Supplementary Material
for
Multi-Content GAN for Few-Shot Font Style Transfer

Samaneh Azadi[1]*, Matthew Fisher[2], Vladimir Kim[2], Zhaowen Wang[2], Eli Shechtman[2], Trevor Darrell[1]
[1]UC Berkeley, [2]Adobe Research
{sazadi,trevor}@eecs.berkeley.edu {matfishe,vokim,zhawang,elishe}@adobe.com

Figure 1: Example synthetic color gradient fonts

1. Font Dataset

To create a baseline dataset of ornamented fonts, we apply random color gradients and outlining on the gray-scale glyphs, two random color gradients on each font of our collected 10K examples, resulting in a 20K color font dataset. A few examples are shown in Figure 1. Size of this data set can be arbitrarily increased through generating more random colors. These gradient fonts do not have the same distribution as in-the-wild ornamentations but can be used for applications such as network pre-training.

2. Network Architectures

We employ our generator (encoder-decoder) architecture based on the image transformation network introduced in [2] and discussed in [1]. We represent a Convolution-BatchNorm-ReLU consisting of k channels with `CRk`, a Convolution-BatchNorm layer with `Ck`, a Convolution-BatchNorm-ReLU-Dropout with `CRDk`, and a Convolution-LeakyReLU with `CLk`. In the above notations, all input channels are convolved to all output channels in each layer. We also use another Convolution-BatchNorm-ReLU block in which each input channel is convolved with its own set of filters and denote it by CR^{26}k, where 26 shows the number of such groups. Dropout rate during training is 50% while ignored at test time. Negative slope of the Leaky ReLU is also set to 0.2.

*Work done during an internship at Adobe Research

2.1. Generators Architecture

Our encoder architecture in GlyphNet is: CR^{26}26-CR64-CR192-CR576-(CRD576-C576)-(CRD576-CR576)-(CRD576-C576) where convolutions are down-sampling by a factor of $1-1-2-2-1-1-1-1-1-1$, respectively, and each `(CRD576-C576)` pair is one ResNet Block.

The encoder in OrnaNet follows a similar network architecture except for in its first layer where the CR^{26}26 has been eliminated.

The decoder architecture in both GlyphNet and OrnaNet is as follows: `(CRD576-C576)-(CRD576-C576)-(CRD576-C576)` - `CR192-CR64` each up-sampling by a factor of $1-1-1-1-1-1-2-2$, respectively. Another Convolution layer with 26 channels followed by a `Tanh` unit is then applied in the last layer of the decoder.

2.2. Discriminators Architecture

As mentioned in the paper in Figure 1, our GlyphNet and OrnaNet discriminators, D_1 and D_2, consist of a local and global discriminator where weights of the local discriminator is shared with the latter. The local discriminator consists of `CL64-CL128` followed by a convolution mapping its 128 input channels to one output. Convolutions here are down-sampling by a factor of $2-1-1$, respectively. The global discriminator has two additional layers before joining the layers in the local discriminator as `CR52-CR52` each down-sampling by a factor of 2. Receptive field size of our local discriminator is 21 while global discriminator covers a larger area than the 64 pixels in the image domain, and thus can capture a global information from each image.

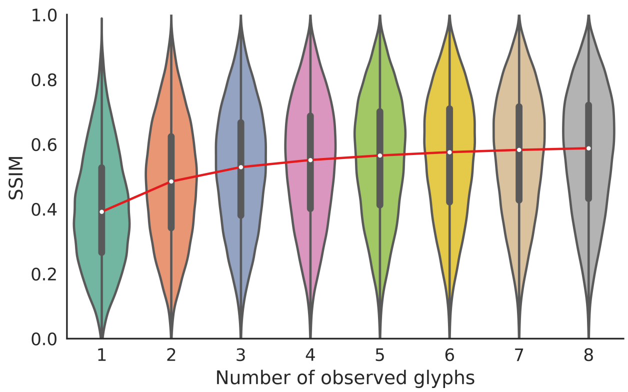

Figure 2: Effect of number of observed glyphs on the quality of GlyphNet predictions. Red line is passing through median of each distribution.

