MOSAIC: Multi-Object Segmented Arbitrary Stylization Using CLIP

Prajwal Ganugula[*†1,2], Y S S S Santosh Kumar[*1], N K Sagar Reddy[*‡1], Prabhath Chellingi[§2], Avinash Thakur[1], Neeraj Kasera[1], and C Shyam Anand[1]

[1]OPPO Mobiles R & D Center, Hyderabad, India
[2]Department of Computer Science and Engineering, IIT Hyderabad, India
{g.prajwal,y.kumar,nallamilli.reddy,avinash.thakur,neeraj.kasera,c.shyam.anand}@oppo.com, cs20btech11038@iith.ac.in

Abstract

Style transfer driven by text prompts paved a new path for creatively stylizing the images without collecting an actual style image. Despite having promising results, with text-driven stylization, the user has no control over the stylization. If a user wants to create an artistic image, the user requires fine control over the stylization of various entities individually in the content image, which is not addressed by the current state-of-the-art approaches. On the other hand, diffusion style transfer methods also suffer from the same issue because the regional stylization control over the stylized output is ineffective. To address this problem, We propose a new method Multi-Object Segmented Arbitrary Stylization Using CLIP (MOSAIC), that can apply styles to different objects in the image based on the context extracted from the input prompt. Text-based segmentation and stylization modules which are based on vision transformer architecture, were used to segment and stylize the objects. Our method can extend to any arbitrary objects, styles and produce high-quality images compared to the current state of art methods. To our knowledge, this is the first attempt to perform text-guided arbitrary object-wise stylization. We demonstrate the effectiveness of our approach through qualitative and quantitative analysis, showing that it can generate visually appealing stylized images with enhanced control over stylization and the ability to generalize to unseen object classes.

1. Introduction

Style transfer has emerged as an essential technique in the field of computer vision and image processing, allowing for the transformation of style from a style or texture image to a reference image while preserving the contents of the reference image. Gatys *et al*. [1] formulated style transfer as an image optimization problem, which was later implemented by Ulyanov *et al*. [2] using a feed-forward neural network to reduce inference time. To improve the visual quality of the results, Johnson *et al*. [3] proposed the use of perceptual loss. However, these techniques were limited to single-styling images. Dumoulin *et al*. [4] addressed this issue by implementing Conditional Instance Normalization layers to extend the stylization network to multiple styles. However, this approach becomes infeasible after reaching a certain number of styles and is limited to the styles the model is trained on. To overcome these limitations, Xun Huang *et al*. [5] proposed the use of Adaptive Instance Normalization to extend style transfer to arbitrary styles. Although several techniques have been proposed for style transfer, each technique has its limitations, and further research is needed to develop a more robust and flexible style transfer algorithm.

Applying different styles to different objects in an image is a challenging problem that requires identifying individual objects and applying corresponding styles to each object. Kurzman *et al*. [9] proposed a Class-Based styling method, where they utilized a segmentation model to identify the objects belonging to the same class and applied styles guided by the segmentation masks. This approach allows for consistent styling of objects in the same class and produces visually appealing results. However, this method does not consider the individual characteristics of each object, which limits its flexibility. Huang *et al*. [10] proposed the Style Mixer method to address this limitation by applying multi-

[*]Equal Contribution.
[†]Prajwal led the project and guided the intern.
[‡]Sagar provided valuable contribution to the pipeline.
[§]Work done during internship at OPPO Mobiles R&D Center.

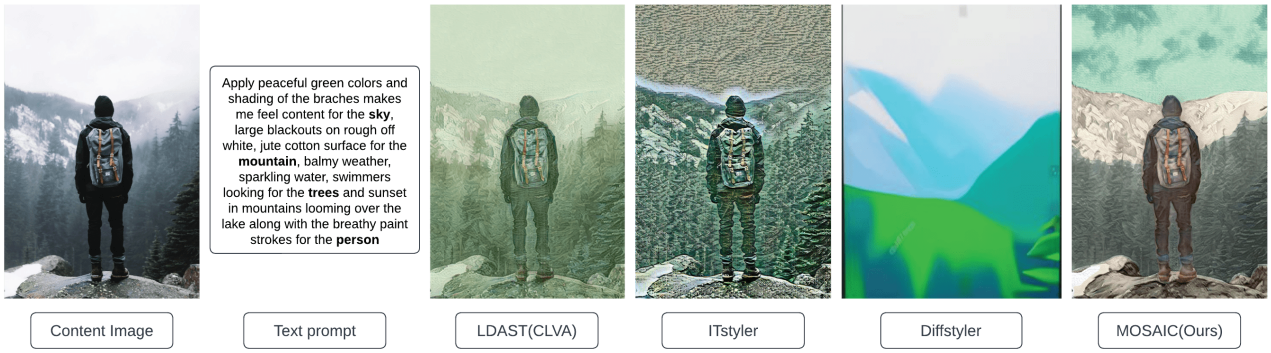

Figure 1: Showing Our output result comparing with *LDAST*[6], *ITsytler*[7], *Diffstyler*[8], *MOSAIC(ours)*.

,

ple styles based on regional semantics. This approach allows for more precise control over the stylization of each object by considering its characteristics, leading to more diverse and creative stylization. However, this method requires additional computational resources and may lead to longer stylization times. Although both Class-Based styling and Style Mixer methods have shown promising results, further research is needed to develop more efficient and flexible techniques for object-level style transfer.

Object recognition models trained on specific object classes tend to perform poorly in recognizing unseen object classes. Additionally, objects belonging to the same class may have different descriptions within an image, requiring the model to understand illustrations of both seen and unseen classes during training. To address this limitation, Lüddecke and Ecker proposed the CLIPSeg[11] method, which utilizes arbitrary text prompts to segment objects. Similarly, Li *et al*. [12] proposed the LSeg method for language-driven semantic segmentation, using the contrastive learning approach with text and image embeddings to predict segmentation classes. Other works like OpenSeg [13] and OVSeg [14] have also addressed similar issues. Following these, a recent work SAM[15](Segment Anything Model) was trained on a new dataset which was created mainly for segmentation. This was designed to take inputs in different ways(point based, text based, bounding box based). CLIP[16] embeddings could be passed as an input to SAM, which makes text-based Image segmentation possible.

