Project Work 1

Implement a Generative Adversarial Network (GAN) from scratch. This GAN is based on the Siamese neural network. Therefore, you will have two generators and two discriminators. One of the generators takes the thermal (grayscale) image as input and the other generator takes the RGB (colored) image as input.

Design of the network:

During training, the contrastive loss is used in the latent embedding subspace to optimize the network parameters, so that latent features of input images from different spectral domains of the same identity are close to each other while the features of different identities are pushed further apart.

Generator: Used recent U-Net densely connected encoder-decoder structure (followed by ResNet-18 for both encoder and decoder).

To design the generator, contrastive loss along with adversarial loss is used. In addition to contrastive loss and adversarial loss, perceptual loss and L2 reconstruction loss are used to guide the generator toward the optimal solutions. The perceptual loss is measured via a pre-trained VGG-16 network.

Discriminator: The discriminator is patch-based (like the PatchGAN classifier). The discriminators are trained to simulate along with the respective generators.

Details to implement:

Follow the architecture of ResNet-18 to implement both the encoder and decoder sections of the U-Net.

In the encoder, each block is designed by applying two 3x3 convolutions, each followed by ReLU.

For down-sampling, it uses a 2x2 max pooling operation with stride 2.

We double the number of feature channels at each down-sampling step.

Similarly, each step in the decoder section, upscale the feature map by applying 2x2 transpose convolution, up-sampling the dimension of the feature map.

Each feature map is concatenated with the corresponding feature map from the encoder, followed by two 3x3 convolutions with a ReLU activation function.

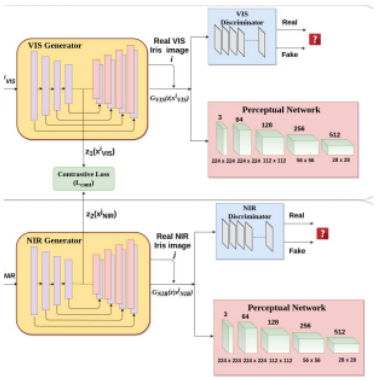
Training:

Both of the frameworks have to be implemented in PyTorch. Train the network with a batch size of 16 and learning rate $2 * 10^{-4}$. Adam optimizer with a first-order momentum of 0.999.

Leaky ReLU is used as an activation function with a slope of 0.35 for the discriminator.

To find the optimal hyper-parameters for our learning algorithms, we have used a random strategy.

For the network convergence, we consider $\lambda_3 = 1$ and $\lambda_4 = 0.3$ and $\lambda_5 = 0.3$. Also, $\lambda_1 = 10^{-6}$ and $\lambda_2 = 2 * 10^{-3}$.


VIS Generator
Real VIS Iris image
VIS Discriminator
Real
Fake
Perceptual Network
3
64
128
256
512
224 x 224
224 x 224
112 x 112
56 x 56
28 x 28
Contrastive Loss
NIR Generator
Real NIR Iris image
NIR Discriminator
Real
Fake
Perceptual Network
3
64
128
256
512
224 x 224
224 x 224
112 x 112
56 x 56
28 x 28