3. Experiments and Results

3.1. Automatic Learning of Correlations between Contents

Automatic learning of the correlations existing between different letters is a key factor in transferring style of the few observed letters in our multi-content GAN. In this section, we study such correlations through the structural similarity (SSIM) metric on a random subset of our 10K font data set consisting of 1500 examples. For each instance, we randomly keep one of the 26 glyphs and generate the rest through our pre-trained GlyphNet.

Computing the structural similarity between each generated glyph and its ground truth, we find 25 distributions over its SSIM scores when a single letter has been observed at a time. In Figure 3, we illustrate the distributions $\alpha|\beta$ of generating letter α when letter β is observed (in blue) vs when any other letter rather than β is given (in red). Distributions for the two most informative given letters and the two least informative ones in generating each of the 26 letters are shown in this figure. For instance, looking at the fifth row of the figure, letters *F* and *B* are the most constructive in generating letter *E* compared with other letters while *I* and *W* are the least informative ones. As other examples, *O* and *C* are the most guiding letters for constructing *G* as well as *R* and *B* for generating *P*.

3.2. Number of Observed Letters

Here, we investigate the dependency of quality of GlyphNet predictions on the number of observed letters. Similar to Section 3.1, we use a random subset of our font data set with 1500 example fonts and for each font, we generate 26 letters given n observed ones from our pre-trained GlyphNet. The impact of changing n from 1 to 8 on the distribution of SSIM scores between each unobserved letter and its ground truth is shown in Figure 2. The slope of the red line passing through the median of each distribution is decreasing as n increases and reaches to a stable point once the number of observations for each font is close to 6. This study confirms the advantage of our multi-content GAN method in transferring style when we have very few examples.

3.3. Results on Synthetic Color Font Dataset

In this section, we compare our end-to-end multi-content GAN approach with the image translation method discussed in Section 5.1 of the paper. In Figure 4, we demonstrate the results on multiple examples from our color font data set where we have applied random color gradients on the gray-scale glyph outlines. By looking at the nearest neighbor examples, we have made sure that the fonts shown in this figure were not used during training of our Glyph Network.

Given a subset of color letters in the input stack of GlyphNet with dimension $1 \times 78 \times 64 \times 64$ including RGB channels, we generate all 26 RGB letters from the pre-trained GlyphNet on our color font data set. Results are denoted as "Image Translation" in Figure 4. Our MC-GAN results are outputs of our end-to-end model fine-tuned on each exemplar font. The image translation method cannot generalize well in transferring these gradient colors at test time by observing only a few examples although other similar random patterns have been seen during training.

3.4. Perceptual Evaluation

As mentioned in Section 5.3 of the paper, we evaluate performance of our end-to-end model in transferring ornamentations by comparing its output against the patch-based model of [3]. Here, glyph outlines for both methods are generated through our pre-trained GlyphNet. We do this evaluation on 33 fonts downloaded from the web[1] and ask 11 users to choose outputs of one of the models by observing the subset of given letters. Full results and percentage of user preferences to each method are represented in Figures 5, 6, 7, 8, with an overall 80% preference to our MC-GAN.

References

[1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016. 1

[2] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 1

[3] S. Yang, J. Liu, Z. Lian, and Z. Guo. Awesome typography: Statistics-based text effects transfer. *arXiv preprint arXiv:1611.09026*, 2016. 2, 5, 6, 7, 8

