Finding the Best Architecture for Fully-Automated Image Colorization

1st Gabriel Nkole
School of Computer Science & Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

2nd Richard Klein (Supervisor)
School of Computer Science & Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

***Abstract*—A variety of architectures can be used for deep learning colorization models and currently there is no concrete review that determines which architecture is the best and why. Therefore, we will undergo an analysis of the different architectures and find the one best suited for colorization after considering a range of metrics. We hypothesize that the diffusion model is the best architecture under all circumstances except speed.**

I. Introduction

Photography was invented in the early 19th century, but the technology was very limited and could only produce black-and-white (monochrome) photographs. If it was necessary for a photograph to be in color, somebody needed to color it manually; this required expertise and was a time-consuming process [1].

Although color photography became available to the public in the 1930s (theoretically eliminating the need for monochrome photography), it was not widely adopted until the 1960s and thus there was still a need for images to be colorized [2].

With the advent of computer technology in the mid-20th century, automated processes for colorization were introduced. However, colorization has been difficult to perform with computers because it is an ill-posed problem. The only available information about a pixel is its intensity and this value needs to be used to predict all 3 RGB values (or the 2 other values in CIELAB space).

Many models have been proposed with varying success, and for this project the deep learning models will be analyzed. Within the relatively few years models of this type have existed (the first was introduced by Cheng et al. [3]), they have shown great promise in solving this problem. Since there are several suitable neural network architectures for such models, it is necessary to compare them to find the one best-suited for colorization. Diffusion models have been performing well for many image tasks (including colorization) and we aim to prove that they are the overall best architecture for colorization.

With color photography being mainstream, the general public's need for colorization has mostly vanished. Today, it is commonly used to restore historic black-and-white photographs and videos, enhance medical images that inherently lack color or (simply) make grayscale images more appealing.

II. Background & Related Works

A. Introduction

There are a variety of architectures to select from when designing a deep learning colorization model. However, the 3 crucial ones providing the best results are Generative Adversarial Networks, Transformers & Generative Diffusion Models. Therefore, we will review, implement and evaluate each architecture in order to compare them against one another.

B. Background

1) Generative Adversarial Networks: GAN was first introduced in the research paper "Generative Adversarial Nets" by Goodfellow et al. [4]. A GAN contains two sub-neural networks, a generator and a discriminator. The generator creates fake data, and the discriminator differentiates the fake data from the real data. The generator's purpose is to deceive the discriminator, and these networks are both trained competitively.

The competitive nature of training between the generator and discriminator has resulted in GANs that can create impressive results. Consequently, GANs became the state-of-the-art model for many fields at the time, one of the most prominent ones being Vision. A common use case is the synthesizing of new images from noise (such as human faces, animals, etc.), and it has become an important model for image colorization by deep learning. GAN-based colorization models can produce realistic results able to fool human observers.

Although GANs have shown promise, the main challenge they face is their instability during training. Since the model is inherently adversarial, issues can arise where the model fails to converge or the generator cycles through a small set of outputs (mode collapse); however, there are techniques available to remedy these issues [5].

2) Transformers: In the 2017 research paper "Attention is All You Need" [6], a novel neural network architecture for Natural Language Processing (NLP) named the Transformer was proposed. As the name suggests, the model is based solely on the mechanism of attention, and the neural networks employed are simple feedforward networks. Attention was not a new concept at the time, Long-Short Term Memory networks (LSTMs) already made use of attention by storing

the previous state(s) of the network and having it influence the new state, however, Transformers rely on self-attention to determine long-term dependencies between tokens better than LSTMs. Moreover, the focus on self-attention has allowed for more parallelism and thus better performance [6].

It was evident that Transformers outperformed previous models in NLP tasks with far less training time, making it the new state-of-the-art in that field. Although purposefully built for NLP tasks, Transformers have found use in the field of Vision on problems such as image classification, object detection, super-resolution and (more importantly) image colorization [7].

3) Generative Diffusion Models: The Diffusion Model was first introduced in "Deep Unsupervised Learning using Nonequilibrium Thermodynamics" [8]. Historically, a difficulty for probabilistic models was how to design a model that can be evaluated and easily fit to the data while still being able to fit structure in arbitrary data. Achieving both was difficult as there was a trade-off between the two designs.

This paper introduces a method that uses a Markov Chain to gradually convert a simple distribution (e.g. Gaussian) to a target distribution by way of diffusion. The probabilistic model is explicitly defined at the end of the chain and because each step has an evaluable probability, this translates to the entire chain having this property [8].

The model is defined by two processes: an iterative forward diffusion process and a reverse diffusion process. In the forward process, a small amount of noise is iteratively added to an element of the target distribution until T diffusion steps where the example fully becomes an element of the noise distribution. The reverse process then attempts to iteratively transform this noise back into the desired output by way of a denoising model. For this to be possible it is necessary to be able to find the element's state at the previous timestep by only using its state at the current timestep, however, this transition function is initially unknown as it requires the entire dataset. Therefore, a model of the probabilities that can facilitate this is learned [9].

Although the process of turning noise into the desired output is longer than in other generative models, learning the target distribution in small steps instead of directly (as other generative models do) allows the model to better learn the target distribution.

This can contribute to better results in any of the tasks that diffusion models are used for (such as image colorization).

