End-to-End Bangla Scene Text Editing System Development Using Deep Learning

Md Aminul Haque Palash[1], Sourav Saha[2], and Faria Afrin[3]

No Institute Given

Abstract. Numerous vision-based applications require the use of text because it gives rich and precise information. The Bangla Scene-Text-Editing (STE) system is a technology that provides a means of editing text in natural images or modifying words in images while still maintaining the image's authenticity. This is a tough process, as both the background and font styles must be retained such that the altered image is visually identical to the source image. The Bangla Scene-Text-Editing (STE) system is made up of three parts: the text conversion module, the background inpainting module, and the fusion module, which are all available. They work together to make an end-to-end system. This component modifies the text in the source image with the text in the targeted text while preserving the source image's original text style. The background inpainting module removes the text and replaces it with a texture that looks like the text area. The Fusion module brings together the information from the two previous modules and makes the altered text pictures that were made before. This is the very first attempt, as far as we aware, to change Bengali text in natural pictures at the word level, and it represents a significant milestone. On a synthetic dataset, both visually effective and quantitative findings were observed.

Keywords: Text Editing Network, Text Erasure, Text Synthesis, GAN

1 Introduction

A considerable part of the present era is structured to be read through the use of labels and other textual clues, and text is among the most basic tools for keeping and transferring information. Image/video text, often known as scene text, is a type of text that appears in images and movies. Scene text includes rich semantic, useful, and relevant information in a number of different vision-based applications. Industrial automation [1], image search [2], robot navigation [3], intelligent inspection [4] are some of the examples. In this paper, we come up with a different task related to scene text, called "Scene Text Editing." Our plan is to replace the text instance in a text image without destroying its realistic appearance. We have implemented this system for the Bangla Language named as Scene Text Editor, as shown in Fig. 1, generates highly detailed text images by modifying each word in the source image while modifying the text and background appearances. Applications like text image synthesis [5], text image

modification, promotional photo editing, and immersive virtual reality translation [6] have sparked increased interest in scene text editing from industry and academia.

Fig. 1. The procedure for changing the scene text.

There have been two major aspects to scene text editing, text style transfer and background texture retention. Text style, in particular, is made up of a variety of factors, including font, orientation, color, spatial point of view, and stroke size making it difficult to accurately capture the entire formatting in the source image and transfer it to the target language text. In the meantime, it's challenging to keep the edited background consistent, and even more so when text makes it appear on a complex background like a street store sign or a menu. Furthermore, if somehow the targeted text is shorter, the overfilled characters should have been wiped away and replaced with a suitable texture.

To resolve these concerns, we recommend a Bangla Scene Text Editing (STE) system, which thus learns from pairs of pictures. As illustrated in Fig. 2, the main concept of Bangla STE is to break down a complex task into multiple easier, modular, and trainable sub-networks: text conversion, background inpainting, and fusion.

In certain tasks, including image-to-image translation as well as style transfer, GAN models [7],[8],[9] have made good advances. Certain techniques usually