Prompt-based arbitrary style transfer is a branch of style transfer networks where input text prompts are used instead of reference style images. Kwon et al. (2021) proposed CLIPStyler[17], which stylizes images based on the input text descriptions of the style. They employed patch-wise CLIP(Contrastive Language-Image Pretraining)[16] loss to moderate the quality of style images in all regions. Subsequent works, including Fast CLIPStyler[18], LDAST[6], and ITstyler[7], have enhanced the ability of stylization networks. Diffusion-based models, such as DiffStyler[8], have also been used for text-guided stylization networks. Despite these advancements, there is still room for further research to develop more efficient and effective prompt-based style transfer techniques.

Existing approaches for style transfer typically require a reference style image, but no complete text-based pipeline for multi-object arbitrary style transfer currently exists. In this paper, we propose a novel text-based pipeline for multi-object arbitrary style transfer, allowing generation of stylized images with a text description of the desired style.

Our contributions can be summarized in three main points:

1. We propose a novel text-based pipeline for multi-object arbitrary style transfer, which utilizes a custom decoder block to segregate text into segmentation and stylization tasks.

2. Employing a combination of style embedding and object segmentation techniques to generate high-quality stylized images. This research has the potential to provide new and efficient ways to create stylized images without the need for reference-style ideas.

3. Conducting experiments to demonstrate the effectiveness of our approach in generating visually appealing stylized images with enhanced control over stylization. We evaluate the results using user study and patch-wise CLIP score to capture object wise stylization capability. Our model's ability to generalize to unseen object classes is shown in Figure 1.

2. Related work

Image-based style transfer. Transferring styles from one image to another has garnered significant attention in recent years due to its ability to create visually appealing and artistic images. Early techniques for image-based style transfer involved pixel-wise updates of content images compared to brush-strokes of style images [19, 20, 21, 22, 23]. However, these techniques were later replaced by neural style transfer methods [24], [25], [1], [26], [27] that enabled the creation of more realistic stylized images. While these methods provided high-quality stylized images, the optimization process was slow. To address this issue, knowledge-distilled transform networks were introduced [3], [28], [29], [30], [2] to speed up the optimization process. However, these methods were limited to single-style per-model transformation. To overcome this limitation, Conditional Instance Normalization (CIN) layers were introduced [4], [31] that enabled the creation of a multiple-style per-model network. However, CIN layers were limited to a fixed set of styles. To enable arbitrary styles per model, the CIN layers were replaced by Arbitrary Instance Normalization (AdaIN) layers [5], [32]. Further improvements were made to preserve various features of a stylized image, including AdaAttn[33], AesUST[34], and All-to-key[35] as well as various other techniques [36, 37, 38].

Text-Based style transfer: Text-based style transfer has emerged as an alternative to image-based techniques due to the limitations of the latter approach in terms of the availability of style images. Language-based image editing using predefined semantic labels, referred to as LBIE, was introduced as a solution to this problem [39]. Kwon proposed a text-style transfer technique using CLIP[16] in CLIPstyler[17]. However, CLIPstyler has a high inference time, leading to the development of various improvements, including FastCLIPstyler [18], LDAST [6], and ITstyler[7], which utilize normalization layers. Furthermore, significant progress has been made using Diffusion models [40] and GANs [41] in DiffStyler [8], styleGAN [42], Pix2pix [43],[44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54]. To improve CLIP's performance, mixed forms with GANs were explored in styleCLIP [55], and NADA [56].

Semantic Image segmentation: Semantic image segmentation is identifying and segmenting specific objects within an image. Traditionally, large models pre-trained over extensive datasets [57, 58] have been used to perform this task, which is computationally expensive. Recently, neural network and transformer-based approaches have been proposed in DabNet[59], TransUNet[60], SETR[61], Segformer[62], and Segmenter[63], to improve the efficiency of the segmentation process. However, these methods have limited class segmentation capabilities. Text-based image segmentation has been proposed to segment images into infinite classes to address this limitation. Several architectures have been proposed based on CLIP, including CLIPseg[11], Langseg[12], and CRIS[64]. Several models based on GANs[41] have been proposed to enhance semantic image segmentation. The recent introduction of the SAM [15] model made a new revolution in image segmentation as it can segment anything. SAM is the world's first massive-scale, promptable, interactive foundation image segmentation model. For this model's training, they have gathered a new dataset of 11 million images and 1.1 billion masks. It can take input in three ways; one is prompt-based input which uses CLIP[16] to encode the prompt and mask the image. The architecture is conveniently designed to produce different masks for different inputs with faster inference once the input image gets encoded. This feature is beneficial in masking multiple objects efficiently.

Content Based Image Style Transfer: Content-Based Image Style Transfer refers to stylizing each object in an image separately. CB-Styling [9] was one of the earliest attempts in this direction, which utilized a combination of segmentation [59] and stylization [27] networks. However, a limitation of CB-Styling was its inability to handle a wide range of classes in segmentation. Several approaches have been proposed to address this limitation that use attention mechanisms to stylize objects in images by comparing them with style images. For instance, StyleMixer [10], Splice [65], MAST [66], and [67] utilize attention mechanisms to stylize objects in images, although they all suffer from the drawback of being limited to the stylization of the content image in the presence of the style image, which falls short of meeting the desired quality benchmarks.

Semantic text segmentation: Semantic text segmentation is a critical task in natural language processing, which involves extracting semantic features and identifying various classes from a given text. Language translation transformers [68] have made this process easy by connecting the words in a sentence to extract semantic meaning. Further advancements have been made using Generative Pre-trained Transformers (GPTs) [69], demonstrating an understanding of the importance and capability to perform specific tasks.

3. Method

We aim to perform object-wise text-style transfer in an input image with a text prompt. To achieve this, we propose a pipeline as shown in Figure 3. LDAST[6](Language Driven Artistic Style Transfer) and SAM[15](Segment anything Model) inspire the blocks in our pipeline. The encoders of LDAST and SAM models use pre-trained CLIP ViT-B/16 text encoder as a backbone and take embeddings from the model to further process their respective tasks. The details and architectures of these models are discussed in this section.

