FastCLIPstyler: Optimisation-free Text-based Image Style Transfer Using Style Representations

Ananda Padhmanabhan Suresh* Sanjana Jain* Pavit Noinongyao Ankush Ganguly
Ukrit Watchareeruetai Aubin Samacoits

Sertis Vision Lab
597/5 Sukhumvit Road, Watthana, Bangkok, 10110, Thailand
{asure,sjain,ponio,agang,uwatc,asama}@sertiscorp.com
*Both authors contributed equally to this work

Abstract

In recent years, language-driven artistic style transfer has emerged as a new type of style transfer technique, eliminating the need for a reference style image by using natural language descriptions of the style. The first model to achieve this, called CLIPstyler, has demonstrated impressive stylisation results. However, its lengthy optimisation procedure at runtime for each query limits its suitability for many practical applications. In this work, we present FastCLIPstyler, a generalised text-based image style transfer model capable of stylising images in a single forward pass for arbitrary text inputs. Furthermore, we introduce EdgeCLIPstyler, a lightweight model designed for compatibility with resource-constrained devices. Through quantitative and qualitative comparisons with state-of-the-art approaches, we demonstrate that our models achieve superior stylisation quality based on measurable metrics while offering significantly improved runtime efficiency, particularly on edge devices.

1. Introduction

The objective of style transfer is to recompose a content image with the semantic texture of a style image. Research in this domain has been inspired by the work of Gatys *et al*. [9], who demonstrated the capability of Convolutional Neural Networks (CNNs) to generate stylised images by extracting content information from arbitrary images and style information from well-known artworks [23]. Their algorithm employs a pre-trained VGG-19 network [22] to define content and style loss and jointly optimise them to create stylised images. Since then, Li *et al*. [16] reformulated the problem as distribution alignment, introducing new loss functions for the same. Ulyanov *et al*. [25] and Li *et al*. [15] presented models that can apply a reference image's style in a single neural network pass, while Ghiasi *et al*. [10], Huang *et al*. [11] (AdaIN) and Chen *et al*. [3] developed models that can transfer style from an arbitrary style image without requiring optimisation at runtime. More recent models, such as SANet [19] and AdaAttn [17], have implemented attention mechanisms to improve the quality of synthesised images. While these approaches successfully create visually pleasing stylised images, they rely on the availability of the desired reference style image, which is not always available.

Recognising this limitation, [14] developed a method called CLIPstyler to solve Language Driven Artistic Style Transfer (LDAST) [7], the concept of which is to stylise images based on a text prompt instead of a reference style image. CLIPstyler employs CLIP [20], an embedding model that projects image and text to a shared embedding space, enabling the application of a style text prompt to a content image. One drawback of this is its time-consuming optimisation procedure at inference time for each text query, making it unsuitable for real-world applications. To address this issue, Fu et al. [7] recently introduced Contrastive Language Visual Artist (CLVA), capable of stylising images using a general text prompt without optimisation. However, their model does not generalise well to unseen prompts and is not able to support resource-constrained devices, losing out on a lot of practical applications for LDAST.

In this work, we introduce FastCLIPstyler, a text-based image style transfer model which, unlike CLIPstyler, eliminates the need for runtime optimisation. We achieve this by incorporating a pre-trained, generalisable, purely-vision based style transfer network into the CLIPstyler framework. The CLIP model facilitates this approach, enabling training without dependence on labelled datasets such as Artemis [1]

