Supplementary Material for
Meta Variance Transfer: Learning to Augment from The Others

Seong-Jin Park [1] **Seungju Han** [1] **Jiwon Baek** [1] **Insoo Kim** [1] **Juhwan Song** [1] **Hae Beom Lee** [2]
Jae-Joon Han [1] **Sung Ju Hwang** [2]

1. Detailed Network Architecture

The detailed network architectures used in our Meta Variance Transfer (MVT) are described in Table 1 and Table 2. Each table shows three networks for backbone θ, variance transfer ϕ, and manifold regularization δ. Table 1 describes the networks based on Conv4 architecture, while Table 2 describes the deeper networks baed on ResNet (He et al., 2016) architecture. For the Conv4 network, we additionally applied a max pooling to reduce the number of parameters for the final fully connected (FC) layer. Note that we used the same parameters for the FC1 and FC2 with simply applying a matrix transposition in the manifold regularization.

Table 1. Network architecture for Meta Variance Transfer using Conv4 backbone.

(a) Conv4-based backbone θ

Layer	Conv4_{+}
conv1	$[3 \times 3, 32]$, stride 2
conv2	$[3 \times 3, 32]$, stride 2
conv3	$[3 \times 3, 32]$, stride 2
conv4	$[3 \times 3, 32]$, stride 2
pooling	Max pooling (3×3, stride 2)
FC1	$(288, 288)$

(b) Variance transfer network ϕ

Layer	Conv4_{+}
FC1	(288 $\times$ 4, 32)
LeakyReLU	-
FC2	(32, 288)

(c) Manifold regularization δ

Layer	Conv4_{+}
FC1_W_{δ}	(288, 32)
$\text{FC2\_W}_{\delta}^{T}$	(32, 288)

*Equal contribution [1]Samsung Advanced Institute of Technology [2]KAIST. Correspondence to: Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Table 2. Network architecture for Meta Variance Transfer using ResNet backbone.

(a) ResNet-based backbone θ

Layer	ResNet-10_{+}	ResNet-34_{+}
conv1	$[7 \times 7, 64]$, stride 2	
pool1	3×3 max pool, stride 2	
conv2	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 1$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$
conv3	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 1$, s2	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$, s2
conv4	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 1$, s2	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$, s2
conv5	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 1$, s2	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$, s2
pooling	Global Average Pooling	
FC1	$(512, 512)$	

(b) Variance transfer network ϕ

Layer	ResNet_{+}
FC1	(512 $\times$ 4, 64)
LeakyReLU	-
FC2	(64, 512)

(c) Manifold regularization δ

Layer	ResNet_{+}
FC1_W_{δ}	(512, 64)
$\text{FC2\_W}_{\delta}^{T}$	(64, 512)

2. Decoder Architecture for Visualization

The decoder architecture used in our experiment for the proof of concept is described in Table 3. Given pre-trained backbone network from our MVT meta-training, we trained the decoder by feeding the feature embedding e and reconstructing the original 112×112 input facial images. For each convolutional block, instead of deconvolution, we used a $2\times$ upsampling by nearest neighbor interpolation followed by a 3×3 convolution. We additionally performed a instance normalization (Ulyanov et al., 2016) and a nonlinear

Table 3. Decoder network architecture for image visualization from a feature embedding e.

Layer	Decoder
FC1	$(512, 7 \times 7 \times 512)$
conv1	$\begin{bmatrix} 2 \times \text{Upsample} \\ 3 \times 3, 256 \\ \text{InstanceNorm} \\ \text{ReLU} \end{bmatrix}$
conv2	$\begin{bmatrix} 2 \times \text{Upsample} \\ 3 \times 3, 128 \\ \text{InstanceNorm} \\ \text{ReLU} \end{bmatrix}$
conv3	$\begin{bmatrix} 2 \times \text{Upsample} \\ 3 \times 3, 64 \\ \text{InstanceNorm} \\ \text{ReLU} \end{bmatrix}$
conv4	$\begin{bmatrix} 2 \times \text{Upsample} \\ 3 \times 3, 32 \\ \text{InstanceNorm} \\ \text{ReLU} \end{bmatrix}$
conv5	$[3 \times 3, 3]$

ReLU for each block. The final output is a $112 \times 112 \times 3$ image.

References

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.