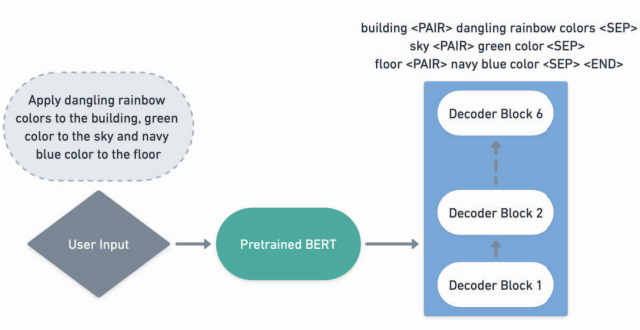

Figure 2: Architecture of the BERT [70] based text segmentation model.

3.1. Text Segmentation Model

Our approach's first step is segmenting the input prompt into objects along with their position description in the image and corresponding styles. We took a pre-trained BERT [70] based model because of its strong ability to understand relationships between different segments of text (objects and their corresponding styles). We added a custom decoder to produce the objects and styles pairs. Each pair is separated by $< SEP >$ token, and the object style in every pair is separated by $< PAIR >$ token. Refer to Figure 2 for the architecture. We trained our model using Text Segmentation Loss.

3.1.1 Text Segmentation Loss

We used cross-entropy loss [71] for text segmentation loss (L).

$$L = -\sum_{c=1} y_c \log(p_c) \tag{1}$$

where y_c is the ground truth and p_c is the predicted probability of the model

We also explored the usage of GPT [72] based models for this task. Refer to [4.2] for more details.

3.2. CLIP Based Models

CLIP[16] is a recent initiative aimed at connecting images and text, with a particular focus on zero-shot capabilities. The CLIP model consists of a ViT [73] based image encoder and a combination of transformer blocks as text encoder. The CLIP model is trained on a large dataset of image-caption pairs using a contrastive learning approach. The model is trained to maximize the similarity between the embeddings of matching image-caption pairs, while minimizing the similarity between the embeddings of non-matching pairs. This is achieved by defining a contrastive loss function that encourages the model to learn embeddings that are discriminative for the given task. The CLIP model is useful in Joint processing of text and images. Since CLIP is contrastively trained on huge dataset, it can generalize well even for downstream zeroshot tasks.

3.3. LDAST

This model aims at text-based editing of an image. It has two modules one is LVA (Language Visual Artist), the other is CR (Contrastive Reasoning), and the combination is CLVA (Contrastive Language Visual Artist)[6]. The process of extracting style from text and applying it to a content image is the core functionality of the Language Visual Artist (LVA) module within the CLVA model. The LVA module enables the network to learn to embed the style text and relate it to the corresponding style image using a discriminator. The Contrastive Reasoning (CR) module further enhances this process by comparing contrasting pairs of style images and text to improve or correct the relativeness between the outputs. To learn the art of style transfer from text, LVA employs a combination of structure reconstruction, patch-wise style discriminator, content matching, and style matching losses. The VGG[75] encoders and decoders are used to extract the feature maps, which are then stitched back together. The CR module improves the learned parameters by applying consistency and relativity losses, resulting in improved content consistency and the relativeness of style in the output. These advanced techniques have the potential to revolutionize text-based image editing and enable more sophisticated image stylization through text.

3.4. Segment Anything Model (SAM)

The objective of SAM[15] is to segment an image into specific classes based on the text prompt. It comes with a separately running image encoder and prompt encoder, which helps in making multiple mask generation efficient. We will cache the image encoding and reuse it whenever a new mask needs to be generated. This mask is generated from the lightweight mask decoder, which runs efficiently even on the CPU. This lightweight mask decoder takes image encoding and prompt encoding as input and produces the respective masked images. The image encoding comes from the cache, and the prompt encoding will be generated instantaneously, even over the CPU. This is all possible because of practical training of the model over the newly developed dataset[15], which comprises 11 million images and 1.1 billion mask captions. This helps us effectively stylize the image object-wise efficiently.

3.5. Architecture Pipeline

Our Architecture pipeline in Figure 3, takes the above ideas to create a unified channel, seamlessly performing the required task. A content image and a text prompt