Although this type of model has existed for several years, its potential to produce high quality results in Vision tasks has only been realized in recent years.

C. Related Works

1) Unsupervised Diverse Colorization via Generative Adversarial Networks (GAN): This paper by Cao et al. [10] is one of the first (and most prominent) to propose a cGAN-based colorization model. A traditional GAN is insufficient for colorization because it is designed to only receive noise as input (and it is necessary to input grayscale image so that a color image can be produced). Seeing that there may be a need to input some pre-existing data into the network, Goodfellow et al. [4] also proposed the conditional GAN (cGAN) architecture, which accepts pre-existing data in addition to noise to generate output. This variant is what is used for all GAN colorization models.

While other image generators consist of convolutional and de-convolutional components, this model only contains convolutional layers with stride set to 1 to prevent the data shape from downsizing. This allows the layer shape to remain consistent, providing the possibility to feed the conditional information to the generator at every layer (multi-layer conditional). Training is still possible because the model learns its own up-sampling/down-sampling method, and the network architecture becomes simpler. This change in the generator's architecture is the main difference between this model and the others at the time. Otherwise, other components of the architecture remain mostly the same [10].

Comparisons were made on RGB vs YUV, single-layer noise vs multi-layer noise and single-layer conditional vs multi-layer conditional. The results showed that the best network configuration was YUV combined with multi-layer noise and conditional. The multi-layer noise especially contributed to more diverse colorizations and the multi-layer conditional contributed to stability [10].

Evaluation for this model was done via Human Study with 80 participants, and 37.5% of the generated images were considered more convincible than the true images.

2) Image-to-Image Translation with Conditional Adversarial Networks (GAN): This paper proposed a unified model (Pix2Pix) for performing image-to-image translation tasks. Although, at the time, GANs had already been trained for a variety of tasks, the architectures and losses differed heavily from model to model. Therefore, the main objective for this paper was to create a model that can perform well on any of these tasks without needing to be tuned for each task. To this end, a U-Net is used for the generator and a convolutional PatchGAN classifier is used for the discriminator [11].

After being trained for colorization on the ImageNet dataset, the generated images managed to fool participants on 22.5% of trials.

3) Colorization Transformer (Transformer): Although the transformer architecture has existed since 2017 and has been used for many Vision tasks since then, the first ever image colorization transformer model (ColTran) was proposed in 2021.

A core idea of this model is the use of a concept called axial attention, which was adopted from the Axial Transformer [12]. Instead of directly computing global self-attention between all tokens of the input like the standard Transformer, local self-attention is computed separately on each axis of the input and hierarchically combined to compute the desired global self-attention. Only working on a subset of the input at a time

in this manner decreases computational cost while preserving the relationships between tokens. Another concept leveraged is conditioning. Aside from the grayscale image, a given pixel of the image is also conditioned on all previous pixels (as per raster order) and this serves to increase the performance of the model. To make conditioning possible, the core Transformer components (Self-attention, MLP, Layer Normalization) need to be converted to their conditional counterparts [13].

ColTran consists of 3 core components: ColTran Core, the Color Upsampler and the Spatial Upsampler. The general idea behind the model is to first produce a coarse coloring for a downscaled version of the grayscale image and then upsample the color and spatial of that image to produce a highly detailed colorization of the original image with the correct dimensions [13].

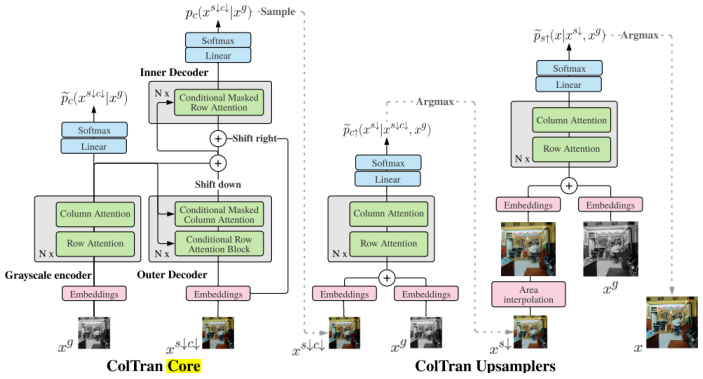

Fig. 1. Coltran Structure

The core itself consists of the grayscale encoder, outer decoder, and inner decoder sub-components. The grayscale encoder learns new representations for all the pixels in the input. It is ran once per channel and information for each channel is encoded independently. The outer decoder then computes a state for all previous rows of a given pixel and the inner decoder computes a state for all previous pixels in the current row. The output of the core is a downscaled colorized 3-bit RGB image. The color upsampler then receives said image and the grayscale image and outputs a new 8-bit representation of the downscaled image. Finally, this result is first stretched to the original shape and is input alongside the grayscale image to the spatial upsampler to produce the high-quality colorized image [13].

Coltran achieved an FID score of 19.370 on the ImageNet 5K test images.

4) Palette Image-to-Image Diffusion Models (Diffusion): Palette is a unified framework for image-to-image translation, and in the original paper, it was tested on four challenging imaging tasks namely: inpainting, uncropping, JPEG restoration and (most importantly) colorization [14]. The framework is based on conditional diffusion models and no task-specific tuning or completely novel techniques were employed.