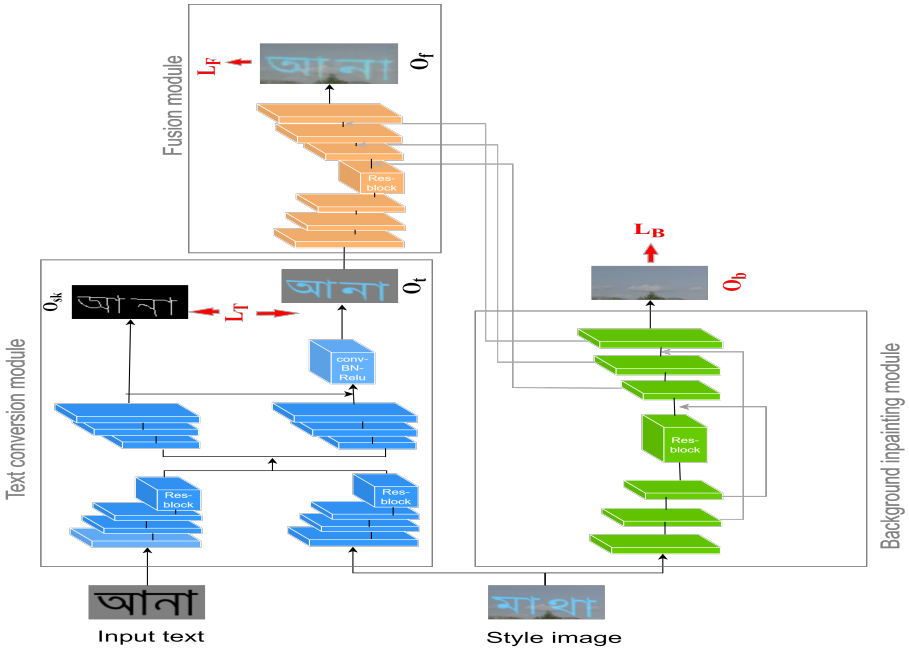

Fig. 2. The overall structure of Bangla STE.

employ the encoder-decoder architecture, which inserts the input into the subspace and then decodes it to produce the final images. An alternative to the single branch architecture that is currently in use is the suggested Bangla STE, which divides the network into modular sub-networks by breaking down a complicated task into a number of simple-to-learn tasks. Recent studies [10], [11] have demonstrated that this technique of network decomposition is effective. Apart from that, the trial results obtained with Bangla STE outperform those obtained with pix2pix [8], a proven method for image-to-image translation, which further demonstrates the effectiveness and resilience of Bangla STE. However, many architectures have modified outstanding texts in recent times. Thanks to the rising application of machine learning and deep learning. SRNeT [12], which is a deep-learning-based framework for the STE system, claims to be able to modify text in the wild. We would really like to adapt this architecture to the Bangla language and develop a deep-learning-based Bangla STE system based on it.

The most significant contributions made by this research are as follows:

– We believe that, to the state of the art, this is the very first time in Bangla an end-to-end easily trained system has been used to handle the challenge of word or text-line level scene text editing, and we prepare a Bangla synthetic text data set for training purposes.

- Our approach is to break down Bangla STE into a number of basic, adaptable, and easily interpretable modules, such as text conversion, background painting, and the final fusion module that allows Bangla STE to produce more accurate outcomes than the majority of image-to-image translations GAN models.
- The suggested system can retain as many semantic features as feasible under the supervision of the stroke skeleton.

2 Related Work

2.1 GAN

GANs [7] have recently gotten a lot of coverage and have made significant advances in a variety of disciplines including image-to-image translation [8], image generation from noise [13], style transfer [9] and so on. The structure of GANs is comprised of two components: the generator and the discriminator, which are interconnected. DCGAN [14] increased the training stability of GAN by initially using convolutional neural networks (CNN) as generator and discriminator components. Conditional-GAN [13] achieved substantial achievements in the pixel-level alignment picture production task by generating the appropriate images under the restrictions of supplied circumstances. Cycle-GAN [9] was responsible for completing the cross-domain conversion job with great results using unpaired style images. The mapping job from image to image was accomplished by Pix2pix [8], which was capable of learning the mapping relationship between input and output domains. On the other hand, some existing work on Bangla where Md Aminul et al. [15] proposed a novel transformer base architecture that would be used for Bangla image captioning and includes an attention mechanism. It is designed to fit the intricate structure of the entire image region. In addition, the Bangla Attentional Generative Adversarial Network (AttnGAN) [16] is what the authors came up with to make high-quality Bangla text-to-image generation. Existing GANs, are challenging to apply directly to text editing tasks since text content varies while text shape varies dramatically, and sophisticated background texturing information must also be kept when changing a scene text image.

