A Controllable Artistic Style Transfer approach to Logo Synthesis

Abhirag Nagpure Shubham Gaikwad
Luddy School of Informatics, Computing and Engineering
Indiana University
{anagpure, shgaikwa}@iu.edu

1. Introduction

We investigate the use of generative adversarial networks for logo generation by artistic style transfer. In short, our aim is to migrate style from a source style image to a text or logo image to create artistic typography which can be used in poster creation, logo generation etc. The stylized logos must find a balanced between the deformity introduced by the style and the legibility of a logo that is logo / text should be identifiable. Such a delicate balance is subjective and hard to attain automatically. Therefore, a practical tool allowing users to control the stylistic degree of the glyph is of great value.

We aim to produce glyphs and logos with real-time control over the degree of stylistic deformations. The Shape-Matching GAN [10] paper claims to be the first paper to provide the real-time control of glyph deformations which is very important in text style transfer.

In this project, we take the initial setup for text style transfer given by [10] and use it to create artistic logos. We conduct experiments in multiple dimensions - input resolutions, network architectures, training pipelines and other areas. This generates diverse, controllable and high-quality stylized logos. Supplementary material is available at this link.

2. Background and related work

The aim of the project is to create stylize text based on the given style with good degree of control over the deformation degree. In artistic text style transfer the style specified by a reference image is rendered into the text as shown in [10]. This is particularly useful in many visual creation tasks such as poster and advertisement design.

The real-time control of glyph deformation is the novel idea in the [10]. We try to use this idea to create stylized logos with better quality. [10] tackles the following two issues in the text style transfer task :

1. Glyph deformation degree is subjective. The glyph must be deformed to certain degree but the text should be identifiable.
2. There is no large dataset available with text images and corresponding stylized text images to directly use in this problem.

Related work regarding this project can be divided into 3 parts as follows:

- Image style transfer:
  A lot of work has been done in the style transfer domain over the past few years. Currently a lot of development in the area is the result of generative adversarial network (GAN). GANs learn the style representation directly from the data and are able to produce more artistically rich results as shown in [5].

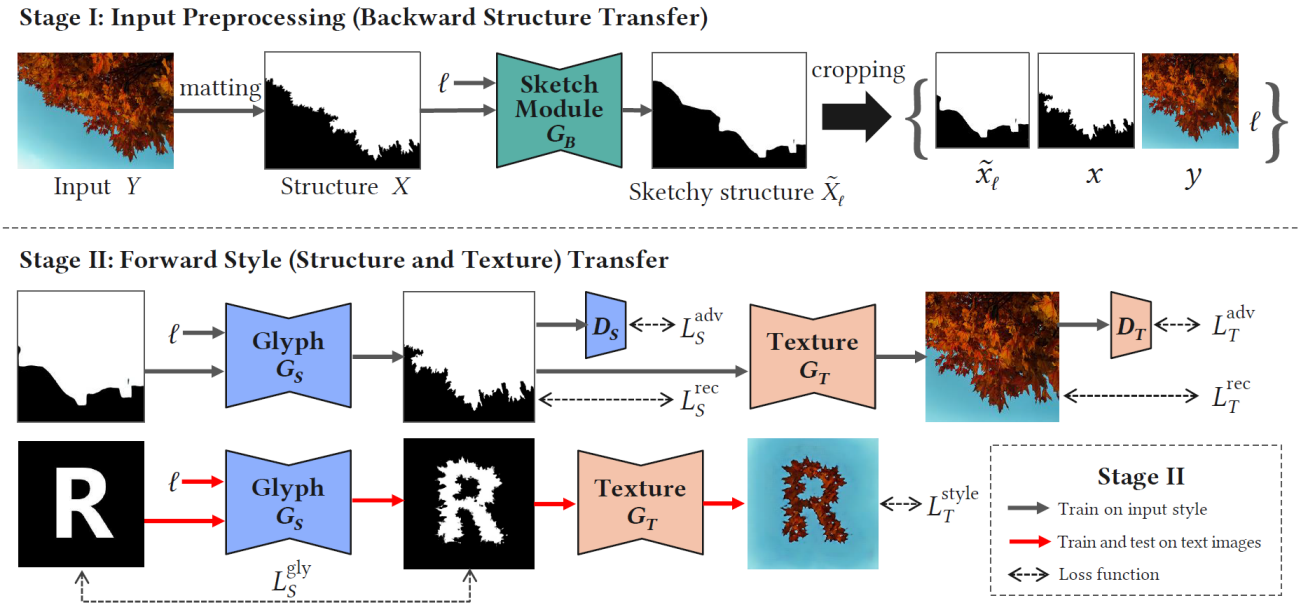
- Artistic text style transfer:
  [2] implemented fast text transfer which can only work on 26 capital English letters. [9] works on more similar task of stylizing the text with different textures and glyph deformation. [10] also uses a GAN-based approach to have a controllable parameter for glyph deformation.

