Supplementary Materials

1 Training Details

We list the hyper-parameters used for training the watermarking model. For our model, we set $\alpha_1^I = 18.0, \alpha_2^I = 0.01, \alpha^M = 0.3, \alpha_1^{adv} = 15.0, \alpha_2^{adv} = 1.0, \alpha_W^{adv} = 0.2$, and $num\_iter = 5$. For the HiDDeN combined model and identity model, we set $\alpha_1^I = 6.0, \alpha_2^I = 0.01, \alpha^M = 1.0$. The message size for our watermarking model is 120 instead of 30, due to the addition of the channel coding layer. We use the same network architecture as in HiDDeN. Namely, the input image I_{co} is first processed by 4 3×3 Conv-BN-ReLU blocks with 64 units per layer. This is then concatenated along the channel dimension with an $H \times W$ spatial repetition of the input message. The combined blocks are then passed to two additional Conv-BN-ReLU blocks to produce the encoded image. For the encoder, we symmetrically pad the input image and use 'VALID' padding for all convolution operations to reduce boundary artifacts of the encoded image. The encoded image is clipped to $[0, 1]$ before passing to the decoder. The decoder consists of seven 3×3 Conv-BN-ReLU layers of size, where the last two layers have stride 2. A global pooling operation followed by a fully-connected layer is used to produce the decoded message.

For both our model and the combined model, the training warm-starts from a pre-trained HiDDeN identity model and stops at 250k iterations. We use ADAM with a learning rate of $1e - 3$ for all models.

For the channel model, we use a two fully connected layers with 512 units each, and train with BSC noise where the noise strength is uniformly sample from $[0, 0.3]$.