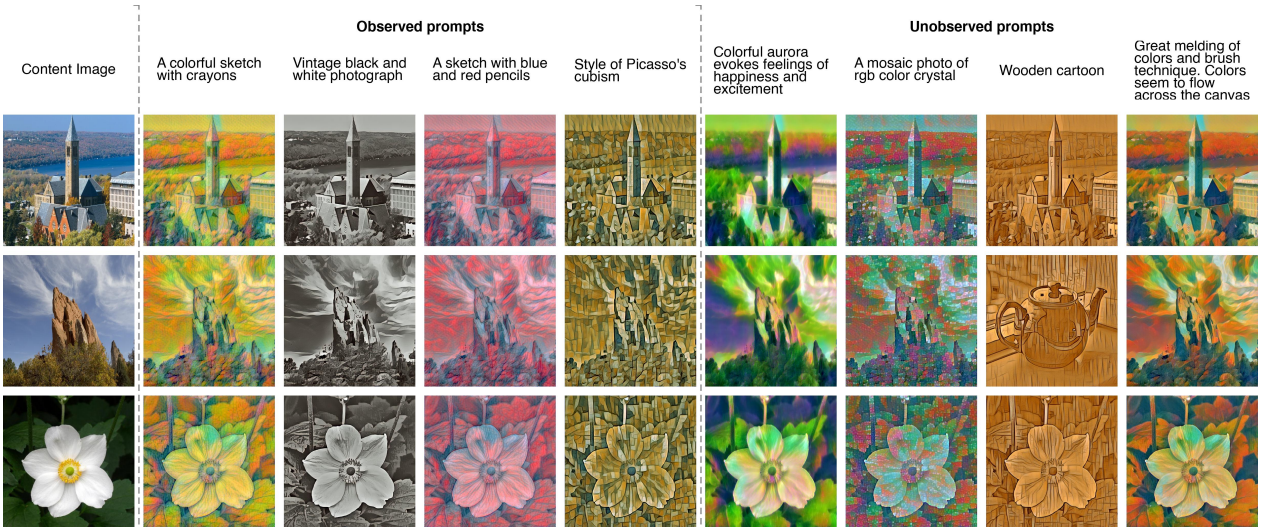


Figure 1. General performance of our model on observed and unobserved prompts. It is correctly able to identify styles across colours, art genres, and generic styles.

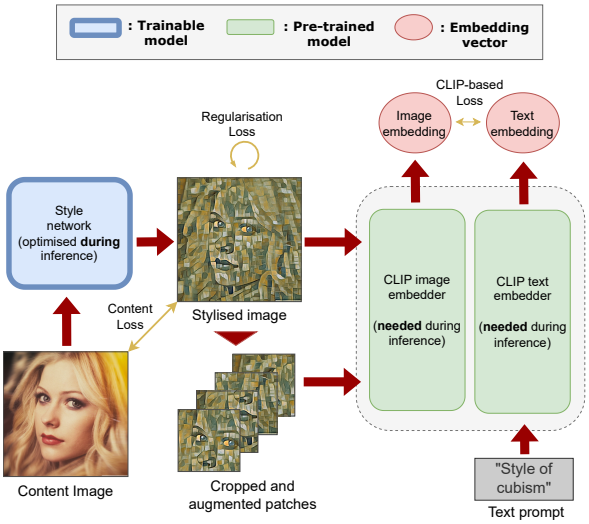


Figure 2. The architecture diagram of CLIPstyler model.

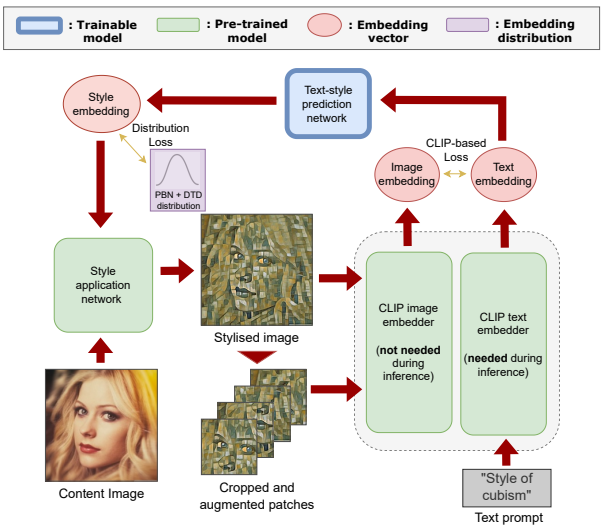


Figure 3. The architecture diagram of our FastCLIPstyler model.

and Describable Textures Dataset (DTD) [5], which usually requires a time-consuming and expensive annotation procedure to create. We also outline a strategy for generating a text prompt dataset, which begins with a set of basic prompts and combines them into more intricate combinations. In addition, we also demonstrate that our model has superior generalisation ability to unseen prompts, especially when compared to the current state-of-the-art in LDAST, CLVA. Furthermore, to broaden our impact, we also introduce EdgeCLIPstyler: a streamlined, edge-compatible iteration of FastCLIPstyler, aptly designed for low-powered devices with minor architectural tweaks. As illustrated in Fig. 1, our approach adeptly applies styles from both familiar and unfamiliar prompts, yielding visually pleasing results.

We highlight the following as the main contributions of our paper:

- We introduce FastCLIPstyler, a text-based image style transfer model that can stylise images in a single forward pass through a network for arbitrary text inputs, without the need for runtime optimisation.
- In our approach to the LDAST problem, we formulate FastCLIPstyler so that it only requires a dataset of

text prompts for training, eliminating the need for corresponding style images. Instead, we employ a novel label generation process, optimising the text-style prediction network for each datapoint individually.

- Using both qualitative and quantitative evaluations, we demonstrate that FastCLIPstyler outperforms CLIPstyler in terms of image generation speed at runtime (730x faster) and surpasses CLVA [7] on stylised image quality.
- We also introduce an edge-compatible variant, Edge-CLIPstyler, which is, to the best of our knowledge, the first model to enable LDAST on edge devices.

2. Method

2.1. Background knowledge

In this section, we outline the key components our model adopts from established works: Ghiasi's style transfer network [10] CLIPstyler [14]

Ghiasi's style transfer network: Ghiasi *et al*. [10] introduced one of the first purely vision-based style transfer networks capable of transferring the style of any reference image to a content image in a single forward pass. The model consists of a style prediction network that converts a reference style image into a 100-dimensional style embedding and a style application network that applies the embedding to the content image. As Ghiasi *et al*.'s style prediction network relies on a style image, it is not applicable to us; we just adopt their style application network, which takes the content image and a style embedding as inputs.

