FastCLIPStyler: Towards fast text-based image style transfer using style representation

Ananda Padhmanabhan Suresh*  Sanjana Jain  Pavit Noinongyao  Ankush Ganguly

Sertis Vision Lab†

Abstract
Artistic style transfer is usually performed between two images, a style image and a content image. Recently, a model named CLIPStyler demonstrated that a natural language description of style could replace the necessity of a reference style image. They achieved this by taking advantage of the CLIP model, which can compute the similarity between a text phrase and an image. In this work, we demonstrate how combining CLIPStyler with a pre-trained, purely vision-based style transfer model can significantly reduce the inference time of CLIPStyler. We call this model FastCLIPStyler. We do a qualitative exploration of the stylised images from both models and argue that our model also has merits in terms of the visual aesthetics of the generated images. Finally, we also point out how FastCLIPStyler can be used to further extend this line of research to create a generalised text-to-style model that does not require optimisation at inference time, which both CLIPStyler and FastCLIPStyler do currently.

1 Introduction
The principal objective of style transfer is to recompose a content image with the semantic texture of a style image, usually an artwork. Recent research in this domain has been inspired by the work of Gatys et al. (2015b) 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 (Singh et al., 2021). The algorithm proposed by Gatys et al. (2015b) makes use of a pre-trained VGG-19 network (Simonyan and Zisserman, 2015) to define a content and style loss and jointly optimise them to create stylised images. While the content loss refers to matching deep feature representations of the content and the stylised image using a suitable distance metric, the style loss aims to match the Gram matrices (Gatys et al., 2015b; Li et al., 2017c) of the feature maps of the style and the stylised image. A Gram matrix captures the spatially-averaged correlations across the feature maps within a given layer of a CNN (Gatys et al., 2015a; Ghiiasi et al., 2017; Portilla and Simoncelli, 2000). These feature correlations of multiple layers aid in obtaining a multi-scale representation of an input image, capturing its texture information but not the global arrangement (Gatys et al., 2016). Thus, matching the Gram matrices of the stylised and the style feature maps helps transfer the texture information of the input style image onto the stylised image. This formulation of the style loss has been adopted in several notable style transfer research, which includes style transfer through pixel optimisation for a single content image (Elad and Milanfar, 2017), a real-time arbitrary style transfer for various style images (Huang and Belongie, 2017; Li et al., 2017b; Liu et al., 2021; Yoo et al., 2019), and feed-forward network optimisation for image stylisation (Ghiiasi et al., 2017; Johnson et al., 2016; Ulyanov et al., 2017).

While these approaches are capable of creating visually pleasing stylised images by transferring styles of famous artworks to arbitrary images, they require a reference style image to achieve the desired style transformation of the content image. In order to overcome this limitation, several methods have been developed to manipulate images with a text condition that conveys the desired style without needing a reference style image. The main idea is to use a pre-trained text-image embedding model to translate the semantic information of a text condition to the visual domain. CLIPStyler (Kwon and Ye, 2022), a recent development in the domain of text-driven style transfer, delivers the semantic textures of input text conditions using CLIP (Radford et al., 2021) – a text-image embedding model. In the case of CLIPStyler, the content image is transformed by a lightweight CNN, trained to express the texture information with input text conditions using a novel loss term.

*Email: asure@sertiscorp.com
†597/5 Sukhumvit Road, Bangkok, 10110, Thailand
referred to as patchCLIP loss and a suitable augmentation scheme. Using this technique, Kwon and Ye (2022) obtained realistic style transfer results by simply changing the text conditions without ever requiring any reference style images. In this work, we propose a text-based style transfer framework with the aim to improve upon CLIPStyler’s processing speed at inference time. We incorporate the CLIPStyler framework with a pre-trained style transfer network, such as the GoogleStyleTransfer network (Ghiasi et al., 2017), and a simple feed-forward network. The goal is to utilize the distribution of the style embeddings from the GoogleStyleTransfer network to sample random art styles that match the input text conditions. Once sampled, these embeddings can be used to transform a given content image into the desired stylised image using the pre-trained style transfer model from Ghiasi et al. (2017).