2.2 Text Style Transfer

Sustaining continuity before and then after modification of the scene text style is incredibly difficult to do. Attempts to transfer or copy text style information from a specific image or stylized text samples have been made. Some techniques concentrate on style transfer at the character level. Consider the following example: Zhang et al. [17] made an attempt to study the stroke-level style transfer capability across Chinese characters. Lyu et al. [18] suggested using an autoencoder driven GAN to create calligraphy images with a particular design using standard Chinese font images. Sun et al. [19] constructed a stylised Chinese

character generator using the VAE architecture. Numerous different methods concentrate on text effects transfer, which could also adapt visual elements from any provided captured images and has a high economic value in applications such as making a special typography library. Yang et al. [20], [5] introduced a patch-based texture synthesis technique which generates image blocks by mapping sub-effect sequences towards the relevant places in the text skeleton. It's worthwhile to mention that this method relies on appropriate statistical processing, which can be sensitive to minor differences and hence impose a considerable computational overhead. TET-GAN [21] employed the GAN to create a minimalist architecture that really enables both stylization and destylization of a range of textual effects at the same time.

Unlike existing approaches, the suggested framework in this research attempts to handle the problem of arbitrary text styles and special effects migration at the word or text-line level instead of at the character level. In actuality, word-level annotations are a lot easier to acquire than character-level annotations, and editing words is far faster than editing characters. Word-level editors generally appreciate layout consistency. Our word-level editor can adapt the positioning of foreground characters towards changing word lengths, whereas character-level techniques disregard this.

2.3 Text Erasure and Editing

Material consistency between the background texture and the scene text is required before editing may begin. A few previous research projects on text erasure have been created, in which the author attempts to erase the scene text stroke pixels while simultaneously finishing image inpainting on the same spots. Text erasure was proposed by Nakamura et al. [22] using an image-patch-based methodology. In contrast, the sliding window-based processing approach leads to significant computing costs. For the very first time, EnsNet [23] demonstrated that it could delete the scene text on the complete image in an end-to-end manner by using a generative adversarial network. The visualization results look better with refined loss than with pix2pix [8]. Generative adversarial networks also influenced our background inpainting module. When text editing, we solely focus on background erasing at the word level. As a result, the Bangla STE background inpainting module can be built to be lighter and have excellent erasing performance.

We observed recent research [24] that attempted to investigate the topic of scene text editing. The color and font of a single character can only be transferred once in a single procedure, with the consistency of the background texture being ignored in the processing. Unlike other techniques, our method incorporates the advantages of both text style transfer and text erasure techniques. In order to make scene text editing more realistic, we came up with a Bangla Scene Text Editing (STE) system that can not only efficiently transmit text style, but also keep or inpaint the whole background area.

3 Methodology

For scene text editing, we introduce a Bangla Scene Text Editing (STE) system. At the beginning of its training phase, the Bangla STE is given a pair of images I_s, I_t, where I_s represents the source style image and I_t represents the target text image. $((T_{sk}, T_t), T_b, T_f)$ are the outputs where T_t is the foreground image with the very same text style as I_s, and T_{sk} is the target text skeleton. The background of I_s is T_b, while the ultimate target text image is T_f. We break down the Bangla STE into three simple and learnable subnetworks in order to successfully tackle the two key issues highlighted in Sec. 1. As shown in Fig. 2, there are three modules: 1) text conversion, 2) background inpainting, and 3) fusion. A skeleton-guided learning process is used to transfer the text style from the source image I_s to the targeted text, with the goal of preserving text semantics (Sec. 3.1). Furthermore, learn an erasure or inpainting activity to fill in the blanks in the background information (Sec. 3.2). Finally,it can be seen (Sec. 3.3) that the text fusion network combines the transmitted target image and the completed background to produce the altered image.

