begin{aligned} L = {} & \lambda_1(l_{MSE}^{rec}) + \lambda_2(l_{FFL}^{rec}) + \lambda_3 l_{VGG}^{rec} + \lambda_4 l_{Style}^{rec} + \lambda_5 l_{GAN} \\ & + \lambda_6\left(l_{MSE}^{Struture} + l_{MSE}^{Texture}\right) + \lambda_7\left(l_{FFL}^{Texture}\right) \end{aligned} \tag{25}$$

where λ_1, λ_2, λ_3, λ_4, λ_5, λ_6, and λ_7 are weighting coefficients of the loss functions, and the values set for our experiments are $\lambda_1 = 10$, $\lambda_2 = 50$, $\lambda_3 = 0.1$, $\lambda_4 = 250$, $\lambda_5 = 0.1$, $\lambda_6 = 1$, and $\lambda_7 = 5$.

3. Experiments

To demonstrate the effectiveness of our method, comparative experiments with Pix2pix [32], CycleGAN [33], S-CycleGAN [34], and EPCGAN [16] are presented. The results of qualitative visualizations show that our method achieves the best results in terms of both structure and texture. In quantitative experiments, three image quality assessment (IQA) metrics are used, namely, the peak signal-to-noise ratio (PSNR), the structural similarity (SSIM) [48], and the chromatic feature similarity (FSIMc) [49]. A higher PSNR indicates higher image quality, and the SSIM and FSIMc reflect the similarity between the pseudo-optical and real optical images, taking a value of 1 if the two images are identical. Experiments show that our method improves the PSNR by 21.0%, the FSIMc by 6.9%, and the SSIM by 161.7% in terms of the average metric values on all test images compared with the next best results, and the considerable SSIM improvement, in particular, proves the superiority of our dual-generator translation network in producing pseudo-optical images with better structure features.

We also present ablation experiments to demonstrate the effectiveness of the adopted loss functions. The superiority of the loss functions is demonstrated by qualitative visualization results that show the gradual texture and structure enhancement of the pseudo-optical images. In addition, quantitative experimental results show that adding the *MSE* loss function to the method presented in [38] can improve the PSNR by 2.3%, the FSIMc by 1.5%, and the SSIM by 13.9% in terms of the average metric values on all test images, whereas