[1] http://www6.flamingtext.com/All-Logos

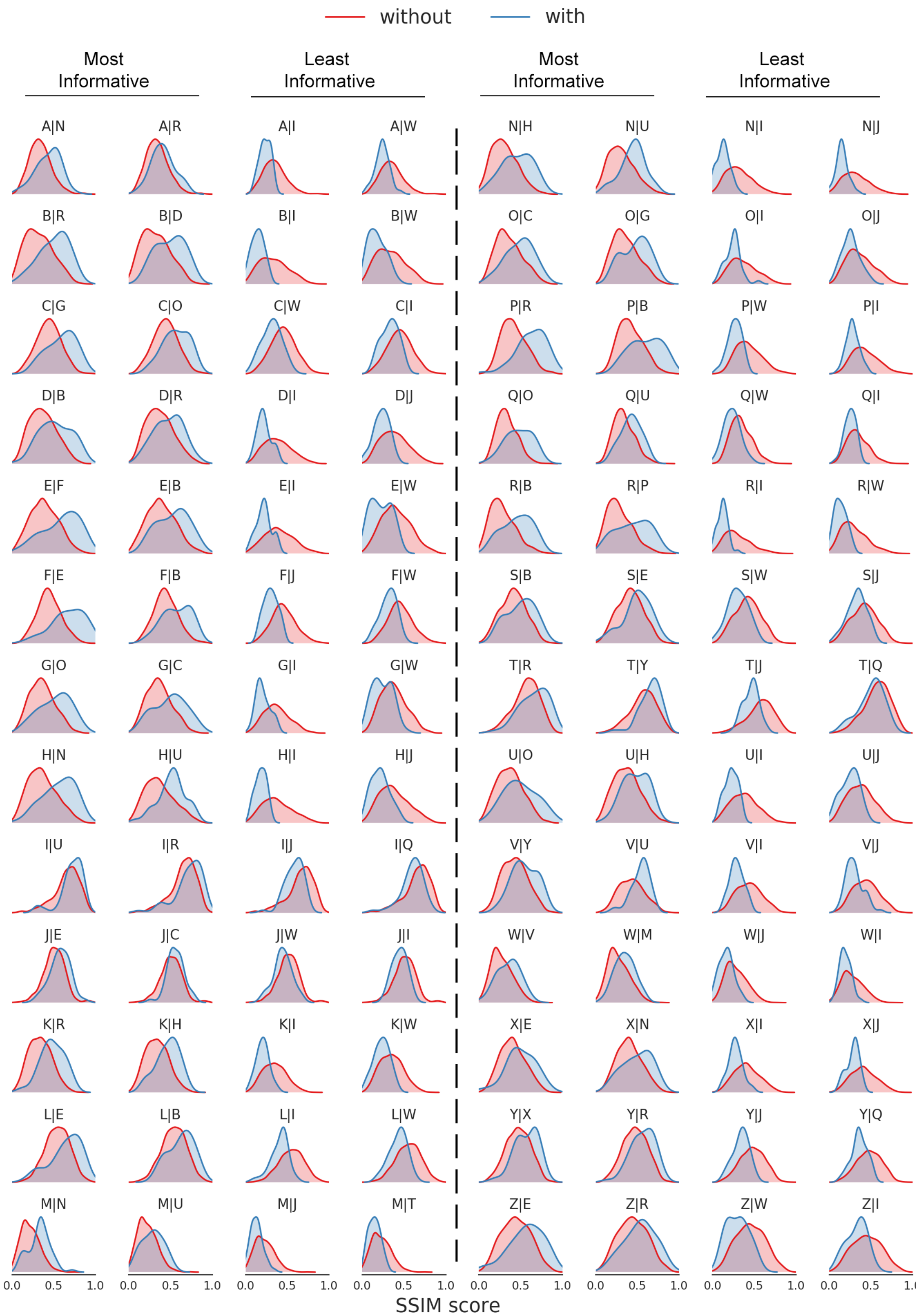

Figure 3: Distributions $(\alpha|\beta)$ over SSIM scores for generating letter α given β in blue and given any other letter rather than β in red. Distributions for the most informative given letters β in generating each glyph α is shown in the left of each column while the least informative givens are presented in the right.

Figure 4: Comparison between image translation and our end-to-end multi-content GAN on our synthetic color font data set. For each example, ground truth and given letters are shown in the **1st row**, image translation outputs in the **2nd row** and MC-GAN in the **last row**.

Figure 5: Comparison of our end-to-end MC-GAN model (**3rd rows**) with the text effect transfer approach [3] using GlyphNet synthesized glyphs (**2nd rows**). Ground truth glyphs and the observed subset are illustrated in the **1st row** of each example font. Scores next to each example reveal the percentage of people who preferred the given results.