• We introduce the FastCLIPStyler, a fast text-based image style transfer that improves upon CLIPStyler with faster processing speed at inference time. We also qualitatively analyse our performance to other purely image-based style transfer methods such as CST (Svoboda et al., 2020), SANet (Park and Lee, 2019), and AdaAttn (Liu et al., 2021) and show that the performance of our method is at par with these state-of-the-art techniques. The main contributions of our proposed text-based image style transfer are as follows:

• As a failure case, Kwon and Ye (2022) demonstrate that CLIPStyler induces certain undesirable artefacts in some of their generated images which FastCLIPStyler manages to avoid.

We organize the paper into the following sections. Section 2 sheds light on the various style transfer research related to this paper. Section 3 describes in detail our proposed FastCLIPStyler framework and how it is improvement over CLIPStyler. Section 4 demonstrates the results from our model and compares them with other state-of-the-art style transfer models with a particular focus on the results from CLIPStyler. Finally, we share our concluding remarks in section 5.

2 Related works

Arbitrary style transfer seeks to transfer any style into any content image without needing a dedicated model for each style or content, especially the unseen ones. To our interest, in this section, we group different style transfer works based on style description (vision-based or text-based style transfer). We also compare the works based on the inference mechanism (optimisation-based and optimisation-free inference).

2.1 Purely vision-based style transfer

Starting from the renowned method (Gatys et al., 2015b), style transfer has gained attention and multiple works have been developed since. Earlier methods employ optimisation-based inference following Gatys et al. (2015b). Li et al. (2017c) reformulate the problem into distribution alignment and propose a new set of loss functions focusing on aligning feature distribution. Li et al. (2017a) also improve upon Gatys et al. (2015b) work by incorporating a new loss called Laplacian loss. Then comes approaches that use a generalised model based on aligning the statistics of content and style images leveraging the established knowledge that feature statistics can well capture style information (Gatys et al., 2016). AdaIN+(Huang and Belongie, 2017) modifies instance normalization to
style normalize (making a feature map invariant to styles) the content image and align its statistics to a given style image resulting in a network that can transfer arbitrary styles. SANet (Park and Lee, 2019) applies an attention mechanism on top of AdAIN to account for local style information and improves the quality of the synthesized image significantly. AdaAttN (Liu et al., 2021), on the other hand, views attention weights as probability distribution from which statistics are derived to align content feature maps with that of the styles. They also incorporate shallow features in the process in addition to high level features leading to better visual appearance. The aforementioned methods perform statistics alignment directly on feature maps output from a pre-trained object classification network (i.e., VGG-19). However, Ghiasi et al. (2017) train a style prediction network especially for style classification, obtaining a dedicated style space in the process. Style embeddings from this space are then input to the style transfer network, which takes the style embedding and a content image to apply the style to the image. Svo-boda et al. (2020) train an auto-encoder to construct latent space representations for both style and content with the use of metric learning. Style transfer is then performed on the latent space and the decoder transforms the embedding into resulting stylised images. One of the advantages of this approach is that it provides us with an explicit representation of styles. This opens the door for many applications regarding style transfer in which representation of styles are needed such as generative modeling, purely new style synthesis, etc. The proposed method utilises this very style representation to improve text-driven style transfer.

Most of the discussed approaches frame the problem in a similar way which is trying to produce a stylised image given a content image and a reference style image. They have made it relatively easy to apply any style to any target image, given that the style reference images are available. However, it is usually not the case that desired reference style images are available. Our method is advantageous over these methods in that it does not require a reference image.