- Multi-scale style control:
  Multi-scale style control deals with strength and stroke size of the texture. [9] deals with texture strength which is texture similarity in result and style image. [10] uses a different parameter glyph deformation degree which is unexplored in previous work.

Apart from the related work about the paper and the above mentioned reference papers, we also had to learn about the PyTorch framework as we were not familiar with PyTorch and wanted to learn about it through this project. We invested few days in understanding the PyTorch framework and completing the basic tutorials on it. Also, we undertook this project as a challenge to learn about and work in the exciting domain of GANs with only the basic theoretical knowledge about it. We read the original paper by Goodfellow [4] which pioneered these generative networks.

Also we studied some of the most popular applications of GANs like image-to-image translation using conditional

Figure 1. Architecture and pipeline as proposed by [10]



GANs [5] and read [6] which approaches the problem of neural style transfer, first tackled by Gatys [3].

3. Experiments

3.1. Dataset

The original paper uses a text dataset as proposed by [8]. This dataset consists of 708 English alphabets along with some glyph structures. Also 5 style images are given to reproduce the results. For our project as we wanted to generate artistic logos we downloaded the logo dataset from [7]. We randomly selected a set of 1052 logos. The logos were selected in such a way that the foreground and background were easily distinguishable and the matting was done using Adobe Photoshop. We asked the authors about the data preparation process [1] and followed the guidelines to preprocess the selected subset of logos. We used the same style images as given in the original paper along with two new styles.

3.2. Architecture details

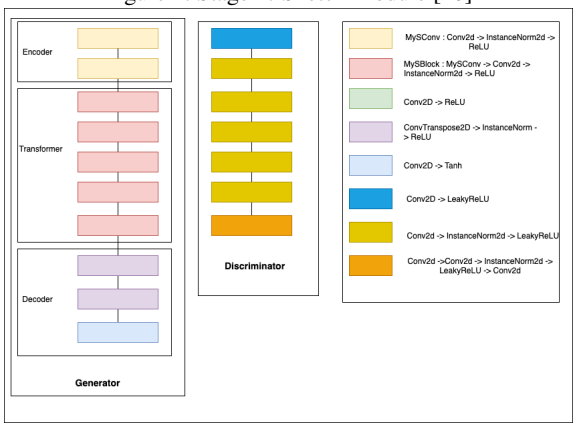
The architecture consists a bidirectional shape matching strategy which has two stages:

1. Input pre-processing (Backward Structure Transfer) : This stage contains a sketch module.
2. Forward Style (Structure and Texture) Transfer : This stage contains a glyph structure module and a texture module.

The paper assumes that the structure mask of the style object is given.

3.2.1 Stage 1 details

Figure 2. Stage 1: Sketch Module [10]



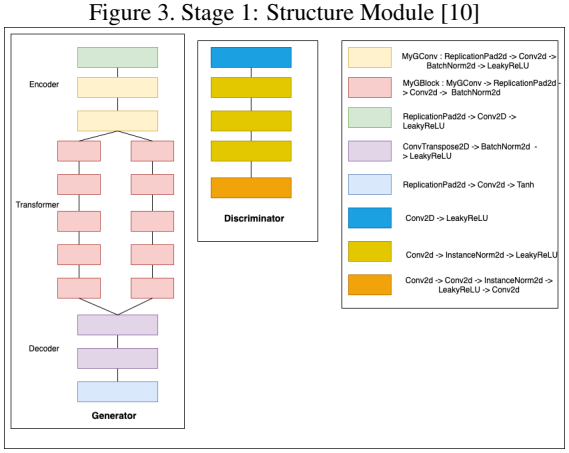
Stage 1 consists of the sketch module (G_B). The aim of this module is to transfer the glyph/logo characteristics to different X with corresponding coarse levels. The sketch module has two parts smoothness block and transformation block. The smoothness block has convolution layers with a Gaussian kernel whose standard deviation is a function of parameter l. This transformation block is trained to achieve structure transfer which maps the smoothed text images back to the text domain to learn the glyph / logo characteristics. This module is trained on the text data in [10]. We trained this module on the combination of text

dataset from [8] and our subset of logos. The results of this trained module are used further while training the glyph structure module and texture module in stage 2.

