Few-shot Compositional Font Generation with Dual Memory – Appendix

A Network Architecture Details

A.1 Memory addressors

Algorithm 1: Unicode-based Korean letter decomposition function

```
Input: A character label $y_c$
Output: Component labels $u_1^c$, $u_2^c$, and $u_3^c$
Data: The number of components for each $i$-th component type $N_i$.
unicode = ToUnicode($y_c$)
// 0xAC00 is the initial Korean Unicode
code = unicode - 0xAC00
$u_3^c$ = code mod $N_3$
$u_2^c$ = (code div $N_3$) mod $N_2$
$u_1^c$ = code div ($N_3 \times N_2$)
```

The memory addressor converts character label y_c to the set of component labels u_i^c by the pre-defined decomposition function $f_d : y_c \mapsto \{u_i^c \mid i = 1 \dots M_c\}$, where u_i^c is the label of i-th component of y_c and M_c is the number of sub-glyphs for y_c. In this paper, we employ Unicode-based decomposition functions specified to each language. We describe the decomposition function for Korean script as an example in Algorithm 1. The function disassembles a character into component labels by the pre-defined rule. On the other hand, each Thai character consists of several Unicodes, each of which corresponds to one component. Therefore, each Unicode constituting the letter is a label itself. The Thai decomposition function only needs to determine the component type of each Unicode.

A.2 Network architecture

The proposed architecture has two important properties: global-context awareness and local-style preservation. Global-context awareness allows the relational reasoning between components to the network, boosting to disassemble source glyphs into sub-glyphs and assemble them to the target glyph. Local-style preservation indicates that the local style of source glyph is reflected in the target.

For the global-context awareness, the encoder adopts global-context block (GCBlock) [3] and self-attention block (SABlock) [19,17], and the decoder employs hourglass block (HGBlock) [16,12]. These blocks extend the receptive field globally and facilitate relational reasoning between components while preserving locality. For the local-style preservation, the network handles multi-level features based on the dual memory framework. The specific architecture overview is described visually in Figure A.1.

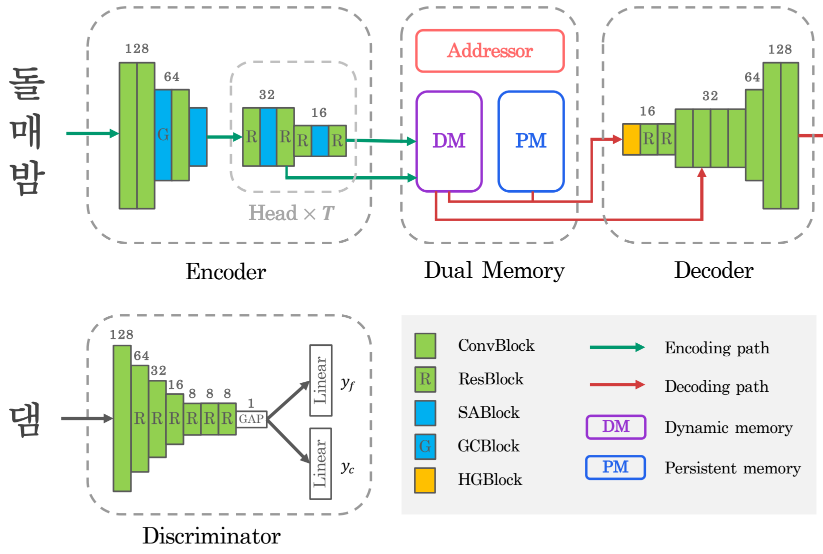

Fig. A.1: The encoder holds multiple heads according to the number of component types T. We denote the spatial size of each block in the figure.

The generator consists of five modules; convolution block (ConvBlock), residual block (ResBlock), self-attention block, global-context block, and hourglass block. Our SABlock is adopted from Transformer [17] instead of SAGAN [19], *i.e.*, the block consists of multi-head self-attention and position-wise feed-forward. We also use two-dimensional relative positional encoding from [1]. The hourglass block consists of multiple convolution blocks and downsampling or upsampling operation follows each block. Through hourglass structure, the spatial size of the feature map is reduced to 1×1 and restored to the original size, which extends the receptive field globally preserving locality. The channel size starts at 32 and doubles as blocks are added, up to 256 for the encoder and 512 for the decoder.