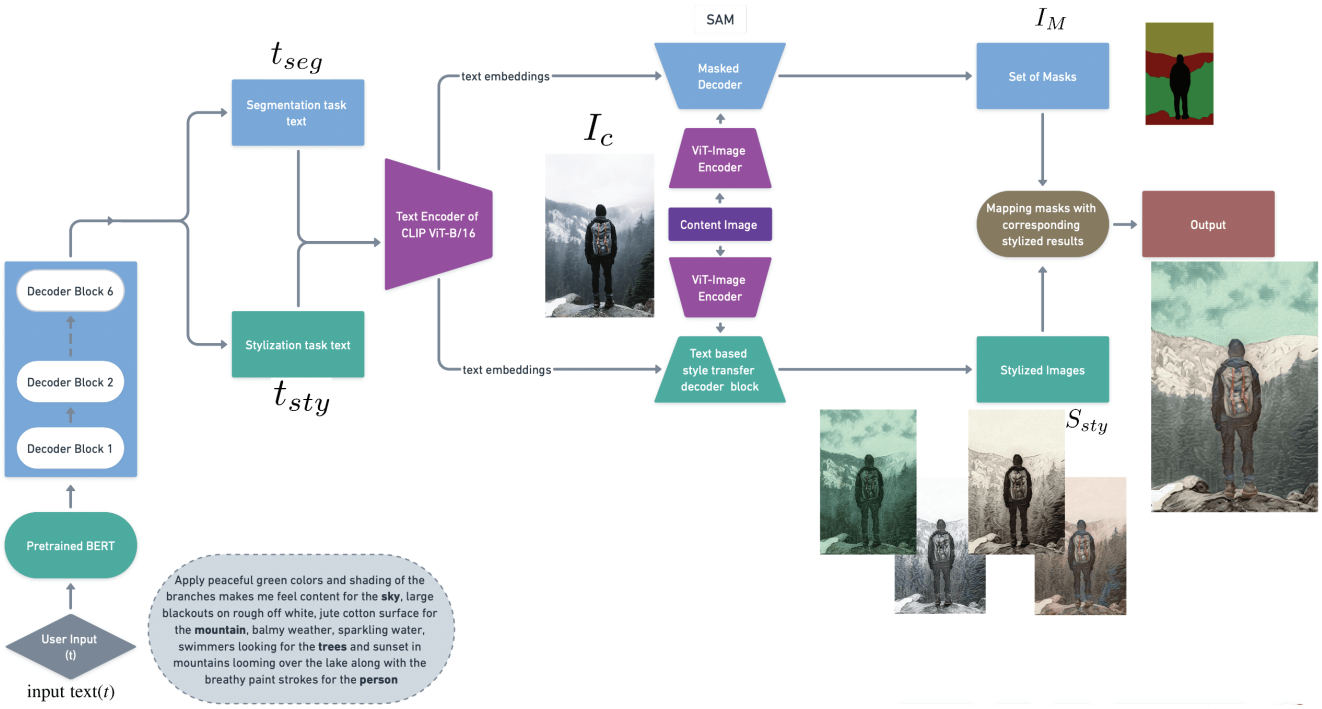

Figure 3: Pipeline of the proposed style transfer method showing the input flow.

are given as input which is then operated on by the modules in the pipeline in a sequential order to produce final object-wise stylized output. In the **text segmentation block**, the input text(t) is segmented into stylization (t_{sty}) and segmentation (t_{seg}) texts. Each object word from t_{seg} is mapped to corresponding style phrases in t_{sty}. Each mapping from the segmented input text ($t_{sty} + t_{seg}$) is passed parallelly (or sequentially) into the next block separately. After segregating the t_{sty} and t_{seg}, we pass them to CLIP-ViT-B/16 pre-trained text encoder to get the corresponding text embeddings for the respective tasks. The two models(Segmentation and stylization networks) take these embeddings and process their respective tasks. For the **image segmentation block**, we take the segmentation part of input text (t_{seg}) and content image (I_c) and produce the required object masks (I_M). Simultaneously at the **stylization block**, we give the t_{sty} and I_c. This produces all the corresponding stylized images for each style in t_{sty}. This gives out a set (S_{sty}) of stylized images of I_c. In the **Object-wise stylization block** of the architecture, we extract the styles of objects from this S_{sty} using the object masks (I_M) and mappings established previously in the first block. Finally, after extracting the corresponding style for each object, we combine the pixel values to produce the final MOSAIC output.

4. Experiments

4.1. Text Segmentation Dataset

To get accurate text segregation for segmentation and style transfer tasks, we need data for which the input text and corresponding segregated text segments for each of the segmentation and stylization tasks are annotated. For this reason we constructed a dataset consisting of 400 classes and 150 styles. Our dataset was carefully designed to ensure that the resulting text prompts closely resembled the inputs typically encountered in real-world scenarios. Using this dataset, we trained a text segmentation model that is capable of accurately segmenting and segregating the text prompts while preserving the relationships between the resulting segments.

4.2. Text Segmentation Model

We adopted a pretrained BERT encoder, which was integrated with our custom decoder comprising 6 decoder layers, the embedding size was set to 512, used 8 heads, as previously established in [68]. To optimize the training process, we maintained the encoder's parameters fixed while exclusively training the decoder on our curated dataset, utilizing the Cross Entropy Loss (Equation 1). Freezing BERT's weights resulted in fast convergence due to its ability to generalize to intricate inputs. Additionally, this led to a substantial reduction in the number of learnable parame-