3.2.2 Stage 2 details

Stage 2 consists of glyph structure module and the texture module. As the names suggest, the structure module transfers structure and the texture module transfers the texture of the style image to the given text/logo image.

Figure 3. Stage 1: Structure Module [10]

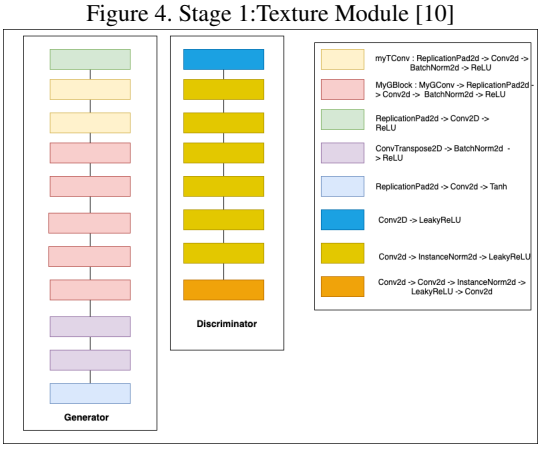


Glyph Module (G_S): This module is trained on a single style image at a time. Different level outputs based on the all the l values are first obtained by performing a forward pass through the trained sketch module. Then based on the l value randomly chosen during each epoch one image is chosen for each input image during each epochs. The training process is setup in such a way that the model first learns to capture the extreme deformities and then all the intermediate deformities by using the mechanism of Controllable Resblock. Randomly cropped sub image pairs of X and the respective results from sketch module are trained using controllable Resblock.

Controllable Resblock: G_S uses architecture inspired from StyleNet [6]. The middle layers of original Resblock in [6] are replaced by controllable resblock. The Controllable ResBlock is essentially a linear combination of two ResBlocks weighted by l. If l= 0 or l= 1 then G_S degrades into the original StyleNet, and is solely tasked with the greatest or tiniest shape deformation to avoid the many-to-one problem. Otherwise, G_S tries to compromise between the two extremes.

Texture module (G_T): This module uses the results of G_S and adds the texture of the original style image. Similar as in training G_S, random cropping is used to get enough training pairs and then G_T is trained using the reconstruction loss and conditional adversarial loss.

Figure 4. Stage 1:Texture Module [10]



Generators and Discriminators Generators are adapted from the Encoder-Decoder architecture of StyleNet [6] with six ResBlocks, except that G_S uses the proposed Controllable ResBlock instead. Discriminators are borrowed PatchGAN [5].

3.3. Experiments

3.3.1 Sketch module with logos

Our main task in this project was to create artistic logos. To achieve this task the main step was creating a logo dataset as per the requirement of the sketch module the training the sketch module so that it can learn to recreate the logos with different levels of smoothness based on the given l value.

The original training process for training the English alphabets and some glyphs was very particular in many aspects as the details specified in paper and the actual implementation had a few differences. For example the dataset also consisted of 5 images named augment images which were just squares of different sizes with 2 images cutting the square diagonally. These images were supposed to help during training of sketch module. We observed that according to given training process these augment images were used as almost half the data during each batch of training. For example if a batch is of 64 then these 5 images would be repeated to form 32 out of the 64 images with remaining being the training images.

We understood such training caveats in the given training style and experimented quite a lot to get a better results as this was the most crucial part to get this working on the logo dataset.

As we are not able to add all the experiments conducted during the training of this module we are only including the ones with significant progress :

1. Initially we did not change given generator discriminator architecture of the Glyph module. Instead we focused on understanding the impact of the augmented image during training process of logos.

2. We tried different size of augmented data in each epoch. Such that the augmented data can have half or one-forth or one-eighth data in every batch. The results with using the one-eighth data as augmented data was much better than using the one half augmented data in each batch. We also created few more augmentation and as the logo dataset is much more complex than the simple text data more augmentation helped in producing the better results.

3. We experimented with training with only the logo dataset and with the combination of logo and text dataset. The combination of the two is a better choice since intuitively, moving on to a more complex problem by tackling it in easier sub-problems seems to be a good idea.

4. Logos are much more complex as compared to text as logos have different very small patterns which model should capture. We experimented with the resolution size of the image. Originally in the paper the resolution used is 256*256. We tried with resolutions 128*128, 320*320 and 64*64 but did not find any significant improvement in the results.

5. Our other experiments involved changing the architecture of the generator and discriminator models. As the dataset is more complex we tried to make the model more suitable for this dataset using more number of feature maps, also by adding few more layers in architecture to make it more deep and by changing the number of training examples used during the each epoch.