CLIPstyler: While most style transfer approaches utilise networks like VGG to model their transfer process, CLIPstyler [14] employs the CLIP model [20] for style transfer, enabling the use of natural language descriptions instead of reference images. The content image is processed through an image-to-image CNN that directly transforms it into a stylised version. CLIP computes a similarity score between the generated image and the user's text query, and the CNN is optimised until the generated image resembles the textual description. The architecture of CLIPstyler is shown in Fig. 2.

2.2. FastCLIPstyler

The CLIPstyler framework trains a CNN model from scratch during inference to transform content images into stylised versions. However, leveraging existing style transfer models can eliminate this ground-up training. For instance, the Ghiasi et al. model [10] uses a style application network that stylises content images based on a 100-dimensional style representation. Instead of training a full CNN model, we train a compact, fully-connected feed-forward network called the 'text-style prediction network', which inputs CLIP text embeddings and predicts their 100-dimensional style representations. The FastCLIPstyler architecture is shown in Fig. 3. We selected Ghiasi's network for its robust generalisation capabilities, demonstrated by its performance on a variety of unseen style images. Our pipeline can be easily adapted to other image-based style transfer methods, provided they offer an explicit style representation.

Our text-style prediction network can be trained to learn the mapping between text embeddings and their corresponding style representation in the Ghiasi-style space. For doing this, the network can be trained using a dataset of text prompts and their corresponding style embeddings. In order to get these style embeddings, we optimise the untrained model to each of the prompts, one at a time, with the guidance of CLIP. Optimising the neural network for each query in this manner is similar to how it was done in CLIPstyler, which served as the inspiration for this approach. While in CLIPstyler, the optimisation procedure for a particular query generated the corresponding final image, what we want from our network at this stage is the intermediate style embedding corresponding to a query. This creates 'labels' for each query in the dataset, which can then be used to train a generalised network. Compared to CLIPstyler, our trainable network has a much more simplified task to learn — it only needs to predict an appropriate input for the pre-trained style application model without worrying about preserving content or artistic quality in the final image.

The text-style prediction network that we use is a simple, fully-connected feed-forward network comprising four hidden layers that takes a 512-dimensional text embedding as input and outputs a 100-dimensional style representation. The activation function in between the layers is the Leaky ReLU [18] with a negative slope of 0.2. The final layer uses the tanh activation function to normalise the style representations between -1 and 1. The full feed-forward model specification, along with the hyperparameters, can be found in the supplementary section.

In Fig. 2 and Fig. 3, the architectural differences between CLIPstyler and our proposed FastCLIPstyler are highlighted. FastCLIPstyler needs one-time training using a dataset and then requires no inference-time optimisation, while CLIPstyler optimises a CNN for each input during inference, making it more time-consuming. This is because CLIPstyler transforms content images into stylised ones by optimising a dedicated network each time, whereas FastCLIPstyler converts an embedding in the CLIP space to the style application network, a less complex task. This results in FastCLIPstyler being more efficient at inference.

2.3. Dataset generation

In training our model, we capitalise on a key advantage: our method does not require a labelled image dataset, as

it can generate its own labels by leveraging CLIP's capabilities. In this regard, CLIP essentially acts as our 'labeller'. The initial dataset we need consists of a diverse set of style prompts, which can be easily generated from a corpus, as opposed to images that must be collected from the real world.

To assemble this list of prompts, we combine keywords encompassing colours, textures, art styles, and objects with distinct textures, ultimately generating a dataset of 4,300 prompts. For example, by pairing art styles and real-world objects, we create prompts like 'mosaic stone wall' or 'acrylic snow'. To further enhance the generalisation and style transfer quality of our model, we employ ChatGPT's [2] language generation capabilities, which allows us to create an additional 1,500 style prompts and increase the dataset's diversity. These prompts cover aspects of nature, emotional states, and examples similar to those found in the ArtEmis dataset. We demonstrate that this relatively simple strategy of data generation is adequate to train the model so that it's capable of generalising across a wide range of seen and unseen prompts.

To the best of our knowledge, our approach is unique in that it does not depend directly on labelled datasets like ArtEmis or DTD. The ability to self-generate a dataset for training an LDAST model represents a significant advantage, as it eliminates the need for a costly data collection phase. Additional information on our data generation strategy can be found in the supplementary section.

2.4. Loss function

In this section, we begin by exploring the loss function we use to optimise the model for each text prompt to generate their corresponding style embedding labels. We later discuss the loss function used to train the generalised model using these style embedding labels.

As proposed by Kwon and Ye [14] and originally suggested by StyleGAN-NADA [8], we adopt the directional loss L_{dir} to measure the closeness of the image generated by the style transfer model with the text input using CLIP in a stable way.