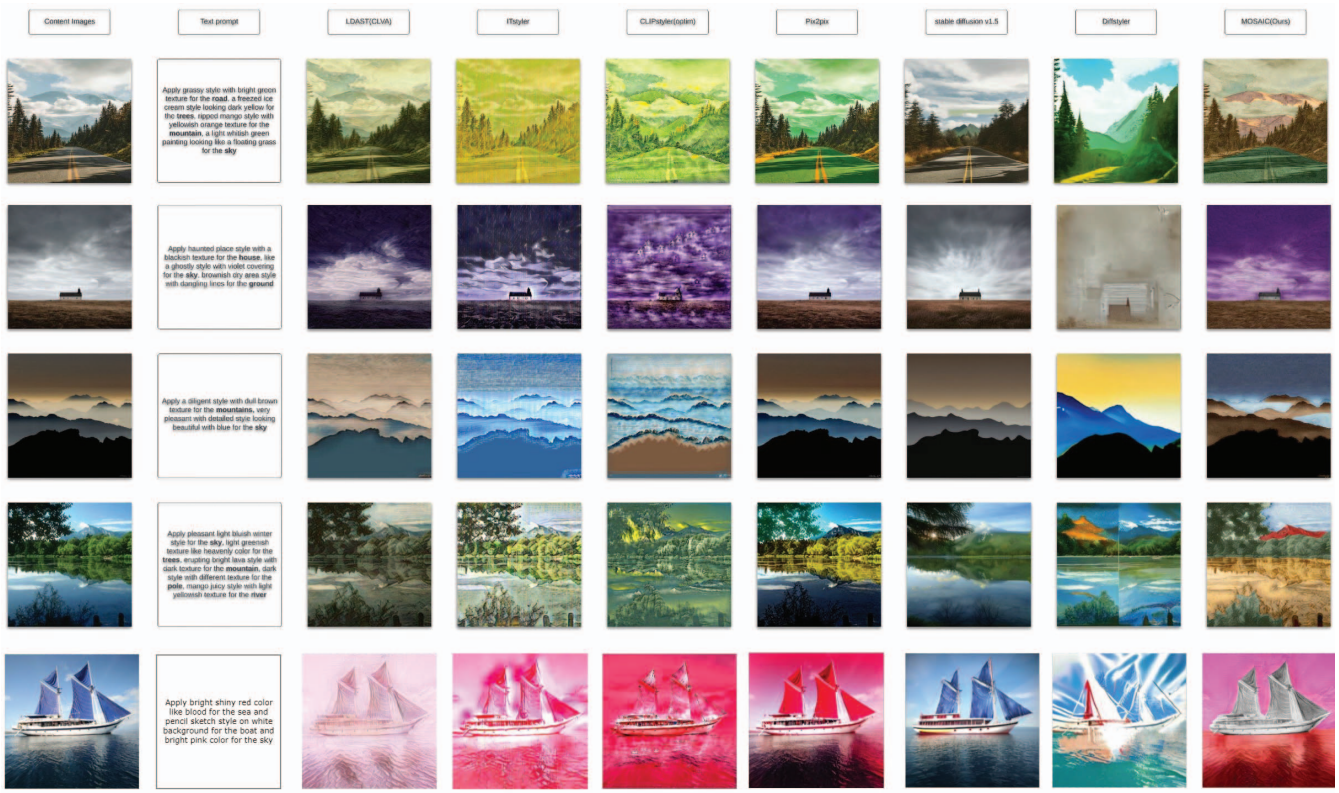

Figure 4: Comparing the output of our model with LDAST[6], ITstyler[7], CLIPstyler(optim.)[17], Pix2pix[43], stablediffusionv1.4[74], Diffstyler[8], MOSAIC(ours). Our model produced results that were in line with user expectations and showed superior results between stylized objects than other models.

ters, enhancing training efficiency.

We used adam optimizer with initial learning rate as 0.001 and Cosine Annealing with warmup phase of 5 epochs and trained it for 400 epochs on 8 V100 GPU cluster. Our model has showed the capability to generalize well to arbitrary text prompts which contain multiple objects and styles. Alternatively, If we want to have a more generalized model that can generalize well to unseen classes and styles, we can also use ChatGPT API [76] to decompose the prompt text into objects and their corresponding styles.

4.3. Qualitative Analysis

We show the results to understand the effectiveness of our pipeline. We divided this study into two stages. Initially, we compare the results with models which come under the same reign $i.e.$, text-based style transfer models. We took some SOTA models, Pix2pix[43], LDAST[6], ITstyler[7], CLIPstyler(optim.)[17], Diffstyler[8], and stablediffusionv1.4[74], for the analysis. We excluded the FastCLIPstyler[18] and CLIPstyler(fast)[17] as the CLIPstyler(optim.) is always better than them. The results of the outputs produced with the same content and text prompts are shown in Figure 4. The problem with these models is that they cannot distinguish objects from the text and its corresponding style phrases, as obtained from our proposed method.

The next stage includes the image-based style transfer models, which use the style images generated by the stable diffusion model of *huggingface*[77](stable-diffusion-v1.4[74]) from the same text prompt. We compared the results with *styleMixer*[10] by generating style images for each text in the text prompt, as shown in Figure 5. The outputs of *styleMixer* are not good because of two drawbacks. One is that the styleMixer needs style images with objects almost matching the content image. The other is that the stable diffusion used here can't generate the styles ideally from the same text prompt given to our model. We need to check whether the output is accurate to text rather than just seeing its uniformity. These problems don't stop our model from giving pleasing results, as it doesn't need any style image.

We also compare our model with Image-based style networks using the comparable style for the text prompt generated using the same stable diffusion model of *hugging-*

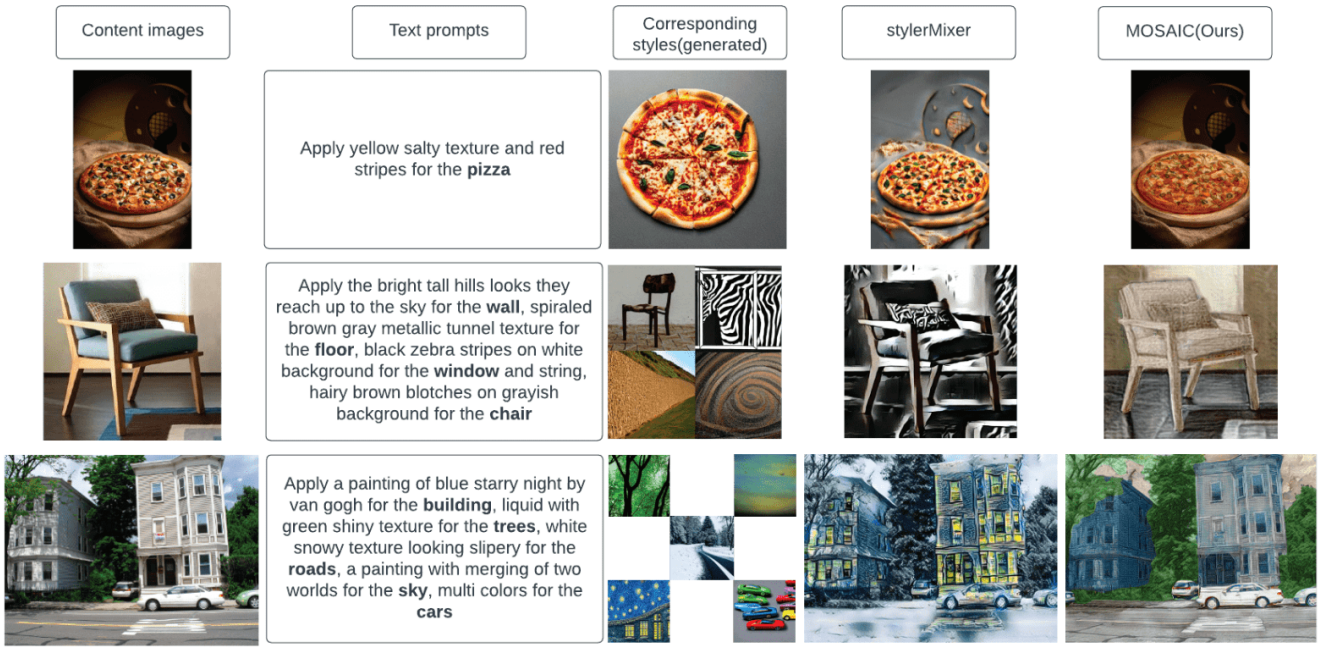

Figure 5: Comparing our model results with *styleMixer*[10], showing that the styles aren't extracted well from the style images by styleMixer