We employ a simple structure for the discriminator. Several residual blocks follow the first convolution block. Like the generator, the channel size starts at 32 and doubles as blocks are added, up to 1024. The output feature map of the last residual block is spatially squeezed to 1×1 size and it is fed to the two linear, font and character discriminators. Each discriminator is a multi-task discriminator that performs binary classification for each target class. Therefore, the font discriminator produces $|\mathbb{Y}_f|$ binary outputs and the character discriminator produces $|\mathbb{Y}_c|$ binary outputs, where $|\mathbb{Y}|$ denotes the number of target classes.

Since the persistent memory (PM) is independent of local styles, we set the size of PM same as the size of high-level features, the final output of the encoder, *i.e.*, 16×16. The learned embedding is refined via three convolution blocks, added to the high-level features of dynamic memory (DM), and then fed to the decoder. The component classifier comprises two residual blocks and one linear layer and identifies the class of the high-level component features from the DM.

B Experimental Setting Details

B.1 DM-Font implementation details

We use Adam [10] with a learning rate of 0.0002 for the generator and 0.0008 for the discriminator, following the two time-scale update rule [7]. The component classifier use same learning rate with the generator. The discriminator adopts spectral normalization [15] for the regularization. We train the model with hinge GAN loss [19,15,2,14,11] during 200K iterations. We employ exponential moving average of the generator [8,18,14,9]. For the Thai-printing dataset, we use a learning rate of 0.00005 for the generator and 0.0001 for the discriminator with 250K training iterations while other settings are same as the Korean experiments.

B.2 Evaluation classifier implementation details

Two different ResNet-50 [5] are separately trained for the content and the style classifiers with Korean and Thai scripts. The classifiers are optimized using the Adam optimizer [10] with 20 epochs. We expect that more recent Adam variants, *e.g.*, RAdam [13] or AdamP [6], further improve the classifier performances. The content classifier is supervised to predict a correct character, while the style classifier is trained to predict a font label. We randomly use 85% of the data points as the train data and the remained data points are used for the validation. Unlike the DM-Font training, this strategy shows all characters and fonts to classifiers. In our experiment, every classifier achieves over 96% of validation accuracy: 97.9% Korean content accuracy, 96.0% Korean style accuracy, 99.6% Thai content accuracy, and 99.99% Thai style accuracy. Note that all classifiers are only used for the evaluation but not for the DM-Font training.

B.3 User study details

30 different styles (fonts) from the Korean-unrefined dataset are selected for the user study. We randomly choose 8 characters for each style and generate the characters with four different methods: EMD [21], AGIS-Net [4], FUNIT [14], and DM-Font (ours). We also provide ground truth characters for the selected characters to the users for the comparison. Users chose the best method in 3 different criteria (content / style / preference). For each question, we randomly shuffle the methods to keep anonymity of methods. To sum up, we got 3,420 responses from 38 Korean natives with 30×3 items.

C Additional Results

C.1 Reference set sensitivity

In all experiments, we select the few-shot samples randomly while satisfying the compositionality. Here, we show that the reference sample selection sensitivity of the proposed method. Table C.1 shows the Korean-handwriting generation results of the eight different runs with different sample selections. The results support that DM-Font is robust to the reference sample selection.

Table C.1: **Reference sample sensitivity.** Eight different runs of DM-Font with different reference samples in the Korean-handwriting dataset.