2.2 Text-driven style transfer

Since a reference style image is not always available at hand and the most natural way to describe artistic styles would be through verbal description, a new way of transferring styles is proposed. ClipStyler (Kwon and Ye, 2022), as shown in Figure 1a employs CLIP (Radford et al., 2021) embedding model to bridge between text and image semantics and applying this to style transfer. It is capable of stylising a content image using only natural language descriptions of styles. However, the calculation of loss involves passing images through VGG and CLIP multiple times leading to considerable additional computing time during inference. Furthermore, a dedicated network is needed to be trained for every query. CLIP loss has also been employed by Liu et al. (2022) for text-driven style transfer. They also further design and train a network in such a way that it does not need online optimisation. However, the text description given to the model is limited to artists’ names rather than open descriptions of a desired style.

2.3 Optimisation-based and optimisation-free inference

Based on their inference mechanism, a subset of style transfer methods work by performing optimisation at inference time. There are methods that directly optimise pixel values of the stylised image as in Gatys et al. (2015b); Li et al. (2017a,c) or those that train a neural network to transform the content image into the style one as in Johnson et al. (2016); Kwon and Ye (2022); Ulyanov et al. (2017). This type of style transfer is more resource demanding due to the need for run-time optimisation and specificity to style and content images. However, the advantage is that it does not need any data for training.

Another branch of style transfer based on inference procedure is optimisation-free inference. This kind of methods formulate the problem as statistics alignment between content’s and style’s features (Ghiasi et al., 2017; Huang and Belongie, 2017; Liu et al., 2021; Park and Lee, 2019). Feature statistics are aligned in feature space and a decoder network is then used to reconstruct them into a stylised image.

As far as we know at the time of writing, there are only two works that aim to do style transfer on the natural language description of a style. Liu et al. (2022) train a generalised model for style transfer and is optimisation-free during inference. However, the text description of styles is limited to artist names. CLIPStyler, on the other hand, is more flexible for text style description, but it uses optimisation-based inference. Our proposal builds on top of CLIPStyler’s architecture. The difference is that instead of online optimising a CNN to produce the stylised image from the content image, we make use of a pre-trained style transfer network for this. And the inference time optimization is done on a simple feed forward network to map from an input text to a style embedding. Although our method still needs optimisation at inference time, doing it this way reduce the computation significantly compared to CLIPStyler.

3 Method

The main idea of this research is to extend the work done by Kwon and Ye (2022) in their CLIPStyler model to take advantage of existing pre-trained, purely vision based style transfer network and enhance the learning procedure in multiple ways. Specifically, we note that in addition to significantly improving our inference time, we are able to remove certain artefacts that appear in some generated images of CLIPStyler. We elaborate on these results in section 4. We are essentially combining two style transfer techniques here: CLIPStyler that can apply a style based on a text description and GoogleStyleTransfer (Ghiasi et al., 2017) that can apply a style based on a reference style image. Since the two models that we use are fundamental to our approach, we give a brief description of both in the following section.
3.1 Models used

3.1.1 Style transfer network

GoogleStyleTransfer (Ghiasi et al., 2017) is the culmination of Gram matrix based style transfer that iteratively improved upon the original style transfer research (Gatys et al., 2015b). It uses a two-stage transfer process, in which a style prediction network converts a given reference style image into a low dimensional style embedding, and a style transfer network applies the embedding on the given content image. These networks are trained jointly using a loss function defined based on the closeness of extracted features from the VGG-19 model. The model was trained on Painter by Numbers³ (PBN) and Describable Textures Dataset⁴ (DTD). PBN is a dataset of 79,433 images of real paintings by different artists and DTD is a dataset of 5640 images containing different textures in the wild. The trained model is capable of doing general style transfer where it can transfer the style of an unseen painting to a given content image. In addition to introducing this model, they also demonstrate that these embeddings have useful properties like the existence of a meaningful interpolation between different styles in the latent space. This model is our choice for the style transfer network because it uses an explicit low dimensional style representation that we are able to take advantage of. We do this by using the style representations of images from the PBN dataset in order to incorporate a prior belief to the system on what a real painting is supposed to look like. A similar notable research in this domain is reported by Jackson et al. (2019) that made use of these embeddings to fit a normal distribution on the style embeddings of the images in the PBN dataset so that they can sample random art styles. They are then able to use the style transfer model to apply these sampled embeddings to a given content image as a data augmentation technique. We acknowledge their work because they provided a PyTorch implementation of the Ghiasi et al. (2017) model that we use in this work. However, this PyTorch implementation was trained only on the PBN dataset, and not the DTD dataset, which reduces some stylisation capabilities of our model.