We also adopt PatchCLIP loss, L_{patch}, a novel loss term proposed by [14] that was shown to greatly improve the stylisation results. CLIPstyler also defines a content loss that is meant to ensure that the content is preserved during the transfer and a total variation regularisation loss to alleviate unwanted artefacts. However, we find that these loss terms are unnecessary for our training since the Ghiasi network has already been trained to preserve content, as well as to apply the style in a manner that is visually pleasing. Hence we only use the CLIP-based losses, L_{dir} and L_{patch}, from [14].

However, using these two loss functions alone poses a practical challenge. The entire 100-dimensional space of real numbers does not constitute a valid input into the Ghiasi network. The samples must be drawn from a specific region of the $\mathbb{R}^{100}$ for the model to work as intended. However, there is nothing stopping the text-style prediction network from predicting style embeddings that are well out of the valid input region of the Ghiasi network. To address this, we construct another loss term that penalises the network for making predictions outside the valid input region of the Ghiasi network. Since the model was trained on the PBN [13] and DTD [5] datasets, we compute the style embeddings of all images in these datasets and assume a normal distribution over these in order to calculate the likelihood of a predicted embedding vector being sampled from this distribution. This likelihood defines a distribution loss term L_{dis}:

$$L_{dis} = (x - \mu_{data})^T \Sigma_{data}^{-1} (x - \mu_{data}), \quad (1)$$

where x is the embedding tensor, μ_{data} and Σ_{data} are the mean and covariance of the embeddings of the PBN and DTD datasets. This term penalises the model for predicting style embeddings that lie far from the distribution of style embeddings of real images.

Putting these together, our overall loss function is formulated as:

$$L_{total} = \lambda_{dir} L_{dir} + \lambda_{patch} L_{patch} + \lambda_{dis} L_{dis}, \quad (2)$$

where λ_{dir}, λ_{patch} and λ_{dis} are coefficients governing the weights of their losses respectively.

To generate text embeddings, we pass the text prompts through the CLIP text embedder. Once we have the dataset of text prompts and their corresponding text and style embeddings, we train the generalised text-style prediction network using a simple mean squared average error loss.

2.5. EdgeCLIPstyler: Edge-compatible FastCLIPstyler

FastCLIPstyler, built upon the powerful CLIP model, yields impressive stylisation results but is computationally expensive, making it unsuitable for resource-constrained edge devices. To address this limitation, we introduce EdgeCLIPstyler, an edge-compatible FastCLIPstyler framework adapted for edge devices using the more resource-efficient Sentence-BERT paraphrase-albert-v2 [21] text-embedder model.

Compared to CLIP's text embedder, the Sentence-BERT paraphrase-albert-v2 embedder is smaller and demands fewer computational resources, making it ideal for our edge-based approach. We train the text-style prediction network using embeddings generated from the Sentence-BERT model. Although the CLIP model is still needed for supervising model training (in the CLIP loss computation), any text embedder can be utilised during model inference.

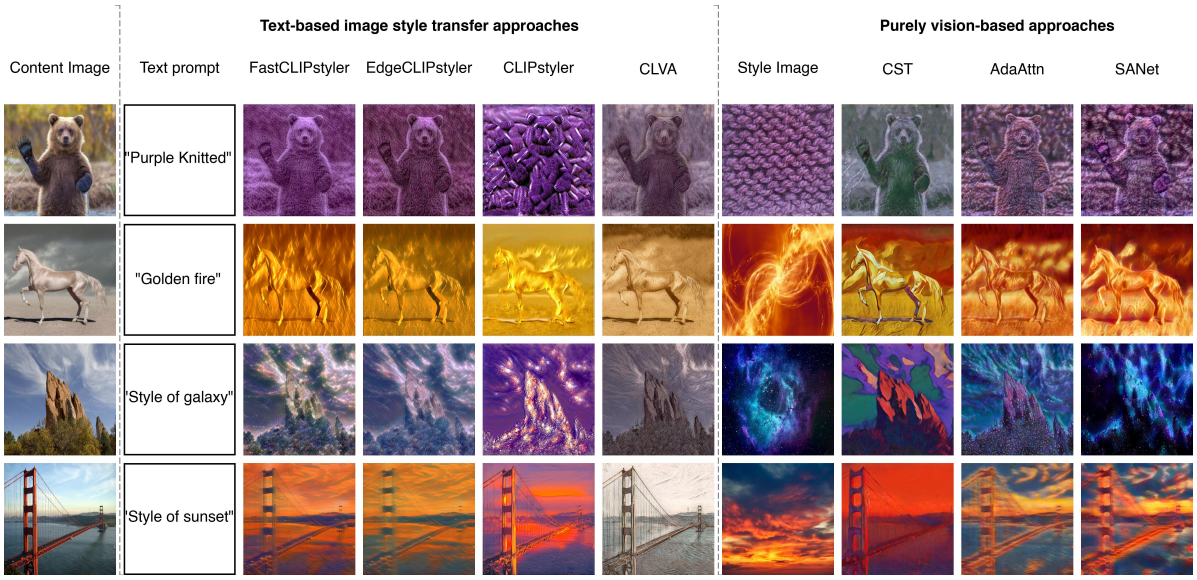


Figure 4. Comparison of our model with other state-of-the-art models in its ability to transfer styles of various complex queries and artworks.