	Pixel-level		Content-aware			Style-aware		
	SSIM	MS-SSIM	Acc(%)	PD	mFID	Acc(%)	PD	mFID
Run 1	0.704	0.457	98.1%	0.018	22.1	64.1%	0.038	34.6
Run 2	0.702	0.452	98.8%	0.016	19.9	64.2%	0.038	37.2
Run 3	0.701	0.456	98.0%	0.018	23.4	66.0%	0.037	35.2
Run 4	0.702	0.451	97.8%	0.019	22.9	65.0%	0.038	36.7
Run 5	0.701	0.453	98.2%	0.018	22.9	64.8%	0.038	36.4
Run 6	0.703	0.460	97.2%	0.020	24.8	67.8%	0.036	34.0
Run 7	0.700	0.447	98.3%	0.018	21.9	64.8%	0.037	36.6
Run 8	0.701	0.451	98.2%	0.018	22.2	65.8%	0.037	35.4
Avg.	0.702	0.453	98.1%	0.018	22.5	65.3%	0.037	35.8
Std.	0.001	0.004	0.4%	0.001	1.4	1.2%	0.001	1.1

Table C.2: **Quantatitive Evaluation on the Korean-unrefined dataset.** Higher is better, except perceptual distance (PD) and mFID.

	Pixel-level		Content-aware		Style-aware	
	SSIM	MS-SSIM	PD	mFID	PD	mFID
EMD [21]	0.716	0.340	0.106	99.2	0.079	93.3
FUNIT [14]	0.711	0.311	0.080	87.0	0.066	79.4
AGIS-Net [4]	0.708	0.334	0.052	67.2	0.089	134.5
DM-Font (ours)	**0.726**	**0.387**	**0.048**	**46.2**	**0.046**	**31.5**

C.2 Results on the Korean-unrefined dataset

Table C.2 shows the quantitative evaluation results of the Korean-unrefined dataset used for the user study. We use the classifiers trained by the Korean-handwriting dataset for the evaluation. Hence, we only report the perceptual distance and mFID while accuracies are not measurable by the classifiers. In all evaluation metrics, DM-Font consistently shows the remarkable performance as other datasets. The example visual samples are shown in Figure C.1.

C.3 Ablation study

Table C.3 shows the full ablation study results including all evaluation metrics. As the observations in the main manuscript, all metrics show similar behavior with the averaged accuracies; our proposed components and objective functions significantly improve the generation quality.

Table C.3: **Ablation studies on the Korean-handwriting dataset.** Higher is better, except perceptual distance (PD) and mFID.

(a) Impact of components. DM, PM, and Comp. G denote dynamic memory, persistent memory, and compositional generator, respectively.

	Pixel-level		Content-aware			Style-aware		
	SSIM	MS-SSIM	Acc(%)	PD	mFID	Acc(%)	PD	mFID
Evaluation on the **seen** character set during training								
Baseline	0.689	0.373	96.7	0.026	33.6	6.5	0.084	132.7
+ DM	0.702	0.424	**99.7**	**0.015**	**19.5**	31.8	0.060	77.6
+ PM	**0.704**	0.435	97.7	0.020	26.9	46.6	0.049	57.1
+ Comp. G	**0.704**	**0.457**	98.1	0.018	22.1	**64.1**	**0.038**	**34.6**
Evaluation on the **unseen** character set during training								
Baseline	0.693	0.375	96.6	0.027	34.3	6.5	0.084	134.8
+ DM	0.705	0.423	**99.8**	**0.015**	**19.5**	32.3	0.060	81.0
+ PM	**0.707**	0.432	97.6	0.022	28.9	45.9	0.050	61.4
+ Comp. G	**0.707**	**0.455**	98.5	0.018	20.8	**62.6**	**0.039**	**40.5**

(b) Impact of objective functions.