When trained for image colorization, Palette achieved an FID score of 15.78 on the ImageNet 5K test images. This is the lowest of any other model (according to Papers With Code [15], making it the current state-of-the-art model.

5) BBDM: Image-to-image Translation with Brownian Bridge Diffusion Models (Diffusion): Generally, conditional diffusion models operate by conditioning the U-Net on the input image at every timestep in order t to produce the slightly less noisy image at timestep t-1. The objective then becomes minimizing the error between the true and estimated Gaussian distributions (ϵ and ϵ_0 respectively) at each timestep, however, since ϵ_0 depends on y, $p(x_t|y)$ needs to be accounted for in the objective function. Since many conditional models omit this, they are technically optimising the incorrect objective function. Although they still work practically, they are not guaranteed to converge. Moreover, this omission can lead to a conditional diffusion model being unable to generalize well, making it better suited for tasks where the input domain is very similar to the target domain (e.g. in-painting). This paper proposes a new method called the Brownian Bridge Diffusion Model for general image-to-image translation tasks in order to remedy this problem [16].

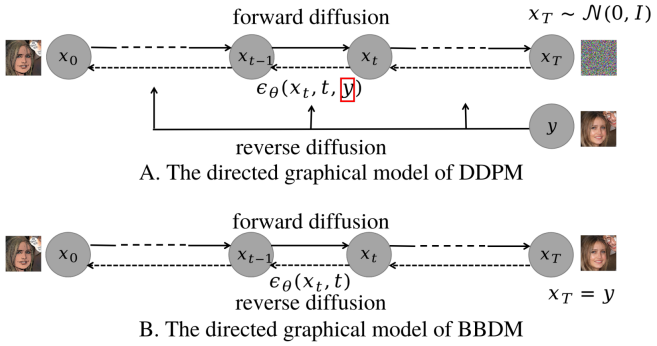

Fig. 2. Conditional vs Brownian Bridge Diffusion

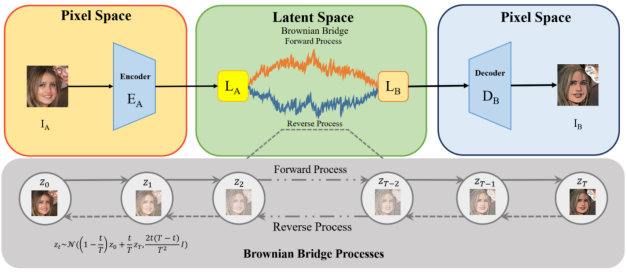

Fig. 3. Latent Brownian Bridge Diffusion Model

This method entails having the image at the end of the forward process be the input image, and to have the model directly learn a mapping from the input domain to the output domain. The probability distribution at each timestep is now calculated using both the input and target images (this is where the term "bridge" is derived). The forward and reverse process equations are modified to accommodate this change. To speed up training and inference, this experiment was conducted in

Latent Space with the aid of a pre-trained VQGAN used in the Latent Diffusion Model (LDM) [17].

Colorization was not thoroughly evaluated and only a few example images were provided, therefore it was uncertain how well this model would perform at this task.

III. Setup

All training is conducted on PCs with an Intel Core i9-10940X CPU (14 cores), NVIDIA RTX3090 GPU (24GB), and 128GB of system RAM.

The GAN model we implemented was Pix2Pix from Pix2Pix & CycleGAN [11]. It was necessary to change the image sizes in the configuration files to 256 and the resulting generator and discriminator contained 54 410 000 parameters and 2 766 000 parameters respectively.

The Transformer model we implemented was ColTran [13]. For the Colorizer we set the batch size to 24 and the number of encoder, inner decoder & outer decoder layers to 1; the resulting network contained 20 566 528 parameters (of which 20 560 384 are trainable). The Upsamplers were unedited and thus the Color Upsampler had 4 017 920 parameters & the Spatial Upsampler had 4 598 528 parameters.

The Diffusion model we implemented was the Brownian Bridge Diffusion Model [16]. We adjusted the image size of the U-Net to 256 and downloaded a VQGAN with an downsampling factor of 4 from LDM [17] to conduct our experiment in Latent Space. The resulting network had 292 410 000 parameters (of which 237 090 000 were trainable). The batch size for this model was initially set to 8 and given that the authors make use of the same GPU as us (an NVIDIA RTX3090), we did not modify it.

The data used for both training and testing are the 100 000 images that make up the test set of ImageNet [18]. We preprocessed them by resizing them to 256x256 and converting them to RGB. Images 1-90000 are set aside for training, 90 001-95 000 are for validation and 95 001-100 000 are for testing.

To compute the FID, PSNR, SSIM & IS metrics on our data, we make use of the ”metrics” package within the PyTorch-Ignite library.

IV. Experiments

We conducted the experiment in two phases. In the first phase we created 4 variations of each model based on different hyper-parameters, trained them for 1 day and evaluated them on the 5000 images validation set. The best performing variation of each model (the one with the lowest FID score) was then chosen to represent that model in the second phase. The training set for this phase consisted of the first 50 000 images out of the entire 90 000 images training set.

In the second phase we trained each model on 90 000 images for 5 days, and on each day we evaluated each model's performance using the 5000 images test set.