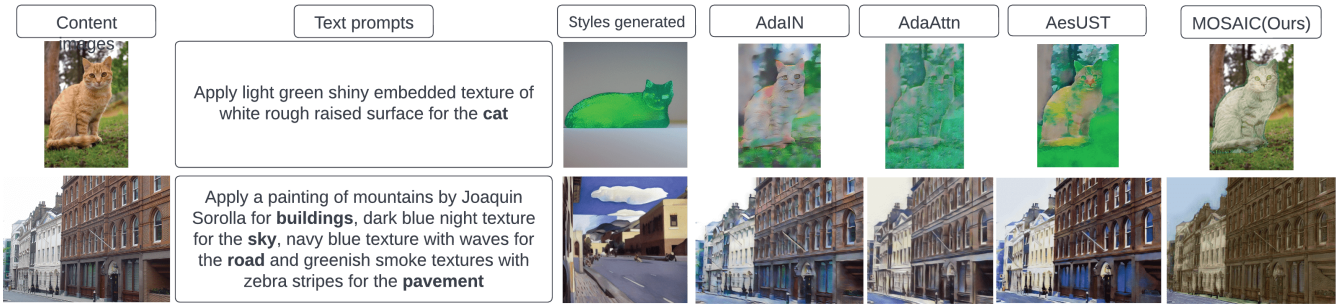

Figure 6: Comparing our model with SOTA Image based style transfer models, *AdaIN*[5], *AdaAttn*[68], *AesUST*[34], *MOSAIC(ours)*. This shows that the previous models cannot perform object-specific style transfer and shows the inaccuracy of generated style image.

face[77](stable-diffusion-v1.4[74]). In examining Figure 6, the results look comparable in quality, but the style transfer is not object-specific, and obtaining the style images which suit the text prompt every time can be challenging.

The advantage of models similar to ours $i.e.$, text-based models, are they don't need style images to transfer the style. Finding a style image that suits a user's description(text prompt) is difficult; hence, an image-based model will always have limitations.

4.4. Quantitative Analysis

4.4.1 User Study

For the quantitative comparison of our model, we have conducted a user study for quantitative analysis. For the user study, we have evaluated 120 Content images with 15 different stylization prompts per image for the following models: LDAST[6], ITStyler[7], CLIPStyler[17], Pix2Pix[43], Diffstyler[8] and MOSAIC. We have gathered responses from 78 users through a form, and collected their ratings for each stylized image. We have given the following scores for the users to select, from 1 being the least, to 10 being the

Method	Rating ↑
LDAST[6]	5.7
ITStyler[7]	7.2
CLIP-Styler[17]	4.6
Pix2Pix[43]	7.9
Diffstyler[8]	5.5
MOSAIC(ours)	**9.1**

Table 1: Analysis of User Ratings (ranging from 1 to 10)

Method	Patch-wise CLIP score ↑
LDAST[6]	0.1630
ITStyler[7]	0.2031
CLIP-Styler[17]	0.1742
Pix2Pix[43]	0.1934
Diffstyler[8]	0.1697
MOSAIC(ours)	**0.2671**

Table 2: Patch-wise CLIPScore comparison between state-of-the-art methods and MOSAIC(ours)

most accurately stylized images. The results of the User study survey can be visualized in Table 1.

4.4.2 Patch-wise CLIP score

Similar to CLIPStyler[17] which uses CLIP score [78] as a metric, we use a modified variant of CLIP score called patch-wise CLIP score as a metric to compare image stylization between different benchmarks. We used the same dataset of image-text pairs as in the previous section, but with simplified stylization prompts. What differentiates patch-wise CLIP score from the CLIP score is that CLIP score takes random crops of the whole image, whereas in patch-wise CLIP score, we take 8 random crops per object using the object wise bounding boxes of the image. The object wise bounding boxes are directly extracted from the object wise masks generated by SAM[15](Segment Anything Model). Due to this differentiation, we now have a well defined style(text) associated with the random crops that we take from the defined objects bounding boxes. We have evaluated the Patch-wise CLIP scores between the stylized images and the prompts for the following models: LDAST[6], ITStyler[7], CLIPStyler[17], Pix2Pix[43], Diffstyler[8] and MOSAIC. We can observe the effectiveness of Object-wise style transfer of MOSAIC from its CLIP score comparison from Table 2. The rest of the benchmarks tend to stylize the whole image with a mixture of styles instead of Object-wise Style transfer.

5. Deployment on Edge Devices

The following section presents the details on performance of each module in the pipeline. To deploy the pipeline on the edge devices we had to optimize each module individually. Refer to table 3 for latency's on individual modules and their efficient counter parts.

In the context of enhancing the performance and efficiency of our pipeline, several crucial modifications were made to key modules, as outlined below:

Segmentation Module: To achieve improved performance, we replaced the previously employed Segment Anything Model (SAM) [15] with a more efficient architecture known as MobileSAM [79]. This architectural change resulted in a remarkable reduction in latency from 456ms to 12ms, leading to an impressive 38x speedup.

Text Encoding Module: For encoding text into a unified space, a robust model such as CLIP was initially considered. However, the high inference time of 286ms associated with CLIP posed potential challenges in terms of pipeline latency. To address this concern, we opted to replace CLIP with the text encoder of MoTIS (Mobile Text to Image Search) [81], which substantially reduced the inference time to 98.6ms, resulting in a nearly 3x speedup.

Text Segmentation Task: While the ChatGPT-based model demonstrated excellent performance and was easily accessible through API calls, we sought to provide an offline version with greater control over output. To accomplish this, custom models, namely Transformer Large (9 Heads) with inference times of 276ms for the large variant and 95.4ms for the smaller variant (3 Heads), were developed.