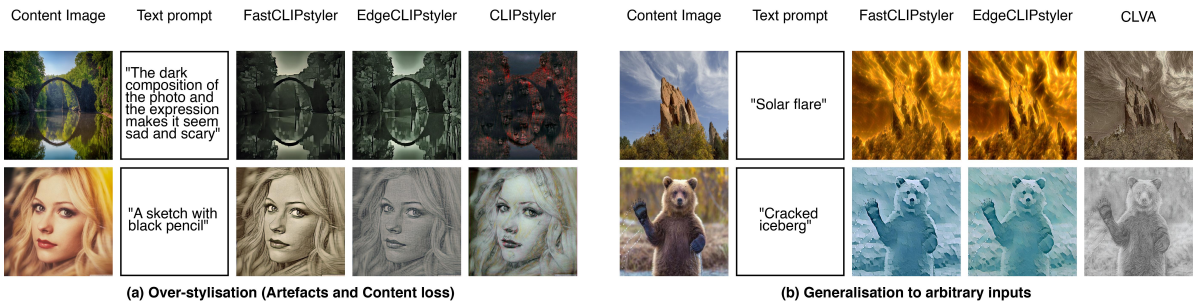


Figure 5. Examples where limitations of CLIPstyler and LDAST are highlighted and compared to our model.

This slight architectural modification enables us to achieve an edge-compatible LDAST model.

As we demonstrate in Sec. 3, EdgeCLIPstyler is capable of running on edge devices while delivering impressive stylisation results; it can be an excellent tool for social media and graphic designing applications. This also finds applications in tools like video conferencing background enhancement, social media filters, photo editing applications etc., directly on mobile devices, without relying on remote servers for processing, which has implications for privacy and data security.

3. Results

All experiments were performed on a 6-core Intel i5 desktop with 16 GB RAM and a single 8GB Nvidia GeForce RTX 2070 GPU. For benchmarking our inference time on low-powered CPU-only devices, we use the Intel NUC Kit NUC7i5BNH with 8 GB RAM and Raspberry Pi 3B+ with 1 GB RAM.

During the dataset generation step, for the CLIP image embedder, we chose the ViT-B/32 backbone to generate the 512-dimensional style embeddings, making our experiments comparable to the CLIPstyler framework. Our choice for the CLIP model is based on the trade-off between computational efficiency and model performance presented by [20].

3.1. Qualitative evaluation

In Fig. 1, we show the style transfer capability of our model. It has an awareness of a wide range of queries and is able to apply colours, textures, art styles, and real-world objects, as well as compound statements combining these. We also show that the model has good performance when it comes to handling prompts that it has never seen during training. Figure 4 shows the comparison of our approaches with other state-of-the-art techniques in style transfer. We

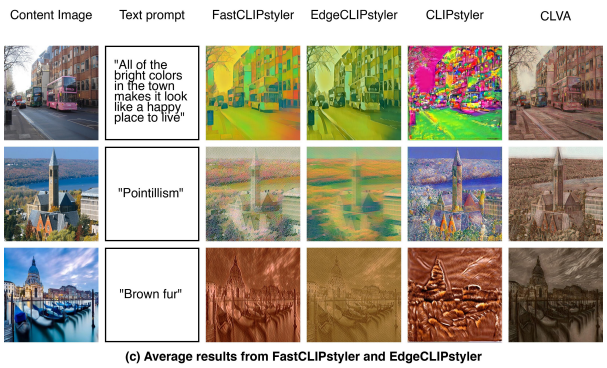


Figure 6. Examples where some of the limitations of our models are highlighted.

also compare our models with recent examples of purely vision-based style transfer techniques, namely, CST [24], SANet [19], and AdaAttn [17]. Despite not having the advantage of an explicit reference style image during style transfer, our model generates images similar to these techniques.

Figure 5 demonstrates the superior performance of our models in certain instances compared to CLIPstyler and CLVA. Specifically, in Figure 5(a), CLIPstyler, influenced by the first prompt, introduces disturbing faces, and the second prompt leads to further undesirable artefacts. In these cases, our models maintain image integrity without such issues.

Figure 5(b) emphasises instances where CLVA's style application fails to align with the given prompt. We observe that CLVA particularly struggles when the prompt significantly deviates from the ArtEmis and DTD datasets, which served as its training source. In contrast, as depicted in Fig. 1, our model demonstrates the capacity to adapt to new, open-ended prompts that are reminiscent of the ones from the ArtEmis dataset, such as 'Colorful aurora evokes feelings of happiness and excitement.'