³https://www.kaggle.com/c/painter-by-numbers
⁴https://www.robots.ox.ac.uk/~vgg/data/dtd/
3.1.2 CLIPStyler

While most of the style transfer approaches extract features from vision networks like VGG to model their transfer process, CLIPStyler mainly uses the CLIP (Radford et al., 2021) model to do style transfer. Unlike other style transfer approaches, they do not require a reference image for style, but a natural language description of the style. The content image is passed through an image-to-image style transfer network that directly converts the content image into a stylised one. This network is optimised at inference time using a loss function defined by the CLIP model (Radford et al., 2021). CLIP simply computes a similarity score between the generated image and the text query from the user, and the style transfer model is optimised to reduce the loss until the generated image starts resembling the textual description. The architecture of CLIPStyler is given in figure 1a. In Kwon and Ye (2022), it was shown that the CLIP model has enough information about textures, paintings, colors and so on, to successfully guide the style transfer network towards excellent results.

The core idea in CLIPStyler is to compute the loss between the generated image and the text query from the user, and the style transfer model is optimised to reduce the loss until the generated image starts resembling the textual description. The architecture of CLIPStyler is given in figure 1a. In Kwon and Ye (2022), it was shown that the CLIP model has enough information about textures, paintings, colors and so on, to successfully guide the style transfer network towards excellent results.

The core idea in CLIPStyler is to compute the loss between the generated image and the text query from the user, and the style transfer model is optimised to reduce the loss until the generated image starts resembling the textual description. The architecture of CLIPStyler is given in figure 1a. In Kwon and Ye (2022), it was shown that the CLIP model has enough information about textures, paintings, colors and so on, to successfully guide the style transfer network towards excellent results.

The core idea in CLIPStyler is to compute the loss between the generated image and the text query from the user, and the style transfer model is optimised to reduce the loss until the generated image starts resembling the textual description. The architecture of CLIPStyler is given in figure 1a. In Kwon and Ye (2022), it was shown that the CLIP model has enough information about textures, paintings, colors and so on, to successfully guide the style transfer network towards excellent results.

3.3 Loss function

This section explores the loss function that we use to train the model. We primarily define $L_{global}$ which measures the closeness of the image generated by the style transfer model with the text input using CLIP. We construct this exactly the same way as recommended by Kwon and Ye (2022). The similarity between the generated image, and a given text string in the CLIP space is given by:

$$L_{global} = D_{CLIP}(f(I_c), t_{sty})$$ (1)

Here $I_c$ is the content image, $t_{sty}$ is the text describing the style, $f$ is the style transfer network that generates the stylised image, and $D_{CLIP}$ is the CLIP distance function that computes the closeness between an image and a text. We also use the following modification to this loss as originally suggested by Gal et al. (2021) to make the learning more stable, similar to how it is done by Kwon and Ye (2022). The directional loss $L_{dir}$ is computed as:

$$\nabla T = E_T(t_{sty}) - E_T(t_{src})$$

$$\nabla I = E_I(f(I_c)) - E_I(I_c)$$

$$L_{dir} = 1 - \frac{\nabla I \cdot \nabla T}{|\nabla T||\nabla I|}$$

Here $E_I$ and $E_T$ are image and text encoders of CLIP, $t_{sty}$ is the input query and $t_{src}$ is simply a constant string set to "A photo" for natural content images.