2 Encoded Image Samples

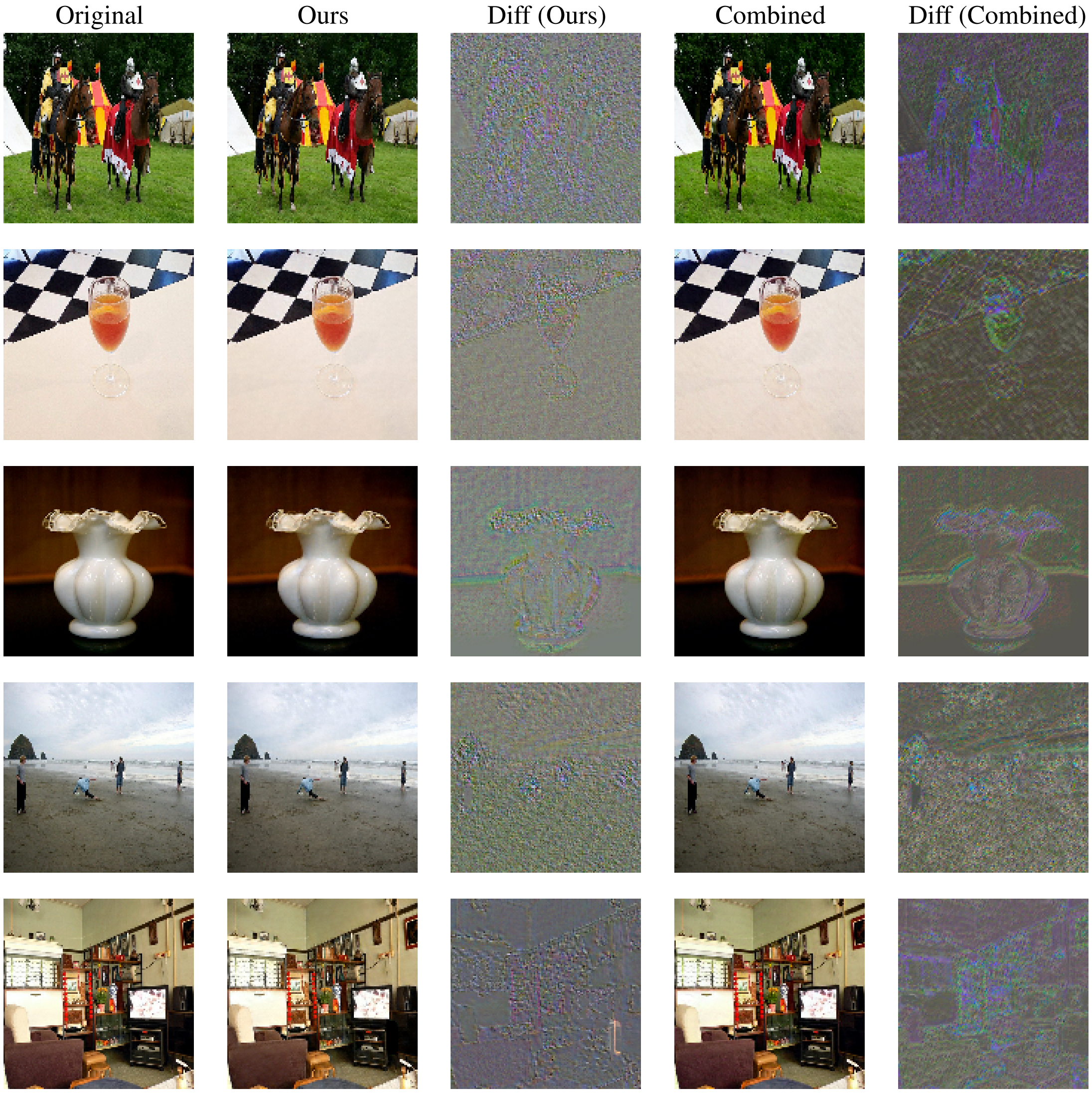

Figure 1: Samples of encoded image from HiDDeN and our model.