6. The best parameters we found were with resolution = 128*128, increasing the initial generator features from 32 to 128, and the discriminator features from 32 to 64, increasing generator layers to 8, increasing number of images used in each epoch from 12800 to 25600, using additional augment Data with replacing one-eighth in every batch instead of one-half and using both the text data and logo dataset.

7. We also tried an experiment where we wanted model to learn first the more granular details and then the finer details in later epochs. We tried training for certain epochs with different image resolutions such as first two epochs with 320 * 320, next 2 with 256 * 256 and so on. Unfortunately this further degraded the results and we did not get enough time to follow up on this methodology.

3.3.2 Structure and texture module

1. Using the default setup for the structure and texture module gave us pretty good results. We further experimented with these two modules but as for overall training of these two modules almost takes 9 hours we were not able to conduct many experiments.

2. The given setup of structure module training consist of training for certain epochs with certain l value. Such that first 30 epochs are trained with $l = 1$, next 40 with l = random value from 0 to 1 except one and next 80 epochs with random value between 0 to 1. This trains the model in such a way that first the one part of Controllable Resblock is trained completely and then the other part starts learning and eventually after the 80 epochs the model learns to balance between two extremes based on current l value.

3. We experimented with just training normally with for each epoch just choosing the l value between 0 to 1 but this gave worse results than the sequential training as mentioned above.

4. As this training process was taking a lot of time we experimented with decreasing the number of layer and the amount of layers in models but this resulted in worse results so we continued with the given model architectures.

4. Results

During testing, the user needs to provide a logo, and the trained structure and texture module checkpoint files on a style for which we are generating the results. Once the training of structure and texture module is done the sketch module is no longer required during the testing with the corresponding style.

The results are as follow : All following results are generated on the Test dataset.

1. Results of original paper and best model on glyph / text. Refer Figure 5. Our results are better on the glyph / text dataset. Using the training with glyph data along with the logo dataset improved the results on the original glyph data also.

2. Results of original paper and best model on logos. Refer Figure 6. It can be clearly seen that our model gives much more realistic logos even at a higher deformity degree. The original paper trained on the text and Glyph dataset does not give good results for logo synthesis at higher value of l.

3. Results of best model with different l values Refer Figure 7. It can be seen that the deformity increases as increases in the value of l parameter. As the deformity increases further the stylized image formed contains more unnecessary patches which makes the logos undesirable or the text unreadable.

4. Some bad results on the logos with higher L values. Refer Figure 8. As the l value is increased, some logos start to deform and become unrecognizable. It is very hard to find a perfect balance which would give good results for all logos with different styles at all l values.

5. Discussions

1. The results obtained with different styles clearly shows that the artistic logo generation works very good when logos are less complex and both the foreground and the background are distinguishable.

2. This approach of artistic style transfer provides a good control over the amount of deformations of the logos and text.

3. The complex logos or the logos with finer details cannot lose the finer details while generating the artistic logos.

4. This method cannot be applied to any logo, as most of the logos have multiple colors or multiple overlapping layers. The input required for this task is the output of image matting produced via Photoshop which converts the foreground and background in the image into a distance-based binary representational image.

5. One of the important drawback of this method is it requires a separately trained model for each style. Thus to get results on a new style will take approximately 9 hours on a machine with a 16 GB GPU.

6. As seen by the results on different styles, the style image needs to be perfectly converted via the image matting algorithm. Otherwise the results may produce a cloud like shapes around the logos.

6. Conclusion

In this project, we took the initial setup for text style transfer given by [10]. We invested a lot of time in understanding this paper and the given code with its different caveats such as data pre-processing, training pipelines etc.

We created a subset of the logo dataset by applying image matting through Photoshop on the randomly selected 1052 logos. We conducted experiments in multiple dimensions - input resolutions, network architectures, training pipelines etc. This gave us fairly high-quality stylized logos with visible different deformity based on the value of deformity factor l. This is very useful in quickly providing the results for many visual creation tasks such as poster and advertisement design.

A future scope involves changing the model pipeline such that a new style can be fine-tuned over the previously trained model or more better approach would be if user only requires to pass the new style image as input to the trained model and get results in a single forward pass.

References

[1] https://github.com/tamu-vita/shapematchinggan/issues/16.

[2] Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer, 2017.

[3] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style, 2015.

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2016.

[6] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

[7] Alexander Sage, Eirikur Agustsson, Radu Timofte, and Luc Van Gool. Logo synthesis and manipulation with clustered generative adversarial networks, 2017.

[8] Shuai Yang, Jiaying Liu, Wenjing Wang, and Zongming Guo. Tet-gan: Text effects transfer via stylization and destylization, 2018.