However, in practice, we found that when using this as the sole loss term, the content sometimes is not preserved, and the generated images are over-stylised. This is not surprising since there is nothing stopping the style prediction network from giving out style embeddings outside the valid input space of the style transfer network. To get better results, we need to regulate the style embedding tensor to remain a valid input to the style transfer network.

We can do this easily by comparing the generated embedding vector with the distribution of embeddings of real paintings that was used to train the style transfer model. Since the model was trained on the PBN dataset, we compute the style embeddings of all these images. We then extract the mean and standard deviation of these embeddings. Then, we compute the mean square error of the generated embedding with respect to this Gaussian distribution. This loss is then added to the $L_{to}$ to ensure that the style embedding is within the valid range.
Figure 3: Comparison of our model with other state-of-the-art purely vision based models in its ability to identify paintings of famous artists and their artworks.

assume a normal distribution over these so that we are able calculate the likelihood of a given embedding vector being sampled from this distribution. This likelihood is incorporated into the loss function of the network so that the text-to-style prediction model is penalised if it generates vectors that lie very far from the distribution of possible real styles, and ensures that it remains a valid input to the style transfer network. Computing this term is cheap during inference as it only involves very few addition and multiplication operations with the mean and precision matrices of a pre-computed normal distribution. We define this distribution loss function as

\[ L_{\text{dis}} = (x - \mu)^T \Sigma^{-1} (x - \mu), \]

(2)

where \( x \) is the embedding tensor, \( \mu \) and \( \Sigma \) are the mean and covariance of the embeddings of the PBN and DTD dataset. We find that using the full equation of the multivariate normal distribution is unnecessary here since we simply have to define a penalty term to the loss function. Putting these together, our overall loss function is formulated as:

\[ L_{\text{total}} = \lambda_{\text{dir}} L_{\text{dir}} + \lambda_{\text{dis}} L_{\text{dis}}, \]

(3)

where \( \lambda_{\text{dir}} \) and \( \lambda_{\text{dis}} \) are coefficients governing the weight each loss term carries.

3.4 Advantages over CLIPStyler

3.4.1 Ability to map from text to style embeddings

One major disadvantage of CLIPStyler is that it needs to run a training loop during inference for each query. In order to alleviate this issue, Kwon and Ye (2022) propose doing pre-training of their style transfer model for a particular query with a small dataset of content images. One can then simply use this model to do a forward pass during inference. However, the trained model does not work for a general query, only on the particular query on which it was trained. A style is hence represented in CLIPStyler by a fully trained CNN model.

Our framework proposes an alternative to this by representing the style as an embedding vector. In our particular case, since we are using the GoogleStyleTransfer model, the style is represented by a 100-D tensor. This style representation has obvious advantages when it comes to storing and transferring a particular style. And while we also need to train a fully-connected feed-forward neural network for each query, this is not required during inference in the presence of a pre-generated embedding vector. Ghiasi et al. (2017) already show the existence of such a meaningful latent space of styles. However, ours is the first work that is able to generate these representations based on a natural language description of the style of interest. This is advantageous because we can use such a mapping to create a generalized text-to-embedding model that can directly predict a style representation based on text description without the need for a training at inference time. Our architecture hence, opens up the possibility of overcoming what is arguably the biggest drawback of CLIPStyler. Note that we do not explore this possibility in this work but simply point how our model can be used for this purpose.