	Pixel-level		Content-aware			Style-aware		
	SSIM	MS-SSIM	Acc(%)	PD	mFID	Acc(%)	PD	mFID
Evaluation on the **seen** character set during training								
Full	**0.704**	**0.457**	**98.1**	**0.018**	**22.1**	**64.1**	**0.038**	**34.6**
Full $-\mathcal{L}_{l1}$	0.695	0.407	97.0	0.022	27.9	53.4	0.046	48.3
Full $-\mathcal{L}_{feat}$	0.699	0.427	97.8	0.020	23.8	51.4	0.047	51.4
Full $-\mathcal{L}_{cls}$	0.634	0.223	3.0	0.488	965.3	16.2	0.082	118.9
Evaluation on the **unseen** character set during training								
Full	**0.707**	**0.455**	**98.5**	**0.018**	**20.8**	**62.6**	**0.039**	**40.5**
Full $-\mathcal{L}_{l1}$	0.697	0.401	97.5	0.023	26.8	54.3	0.046	52.3
Full $-\mathcal{L}_{feat}$	0.701	0.423	97.8	0.020	24.1	51.2	0.048	56.0
Full $-\mathcal{L}_{cls}$	0.636	0.220	3.2	0.486	960.7	15.9	0.082	123.7

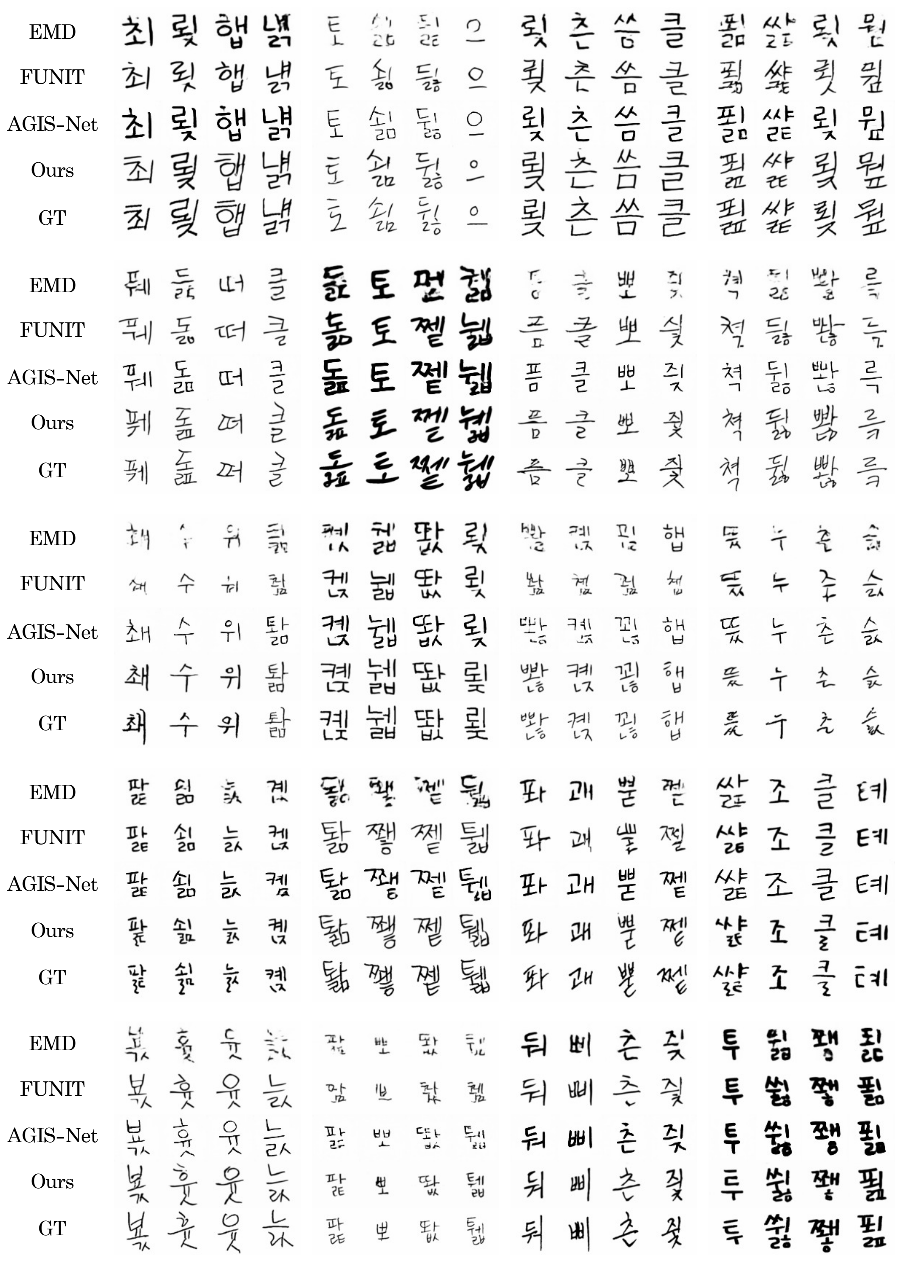

Fig. C.1: **Samples for the user study.** The Korean-unrefined dataset is used.

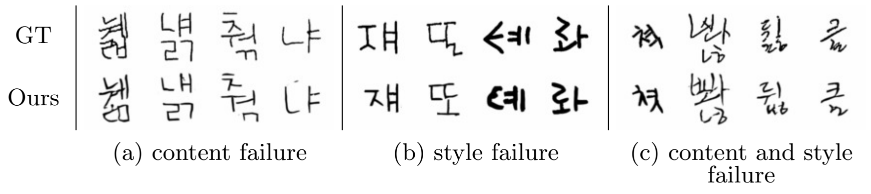

Fig. C.2: **Failure cases.** Examples of generated samples by DM-Font with incorrect content or insufficient stylization.

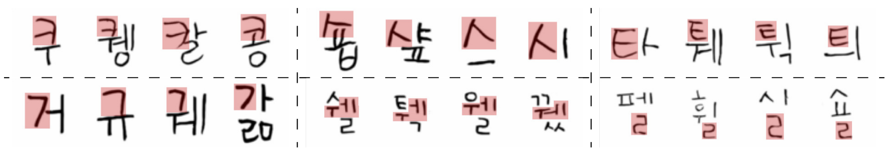

Fig. C.3: **Varying shape of single component.** Six visual examples show the variety of a component (with red boxes) across different characters. The results show that DM-Font generates samples with various component shapes by the compositionality.

C.4 Failure cases