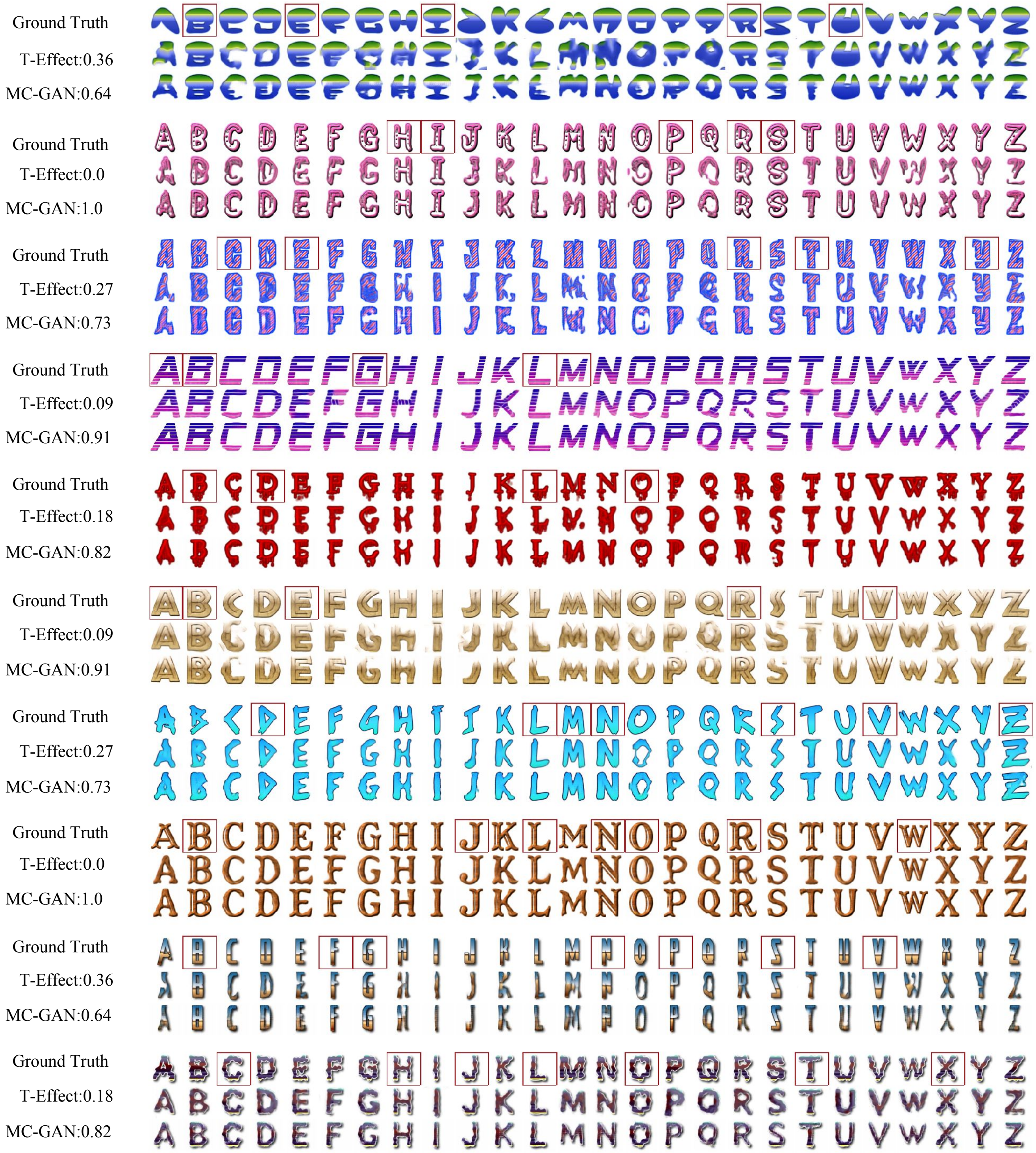

Figure 6: Continue - Comparison of our end-to-end MC-GAN model (**3rd rows**) with the text effect transfer approach [3] using GlyphNet synthesized glyphs (**2nd rows**). Ground truth glyphs and the observed subset are illustrated in the **1st row** of each example font. Scores next to each example reveal the percentage of people who preferred the given results.