3.1 Text Conversion Module

The target text is converted into a fairly standard image with a specific font and then a background number of pixels of 127, and the resulting image is referred to only as the target text image I_t. The text conversion component (blue section in Fig. 2) receives as inputs the source image I_s and the target text image I_t, with the goal of extracting the foreground style from the source image I_s as well as transferring it to the target text image I_t. Text style, such as color, font, geometric distortion, and so on, is included in the foreground style. As a result, the text conversion method generates an image O_t with the target text's semantics and the source picture's text style. This research employs an FCN encoder-decoder. Three down-sampling convolutional layers plus four residual blocks [25] are used to encode the source image I_s, the very same structure is used to encode the input text images I_t, after which two features are combined together across the depth axis. To produce the outputs O_t, there are three up-sampling transpose convolutional layers and one Convolution-BatchNorm-LeakyReLU block. In addition, to start generating more powerful text, we offer a skeleton-guided learning process. The text conversion modules are denoted by the G_T, and the output is written as:

$$O_t = G_T\left(I_t, I_s\right) \tag{1}$$

Skeleton-guided Learning Mechanism: Humans differentiate various texts generally based on the skeleton or glyph of the text, unlike other natural items. It is important to keep the text skeleton in I_t after trying to transfer the text style from the source style image I_s. We have used a skeleton-guided learning technique to accomplish this. To anticipate a single channel skeleton map, we add a skeleton response block made up of three up-sampling layers and one convolutional layer, accompanied by a sigmoid activation function, and afterwards

combine the skeleton heatmap as well as decoder outputs along its depth axis. We utilize the dice loss [26] rather than the cross-entropy loss to assess the skeleton response map's reconstructive quality even though it produces more precise information.

3.2 Background Inpainting Module

The primary objective of this module is to get the background by doing a word-level erasing job. As seen in the green portion of Fig. 2, this module accepts only its source image I_s as input and produces a background image O_b in which all text stroke pixels have been deleted and replaced with an appropriate texture. The input image is encoded using three down-sampling convolutional layers having stride 2, including four residual blocks, after which the decoder employs three up-sampling convolutional layers to construct the output image well within the original size. After every layer, we apply the leaky ReLU activation function, whereas the output layer uses the tanh function. Here, the background generator is abbreviated as G_B. In addition to making the visual effects appear more genuine, we must restore as much of the texture of the background as we possibly can. U-Net [27], which suggests adding skip connections across mirror layers to solve object segmentation and image-to-image translation issues, has been shown to be extremely efficient and robust. This method is used in the up-sampling procedure, where previously encoded feature maps of about the same size are combined to save better textures. This aids in the recovery of missing background data during the down-sampling procedure.

Unlike earlier full-text image erasing approaches [22], [23], our method focuses on image inpainting at the word level. Because the text in word-level images has a fairly uniform scale, our network architecture has a basic and tidy appearance. Adversarial learning is included to establish a much more refined look, influenced by the success of this research [23].

3.3 Fusion Module

The fusion component combines the target text image and, indeed, the background texture into a systematic approach to managing an edited scene text image. The fusion strategy incorporates the encoder-decoder FCN structure, as shown in the orange part of Fig. 2. The encoder, consisting of three down-sampling convolutional layers and residual blocks, receives the foreground picture produced by the textual conversion component. The final adjusted image is then generated using a decoder with three upsampling transposed convolutional layers and Convolution-BatchNorm-LeakyReLU blocks. In the up-sampling step of the fusion decoder, we relate the decoding feature maps of the background inpainting module to the corresponding feature maps with the same quality. In this method, the fusion network produces images with significantly restored background features; text objects and background are really well merged while achieving synthesis realism in appearances.

3.4 Discriminators

In our Bangla STE system, two discriminators with the very same architecture as PatchGAN [8] are used. They are constructed from five convolution layers in order to lower the scale to 1/16 of their original size. To determine whether the erased result O_b and the intended background T_b are identical, the discriminator D_B in the background inpainting module concatenates the inputs of I_s with the inputs of O_b or T_b. In contrast, the discriminator D_F in the fusion module concatenates two or more images like I_t and O_f or T_f in order to measure the consistency of the final outcome O_f with the ground truth image T_f.