A. Phase 1

TABLE I
PIX2PIX HYPER-PARAMETER TUNING

	default	batch size = 32	beta1 = 0.7	lr = 5e-4
FID:	16.33	27.53	17.87	18.30

default: batch size = 4, beta1 = 0.5 & lr = 2e-4

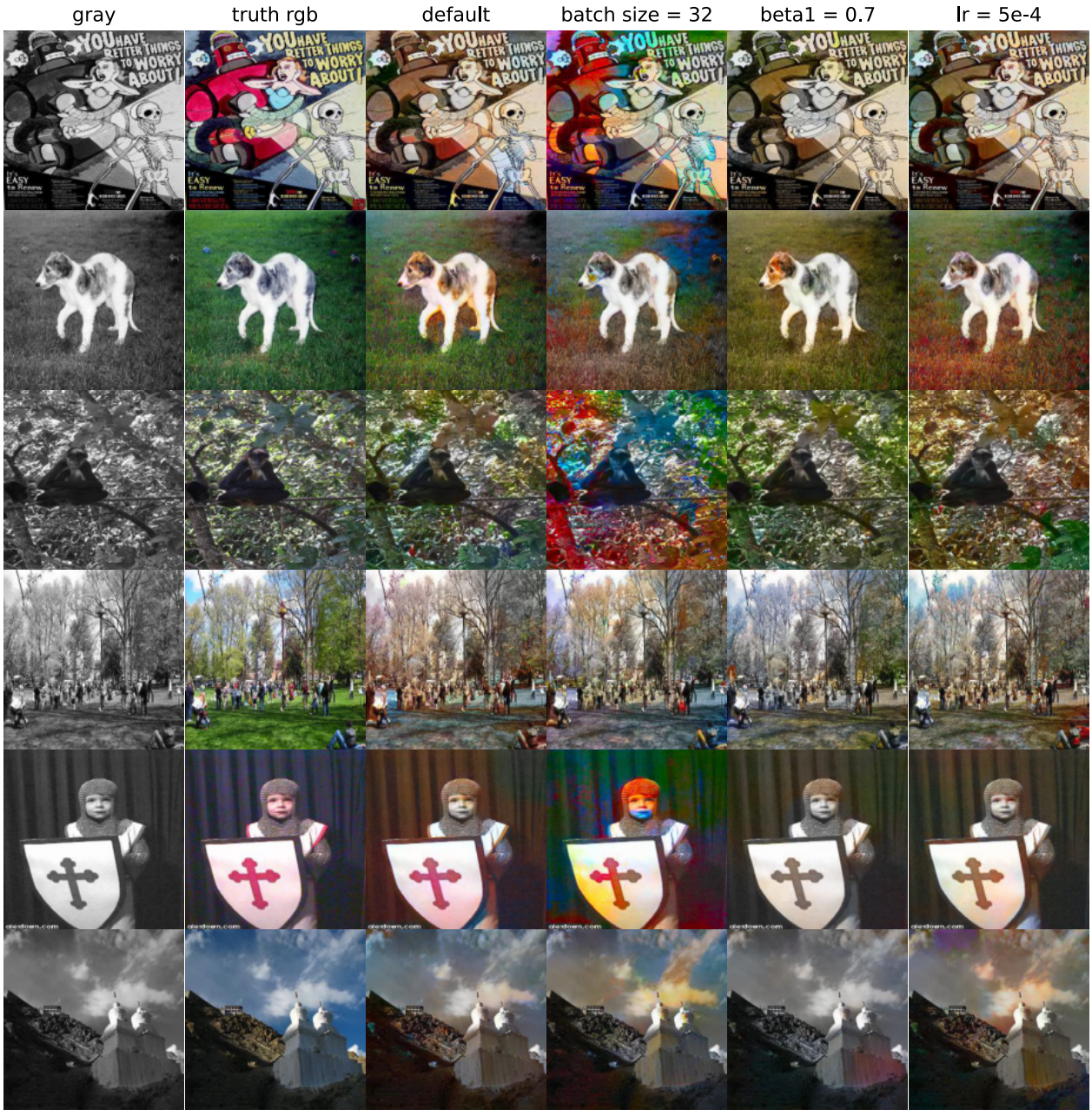

Fig. 4. Pix2Pix colorization on validation set

TABLE II
COLTRAN HYPER-PARAMETER TUNING

	default	loss factor = 0.9	lr = 5e-4	optimizer type = 'adam'
FID:	26.00	21.11	22.93	21.22

default: loss factor = 0.99, lr = 3e-4 & optimizer type = 'rmsprop'

Fig. 5. ColTran colorization on validation set

TABLE III
BBDM HYPER-PARAMETER TUNING

	default	beta1 = 0.7	loss type = 'l2'	lr = 5e-4
FID:	46.76	46.28	50.56	48.11

default: beta1 = 0.9, loss type = 'l1' & lr = 1e-4

Fig. 6. BBDM colorization on validation set

TABLE IV
AVERAGE INFERENCE TIMES FOR 5K IMAGES [HOURS(H):MINUTES(M)]

Pix2Pix	**BBDM**	**ColTran**		
		Colorizer	***Color Upsampler***	***Spatial Upsampler***
3M	2H:40M	6H:50M	4M	42M

For Pix2Pix, each variant was trained for 130 epochs. The default parameters produced the lowest FID score, and the variations where beta1 and the learning rate were modified produced similar FID scores. Looking at Fig. 4, the colorized images were also of similar quality. However, increasing the batch size from 4 to 32 significantly increased the FID score, and produced inconsistent colorizations.