3.4.2 Faster inference

In CLIPStyler, several loss terms are carefully constructed and minimized simultaneously to achieve visually pleasing results as explained in section 3.1.2. Computing all these losses are necessary because the style transfer model, by itself, has no information on what style is or how it is to be applied to a content image. It does not have the advantage of a style being defined by a Gram matrix as
is done in other forms of image-based style transfer research. Hence, these loss functions are the ones that guide the learning process in such a way that the applied style is not only consistent with the given text input, but also visually pleasing to the viewer. However, these are not cheap to compute. The patch loss requires the splitting of the image into multiple patches and querying each patch with CLIP. The content loss requires the content image and the generated style image to be run through the VGG network to compute content features. In our formulation, we can completely avoid these losses except for the global CLIP loss, which simply measures how closely the generated image matches with the text description. This is because the style transfer network has already been trained so that it is able to retain the content and apply style in a visually pleasing manner. It is observed that removing these improves the speed of computation as detailed in 4.3.1.

### 3.4.3 Inclusion of priors of real paintings

The style transfer network used in CLIPStyler is a model that is trained from scratch at inference based on the loss function that CLIP provides. Its awareness of painting styles comes solely from the CLIP model. However, as our style transfer model was pre-trained on a dataset of real paintings, it starts off with a bias towards generating images that resemble the ones that it has encountered during its training. That is to say, our model has a natural prior over a distribution of real paintings. The role of CLIP then, is to guide the search in that distribution for a sample that most appropriately describes the given input prompt. This can have a dramatic impact on the aesthetics of certain generated images, both positively and negatively. We present the results of such queries and their generated examples in section 4.4.

### 4 Results

#### 4.1 Experiment settings

In order to be consistent with the results that were shared in Kwon and Ye (2022), we set the image resolution of the content image to be $512 \times 512$. Our text-to-style prediction network is a fully connected feed-forward network that takes a $512$ dimensional text embedding as input and outputs a $100$ dimensional style representation. It has two hidden layers of $256$ and $128$ nodes. The activation function in between the layers is the leaky relu with a negative slope of $0.2$. At the final node, the tanh activation function is used to normalise the style representations between $-1$ and $1$.

While training, we set $\lambda_{dis}$ to $5 \times 10^2$ and $\lambda_{dis}$ to $10^{-3}$. The Adam optimiser is used with a learning rate of $5 \times 10^{-3}$ with the number of epochs set to $150$. These experiments were performed on an Intel Core i5 desktop with a single 8GB GeForce RTX 2070 GPU. Instead of directly using the CLIP embedding of the input string, Radford et al. (2021) suggest creating a set of different texts with the same meaning and averaging out their representations for increased stability of the training processes. Same as Kwon and Ye (2022), we use this technique here as well.

#### 4.2 Qualitative evaluation

In figure 2, we demonstrate the basic capabilities of our model. It has the awareness of a wide range of queries and is able to apply color, different textures, and even compound statements combining these. When given statements like "blue lines" and "black pencil", the model is able to create a style that gives importance to both entities mentioned in the text. It also has awareness of various art-genres such as "mosaic" or "cubism". Additionally, the content is preserved well across stylisation without an explicit requirement for a content loss like CLIPStyler. Figure 3 shows the awareness that our model has of various artists and their important artworks. We show this in comparison to other purely vision based state-of-the-art style transfer techniques from recent years. The style transfer models that we used for this comparison are CST (Svoboda et al., 2020), SANet (Park and Lee, 2019), and AdaAttn (Liu et al., 2021). We notice that even though our technique does not have the advantage of having the reference style image when doing style transfer, the general performance is comparable to the state-of-the-art.

Figure 4 illustrates a direct comparison between our model and CLIPStyler since our work is a direct extension of their model. We have listed some of the queries that Kwon and Ye (2022) chose to highlight in their paper along with some additional ones. We can observe that the quality of the stylised images from our model is comparable to that of CLIPStyler. The rest of the result section is dedicated to doing a deeper analysis of the generated images from both these models.

| Operation                                         | CLIPStyler | FastCLIPStyler |
|---------------------------------------------------|------------|---------------|
| Backward propagation                              | 124.32     | 56            |
| Cropping and augmenting images into patches       | 109.9      | -             |
| Extracting features from VGG                      | 19.3       | -             |
| Extracting features from CLIP                     | 13.9       | 7.6           |
| Forward propagation through style transfer model   | 12.1       | 4.3           |
| Forward propagation through style prediction model| -          | 0.23          |