3.5 Training and Inference

The entire network can be trained from start to finish in the training phase. We train the generator and discriminators in turn, adopting GAN's training techniques. With our training data, we synthesize sets of images in a similar approach but without the text. Text stroke segmentation masks can also be used to get the foreground, background, as well as text skeleton images. When the generator receives the inputs, I_t and I_s are changed with the text O_t under the observation of the images T_{sk}, T_t, T_b, and T_f. D_B is fed with (I_s, O_b) and (I_s, T_b) for adversarial training in order to pursue background consistency. To ensure appropriate findings, (I_t, O_f) as well as (I_t, T_f) are put into D_F.

The generator may produce the erased outcome of the style image as well as a modified image during the inference stage, provided the standard text image and even the style image. To visualize the entire image, we trim out the targeted patches as per the boundary box annotations and feed them to our system. The findings are then pasted back into their original positions.

4 Experiments

Our findings indicate that our model has a considerable level of ability to edit text in scenes, which is shown in the following section.

4.1 Datasets

The datasets which were utilized in this research described are as follows, with a brief introduction:

Synthetic Data: In terms of improving text synthesis technology [28], we propose synthesizing data in a pair of styles but with varied content. By picking fonts, colors, and other parameters at random, we can make text that looks like it's been styled. Then, the text should be placed on top of a background image. Our dataset [29] provided us with the background image we used for this project. As a result of image skeletonization [30] as ground truth (Fig. 3) we may obtain the relevant background, foreground text, and text skeletons. Here, Fig. 4 illustrates the process of creating synthetic data. In our experiment, we

resize the height of the text image to 64 pixels in our experiment while keeping the same aspect ratio. With a total of 100,000 images in the training set and only 1,000 images in the test set, the system is trained.

Fig. 3. Synthetic data examples.

4.2 Implementation Details

On the basis of pix2pix [8], we established our network architecture. With $\beta 1$= 0.5, and $\beta 2$= 0.999, the Adam [31] optimizer is being used to train a model until the output begins to become more stable during the training stage. The learning rate is initially set at 2×10^{-4}, then after 30 epochs, it is gradually reduced to 2×10^{-6}. On both the generator and the discriminator, we perform spectral normalization [32]. We employ batch normalization [33] on the generator only. Here, the batch size is set to 8, and the input images are resized in size to approximately $\mathcal{W} \times 64$ while maintaining the aspect ratio. At the beginning of training, we randomly select the batch data and resize the image width so that it is the average width of the data. When testing, we might use images of varying widths to achieve the desired results. When using a single NVIDIA TITAN Xp graphics card, it takes approximately eight hours to train the model.

4.3 Evaluation Metrics

To assess our approach, we employ the most generally accepted metrics in image generation, including the following:

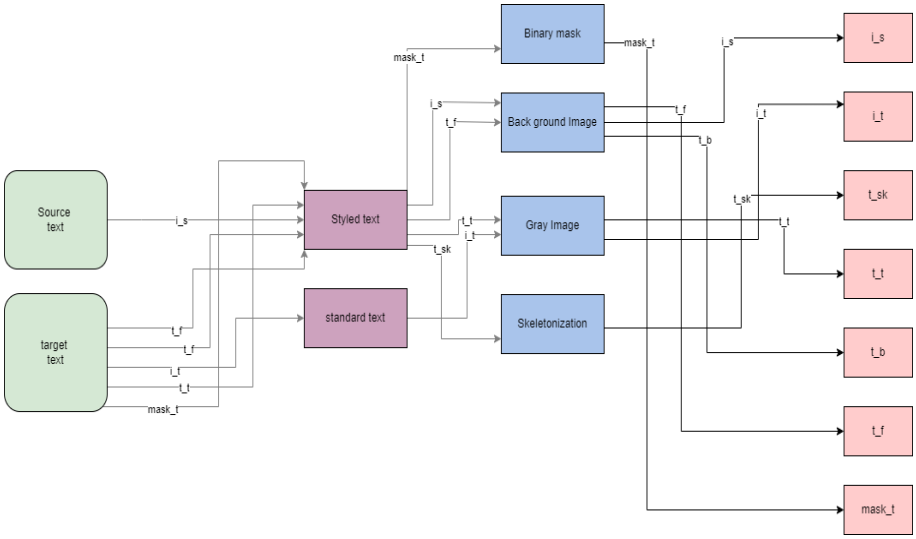

Fig. 4. Visualization of synthetic data generation process.