For ColTran, each variant was trained for approximately 100 000 training steps, and we trained a Color Upsampler and a Spatial Upsampler separately for 800000 (43 hours) and 200000 (36 hours) training steps respectively. The results were generally consistent among all variations. Although the Color Upsampler was trained for a long duration, it produced poor results, therefore the pretrained Color Upsampler was used. The Spatial Upsampler, however, produced excellent results as seen in Fig. 5.

For BBDM, each variant was trained for 18 epochs. The FID scores for all versions of BBDM were consistently high at approximately 45-50. When inspecting Fig. 6, one can see that the colorized images are noticeably distorted from the original, especially regarding the finer details. E.g., the text is unreadable in the first image and, in the fourth image, although trees can be seen, it is impossible to distinguish that there is a crowd of people in the image. However, with simpler images such as the one with the dog, the results are of surprisingly high quality, with consistent colors and a lack of obvious color bleeding.

Table IV shows that the ranking in terms of inference speed is Pix2Pix, BBDM, ColTran in that order, meaning that the Diffusion Model was not the slowest as we expected. We do note that it does inference relatively quickly because we conduct the experiment in Latent Space. When attempting to conduct the experiment in Pixel Space, the projected time for the inference of 5000 images with our current 200 sample steps would be 30+ hours; and the time for an epoch would be 20+ hours.