Table 1: Time in milliseconds spend by each model per optimisation iteration performing their most expensive operations.
4.3 Comparisons with CLIPStyler

4.3.1 Inference time
A significant advantage that our model has over CLIPStyler is the reduced inference time owing to a reduced model complexity and a simplified loss function. In Table 1, we demonstrate a breakdown of the time spent by each model in their most expensive operations per iteration. As shown, we are able to avoid a lot of the performance bottlenecks that CLIPStyler has. Since the model that we are optimising is a simple fully-connected feed-forward neural network, the backpropagation itself takes relatively less time. The operations for cropping and augmenting the generated images into patches is also a significant bottleneck in performance that we are able to avoid. In addition, we do not have to spend time passing the images through VGG to compute the content features. The time spent by the model querying CLIP is lesser since we only have to query it once per iteration for the final generated image and not all the patches of the image. The main component that we have that CLIPStyler does not is the style prediction model; which, owing to its simple architecture, also has a very fast forward propagation speed.

We find that in general, while CLIPStyler is able to converge in around a minute, FastCLIPStyler does not take more than 10 seconds, a speed up of around 6 times.

4.3.2 Introduction of undesirable artefacts in generated images
Since the style transfer network in CLIPStyler is trained from scratch during inference, it has a lot of flexibility in being able to convert the content image in a variety of different and novel ways. While this flexibility enables it to create exceptional results, there are some instances where this introduces a large number of undesirable artefacts throughout the image. We demonstrate this in Figure 5. Artefacts that looks like eyes, nose, and ears appear all throughout the generated images. It appears that what should be the content of the image is somehow "leaking" into what should be the style. The precise reason why these show up in some images is unclear, but we suspect that the flexibility of the style transfer network combined with the patching of the generated image into multiple smaller images is at the root of this. As shown in 5 this is something that can be completely avoided with our framework. Since the style transfer model is pre-trained to apply style in a relatively predictable manner, it is nearly incapable of altering the content image to the degree with which the style transfer model in CLIPStyler is able to. In addition, the splitting of the image into patches is also something that is not done in our model. Both these facts combined, we observe that the additional artefacts that might make some of the generated images of CLIPStyler completely unusable, does not appear in our stylised image.

4.4 Effect of having a prior of real images
As explained in section 3.4.3, our model has a natural prior over a dataset of real paintings. This can either work for the advantage or disadvantage for the model. To demonstrate its effect, we present some examples of queries in
Figure 5: We observe that in CLIPStyler, there are instances where some facial features like eyes, ears etc are applied throughout the content image in random patches.

Figure 6. As can be seen from the figure, there are cases where directly training the style transfer network allows CLIPStyler to generate really impressive styles. For example, when given the prompt "lightning", CLIPStyler is able to generate sharp edges defining lightning. Since style, as understood by GoogleStyleTransfer model, refers more towards the broader brushstrokes and color-schemes, our model is unable to consistently produce these sharp edges required. Nonetheless, it should be noted that our model is also getting the general aesthetics of such queries correctly, even though the results are visually less impressive.

The figure also shows instances where having the flexibility of being unbiased towards how real humans paint puts CLIPStyler at a disadvantage. When given the prompt "style of Mona Lisa", CLIPStyler wrongly interprets that to meaning trying to put the face of Mona Lisa on the given content image. This is likely not how a human would interpret the query. Similarly, when given the prompt, "Style of Thawan Duchanee", an important Thai painter, CLIPStyler seems to replace the original text at the front of the ship with text resembling the spelling of “Thawan". This is likely because CLIP has not been trained on as many works of Thawan Duchanee as Van Gogh’s, for example. In cases like this, instead of returning a failure case with a high loss as our model does, CLIPStyler tries to find creative ways to drive down the loss, as discussed in the examples above.