1) UQI [34], luminance distortion, loss of correlation, and contrast distortion are used by UQI to assess the resolution of the image;

2) VIFP is a comprehensive image enhancement evaluation measure in a pixel domain depending on actual scene data and the concept of how images are collected by the human visual system;

3) PSNR, typically calculates the peak signal to noise ratio. To compute an image's PSNR, we need an ideally cleaned image with the highest potential power to analyze that image;

4) SSIM [35], a program that calculates the architectural similarity measure among two images.

As we have made a comparison of images created by our Bangla STE system with real-life images with evaluation values, shown in Fig. 5. The larger PSNR and SSIM, indicate that the findings are close to ground reality. We solely use synthetic test measurements to compute the metrics indicated above, because there are very few paired data points in the real-world dataset. We assess the recognition performance on real data to assess the quality of the processed outcome. We only estimate such indicators in the clipped regions because our network's inputs are indeed cropped images. Visual evaluation is also utilized in a real-world dataset to distinguish between the multiple approaches qualitatively.

From Table. 1, we can see the evaluation results depending on PSNR, UQI, SSIM1, SSIM2, VIFP for our Bangla STE system. The higher the PSNR value, the more effective the compression or filter approach will be. Basically, SSIM scores always stay in the range from 0 to 1, while 1 always denotes that something is ideal between the rebuild and actual images.

ground truth	output	PSNR	UQI
		12.54	0.891
		14.99	0.79
		13.14	0.69
		11.48	0.53
		13.56	0.68
		12.07	0.48
		11.79	0.87

Fig. 5. Comparison between our Bangla STE system's generated images and real images with evaluation values.

	Mean	std	25%	50%	75%
PSNR	13.82	2.49	12.23	13.71	15.39
UQI	0.72	0.09	0.61	0.75	0.86
SSIM1	0.098	0.097	0.032	0.066	0.136
SSIM2	0.107	0.0103	0.03	0.072	0.15
VIFP	0.009	0.015	0.004	0.006	0.010

Table 1: Evaluation Results

Ground truth	Output

Fig. 6. Failed Cases

4.4 Failed Cases

Despite the fact that our technique can handle most scene photos, it does have significant restrictions. Whenever the text has exceedingly complicated structures or unusual font patterns, our approaches may fail. Our technique has failed in a few circumstances, as shown in Fig. 6.

5 Conclusion and Future Work

The Bangla Scene-Text-Editing (STE) that uses an end-to-end infrastructure to consider replacing text from a scene text image while preserving the original style. To try to accomplish this system, we start by dividing it into three main steps: (1) Using skeleton, extract the foreground text style and apply it to the input text. (2) To obtain the background image, wipe away the style image with the appropriate appearance. (3) Combine the transferred text with the removed background. Towards the state of the art, this is the very first work in Bengali to edit text images at the level of words. On the synthetic dataset, our system produced impressive results, including both subjective visual reality and objective quantitative scores. Simultaneously, the network can wipe away text and edit in cross-language situations, and the efficiency of our system has already been proven through extensive ablation research.

In the coming days, we expect to deal with text editing problems and even more complicated situations while constructing the structure that is relatively easy to make use of. To fully utilize the suggested model's capability, we would therefore edit text between so many language pairs. We're going to try to come up with new evaluation metrics so that we can fully measure the quality of text editing.