Stylization Network: Through rigorous testing, it was observed that the Stylization Network exhibited consistent inference times of 30ms. As a result, the overall pipeline's inference time was determined to be 236ms. This is a noteworthy 5x speedup compared to any Diffusion Based Models that typically require 1 second to generate an image, considering the use of 50 sampling steps.

These adjustments demonstrate substantial enhancements to our pipeline's overall efficiency, making it well-suited for various real-world applications.

6. Limitations

The main limitation of this model comes with segmentation. The performance of the segmentation model aids in producing more pleasant stylized images. The output sometimes deteriorates due to unpleasant segmentation masks, as shown in Figure 7.

The other thing is about the long input prompts. This pipeline demands longer prompts for giving better pleasing outputs. This may look clumsy, as shown in Figure 8.

Module	Server(T4 GPU)		Edge	
	Architecture	Latency	Architecture	Latency
Segmentation	SAM [15]	456 ms	MobileSAM [79]	12 ms
Text Encoder	CLIP [80]	286 ms	MoTIS [81]	98.6 ms
Text Segmentation	Transformer Large	176 ms	Transformer Small	75.4 ms
CLIP Score	0.2671		0.2245	

Table 3: Comparision of Latencies of pipeline. For Server deployment we used a single T4 GPU and for edge deployment we benchmarked on Qualcomm SM8450 Snapdragon 8 Gen 1 Processor

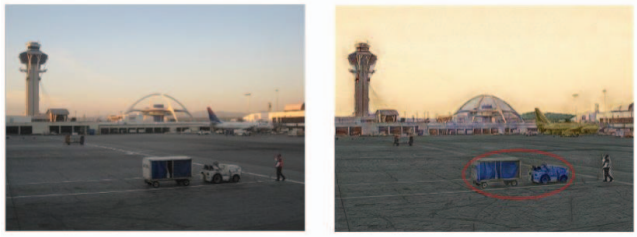

Figure 7: Case of imperfect mask from Image Segmentation Model

Figure 8: This shows the long prompt even for an Image with fewer objects.

7. Future Goals

In the future, we plan to improve the quality of our outputs and streamline our pipeline by integrating multiple modules into a single model using architectures such as GANs[41]. Additionally, we aim to enhance the model's ability to stylize individual objects rather than the entire image. The stylization of the whole picture results in many unseen artifacts are being produced due to the mixing of content from the out-of-mask regions. We can achieve more pleasing results by extending this method to do mask-aware style transfer. Due to masks, there was a sudden change in texture or color, sometimes creating an unpleasant look. For this, we can use bilateral grid mapping[82] and implement a smooth transition. Using bilateral grid mapping to generate the final output gives us aesthetically looking images as they are good at producing photo-realistic images. We also intend to investigate the optimal sequence of blocks through which the input data should flow to achieve the best results. This mask problem can also be addressed using Diffusion models as shown in [83]. Our method can be further extended to future language-based stylization models, so that the stylization would be object wise and well controlled by the user. By performing these modifications, we aim to make further advancements in object-wise stylization using text and contribute to the field's development.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *CoRR*, vol. abs/1508.06576, 2015.

[2] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[3] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," *Lecture Notes in Computer Science*, p. 694–711, 2016.

[4] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," *ArXiv*, vol. abs/1610.07629, 2016.

[5] X. Huang and S. J. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1510–1519, 2017.

[6] T.-J. Fu, X. E. Wang, and W. Y. Wang, "Language-driven image style transfer," *ArXiv*, vol. abs/2106.00178, 2021.

[7] Y.-H. Bai, J. Liu, C. Dong, and C. Yuan, "Itstyler: Image-optimized text-based style transfer," *ArXiv*, vol. abs/2301.10916, 2023.

[8] N. Huang, Y. xin Zhang, F. Tang, C. Ma, H. Huang, Y. Zhang, W. Dong, and C. Xu, "Diffstyler: Controllable dual diffusion for text-driven image stylization," *ArXiv*, vol. abs/2211.10682, 2022.

[9] L. Kurzman, D. Vazquez, and I. Laradji, "Class-based styling: Real-time localized style transfer with semantic seg-

mentation," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

[10] Z. Huang, J. Zhang, and J. Liao, "Style mixer: Semantic-aware multi-style transfer network," *Computer Graphics Forum*, vol. 38, pp. 469–480, oct 2019.

[11] T. Lüddecke and A. Ecker, "Image segmentation using text and image prompts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7086–7096, 2022.

[12] B. Li, K. Q. Weinberger, S. J. Belongie, V. Koltun, and R. Ranftl, "Language-driven semantic segmentation," *ArXiv*, vol. abs/2201.03546, 2022.

[13] G. Ghiasi, X. Gu, Y. Cui, and T. Lin, "Open-vocabulary image segmentation," *CoRR*, vol. abs/2112.12143, 2021.

[14] F. Liang, B. Wu, X. Dai, K. Li, Y. Zhao, H. Zhang, P. Zhang, P. Vajda, and D. Marculescu, "Open-vocabulary semantic segmentation with mask-adapted clip," *ArXiv*, vol. abs/2210.04150, 2022.

[15] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023.

[16] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, 2021.

[17] G. Kwon and J.-C. Ye, "Clipstyler: Image style transfer with a single text condition," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18041–18050, 2021.

[18] A. P. Suresh, S. Jain, P. Noinongyao, and A. Ganguly, "Fast-clipstyler: Optimisation-free text-based image style transfer using style representations," 2022.

[19] J. E. Kyprianidis, J. Collomosse, T. Wang, and T. Isenberg, "State of the "art": A taxonomy of artistic stylization techniques for images and video," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 5, pp. 866–885, 2013.

[20] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 1033–1038 vol.2, 1999.

[21] M. Elad and P. Milanfar, "Style transfer via texture synthesis," *IEEE Transactions on Image Processing*, vol. 26, p. 2338–2351, May 2017.

[22] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 341–346, 2001.

[23] D. J. Heeger and J. R. Bergen, "Pyramid-based texture analysis/synthesis," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 229–238, 1995.

[24] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

[25] M. Ruder, A. Dosovitskiy, and T. Brox, "Artistic style transfer for videos," *CoRR*, vol. abs/1604.08610, 2016.