In Fig. 6, we acknowledge some limitations of our model compared to CLIPstyler and CLVA. Our models tend to produce more monochromatic images when a specific colour is implied directly or indirectly in the prompt, as seen in the 'brown fur' and 'wooden cartoon' examples. EdgeCLIPstyler occasionally falls behind FastCLIPstyler, as in the 'pointillism' case. Additionally, CLVA's images exhibit a superior 'artistic quality,' as demonstrated in the 'bright colours in the town' prompt. However, CLVA fails to maintain consistency with the prompt when moving beyond the training distribution of ArtEmis and DTD, whereas our models remain more robust.

3.2. Quantitative evaluation

For the quantitative evaluation of our proposed text-based image style transfer model, we create a dataset containing 80 prompts, including general prompts and prompts inspired by the DTD [6] and ArtEmis [1] datasets. In order to perform an automatic metric-based evaluation, we compute four key metrics: Vision Language Semantic (VLS), Structural SIMilarity (SSIM), Percept, and FAD. VLS [27] is the CLIP [20] cosine similarity between style instructions and the stylisation results. SSIM [26] compares the image quality based on contrast, luminance, and structural aspects between two images. Percept [7] computes the style reconstruction loss [12] from the gram matrix of visual features between two images. Lastly, FAD [7] computes the L2 distance between the activation maps of stylised images obtained from InceptionV3. To compute the SSIM, Percept, and FAD metrics, we utilise a stylised image ground truth for comparison. Following [7], we generate semi-ground truth (Semi-GT) results using the state-of-the-art purely vision-based style transfer method, AdaAttn [17], directly from style images. Due to the inherent limitations of the automatic metrics, we also conducted a user study for the quantitative evaluation through an online user survey performed with 75 participants. Each participant was shown a set of 20 randomly selected prompts, each consisting of results from the four models: CLIP, CLVA, FastCLIPstyler, and EdgeCLIPstyler. They were asked to rate the quality of the stylised images on a scale of 1 to 5. More details on how the quantitative experiments were set up can be found in the supplementary section.

In Tab. 1, we show the quantitative evaluation of our model in comparison to other baselines for the LDAST task. From the automatic metrics evaluation, our FastCLIPstyler excels in preserving structural similarity (highest SSIM) with the Semi-GT. CLIPstyler demonstrates the closest match to the text prompt based on the VLS metric, as it directly optimises for VLS for each datapoint. At the same time, our model achieves the lowest stylisation Percept loss, indicating strong stylisation performance on par with the Semi-GT. CLIPstyler, however, experiences the highest Precept loss, likely indicating over-stylisation. EdgeCLIPstyler achieves the lowest overall similarity distance (indicated by FAD) with the Semi-GT, while FastCLIPstyler achieves a close and comparable FAD distance. In terms of user study results, our methods surpass other techniques. One contributing factor seems to be that CLVA struggles to perform well when faced with unseen prompts. Another reason is that while CLIPstyler is a viable option, it occasionally generates stylised images containing random artefacts, leading to odd-looking results.

Although EdgeCLIPstyler is capable of achieving stylisation mostly on par with FastCLIPstyler, it is worth noting that EdgeCLIPstyler faces challenges when dealing with

Method	VLS ↑	SSIM ↑	Percept ↓	FAD ↓	Human evaluation ↑
CLVA	0.092	0.375	24.93	1.032	2.96
CLIPstyler	**0.194**	0.218	23.56	1.035	3.17
FastCLIPstyler	0.113	**0.418**	**21.83**	1.033	**3.39**
EdgeCLIPstyler	0.091	0.368	23.14	**1.024**	3.18

Table 1. Various metrics that we use to quantify the performance of different models

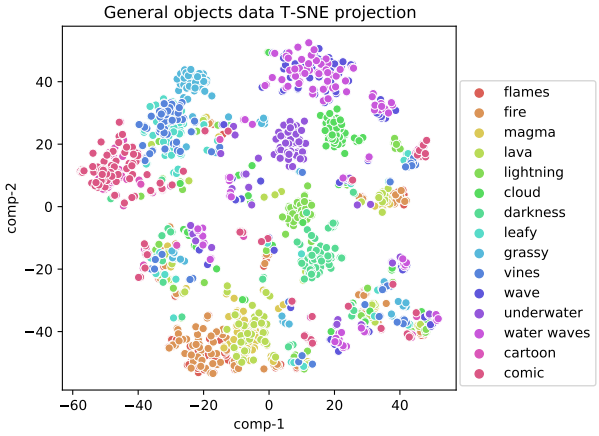


Figure 7. General objects t-SNE visualisation

some specific queries. This is because the underlying BERT model lacks the robustness in capturing visual perception while associating text with images like a CLIP text embedder [4].