[9] Shuai Yang, Jiaying Liu, Wenhan Yang, and Zongming Guo. Context-aware text-based binary image stylization and synthesis. *IEEE Transactions on Image Processing*, 28(2):952–964, Feb 2019.

[10] Shuai Yang, Zhangyang Wang, Zhaowen Wang, Ning Xu, Jiaying Liu, and Zongming Guo. Controllable artistic text style transfer via shape-matching gan, 2019.

Figure 5. Result comparison between our implementation and original paper on Glyph dataset [10]

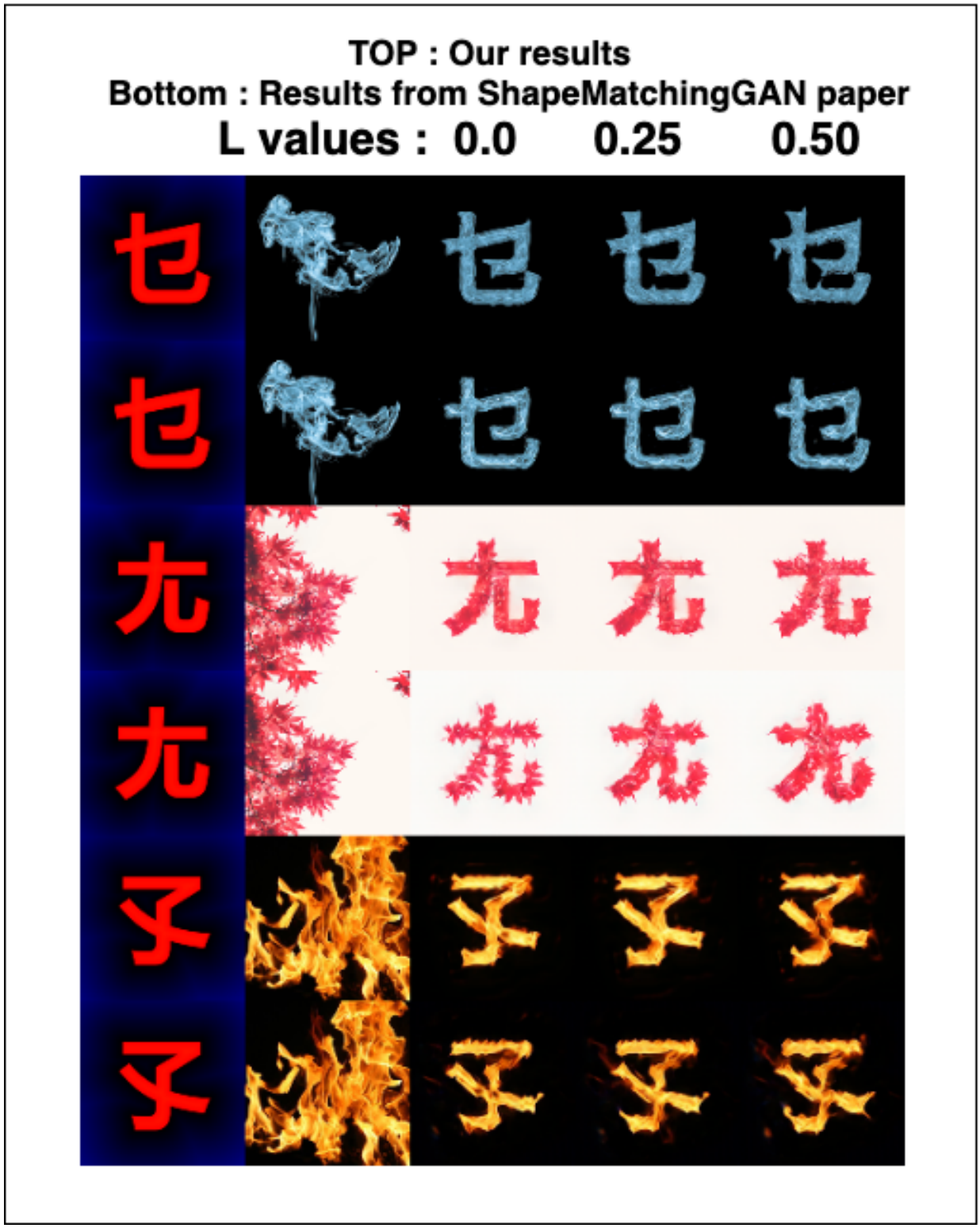


Figure 6. Result comparison between our implementation and original paper on logos [10]

Figure 7. Result comparison of our model at different l values such as (-0.75, -0.50, -0.25, 0.00, 0.25, 0.50) [10]

Figure 8. Result comparison of our model with original paper at high l values [10]