We illustrate the three failure types of our method in Figure C.2. First, DM-Font can fail to generate the glyphs with the correct content due to the high complexity of the glyph. For example, some samples lose their contents – See from the first to the third column of Figure C.2 (a). In practice, developing a content failure detector and a user-guided font correction system can be a solution. Another widely-observed failure case caused by the multi-modality of components, *i.e.*, a component can have multiple styles. Since our scenario assumes that a model only observes one sample for each component, the problem is often ill-posed. Similar ill-posedness problem is also occurred in the colorization problem, and usually addressed by a human-guided algorithm [20]. Similarly, a user-guided font correction algorithm will be an interesting future research.

Finally, we report the cases caused by the errors in the ground truth samples. Note that the samples in Figure C.2 are generated by the Korean-unrefined dataset which can include inherent errors. When the reference glyphs are damaged as the two rightmost samples in the figure, it is difficult to disentangle style and content from the reference set. Due to the strong compositionality regularization by the proposed dual memory architecture, our model tries to use the memorized local styles while ignoring the damaged reference style.

C.5 Examples of various component shape in generated glyphs

We provide more examples of the generated samples by DM-Font with the same component in Figure C.3. The figure shows that the shape of each component

Table C.4: **Style generalization gap on the Korean-handwriting dataset.** We compute the differences of style-aware scores between seen and unseen font sets. The evaluation uses unseen character set. Smaller gap indicates better generalization.