3.3. Inference time benchmarking

We perform inference time benchmarking for various models on different platforms, which can be found in Tab. 2. Our FastCLIPstyler, though slower than CLVA on the powerful RTX 2070 SUPER GPU, still outperforms CLIPstyler by 730x. Additionally, it is more adaptable to low-powered CPU-only devices, running three times faster than CLVA on the edge devices we tested on. Our proposed EdgeCLIPstyler model, which uses the Sentence-BERT paraphrase-albert-v2 text embedder [21] during inference, is edge-compatible and performs comparably to CLVA on the GPU. Most importantly, it is ∼15x faster than CLVA on resource-constrained devices, taking just 0.39 seconds on the 6x i5-9500 CPU and 1.3 seconds on the Intel NUC device, as opposed to the respective durations of 5 seconds and 20.9 seconds for CLVA. Moreover, we successfully demonstrate that our EdgeCLIPstyler model can load and run on a Raspberry Pi 3B+ device in 15 seconds. While not real-time, it is important to note that Raspberry Pi 3B+ is a very low-powered device and could not support loading the CLIPstyler, FastCLIPstyler and CLVA models due to memory issues. It should also be noted that though [14] presented a faster version of CLIPstyler, it was trained to support a single style at a time, making it incomparable to the other approaches. As a result, we have not included this model in our inference time benchmarking experiments.

The results demonstrate the high adaptability of our pipeline and its ability to achieve impressive performance even on resource-constrained devices. This opens up exciting possibilities for deploying our model in various settings, including low-power edge devices, embedded systems, and mobile applications.

3.4. Embedding space mapping

Ghiasi *et al*. [10] have successfully demonstrated that the embedding space of their style transfer network captures semantic information about styles. As we adopt their style embedding space to fit our text-style prediction network, we verify that our prediction network is also able to preserve the semantic information upon mapping from the text embeddings.

To do so, we generate the text embeddings and corresponding style embeddings for various text prompts using our generalised text-style prediction network. Figure 7 illustrates the two-dimensional t-SNE plot of the style embeddings obtained by passing various combinations of generated text prompts through our text-style prediction network. As can be seen, our prediction network successfully maps semantically similar queries closer together in the dimensionally-reduced style embedding space. More explorations on the embedding space visualisations can be seen in the supplementary section.

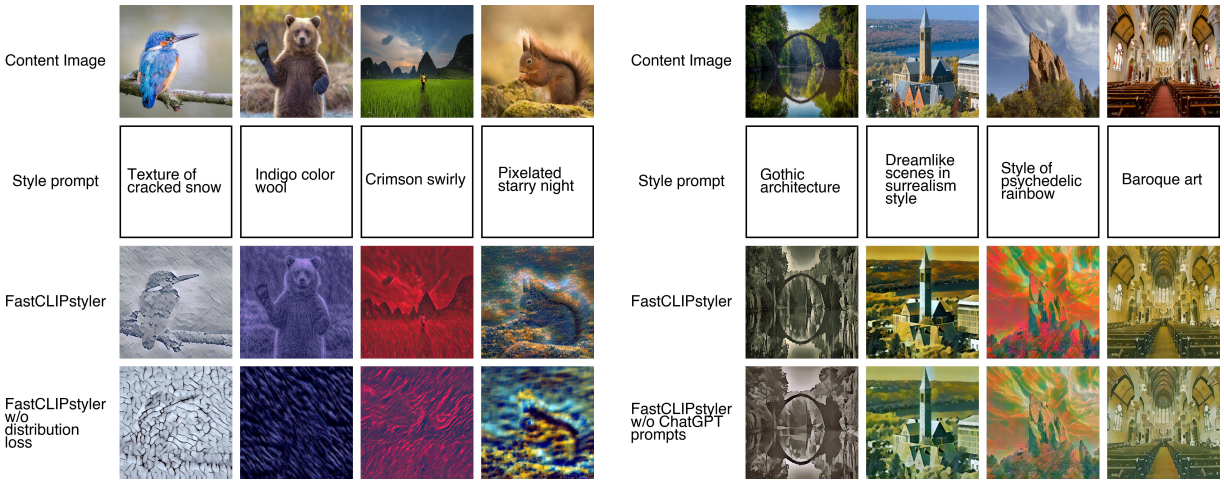
4. Ablation Study

4.1. Effect of distribution loss

Our proposed frameworks involve a text-style prediction network with a 100-dimensional output embedding space that is much larger than the valid region of Ghiasi network's embedding input. Hence, we propose the distribution loss to constrain the embedding towards the valid region of the input space of Ghiasi network, leading to much more visually pleasing results. This impact can be observed in Fig. 8 (a), where the stylised images have a more difficult time achieving content preservation and are heavily over-stylised.

Device	FastCLIPstyler	EdgeCLIPstyler	CLVA	CLIPstyler
RTX 2070 SUPER	70.444	24.891	25.088	51.360×10^3
6X i5-9500 CPU	1770.740	390.328	5520.347	13.098×10^5
Intel NUC	7138.115	1297.359	20998.551	44.26×10^5
Raspberry PI 3B+	NA	15.011×10^3	NA	NA

Table 2. Benchmarking for the time taken (in milliseconds) for various techniques.