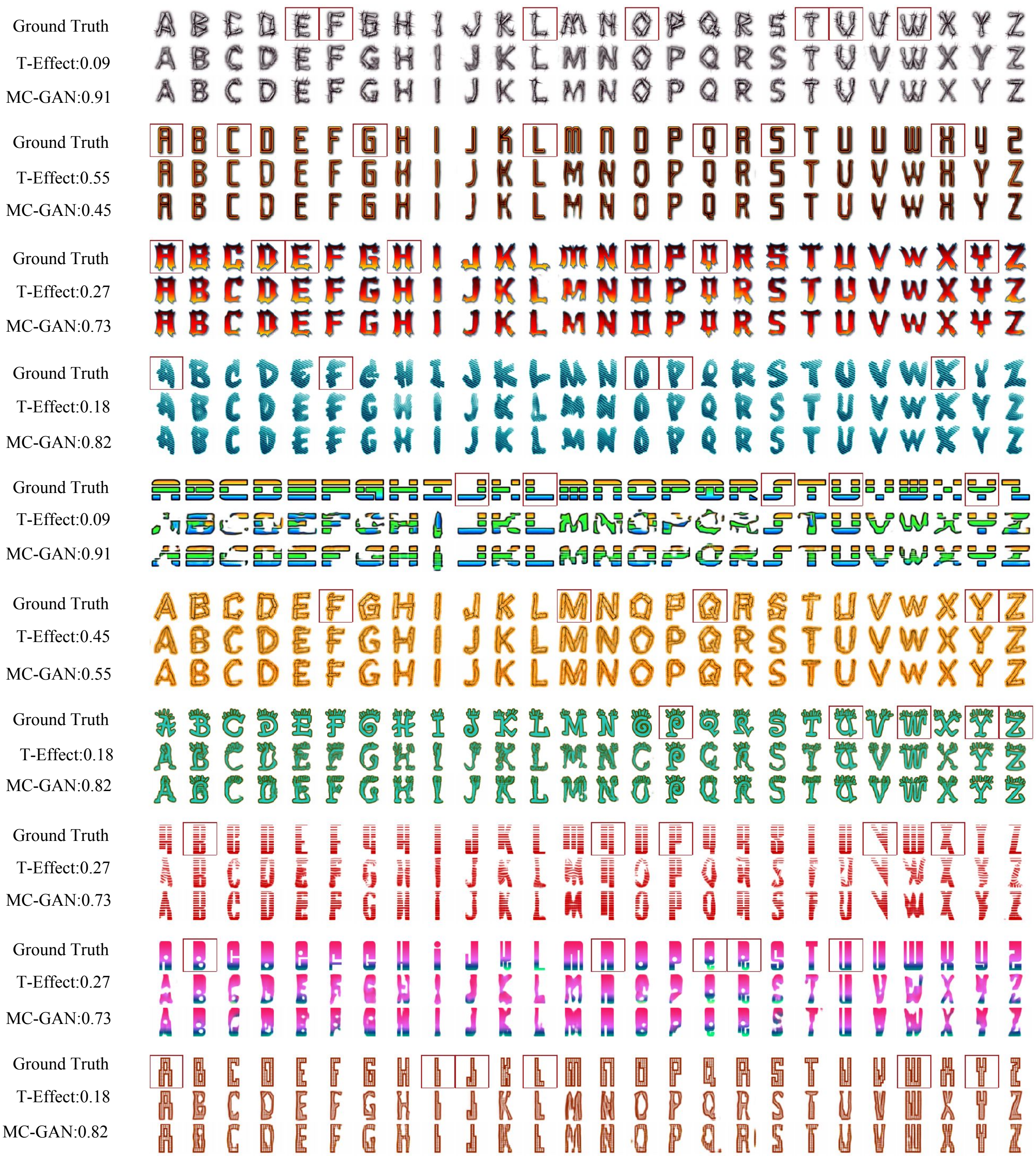

Figure 7: Continue - Comparison of our end-to-end MC-GAN model (**3rd rows**) with the text effect transfer approach [3] using GlyphNet synthesized glyphs (**2nd rows**). Ground truth glyphs and the observed subset are illustrated in the **1st row** of each example font. Scores next to each example reveal the percentage of people who preferred the given results.

Figure 8: Continue - Comparison of our end-to-end MC-GAN model (**3rd rows**) with the text effect transfer approach [3] using GlyphNet synthesized glyphs (**2nd rows**). Ground truth glyphs and the observed subset are illustrated in the **1st row** of each example font. Scores next to each example reveal the percentage of people who preferred the given results.