[26] C. Li and M. Wand, "Combining markov random fields and convolutional neural networks for image synthesis," *CoRR*, vol. abs/1601.04589, 2016.

[27] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman, "Controlling perceptual factors in neural style transfer," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[28] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stylebank: An explicit representation for neural image style transfer," *CoRR*, vol. abs/1703.09210, 2017.

[29] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," *Lecture Notes in Computer Science*, p. 702–716, 2016.

[30] X. E. Wang, G. Oxholm, D. Zhang, and Y. fang Wang, "Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7178–7186, 2016.

[31] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Diversified texture synthesis with feed-forward networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 266–274, 2017.

[32] T. Q. Chen and M. W. Schmidt, "Fast patch-based style transfer of arbitrary style," *ArXiv*, vol. abs/1612.04337, 2016.

[33] S. Liu, T. Lin, D. He, F. Li, M. Wang, X. Li, Z. Sun, Q. Li, and E. Ding, "Adaattn: Revisit attention mechanism in arbitrary neural style transfer," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6629–6638, 2021.

[34] Z. Wang, Z. Zhang, L. Zhao, Z. Zuo, A. Li, W. Xing, and D. Lu, "Aesust: Towards aesthetic-enhanced universal style transfer," *Proceedings of the 30th ACM International Conference on Multimedia*, 2022.

[35] M. Zhu, X. He, N. Wang, X. Wang, and X. Gao, "All-to-key attention for arbitrary style transfer," *ArXiv*, vol. abs/2212.04105, 2022.

[36] P. Wilmot, E. Risser, and C. Barnes, "Stable and controllable neural texture synthesis and style transfer using histogram losses," *ArXiv*, vol. abs/1701.08893, 2017.

[37] X. Peng and K. Saenko, "Synthetic to real adaptation with deep generative correlation alignment networks," *ArXiv*, vol. abs/1701.05524, 2017.

[38] Y. Li, N. Wang, J. Liu, and X. Hou, "Demystifying neural style transfer," *ArXiv*, vol. abs/1701.01036, 2017.

[39] G. P. Laput, M. Dontcheva, G. Wilensky, W. Chang, A. Agarwala, J. Linder, and E. Adar, "Pixeltone: A multimodal interface for image editing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2185–2194, 2013.

[40] L. Yang, Z. Zhang, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, M.-H. Yang, and B. Cui, "Diffusion models: A comprehensive survey of methods and applications," *ArXiv*, vol. abs/2209.00796, 2022.

[41] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[42] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

[43] 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*, pp. 1125–1134, 2017.

[44] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," *ArXiv*, vol. abs/1611.02200, 2016.

[45] M.-Y. Liu, T. M. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," *ArXiv*, vol. abs/1703.00848, 2017.

[46] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *ArXiv*, vol. abs/1701.07875, 2017.

[47] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," *Advances in neural information processing systems*, vol. 29, 2016.

[48] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *International conference on machine learning*, pp. 1060–1069, PMLR, 2016.

[49] A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International conference on machine learning*, pp. 1747–1756, PMLR, 2016.

[50] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[51] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5077–5086, 2017.

[52] E. L. Denton, S. Chintala, R. Fergus, *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," *Advances in neural information processing systems*, vol. 28, 2015.

[53] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3722–3731, 2017.

[54] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *ArXiv*, vol. abs/1606.03498, 2016.

[55] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, "Styleclip: Text-driven manipulation of stylegan imagery," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2085–2094, 2021.

[56] R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or, "Stylegan-nada: Clip-guided domain adaptation of image generators," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–13, 2022.

[57] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

[58] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.

[59] G. Li, I. Yun, J.-H. Kim, and J. Kim, "Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation," *ArXiv*, vol. abs/1907.11357, 2019.

[60] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, "Transunet: Transformers make strong encoders for medical image segmentation," *CoRR*, vol. abs/2102.04306, 2021.

[61] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, *et al.*, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6881–6890, 2021.

[62] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[63] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7262–7272, 2021.

[64] Z. Wang, Y. Lu, Q. Li, X. Tao, Y. Guo, M. Gong, and T. Liu, "Cris: Clip-driven referring image segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11686–11695, 2022.

[65] N. Tumanyan, O. Bar-Tal, S. Bagon, and T. Dekel, "Splicing vit features for semantic appearance transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10748–10757, 2022.

[66] J. Huo, S. Jin, W. Li, J. Wu, Y.-K. Lai, Y. Shi, and Y. Gao, "Manifold alignment for semantically aligned style transfer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14861–14869, 2021.

[67] K. Hong, S. Jeon, H. Yang, J. Fu, and H. Byun, "Domain-aware universal style transfer," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14589–14597, 2021.

[68] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.

[69] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell,

et al., "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[70] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[71] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[72] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[73] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[74] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

[75] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[76] OpenAI, "Chatgpt." `https://beta.openai.com/docs/guides/chat/index`, 2021. [Accessed: 2023-03-8].

[77] "Hugging face website community." `https://huggingface.co/`, note = Accessed: 2023-03-08.

[78] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi, "Clipscore: A reference-free evaluation metric for image captioning," *arXiv preprint arXiv:2104.08718*, 2021.

[79] C. Zhang, D. Han, Y. Qiao, J. U. Kim, S.-H. Bae, S. Lee, and C. S. Hong, "Faster segment anything: Towards lightweight sam for mobile applications," *arXiv preprint arXiv:2306.14289*, 2023.

[80] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible scaling laws for contrastive language-image learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2818–2829, 2023.

[81] S. Ren and K. Q. Zhu, "Leaner and faster: Two-stage model compression for lightweight text-image retrieval," *arXiv preprint arXiv:2204.13913*, 2022.

[82] J. Chen, A. Adams, N. Wadhwa, and S. W. Hasinoff, "Bilateral guided upsampling," *ACM Transactions on Graphics (TOG)*, vol. 35, pp. 1 – 8, 2016.

[83] Á. B. Jiménez, "Mixture of diffusers for scene composition and high resolution image generation," *ArXiv*, vol. abs/2302.02412, 2023.