(a) Distribution loss (b) ChatGPT prompts for training

Figure 8. Comparison of the stylised images with various content and text prompts. The results from the absence of crucial components in our pipelines are shown in different columns.

4.2. Effect of dataset generation using ChatGPT prompts

The inclusion of ChatGPT-based prompts in our dataset generation step helps the model to generalise better and support a wide range of style queries without the requirement for a reference style image. We demonstrate the impact of ChatGPT-based prompts in Fig. 8 (b), where the inclusion of ChatGPT queries enhances the model performance, ensuring that the intricate details of the style prompt are captured better.

5. Discussion and conclusion

By integrating pre-trained models from the purely vision-based style transfer domain into the CLIPstyler framework, we have developed the FastCLIPstyler model, capable of stylising a content image in a single forward pass. This model adeptly applies various styles and textures, as described in natural language, resulting in visually appealing images. We also introduce EdgeCLIPstyler, a model specifically designed for efficient LDAST execution on resource-constrained devices. As far as we know, this is the only model capable of effectively performing this task on low-power devices. Our experiments revealed that our models could generate stylised images free from undesirable artefacts often found in CLIPstyler outputs, all while operating several orders of magnitude faster. Additionally, our models outperform CLVA on numerous measurable metrics, including a human evaluation derived from a user survey.

The capability of our models, especially EdgeCLIPstyler, provides the foundation for a range of practical applications including real-time video conferencing background enhancements, sophisticated social media filters, and advanced photo editing tools. Users can now execute local image editing directly on their mobile devices, eliminating the traditional dependency on remote server processing, and ensuring enhanced user experience along with stringent data privacy and security standards.

Despite these promising results, it is essential to address the limitations of our model to identify potential areas for improvement and future research. One notable limitation is the reliance on leveraging a pre-existing vision-based style transfer model with an explicit low-dimensional representation of the style. This constraint could be seen as restrictive, as recent research trends in the field have shifted towards models learning high dimensional implicit style representations by employing attention networks.

References

[1] Panos Achlioptas, Maks Ovsjanikov, Kilichbek Haydarov, Mohamed Elhoseiny, and Leonidas Guibas. ArtEmis: Affective language for visual art. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11564–11574, 2021. 1, 6

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. 4

[3] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016. 1

[4] Zhihong Chen, Guiming Hardy Chen, Shizhe Diao, Xiang Wan, and Benyou Wang. On the difference of BERT-style and CLIP-style text encoders. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023. 7

[5] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014. 2, 4

[6] Mirecea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 6

[7] Tsu-Jui Fu, Xin Eric Wang, and William Yang Wang. Language-driven artistic style transfer. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVI*, pages 717–734. Springer, 2022. 1, 3, 6

[8] Rinon Gal, Or Patashnik, Haggai Maron, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. StyleGAN-NADA: CLIP-guided domain adaptation of image generators. *ACM Transactions on Graphics*, 41(4), Jul 2022. 4

[9] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:.1508.06576*, 2015. 1

[10] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. In *British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017*. BMVA Press, 2017. 1, 3, 7

[11] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 1

[12] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision – ECCV 2016*, pages 694–711. Springer International Publishing, 2016. 6

[13] Kaggle. Painter by Numbers, 2016. Accessed: 2 May 2023. 4

[14] Gihyun Kwon and Jong Chul Ye. CLIPstyler: Image style transfer with a single text condition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18041–18050, 2022. 1, 3, 4, 7

[15] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with Markovian generative adversarial networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 702–716. Springer, 2016. 1

[16] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 2230–2236. AAAI Press, 2017. 1

[17] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. AdaAttN: Revisit attention mechanism in arbitrary neural style transfer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6629–6638, 2021. 1, 6

[18] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, volume 30, page 3, 2013. 3

[19] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5880–5888, 2019. 1, 6

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1, 3, 5, 6

[21] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. 4, 7

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[23] Akhil Singh, Vaibhav Jaiswal, Gaurav Joshi, Adith Sanjeeve, Shilpa Gite, and Ketan Kotecha. Neural style transfer: A critical review. *IEEE Access*, 9:131583–131613, 2021. 1

[24] Jan Svoboda, Asha Anoosheh, Christian Osendorfer, and Jonathan Masci. Two-stage peer-regularized feature recombination for arbitrary image style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13816–13825, 2020. 6

[25] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. *arXiv preprint arXiv:1603.03417*, 2016. 1

[26] Zhou 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*, 13(4):600–612, 2004. 6

[27] Chenfei Wu, Lun Huang, Qianxi Zhang, Binyang Li, Lei Ji, Fan Yang, Guillermo Sapiro, and Nan Duan. GODIVA: Generating open-domain videos from natural descriptions. *arXiv preprint arXiv:2104.14806*, 2021. 6