B. Phase 2

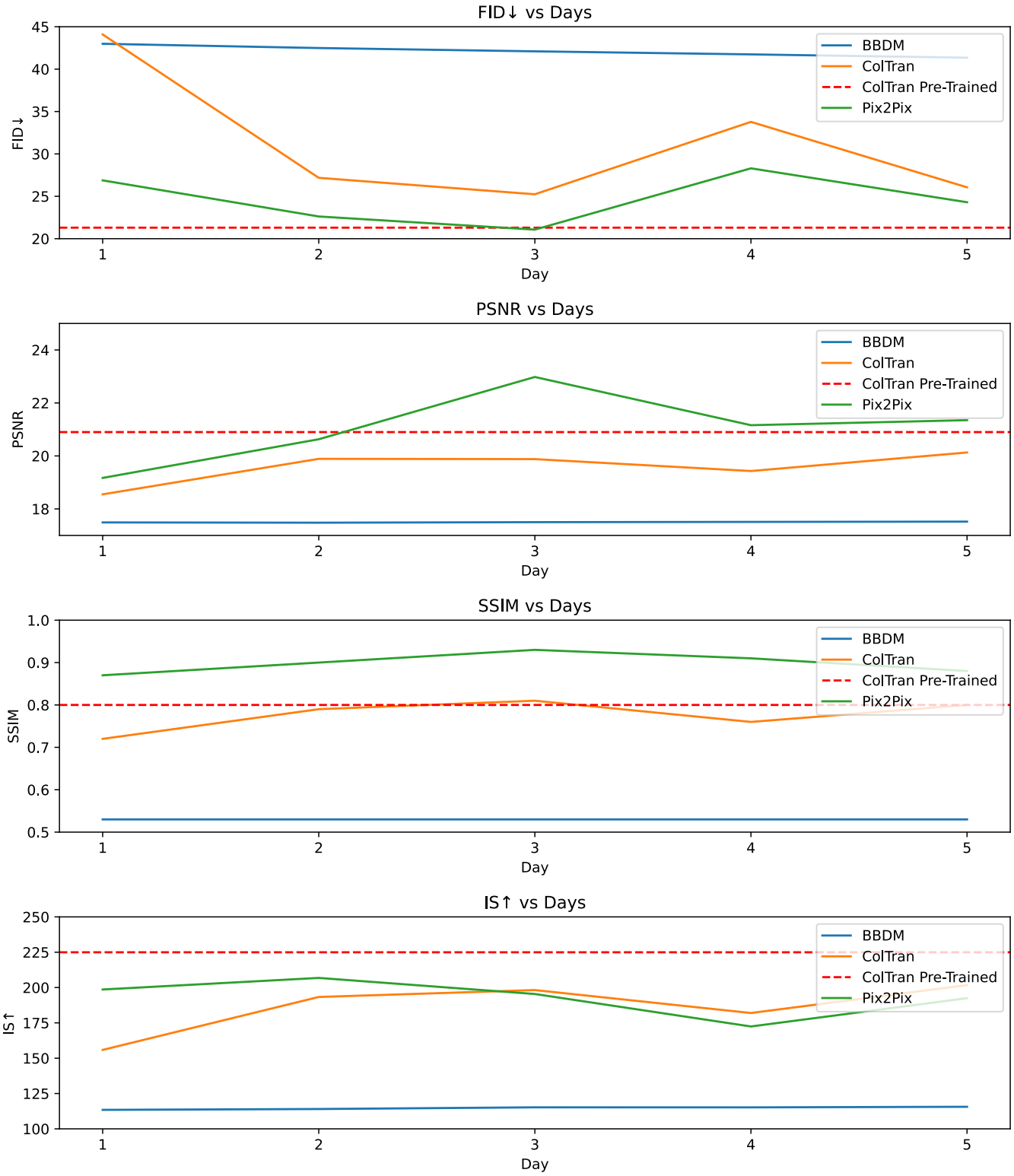

Fig. 7. Plots for evaluation metrics on all models over the number of days

Fig. 8. Pix2Pix colorization on test set

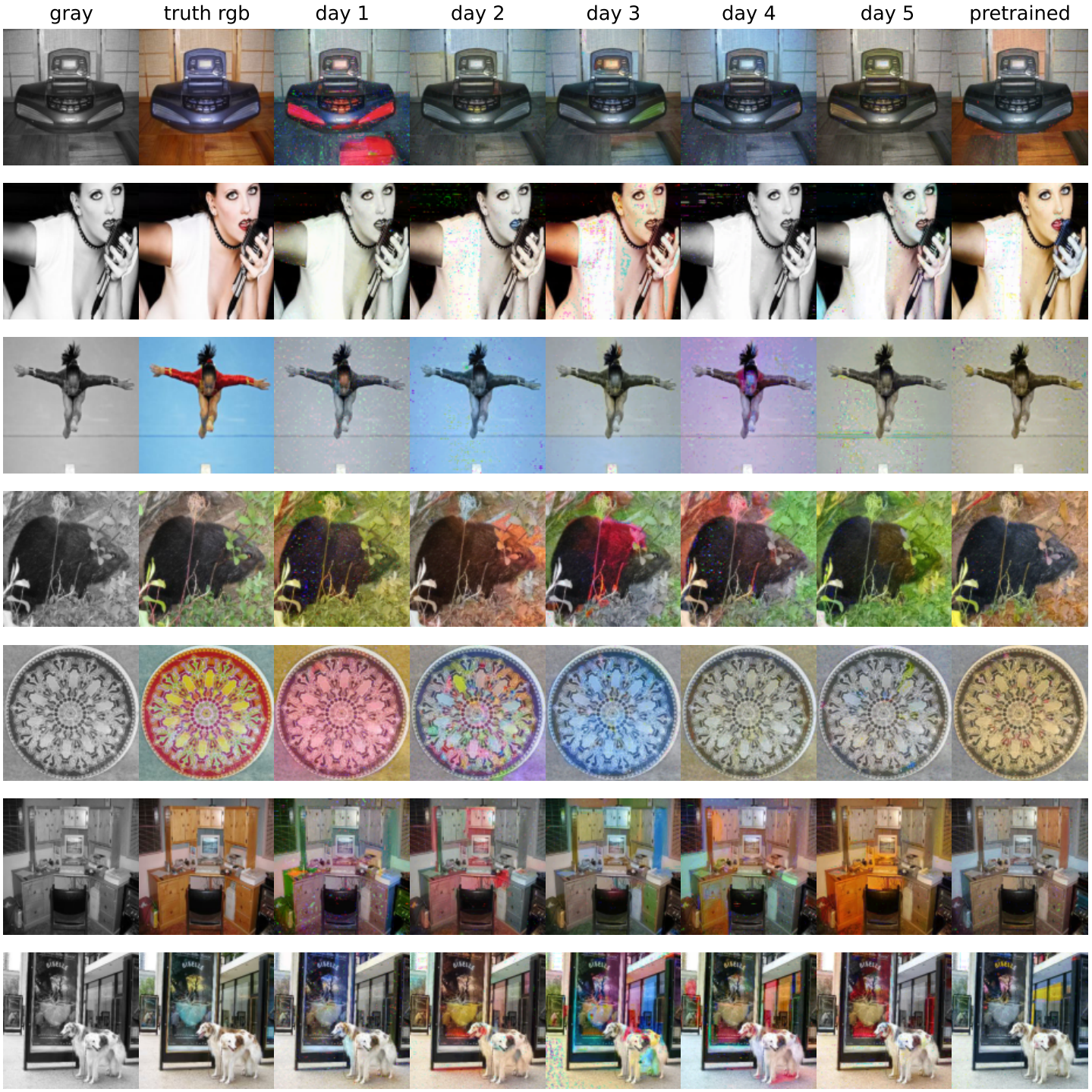

Fig. 9. ColTran colorization on test set

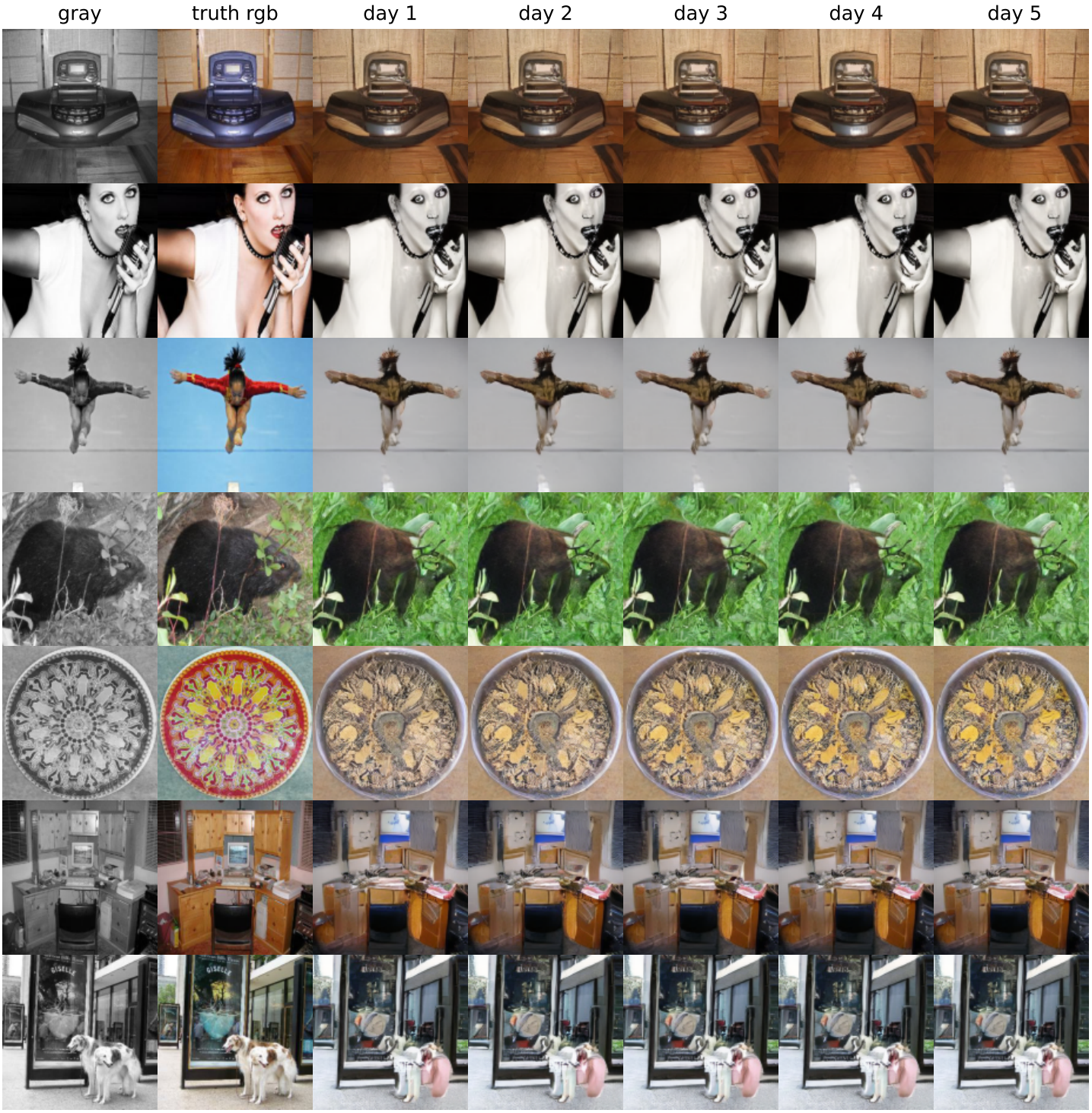

Fig. 10. BBDM colorization on test set

To have a suitable baseline for understanding performance, we evaluate a pre-trained ColTran model on the test set and compare it to our 3 models. These are the dashed lines seen in Fig. 7.

For Pix2Pix, the number of epochs trained on in a day lowered to 80. When reviewing Fig. 8, the colorizations produced across all days of training were mostly grayscale images with random splotches of colour. When investigating the training logs, the loss for the discriminator converged to 0 during day 2 and remained at 0 for the rest of the training. This meant that we encountered a GAN failure mode and this led to consistently bad results. Interestingly, on most metrics Pix2Pix scored relatively close to the pre-trained model. This serves to highlight the necessity to both analyze the metrics and visually inspect the the colorizations in order to determine the quality of our models.

For ColTran, the number of training steps in a day remained the same as the training steps are independent of the size of the dataset. The metrics in Fig. 7 showed that ColTran performed the poorest on day 1, but improved greatly by day 2. This does not accurately reflect on the images for day 2 we have chosen as we see on Fig. 9, but looking at images throughout the days we can see that some of them are close to the quality of the pre-trained model: such as the 2nd image on day 3, and the 6th and 7th images on day 5. This version of ColTran was able to get fairly close to the pretrained version in terms of quality. We retrieved our previously trained Color & Spatial Upsamplers and further trained them on the new training set to 2 000 000 & 500 000 training steps respectively. However, even with this extra training, the Color Upsampler still performed poorly, therefore, we conclude that it is impossible for it to train well with our current number of images and we again relied on the pre-trained Color Upsampler.