	Seen fonts			Unseen fonts			Gap		
	Acc(%)	PD	mFID	Acc(%)	PD	mFID	Acc(%)	PD	mFID
Evaluation on the **Korean-handwriting** dataset									
EMD	74.0	0.032	31.9	5.2	0.089	139.6	68.9	0.057	107.8
FUNIT	**98.6**	**0.015**	**8.3**	5.6	0.087	149.5	93.0	0.072	141.2
AGIS-Net	95.8	0.018	13.4	7.5	0.089	146.1	88.3	0.071	132.7
DM-Font	82.1	0.026	16.9	**62.6**	**0.039**	**40.5**	**19.5**	**0.013**	**23.6**
Evaluation on the **Thai-printing** dataset									
EMD	**99.5**	**0.001**	**1.0**	3.4	0.087	171.6	96.1	0.086	170.6
FUNIT	97.0	0.004	5.0	4.7	0.084	166.9	92.4	0.080	161.9
AGIS-Net	84.6	0.016	28.6	15.8	0.074	145.1	68.8	0.058	116.5
DM-Font	90.2	0.009	13.5	**50.6**	**0.037**	**69.6**	**39.6**	**0.029**	**56.1**

is varying by different sub-glyphs compositions as described in Figure 2 of the main manuscript. Note that all components are observed a few times (usually once) as the reference. These observations support that our model does not simply copy the reference components, but can properly extract local styles and combine them with global composition information and intrinsic shape stored in persistent memory. To sum up, we conclude that DM-Font disentangles local style and global composition information well, and generates the high quality font library with only a few references.

C.6 Generalization gap between seen and unseen fonts

We provide additional benchmarking results on the seen fonts in Table C.4. Note that Table 1 and 2 in the main manuscript are measured in the unseen fonts only. Simply, "seen fonts" can be interpreted as the training performances, and "unseen fonts" as the validation performances. The comparison methods such as EMD [21], FUNIT [14], AGIS-Net [4], show remarkably good results on the training data (seen fonts) but fail to generalize the performance on the validation set (unseen fonts). We also report the generalization gap between the seen and unseen fonts in Table C.4. The results show that comparison methods suffer from the style memorization issue, what we discussed in the nearest neighbor analysis, and cannot be generalizable to the unseen font styles. In contrast, our method shows significantly better generalization gap comparing to others.

References

1. Bello, I., Zoph, B., Vaswani, A., Shlens, J., Le, Q.V.: Attention augmented convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3286–3295 (2019) 2
2. Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. In: International Conference on Learning Representations (2019) 3
3. Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H.: GCNet: Non-local networks meet squeeze-excitation networks and beyond. In: IEEE International Conference on Computer Vision Workshops (2019) 1
4. Gao, Y., Guo, Y., Lian, Z., Tang, Y., Xiao, J.: Artistic glyph image synthesis via one-stage few-shot learning. ACM Transactions on Graphics (2019) 3, 4, 8
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (2016) 3
6. Heo, B., Chun, S., Oh, S.J., Han, D., Yun, S., Uh, Y., Ha, J.W.: Slowing down the weight norm increase in momentum-based optimizers. arXiv preprint arXiv:2006.08217 (2020) 3
7. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems (2017) 3
8. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (2018) 3
9. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2019) 3
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015) 3
11. Lim, J.H., Ye, J.C.: Geometric gan. arXiv preprint arXiv:1705.02894 (2017) 3
12. Lin, T.Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (2017) 1
13. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. In: International Conference on Learning Representations (2020) 3
14. Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J.: Few-shot unsupervised image-to-image translation. In: IEEE International Conference on Computer Vision (2019) 3, 4, 8
15. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (2018) 3
16. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: European Conference on Computer Vision (2016) 1
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017) 1, 2
18. Yazıcı, Y., Foo, C.S., Winkler, S., Yap, K.H., Piliouras, G., Chandrasekhar, V.: The unusual effectiveness of averaging in GAN training. In: International Conference on Learning Representations (2019) 3

19. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative adversarial networks. In: International Conference on Machine Learning (2019) 1, 2, 3
20. Zhang, R., Zhu, J.Y., Isola, P., Geng, X., Lin, A.S., Yu, T., Efros, A.A.: Real-time user-guided image colorization with learned deep priors. ACM Transactions on Graphics (2017) 7
21. Zhang, Y., Zhang, Y., Cai, W.: Separating style and content for generalized style transfer. In: IEEE Conference on Computer Vision and Pattern Recognition (2018) 3, 4, 8