References

1. Y. K. Ham, M. S. Kang, H. K. Chung, R.-H. Park, and G. T. Park, "Recognition of raised characters for automatic classification of rubber tires," *Optical Engineering*, vol. 34, no. 1, pp. 102–109, 1995.
2. S. S. Tsai, H. Chen, D. Chen, G. Schroth, R. Grzeszczuk, and B. Girod, "Mobile visual search on printed documents using text and low bit-rate features," in *2011 18th IEEE International Conference on Image Processing*. IEEE, 2011, pp. 2601–2604.
3. G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 2, pp. 237–267, 2002.
4. R. Chen, B. C. Desai, and C. Zhou, "Cindi robot: an intelligent web crawler based on multi-level inspection," in *11th International Database Engineering and Applications Symposium (IDEAS 2007)*. IEEE, 2007, pp. 93–101.
5. S. Yang, J. Liu, W. Yang, and Z. Guo, "Context-aware unsupervised text stylization," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1688–1696.

6. V. Fragoso, S. Gauglitz, S. Zamora, J. Kleban, and M. Turk, "Translatar: A mobile augmented reality translator," in *2011 IEEE workshop on applications of computer vision (WACV)*. IEEE, 2011, pp. 497–502.
7. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.
8. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
9. J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
10. J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Learning to compose neural networks for question answering," *arXiv preprint arXiv:1601.01705*, 2016.
11. G. Balakrishnan, A. Zhao, A. V. Dalca, F. Durand, and J. Guttag, "Synthesizing images of humans in unseen poses," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8340–8348.
12. L. Wu, C. Zhang, J. Liu, J. Han, J. Liu, E. Ding, and X. Bai, "Editing text in the wild," in *Proceedings of the 27th ACM international conference on multimedia*, 2019, pp. 1500–1508.
13. M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.
14. A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
15. M. A. H. Palash, M. A. A. Nasim, S. Saha, F. Afrin, R. Mallik, and S. Samiappan, "Bangla image caption generation through cnn-transformer based encoder-decoder network," 2021. [Online]. Available: https://arxiv.org/abs/2110.12442
16. M. A. H. Palash, M. A. A. Nasim, A. Dhali, and F. Afrin, "Fine-grained image generation from bangla text description using attentional generative adversarial network," 2021. [Online]. Available: https://arxiv.org/abs/2109.11749
17. Y. Zhang, Y. Zhang, and W. Cai, "Separating style and content for generalized style transfer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8447–8455.
18. P. Lyu, X. Bai, C. Yao, Z. Zhu, T. Huang, and W. Liu, "Auto-encoder guided gan for chinese calligraphy synthesis," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1095–1100.
19. D. Sun, T. Ren, C. Li, H. Su, and J. Zhu, "Learning to write stylized chinese characters by reading a handful of examples," *arXiv preprint arXiv:1712.06424*, 2017.
20. S. Yang, J. Liu, Z. Lian, and Z. Guo, "Awesome typography: Statistics-based text effects transfer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7464–7473.
21. S. Yang, J. Liu, W. Wang, and Z. Guo, "Tet-gan: Text effects transfer via stylization and destylization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1238–1245.
22. T. Nakamura, A. Zhu, K. Yanai, and S. Uchida, "Scene text eraser," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 832–837.

23. S. Zhang, Y. Liu, L. Jin, Y. Huang, and S. Lai, "Ensnet: Ensconce text in the wild," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 801–808.
24. P. Roy, S. Bhattacharya, S. Ghosh, and U. Pal, "Stefann: scene text editor using font adaptive neural network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 228–13 237.
25. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
26. F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 565–571.
27. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
28. A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2315–2324.
29. ——, "Synthetic data for text localisation in natural images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
30. T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.
31. D. P. Kingma and J. B. Adam, "A method for stochastic," *Optimization. In, ICLR*, vol. 5, 2015.
32. T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.
33. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.
34. Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE signal processing letters*, vol. 9, no. 3, pp. 81–84, 2002.
35. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.