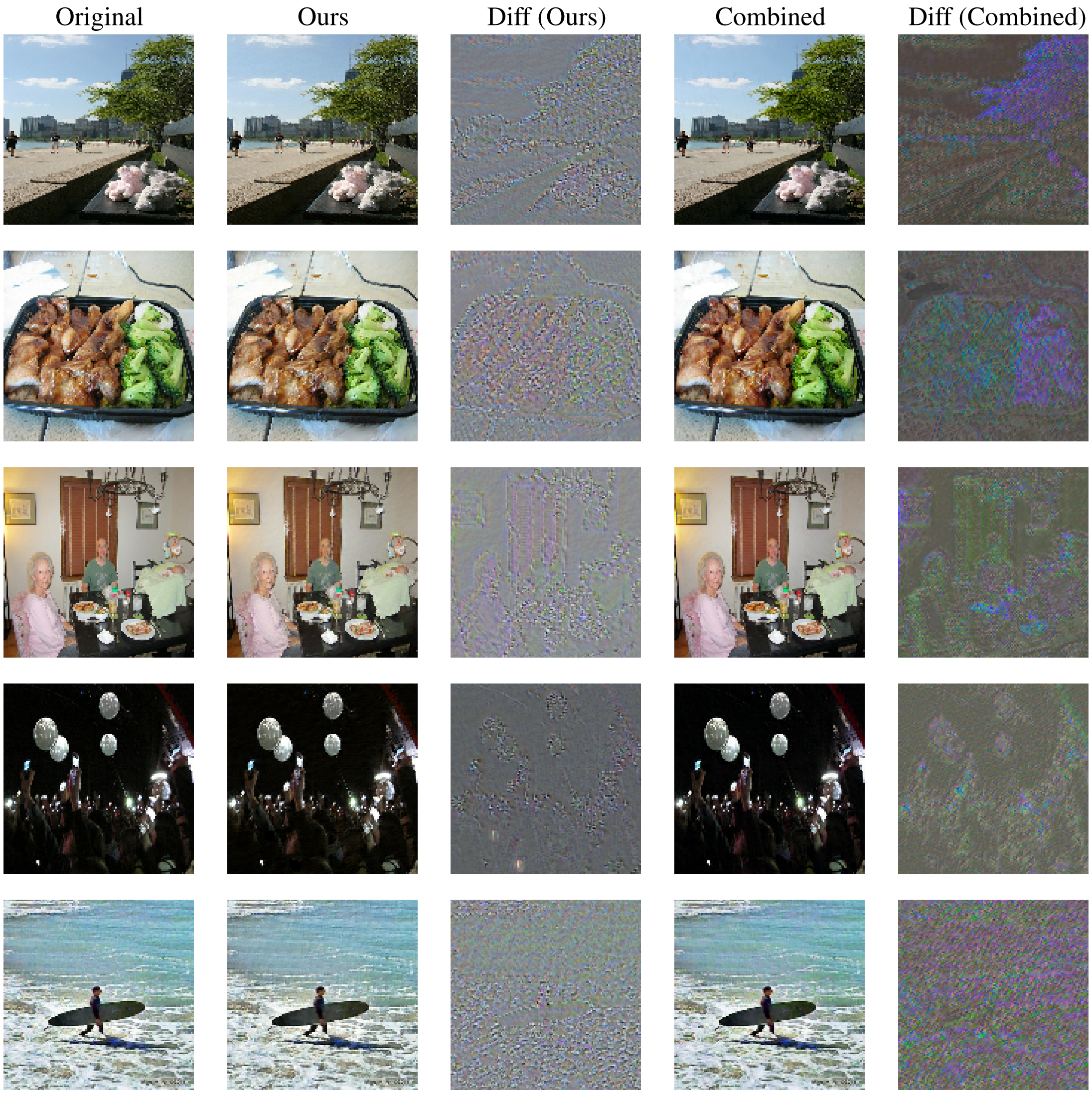

Figure 2: More samples of encoded image from HiDDeN and our model.

3 Adversarial Example Samples

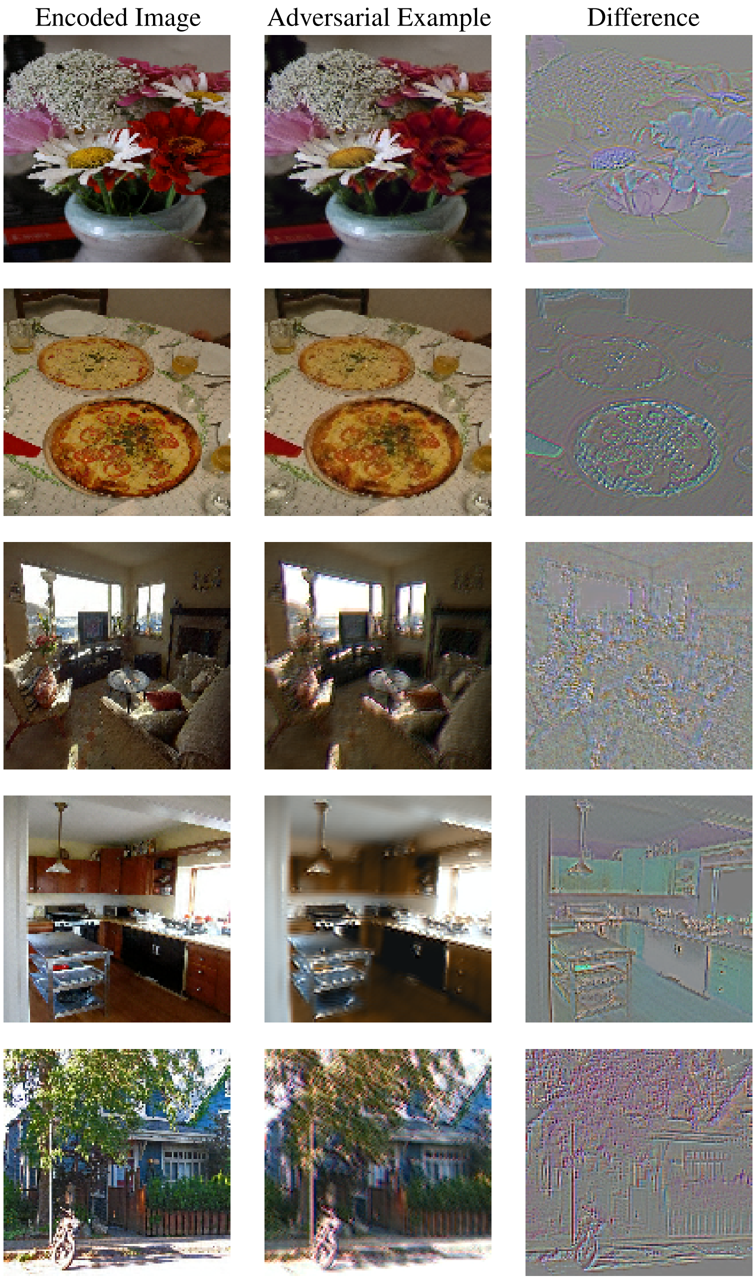

Figure 3: Samples of adversarial examples generated by the attack network.