Splicing ViT Features for Semantic Appearance Transfer Supplementary Material (SM)

ANONYMOUS AUTHOR(S)

A ARCHITECTURE

A.1 Splice Generator Architecture

We base our generator G_θ network on a `U-Net` architecture [5], with a 5-layer encoder and a symmetrical decoder. All layers comprise 3×3 Convolutions, followed by `BatchNorm`, and `LeakyReLU` activation. The encoder's channels dimensions are $[3 \rightarrow 16 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 128]$ (the decoder follows a reversed order). In each level of the encoder, we add an additional 1×1 Convolution layer and concatenate the output features to the corresponding level of the decoder. Lastly, we add a 1×1 Convolution layer followed by `Sigmoid` activation to get the final RGB output.

A.2 SpliceNet Generator Architecture

We design our feed-forward model F_θ based on a `U-Net` architecture [5]. The input image is first passed through a 1×1 convolutional layer with 32 output channels. The output is then passed through a 5-layer encoder with channel dimensions of $[64 \rightarrow 128 \rightarrow 256 \rightarrow 512 \rightarrow 1024]$, followed by a symmetrical decoder. Each layer of the encoder is a downsampling residual block that is comprised of two consecutive 3×3 convolutions and a 1×1 convolution for establishing the residual connection. The decoder consists of upsampling residual blocks with a similar composition of convolutions and residual connection as in the encoder. In the decoder, the weights of the 3×3 convolutions are modulated with the input [CLS] token. In each layer of the encoder, in order to establish the skip connections to the decoder, the output features are passed through a resolution-preserving residual block, which is concatenated to the input of the decoder layer. The residual blocks in the skip connections have a similar composition of convolutions and modulations as the decoder residual blocks. Finally, the output of the last decoder layer is passed through a modulated 1×1 convolutional layer followed by a `Sigmoid` activation that produces the final RGB output. `LeakyReLU` is used as an activation function in all the convolutional layers of the model.

Our mapping network M is a 2-layer MLP that takes as input the [CLS] token $t_{[\text{CLS}]} \in \mathbb{R}^{768}$ extracted from DINO-ViT, and passes it through one hidden layer and an output layer, both with output dimensions of 768 and with `GELU` activations. Following [2], for each modulated convolution in the feed-forward model, an affine transformation is learned that maps the output of the mapping network M to a vector used for modulating the weights.

B VIT FEATURE EXTRACTOR ARCHITECTURE

As described in Sec. 3, we leverage a pre-trained ViT model (DINO-ViT [1]) trained in a self-supervised manner as a feature extractor. We use the 12 layer pretrained model in the 8×8 patches configuration (`ViT-B/8`), downloaded from the official implementation at GitHub.

C TRAINING DETAILS

We implement our framework in PyTorch [4] (code will be made available). We optimize our full objective (Eq. 4, Sec. 3.3), with relative weights: $\alpha = 0.1$, $\beta = 0.1$ for Splice, and $\alpha = 2$, $\beta = 0.1$ for SpliceNet. We use the Adam optimizer [3] with a constant learning rate of $\lambda = 2 \cdot 10^{-3}$ and with hyper-parameters $\beta_1 = 0$, $\beta_2 = 0.99$. Each batch contains $\{\tilde{I}_s, \tilde{I}_t\}$, the augmented views of the source structure image and the target appearance image respectively. For Splice, every 75 iterations, we add $\{I_s, I_t\}$ to the batch (i.e., do not apply augmentations). All the images (both input and generated) are resized down to 224[pix] (maintaining aspect ratio) using bicubic interpolation, before extracting DINO-ViT features for estimating the losses. The test-time training of Splice on an input image pair of size 512×512 takes $\sim$20 minutes to train on a single GPU (Nvidia RTX 6000) for a total of 2000 iterations.

D DATA AUGMENTATIONS

At each training step, given an input pair $\{I_s, I_t\}$, we apply on them the following random augmentations: Augmentations to the source structure image I_s:

- cropping: we uniformly sample a NxN crop; N is between 95% - 100% of the height of I_s (for SpliceNet, we fix N=95%)
- horizontal-flipping, applied in probability p=0.5.
- color jittering: we jitter the brightness, contrast, saturation and hue of the image in probability p, where p=0.5 for Splice and p=0.2 for SpliceNet,
- Gaussian blurring: we apply a Gaussian blurring 3x3 filter (σ is uniformly sampled between 0.1-2.0) in probability p, where p=0.5 for Splice and p=0.1 for SpliceNet,

Augmentations to the target appearance image I_t:

- cropping: we uniformly sample a NxN; N is between 95% - 100% of the height of I_t (for SpliceNet, we fix N=95%).
- horizontal-flipping, applied in probability p=0.5.

REFERENCES

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.*

[2] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and Improving the Image Quality of StyleGAN. In *Proc. CVPR.*

[3] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[4] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention.* Springer.