5 Conclusion
In this paper, we introduced the FastCLIPStyler, an improvement over the CLIPStyler framework in inference speed. We incorporated the CLIPStyler framework with a pre-trained, purely vision-based style transfer model and a simple feed-forward network optimized to a particular query at inference time. Thus, enabling a significant reduction in the processing speed during inference. Our style transfer model is also able to apply a variety of styles and textures based on their natural language description in a visually pleasing manner. Additionally, our experiments demonstrated that the FastCLIPStyler could generate stylised images without undesirable artefacts like in some instances of the generated images from CLIPStyler. Furthermore, a qualitative analysis showed the performance of our framework is comparable to other state-of-the-art style transfer techniques in producing visually pleasing stylised images.

6 Acknowledgements
We are grateful to our colleagues Aubin Samacoits and Ukrit Watchareeruetai for their constructive input and feed-
Figure 6: Effect of the style transfer model being exposed to a prior distribution of artworks. The generated artwork in CLIPStyler can stop resembling real artwork in certain situations. However, our model does not have this problem.

back during the writing of this paper. We also thank Sertis Vision Lab for eagerly supporting research endeavours in Machine Learning.

References

M. Elad and P. Milanfar. Style transfer via texture synthesis. *IEEE Transactions on Image Processing*, 26(5):2338–2351, 2017. doi: 10.1109/TIP.2017.2678168.

R. Gal, O. Patashnik, H. Maron, G. Chechik, and D. Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021.

L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015a.

L. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015b.

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)*, pages 2414–2423, 2016. doi: 10.1109/CVPR.2016.265.

G. Ghiasi, H. Lee, M. Kudlur, V. Dumoulin, and J. 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.

X. Huang and S. 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.

P. T. Jackson, A. A. Abarghouei, S. Bonner, T. P. Breckon, and B. Obara. Style augmentation: Data augmentation via style randomization. In *CVPR Workshops*, volume 6, pages 10–11, 2019.

J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision – ECCV 2016*, pages 694–711. Springer
G. Kwon and J. C. 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. doi: 10.1109/CVPR52688.2022.01753.

S. Li, X. Xu, L. Nie, and T.-S. Chua. Laplacian-steered neural style transfer. In Proceedings of the 25th ACM International Conference on Multimedia, pages 1716–1724, 2017a.

Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Universal style transfer via feature transforms. arXiv, 2017b. doi: 10.48550/arXiv.1705.08086.

Y. Li, N. Wang, J. Liu, and X. Hou. Demystifying neural style transfer. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI’17, page 2230–2236. AAAI Press, 2017c. ISBN 9780999241103.

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. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 6629–6638, 2021. doi: 10.1109/ICCV48922.2021.00658.

Z.-S. Liu, L.-W. Wang, W.-C. Siu, and V. Kalogeriton. Name your style: An arbitrary artist-aware image style transfer. arXiv preprint arXiv:2202.13562, 2022.

D. Y. Park and K. H. 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.

J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. International Journal of Computer Vision, 40(1):49–71, 2000.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning, pages 8748–8763. PMLR, 2021.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations (ICLR 2015), pages 1–14, 2015.

A. Singh, V. Jaiswal, G. Joshi, A. Sanjeeve, S. Gite, and K. Kotecha. Neural style transfer: A critical review. IEEE Access, 9:131583–131613, 2021. doi: 10.1109/ACCESS.2021.3112996.

J. Svoboda, A. Anoosheh, C. Osendorfer, and J. 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.

D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Improved Texture Networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, pages 4105–4113. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.437.

J. Yoo, Y. Uh, S. Chun, B. Kang, and J.-W. Ha. Photorealistic style transfer via wavelet transforms. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 9035–9044, 2019. doi: 10.1109/ICCV.2019.00913.