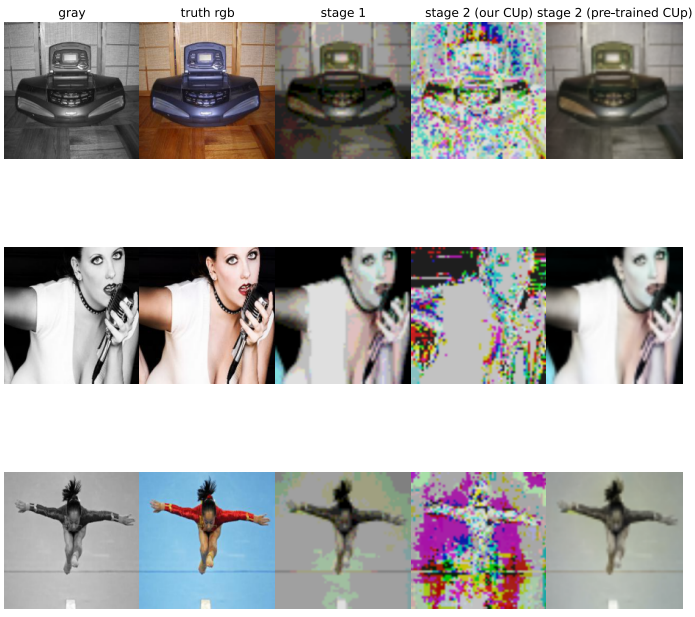

Fig. 11. Day 5 Color Upsampler (CUp) Results

For BBDM, the number of epochs trained on in a day lowered to 14. BBDM's performance remained consistent throughout. The scores for the metrics barely changed and the images are almost interchangeable from day 1 through to day 5. However, the model did not necessarily converge and, in particular, when we inspected the FID, it decreased by roughly 0.3-0.5 each day. We had hoped that training for a longer period of time would remedy the distortion but that was not the case. We hypothesise that this distortion is the result of conducting the experiment in Latent Space. The VQGAN is likely producing an incorrect latent approximation of our images, causing us to lose the finer details. This also seems to be present in the example colorized images in the original BBDM paper. This was likely not mentioned given that that research paper focused on tasks where the input image does not possess fine details (e.g. sketch-to-photo) and the input and output images are very different (unlike colorization where they are very similar).

V. Conclusion

According to our experiments, the Transformer model is the best model for fully-automated image colorization. Even considering that we had to utilize a pre-trained Color Upsampler, the poor performance of our Color Upsampler was a result of the relatively small dataset we trained on and not the model itself. The pre-trained Colorizer & Color Upsampler were both trained for roughly 600 000-700 000 steps, meaning that we could have achieved the state-of-the-art results at no extra computation cost since the rate of the number of training steps is independent of the size of the dataset. Although inference time is long, the results have been the most consistent.

As we have experienced, GANs are hard to train because of their instability, therefore, they are generally not the best model to use if better alternatives are available.

Although the finer details of images were lost, the consistency of the outputs from BBDM suggest that it could outperform ColTran if a higher downsampling factor is used. Given the research papers that exist, we are aware of Diffusion Models that can outperform Transformers at colorization (and image-synthesis in general). However, we were unable to find well-documented/official Conditional Diffusion Models (CDMs) and had to solely rely on the BBDM, which had not been extensively tested on colorization and was designed for a different set of image-to-image tasks. Such developed CDMs being hard to find is likely a result of Text-Based Diffusion Models being more prevalent than pure Image-to-Image Diffusion Models, given that with the former, one can adjust an output image to their specification by using text prompts in addition to other images.

A. Future Work

We are interested in observing the effects that higher downsampling factors could have on a colorization BBDM, largely because we only used the lowest downsampling factor of 4. Furthermore, to our knowledge, the use of Latent Space for colorization models in general is an area yet to be

explored, therefore, we would also be interested in seeing its application on other types of models besides Diffusion.

REFERENCES

[1] Bo Li, Yu-Kun Lai, and Paul L Rosin. A review of image colourisation. Handbook Of Pattern Recognition And Computer Vision, pages 139–157, 2020.

[2] Peerspace. When was Color Photography Invented. https://www.peerspace.com/resources/when-was-color-photography-invented., 2015. Accessed: 2023-05-14.

[3] Zezhou Cheng, Qingxiong Yang, and Bin Sheng. Deep colorization. In Proceedings of the IEEE international conference on computer vision, pages 415–423, 2015.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. Communications of the ACM, 63(11):139–144, 2020.

[5] Jason Brownlee. How to identify and diagnose gan failure modes. Internet: https://machinelearningmastery. com/practical-guide-to-gan-failuremodes, 2019.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[7] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. ACM computing surveys (CSUR), 54(10s):1–41, 2022.

[8] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In International Conference on Machine Learning, pages 2256–2265. PMLR, 2015.

[9] Lilian Weng. What are diffusion models? lilianweng.github.io, Jul 2021.]

[10] Yun Cao, Zhiming Zhou, Weinan Zhang, and Yong Yu. Unsupervised diverse colorization via generative adversarial networks. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 10, pages 151–166. Springer, 2017.

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei 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.

[12] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. arXiv preprint arXiv:1912.12180 (2019)

[13] Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. Colorization transformer. arXiv preprint arXiv:2102.04432, 2021.

[14] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In ACM SIGGRAPH 2022 Conference Proceedings, pages 1–10, 2022

[15] Papers With Code. Colorization on ImageNet val. https://paperswithcode.com/sota/colorization-on-imagenet-val, 2022. Accessed: 2023-05-14.

[16] Bo Li, Kaitao Xue, Bin Liu, and Yu-Kun Lai. BBDM: Image-to-image translation with Brownian bridge diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1952-1961. 2023.

[17] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10684-10695. 2022.

[18] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, San- jeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International journal of computer vision, 115:211–252, 2015

Code will be made publicly available at:
https://github.com/gabriel-nkole/FAIC